Deep Learning Neural Networks based Algorithmic Trading Strategy using Tick by Tick and Order Book Data

Thesis · January 2019

DOI: 10.13140/RG.2.2.31950.31040

CITATIONS
0

READS
4,072

1 author:

Jaime Nino
National University of Colombia
10 PUBLICATIONS **106** CITATIONS

SEE PROFILE

UNIVERSIDAD NACIONAL DE COLOMBIA

Deep Learning Neural Networks based Algorithmic Trading Strategy using Tick by Tick and Order Book Data

Jaime Humberto Nino Pena

Universidad Nacional de Colombia
School of Engineering
Department of Industrial and Systems Engineering
Bogotá, Colombia

2018

Deep Learning Neural Networks based Algorithmic Trading Strategy using Tick by Tick and Order Book Data

Jaime Humberto Nino Pena

Thesis submitted as requirement to obtain a:
PhD. In Systems and Computer Engineering

Advisor:
PhD. German Jairo Hernandez Perez
Research Field:
Machine Learning, Computational Finance
Research group:
Algos

Universidad Nacional de Colombia
School of Engineering
Department of Industrial and Systems Engineering
Bogotá, Colombia
2018

Dedication

To my father and mother, they are magnificent, I am who I am thanks to them, their teachings and their values are a key stone for me as a person.

To my sister, what she has done motivates me to continue challenging myself.

To Rafa, my best friend who has been always on my side.

Acknowledgements

I thank to Colombian Government for the funding given through Colciencias, to the University for welcoming me again, to German, my advisor, who challenged and triggered myself to dive into this path, to my fellow classmates and Algos Research Group members for their insights, opinions and offered collaboration while doing my PhD.

Awards, Publications and Conferences

Awards

- Selected Postgraduate Student from Systems and Industrial Engineering Department to be part of Summer Data Science Course at Stevens Institute of Technology, NJ, USA. June - July 2015
- Fees exemption due to the fact of having the highest academic average for semester I in 2015. Meeting Number 013 of August 13th, 2015, Council of Engineering Faculty.
- Best paper award WEA 2016.
- Top 14 Quantopian Algorithmic Trading Contest No. 19 July to December 2016.
- Ingenio Awards for Machine Learning Implementation at Fondo Nacional del Ahorro for Customer Service, October 2018.

Publications

- Sandoval J., **Nino J.**, Cruz A., Hernandez G., Detecting Informative Patterns in Financial Market Trends based on Visual Analysis, International Conference on Computational Science (ICCS) San Diego, California, Procedia Computer Science Volume 80, pages 752-761, $https://doi.org/10,1016/j.procs$,2016,05,365, ISSN:1877-0509, 2016.
- Arevalo A., **Nino J.**, Hernandez G., High Frequency Trading Strategy based on Deep Neural Networks, Intelligent Computing Methodologies, ICIC 2016, Lecture Notes in Computer Science, vol 9773, Springer. $https://doi.org/10,1007/978-3-319-42297-8\_40$, Print ISSN: 0302-9743, Online ISSN: 1611-3349, 2016.
- **Nino J.**, Hernandez G., Price direction prediction on high frequency data using Deep Belief Networks, Applied Computer Sciences in Engineering, WEA 2016, Communications in Computer and Information Science Series, Volume 657, $https://doi.org/10,1007/978-3-319-50880-1\_7$, Print ISSN: 1865-0929, Online ISSN: 1865-0937, 2016.
- **Nino J.**, Garcia, C., Hernandez, G., A Complex Network Approach to Identify Potential Financial Scandals the Colombian Market Case, Polibits Research Journal on Computer Science and Computer Engineering with Applications, Issue 56, pages $39-44$, $https://doi.org/10,17562/PB-56-5$, Print ISSN: 1870-9044, Online ISSN: 2395-8618, 2017.

- Arevalo A., **Nino J.**, Hernandez G., Sandoval J., Leon D., Aragon A., Algorithmic Trading Using Deep Neural Networks on High Frequency Data, Applied Computer Sciences in Engineering, WEA 2017, Communications in Computer and Information Science, vol 742. Springer, Cham, $https://doi.org/10,1007/978-3-319-66963-2\_14$, Print ISSN: 1865-0929, Online ISSN: 1865-0937, 2017.

- Leon, D., Aragon, A., Hernandez, G., Sandoval, J., Arevalo, A., **Nino J.**. Clustering algorithms for Risk Adjusted Portfolio Construction, Procedia Computer Science, Vol 108, pp $1334-1343$, $https://doi.org/10,1016/j.procs$,2017,05,185, ISNN: 1877-0509, 2017.

- Arevalo A., **Nino J.**, Leon D., Hernandez G., Sandoval J. (2018) Deep Learning and Wavelets for High Frequency Price Forecasting. In: Shi Y. et al. (eds) Computational Science ICCS 2018. Lecture Notes in Computer Science, vol 10861. Springer, Cham, $https://doi.org/10,1007/978-3-319-93701-4\_29$, Print ISSN: 0302-9743, Online ISSN: 1611-3349, 2018.

- **Nino J.**, Arevalo A., Leon D., Hernandez G., Sandoval J., Price Prediction with CNN and Limit Order Book Data. Applied Computer Sciences in Engineering, Part 1, pp $124-135$, $DOI : 10,1007/978-3-030-00350-0\_11$, Print ISSN: 1865-0929, Online ISSN: 1865-0937, 2018.

- **Nino J.**, Hernandez G., Arevalo A., Leon D., Sandoval J., CNN with Limit Order Book Data for Stock Price Prediction, Future Technologies Conference, FTC 2018, Advances in Intelligent Systems and Computing, vol 1, pp $444-459$. Springer, Cham, $https://doi.org/10,1007/978-3-030-02686-8$, Print ISSN: 2194-5357, Online ISSN: 2194-5365, 2018.

Conferences

- Price Prediction on high frequency data using Deep Belief Networks, Workshop Engineering and Applications, WEA Bogota, October 2016.

- A Complex Network Approach to Identify Potential Financial Scandals the Colombian Market Case, Mining Intelligence and Knowledge Exploration Conference Series, Mexico City, November 2016.

- Sampling and Streaming Techniques for Big Data of Systematic Risk Surveillance and Regulatory Control of Financial Systems. Latin-American Congress of Probability and Mathematical Statistics CAPLEM, Costa Rica, December 2016.

- High Frequency Trading Strategy based on Deep Neural Network, 6th High Frequency Trading Strategy, Hoboken NJ, November 2016.
- Guest Speaker High Frequency Trading Strategy based on Deep Neural Network Deep Learning Finance Summit, June $1-2$ 2017, London.
- Guest Speaker Machine Learning for Finance, Bolsa de Valores de Colombia, April 2018.
- Price Prediction with CNN and Limit Order Book Data, WEA Medellin, October 2018.
- CNN with Limit Order Book Data for Stock Price Prediction, FTC SAI Conference, Vancouver, Canada, November 2018.

Abstract

This work presents an innovative and highly competitive Algorithmic Trading (AT) Strategy, based on a Convolutional Neural Network price direction predictor that uses High Frequency (HF) transactions and Limit Order Book (LOB) data. Information used includes data from US and Colombian market. Data processing include more than 5 million raw data files of 21 stocks from different industries (Energy, Finance, Technology, Construction, among others).

Since data include two different sources (Transaction and LOB), applying feature engineering is necessary to homogenize inputs. For transaction data, an image-like representation (Grammian Angular Field GAF) is used. It converts Financial Time Series (FTS) to polar coordinates and creates a kernel based on cosine differences. Additionally, this work proposes a transformation for LOB data. This representation includes all available information deviated from LOB raw data and it will create an image-like representation of LOB. These two sources will feed up into a proposed 3D-Convolutional Neural Network (3D-CNN) architecture that generates price direction predictions.

These predictions will serve as a trading signal generator for two Algorithmic Trading Strategies. Both of them take real market constrains into consideration, such as liquidity provision, transaction costs, among others. The two proposed strategies works under different risk aversion constrains.

Results from the proposed 3D-CNN predictor present a strong performance, ranging between 70 % and 74 % in Directional Accuracy (DA), while reducing model parameters as well as making inputs time invariant. Moreover, trading strategies results illustrate that the proposed CNN predictor can lead to profitable trades and liquidity improvement in the Colombian Market.

Testing results for both AT strategies on Colombian Market Data lead to interesting findings. Under different constrains of take profit, stop loss and transaction cost, both strategies aggressive and conservative lead to positive returns over the same period of time. Moreover, results of number of trades performed by the aggressive AT helps to understand how AT may impact positively liquidity provision in developing financial markets.

Keywords: Computational Finance, Deep Learning, Convolutional Neural Networks, Multimodal Representation Learning, LOB Data, Tick Data, Algorithmic Trading

Strategies, High Frequency Trading.

Resumen

Este trabajo presenta dos estrategias algorítmicas de trading, basadas en un método innovador y altamente competitivo de redes convolucionales para predecir de la dirección en los precios de series financieras de tiempo de alta frecuencia, tanto del Libro de Ordenes como en las Transacciones. La información usada incluye datos del mercado americano y colombiano. Se procesaron más de cinco millones de archivos con información de 21 acciones de diferentes sectores (energía, financiero, tecnología, construcción, entre otros).

La información de entrada incluye dos fuentes de datos diferentes (Transaciones y Libro de Ordenes), por lo cual se hace necesario aplicar ingeniería de características para homogenizarla. Para la información de las transacciones, se usó una representación basada en imágenes con una transformación conocida como Gramian Angular Field (GAF). Ésta convierte una serie de tiempo en coordenadas polares y crea un kernel basado en diferencia de cosenos. Además, este trabajo propone una transformación del Libro de Órdenes. Esta representación incluye toda la información disponible del Libro de Órdenes y la transforma a una imagen. La información representada se pasa a una arquitectura de red convolucional propuesta, la cual genera predicciones de la dirección de los precios. Las predicciones servirán de señales de negociación para dos estrategias de trading algorítmico. Ambas incluyen restricciones reales de mercado, como niveles de liquidez y costos de transacción. Las dos estrategias propuestas trabajan bajo differentes condiciones de riesgo.

Los resultados de predicción de la red convolucional propuesta presenta un desempeño entre el 70 % al 74 % de precición direccional; a la vez que reduce los paramétros del modelo y hace las entradas invariantes en el tiempo. Adicionalmente, los resultados de las estrategias de negociación ilustran que el predictor convolucional puede liderar a generación de ganacias y mejoras de liquidez en el mercado colombiano.

Las pruebas realizadas para las dos estrategias de trading en el mercado colombiano conllevan interesantes hallazagos. Bajo diferentes condiciones de take profit, stop loss y costos de transacción, tanto la estrategia agresiva como la conservadora reportaron retornos positivos para el mismo período de tiempo. Adicionalmente, la estrategia agresiva permite entender el impacto positivo en liquidez para mercados financieros emergentes.
Palabras clave: Finanzas computacionales, Aprendizaje profundo, Redes convolucionales, Aprendizaje de representación, Libro de órdenes, Transacciones, Estrategias de Negociación algorítmica, Negociación de alta frecuencia.

Table of Contents

Abbreviations and Symbols

Abbreviations

Abbreviate	Meaning
ADR	American Depository Receipt
ANN	Artificial Neural Network
AT	Algorithmic Trading
ASX	Australian Stock Exchange
BP	Back Propagation
BVC	Bolsa de Valores de Colombia (Spanish name for Colombian Stock Exchange)
CBT	Computer-Based Trading
CNN	Convolutional Neural Network
DBN	Deep Belief Network
DAX	Deutsche Bourse
DL	Deep Learning
$DLNN$	Deep Learning Neural Networks
DNN	Deep Neural Networks
ETF	Exchange Tradeable Fund
$FFNN$	Feed Forward Neural Networks
FTS	Financial Time Series
GAF	Gramian Angular Field
GPU	Graphic Processing Unit
GRU	Gate Recurrent Unit
HFT	High Frequency Trading
LFT	Low Frequency Trading
LOB	Limit Order Book
LSE	London Stock Exchnage
$LSTM$	Long Short Term Memory
ML	Machine Learning
MLP	Multi-layer Perceptron

Abbreviate	**Meaning**
MTF	Markov Transition Field
$NYSE$	New York Stock Exchange
RBM	Restricted Boltzmann Machine
RNN	Recurrent Neural Networks
SL	Supervised Learning
SVM	Support Vector Machines
TSE	Toronto Stock Exchange
UL	Unsupervised Learning

1. Introduction

Recent advances in computer science and technology, including hardware and software with enormous processing capabilities, have facilitated the fact that machine-based agents make trading decisions on short time-scales. This activity is known as Algorithmic Trading (AT), an emergent discipline that combines Computer Science with Finance, and involves portfolio management and allocation [65], modelling of Financial Time Series (FTS) and trade execution on Financial Markets [101].

In general, Computer-Based Trading (CBT) is a trading system used for financial firms and individual investors to trade financial instruments based on trading strategies. CBT has two main categories: High Frequency Trading (HTF) and Algorithmic Trading (AT). HFT uses an extraordinary high speed computer software for generating, routing and executing trade orders in very short time horizons (milliseconds to seconds). This kind of CBT includes the use of co-location services to minimize network latency [103]. Additionally, AT, as defined by the European Commission Directorate General Internal Market and Services (DG MARKT), is the use of computer programs to enter trading orders where the algorithm decides on order execution aspects such as time, quantity and price.

Financial Markets are a very important component of the economy, because they allow improving capital allocation. This is channeling financial resources to real actors of the economy. If well done, such actor can grow its business, generates new jobs, helps to improve communities and adds value to the economy and the society [33], [92], [95]. In order to make this kind of impact, financial markets need to become very efficient. Efficiency aims to improve liquidity provision, to reduce transaction costs and improve price discovery [15], [72], [103]. **AT** helps to improve such wanted characteristics [15], [72].

As any other activity, trading on financial markets should generate profits for the companies and individuals engage on it, therefore AT must be profitable. As a result, AT must include good analysis of FTS. Since Financial Markets exhibit complex systems properties [76], modelling of FTS continues to be a recurrent problem to mathematicians, engineers and statisticians, due to their noisiness and non-stationary behavior [1], [9], [106] that is present

also in High Frequency Time Series [30].

Over the decades, different modelling techniques, including Statistical and Machine Learning (ML), have been applied for analyzing FTS [9]. Research during the last 20 years shows that ML approaches are more effective than classical statistical techniques. The reason is that ML deals much better with complexity and non-linearity of FTS [38], [53], [79], [96], [108], [111]. ML offers a wide variety of modelling techniques, including clustering, Support Vector Machines (SVM), Decision Trees and Artificial Neural Networks (ANN) among others. Within this set of elements, ANN are the most popular [38], [53], [79], [96], [108], [111].

1.1. Problem Statement

This work includes topics from different topics: machine learning, financial markets, FTS analysis and AT. As a result, it is necessary to understand constrains that emerge from each of those. Having said that, it is important to mention that FTS are challenging due to their characteristics, including noisiness, non stationary behaviour and the impact of externalities [1], [9], [30], [106].

Moreover, financial markets exhibit characteristics of complex systems [76] and efficiency improving is the most important fact. This means that characteristics such as liquidity provision, price discovery, transaction cost reduction are fundamental to make markets more efficient. Having said that, AT insertion has effectively lead to efficiency improvements within financial markets. In fact, automated market agents have affected positively transaction costs, liquidity and price discovery processes [103]. However, AT has been devoted mostly to speed action, which reduces on one hand the use of computation intelligence and on the other important constrains such as market impact while doing [21], [15], [37], [101], [103]. Finally, emergence of applications to model FTS using ML techniques from late 80's is notorious, given the fact that ML handles non-linearities much better than analytical models. However, limited learning capabilities of most ML techniques makes prediction of FTS an on going challenging problem [42], [94], [88], [115].

In consequence, in order to overcome these set of issues, this work will apply DL, the latest development wihtin ML techniques, in order to model FTS in high frequency domain (seconds) to generate trade signals that can be feed or used for an AT system that generates profits while testing its ability to handle financial markets constrains (liquidity, transaction cost and price discovery). Table **1-1** presents a summary of the problem scope for this work.

Table 1-1.: Problem Scope.

Topic	**Issue**
Financial Markets	Complexity [76]. Constant need to become more efficient [15], [33], [72], [92], [95], [103].
FTS	Noisiness and non stationary behavior [1], [9], [30], [106].
ML	ANN learning limitations [29], [100].
CBT	Constrains regarding order quality vs profitability generation [15], [103].

Therefore, the issue of AT that uses HF data is not only about computational speed or profitability constrains, but also computational intelligence. Since, FTS are complex, chaotic and noisy, it is necessary to explore a different ML approach that can overcome known issues of ANN, such as poor generalization due to gradient vanishing in multilayered architectures [29], [100]. Deep Learning (DL) appearance gives a new possibility to model these noise and non-linear data (FTS). Since DL mimics how brain of mammals process information in a hierarchical way, allowing to build complex representations of sensory inputs based on regularity learning of such inputs [4], [45], [61], [76], [75]. In other words, DL may learn non linear patterns by combining initial simple sparse representation into more complex ones layer by layer as a mammals brain processes information.

1.2. Justification

Even tough forecasting FTS continues to be a challenging task [1], [9], [106], DL, through sparse representation of features, can provide a further development to CBT applications. Having the major role that financial markets plays [33], [92], [95], it is very important to explore possible impacts of CBT under near close market conditions by building an AT strategy that integrates computational intelligence and market constrains, while trading on short time frames.

1.3. Methodology and Objectives

This research is an exploratory study about building an AT strategy capable of generate trade orders for short time frames, with trading signal generated by a Convolutional Neural Network. This ML technique takes two different sources of information: LOB and transaction data. Given the fact that these sources are different modality, it is necessary to homogenize

them under a common representation. Results of the AT will be tested through simulation. As a result, the following list shows the central activities for this work:

- To build a DL architecture that predicts price direction, which includes:
  - Data preprocessing and representation.
  - DL topology selection including number of layers and associated characteristics of chosen topology.
  - Definition of software framework and hardware infrastructure to use for training and testing purposes.
- To elaborate an AT strategy that includes trading of different assets, handles real market constrains and integrates predictions from the DL architecture.
  - Analyze market data to find similarities and differences among instruments.
  - Analyze financial market aspects such as liquidity
  - Analyze aspects of security trading such as entry points, exit strategies, trading times, among others.
- To evaluate AT strategy performance.
  - Definition of time span for evaluation.
  - Collect and analyze data from trading simulation environment.
  - Analyze results.

1.4. Document Organization

This works continues as follows:

- Chapter two provides definitions on AT.
- Chapter three informs about Multimodal Learning Representation of Financial Data (Transaction and LOB), and proposes a representation of these data. This is because they are different sources of information for this work.
- Chapter four gives information about ML techniques in general and DL in particular, and proposes a 3D-CNN Multimodal architecture for Algorithmic Trading Strategies.
- Chapter five provides results on experiments conducted, both for FTS modelling and its applications to an AT.
- Chapter six presents the main conclusions and recommendations.

2. Algorithmic Trading

Financial markets in general, and AT in particular are key topics for this work. Therefore, this chapter makes a literature review in order to introduce key financial concepts about: Algorithmic Trading, Algorithmic Trading Strategies and their relevance for Financial Markets. Moreover, it presents important considerations of AT from two different perspectives: financial and computational.

2.1. Definitions

2.1.1. Algorithmic Trading

The European Commission Directorate General Internal Market and Services (DG MARKT) [103] defines AT as the use of computer programs to enter trading orders, where the algorithm decides on aspects of the order execution such as time, quantity and price. Otherwise, the US Commodity Futures Trading Commission (CFTC) [103] defines AT as the use of algorithms to place trade orders. For [15], AT is the use of computer algorithms to make trading decisions without human intervention.

AT development is possible thanks to the evolution of technology as a whole, hardware and software with enormous processing capabilities, allowing non-human-based agents to make trading decisions on very short time-scales. AT decisions may include analyses of factors such as economic fundamentals, news, price changes. In order to process sources of information, AT can use different methods or techniques, including statistical or machine learning, to learn from monitoring sequences of events in a particular market [56], [103]. Moreover, AT involves dynamic planning and market reasoning [101] maximize profits under two major constraints [37], [49]:

- Reducing market impacts due to the possibility of sending large orders.
- Optimizing transactions costs given the frequency of the trades.

2.1.2. High Frequency Trading

European Securities and Markets Authority (ESMA) [15] defines HFT as trading activities that employ sophisticated algorithms to interpret signals from markets and trading strategies implementation that generates a huge number of orders. These orders are transmitted with low latency to venues or exchanges in very short time frames. Moreover, positions are closed out at the end of the trading day and involve the own capital of the brokerage firm, not its clients.

The Securities and Exchange Commission of the US (SEC) [15] defines HTF as an activity made by professional traders in which they generate a large number of trades on a daily basis, using sophisticated algorithms for routing and executing orders in very short time frames, without carrying in positions for the next day. They also use co-location services and data feed services from exchanges to minimize latency. Order execution time-frame ranges from microseconds to seconds.

In fact, developments in telecommunications, hardware capabilities and computer science during the last ten years have enabled computers to make autonomous trading decisions in financial markets across the world. Technological advances have made possible to process huge quantities of data to act in milliseconds in order to take advantage of any market opportunity. HFT is mostly devoted to arbitrages [72], [103], making markets more efficient, as well as more liquid, due to submission of huge quantities of orders to different exchanges [67].

2.1.3. Low Frequency Trading

LFT differs from HFT in order execution time-frames. This umbrella qualifies systems that take minutes to weeks to submit orders. LFT is mostly based in quality trading signals rather than speed, As a result, aspects such as collocation and latency (network, hardware, software) are not as relevant as in HFT. However, data analysis and modelling of FTS behavior are more relevant [101].

2.1.4. Trading Strategies

ESMA defines a trading strategy as a disciplined method to buy or sell a financial asset, which involves working with a pre-defined set of rules [15]. Trading strategies categories include:

- Fundamental strategies based on macroeconomic factors as well as industry and company specific factors such as financial reports, management members, business strategy, rules to conduct businesses, political stability of operating regions, among others [49], [94], [101].
- Technical strategies based on visual analysis of historic prices and volume data. They include use of statistical formulas over the price and/or volume time series and their main objective is to find common patterns across data [49], [94], [101].
- Quantitative strategies include intensive use of computational tasks. This category of trading strategies focuses on stochastic or machine learning techniques [101] to model assets price behavior.

2.2. Algorithmic Trading Advantages

Nevertheless, the trading frequency of algorithms, and inception of machines into trading activities have radically changed financial markets from different perspectives [15], [20], [49], [67], [72], [103].

2.2.1. Perspective 1: Financial Markets Principles

Liquidity Provision

Liquidity is a fundamental characteristic of any financial market, since its lack is core for many financial crisis. Since High Frequency Traders are usually large financial institutions, they act as market makers, therefore there are orders at many different price levels and algorithms are ready to execute to balance any liquidity unbalance to either side (bid/ask).

Several studies in different markets [15], [20], [49], [67], [103] present a consistent reduction in spreads, as well as a persistent growth of traded volumes, when compared to Limit Order Book (LOB) behavior during different years with AT.

Price Discovery Improvement

Price discovery by definition [15] is the capacity of a market to find the best price to buy/sell any asset. Given the fact that AT analyzes market data continuously, it can better predict market order flow; i.e. how likely prices will change in the near future. Therefore, it can better adjust orders to discover the best possible prices to execute transactions [15], [20], [67], [103].

Transaction Costs Reduction

Better liquidity and price discovery improvement translates into more market interest; that is, more investors will be willing to participate and trade, thus more transactions occur in the exchanges. In consequence, AT is a key driver to transactions cost reduction during the last ten years. This happens because exchanges are able to reduce their marginal operating cost due to more transactions from automated market participants [20], [15], [67], [103].

2.2.2. Perspective 2: Computer Engineering

Big Data

Financial information is widely available, from market data, to analyst researches, and news. As a result, financial markets generate huge quantities of data on a daily basis. To give an example of this phenomena, one day Limit Order Book events of NYSE accounts for nearly 25 PB, bringing Big Data processing and analysis to the industry[1].

Machine Learning

Analyses and decisions of these large data sets involve machine-learning modelling [20]. As explained in the next chapter, the literature is full of papers applying different ML techniques intending to model financial data to gain an edge to predict future price direction.

Hardware and Networking

Latency reduction is necessary when doing AT. Use of GPA, GPUs, parallel processing, collocation, fiber channels and any other hardware improvement is a common practice in this industry [20]. In fact, [67] shows that matching engine latency of LSE has reduced from 0.6 microseconds in 2000 to 0.113 nanoseconds in 2011.

Despite its advantages, CBT in general is not apart from criticism. Studies of [15], [49], [67], [72] and [103] argue that, as improvements of liquidity and price discovery occur very fast, they become short lived or limited in time, therefore improvements lack of continuity. This generates an adverse position for low frequency traders when trying to liquidate their positions in particular situations, because they cannot compete in terms of speed with HF traders.

[1] Information collected from NYSE web site

From the Engineering point of view, some authors [15], [20] and [67] explain that algorithms tend to overreact in a very short term scales, causing huge imbalances in LOB, liquidity evaporation or spread changes. In spite of some disadvantages, CBT is still a major trend not only in developed but also in developing markets [20], [49].

2.3. Categories of Algorithmic Trading

2.3.1. Arbitrage

Arbitrage is the opportunity to profit from the simultaneous purchase of an asset in one market and the sale of the same asset in a different market, given price differences resulting in a risk-free profit [103]. Given the fact that CBT involves the analysis of huge quantities of data at very high speeds [37], strategies related to find statistical arbitrages between correlated assets are a common practice for CBT practitioners, particularly for those involved in HFT [103].

2.3.2. Market Impact Reduction

Market impact refers to the action of sending large orders of a particular asset to an exchange. As mentioned before, it is one of the issues when doing trading in general and AT in particular. As a result, a complete field of research have been devoted to market impact reduction. Algorithms specialize in splitting large orders, while keeping constraints of price. The main objective is to move large volumes of shares without giving any hint to market participants, particularly other algorithms [21], [37], [51], [67], [72], [101].

2.3.3. Reversal Engineering Applications

This category includes algorithms that monitor market activity in order to find or decipher what others trading algorithms do. This research filed is related to adversarial agents finding [20], [67], [72].

2.3.4. Intelligent Trading Execution

Due to the fast growth of CBT systems and competition among market participants [15], [20], [21], [67], [72], [94], [103], more attention has been devoted to add intelligence to trading algorithms, in order to reduce the lack of diversity of trading strategies when using AT [15],

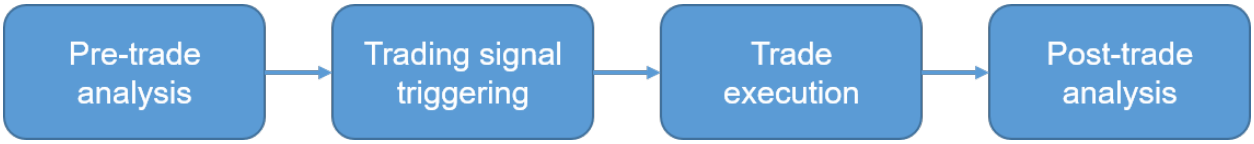

Figure 2-1.: AT System components. Source: [101].

[20], [67]. As a result, this category is for algorithms that trigger trading actions, that is, open and close positions while monitoring market activity. Trading of financial instruments occurs when market participants submit buy and sell orders using a centralized order book [25].

In summary, AT is a major ongoing trend in financial market worldwide. When execution time and frequency is a key driver, AT is known as HFT. In contrast, when there are no time constrains, AT is known as LFT. In both cases, AT helps financial markets efficiency. Machines make markets more efficient by increasing liquidity and improving price discovery. As more machines participate, data flow increase considerably. Consequently, big data and ML techniques flourish within the financial industry, resulting in a wide variety of specialized algorithms. Some of them reduce market impact, others specialize in eliminating arbitrages, others in trading execution and others in discovering trading rules of their automated counterparts [72].

Despite criticism CBT is well positioned. A literature review from [20] reflects that CBT systems account for almost 90 % of the transactions in NYSE, followed nearby LSE and TSE, where automated systems account for more than 80 % of the daily market volume. This trend is highly expanded in other developed markets, including DAX and ASX. Even developing markets as India are starting to take advantage of this trend, as quoted in [49]. Moreover, internet and technological advances have made possible that retail traders can access algorithmic trading platforms in different markets including currencies, commodities, stocks, bonds and others. This list includes names such as Dukascopy, Oanda, SaxoBank, Quantopian, QuantConnect, InteractiveBrokers just to name a few.

Finally, any AT system should include the components highlighted in Figure **2-1**, where pre-trade analysis refers to analyzing market constrains, such as liquidity, market impact; trading signal is related to modeling of FTS; trade execution refers to the ability of merging signals with market constrains and investor profitability needs; and post-trade analysis refers to open position management [72].

3. Multimodal Learning Representation of Transaction and LOB Data

Another important topic for this work is FTS. As mentioned earlier, this work uses multimodal inputs: LOB an transaction data. Therefore, this chapter will present first definitions for both LOB and transaction data. Because of their differences, it is necessary to homogenize them, therefore they are readable for a ML technique. In order to understand this process, this chapter continues presenting how represent these two different sources of information. In consequence, the reader will find information about a previous work [86] on LOB representation. This representation will be extended, and the new proposed representation (extended one) will include all information from LOB data. For transaction data this work will use a method introduced in [107]. The original paper is oriented to time series in general therefore, this will be the first application on FTS, as long as the author knowledge based on literature reviewed. Once data representation are introduced, data can be homogenized in order to feed them up into a ML technique. It is worth mentioning that one of the contributions of this work is the proposed representation for LOB data.

3.1. Definitions

3.1.1. Transaction Data

As shown in Figure **3-1**, a transaction occurs when a buy order equals the price of a sell order. Formally, a transaction $T = (p, q, t)$, where p_T is the price agreed between buyer and seller, q_T is the quantity exchanged and t_T is the transaction time stamp. Since transactions do not occur continuously, their mathematical expression is that of a time series [107]:

$$X = \{x_1, x_2, ..., x_n\} \tag{3-1}$$

FTS collect data of price changes over time due to the interaction of market participants (See Figure **3-1**). Therefore, forecasting FTS is a very challenging problem because they exhibit the following characteristics [100]:

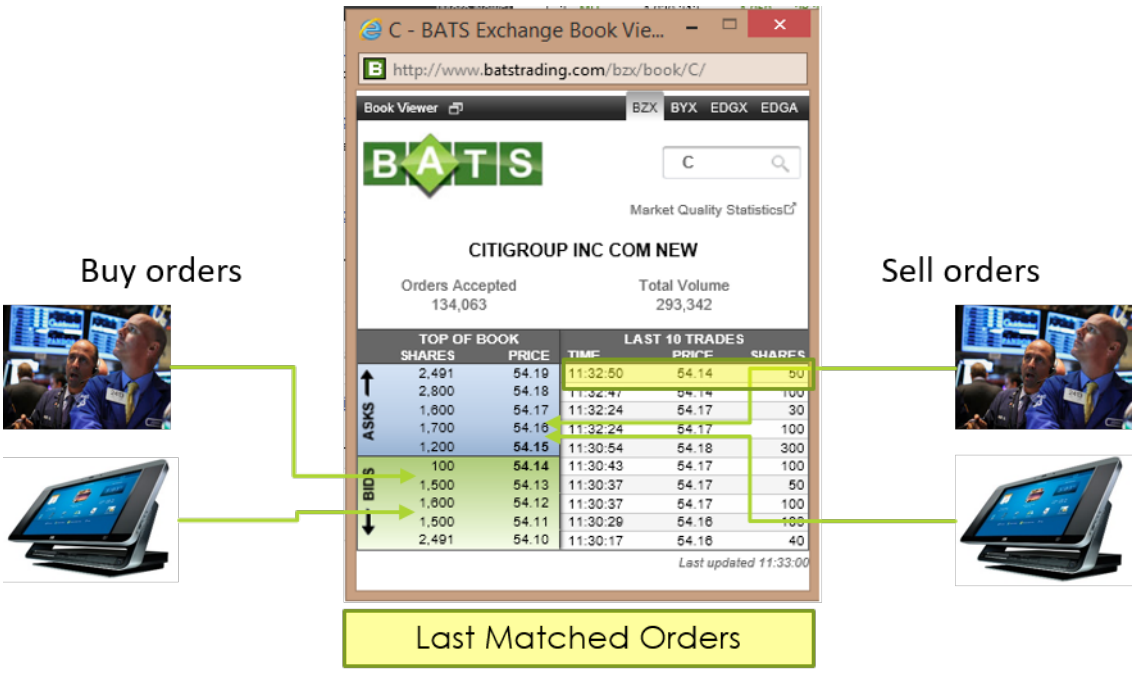

Figure 3-1.: Illustration of how market participants submit buy/sell orders and their matching. LOB image taken from www.batstrading.com.

- Unavailability to capture the full dependency between future and past prices, also known as noise.
- Non-stationary behavior since their distribution changes over time.
- Short-term randomness and long-term determinism, that is deterministically chaotic.

Information content in equation 3-1 is uni-variate, usually prices. However, sometimes transaction data may include prices and traded quantities, generating a bi-variate time series. This work will use uni-variate information when referring to transaction data.

3.1.2. Limit Order Book

Limit Order Book (LOB) is the recording of buy/sell intentions of all market agents in a given exchange. LOB recording centralizes all market participants' intentions. It includes a time-stamp, quantity and price to buy/sell. This information is very important for HF traders, because it includes the foundation principles of financial markets: liquidity and price discovery; therefore, it gives a sense of volume, volatility, market depth among others market properties [51]. LOB is organized by best price and by arrival time, and an order

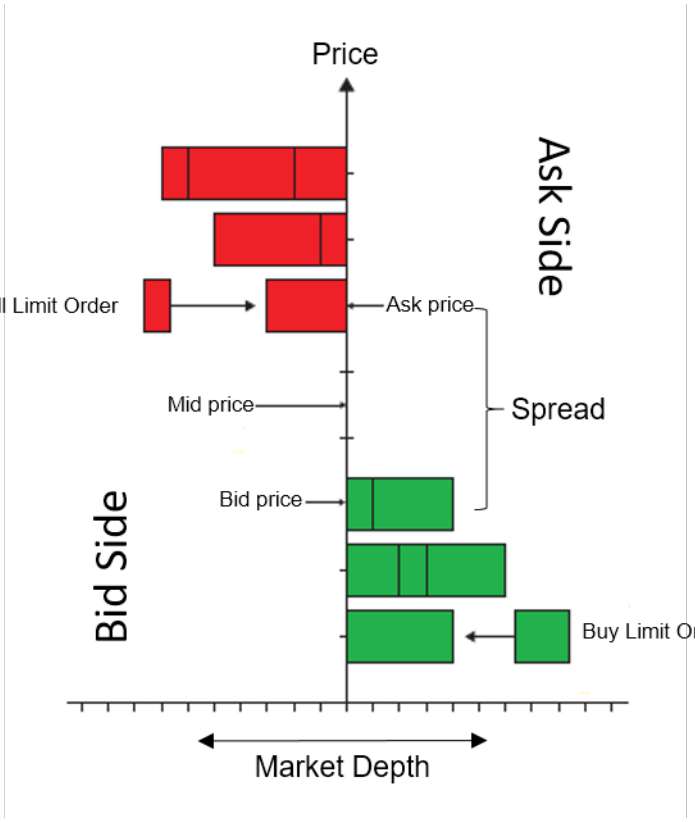

Figure 3-2.: LOB concepts adapted from [40].

matches when the buy price is equal to the sell price [25], [27], [40], [80], [85]. Order matching generates transaction data, as defined previously. Figure **3-1** illustrates how this interaction happens among market participants (human and/or non-human).

Formally, an order $x = (p, q, t, s)$, sent at time t_x, price p_x, quantity q_x (number of shares) and side (buy / sell), is a commitment to buy/sell up to q_x units of an asset at price p_x. Arrival time and quoted price determine orders sorting. Sell orders have larger prices than buy orders [27], [40], [84], [82], [104]. There are a particular vocabulary strongly associated to LOB [40], [84]:

- **Spread size**: It is the difference between the best sell and buy prices.
- **Bid price**: It is the highest price among all active buy orders at time.
- **Ask price**: It is the lowest price among all active sell orders at time.
- **Best quotes**: It is a tuple, formed by the best Bid Price, and the best Ask Price.

- **Market deep**: It is the number of different prices quoted (registered with intentions) on LOB.
- **Order type**: Submitted orders can be either insert, modify or cancel. An *insert* order is a new order send by a market agent; a *modify* order is an update over an existing order; and a *cancel* order deletes an order sent previously.

Data for this work do not include modify or cancel orders, therefore, a LOB $L(t)$ is the set of all active orders x at time t. Dynamics of LOB are complex [27], [40], [113], since they capture interactions among market agents with different points of view and different trading strategies. LOB concepts are illustrated in Figure **3-2**, For a particular time t. General speaking, LOB data can be seen as a collection of FTS. Transactions occur between best quotes, therefore LOB data is naturally multivariate. Given the fact that, at each point in time t LOB data may have different market deep (price lines), computationally speaking it is a collections of lists, where each list is a set of orders. In summary, at any particular time t_k, a LOB $L(t_k)$ contains the following information:

- A list of orders $x = (p, q, t_k, s)$ sorted by price and arrival time.
- Prices p can be repeated n times, this means that different market participants are willing to buy or sell at the same price.
- Quantities willing to be traded q differ depending of market participant characteristics. There fore, quantities q also vary even among orders at same prices p.

In consequence, LOB data present different modalities if compared to transaction data. As a result, analysis and modelling of these data is difficult. Moreover, this work adds an additional challenge, because LOB data and transaction data are different, resulting in a Multimodal Representation problem [8]. In other words, transaction data can be seen as a uni-variate or bi-variate data (price and quantities), whereas LOB data is a collection of time series, with variable dimensionality, because there is no market deep homogeneity for all times t.

3.2. Transaction Data Representation

Given the fact that this work uses multimodal data (transaction and LOB data), it is necessary to homogenize them, therefore, transaction and LOB can be fetch into the proposed computational model to predict price direction (Chapter 4). Because, this work proposes a 3D-CNN for prediction with two different inputs (transaction and LOB), data representation

becomes important in order to be able to use such data. In consequence, both transaction and LOB data will be represented as images.

In this section, transaction data will be represented as a matrix, which can be converted to an image. There are different techniques that allow representing uni-variate time series as images. In [107], authors explain an approach that underlines a kernel filter, to transform uni-variate time series into an image, named Gramian Angular Field, which takes advantages of angular measures (cosines) to represent Cartesian points into polar coordinates. Transactions to be represented will be those happening during the time interval of LOB taken for analysis. (See chapter five for specific details).

3.2.1. Gramian Angular Field

It is a transformation that maps Cartesian to Polar coordinates to then build a Gramian Matrix, or kernel that represents cosine angles between points [107]. This transformation offers the following properties [107]:

- It is bijective as $cos(\phi)$ is monotonic when $\phi \in [0, \pi]$.
- Polar coordinates preserve absolute temporal relationship. Moreover, they have the advantage that they differentiate an area under the curve for time stamps $[i, i+k]$ and $[j, j+k]$ that have the same values in cartesian coordinates. This happens due to area under the curve definition S. In Cartesian, $S_{i,j} = \int_{x(i)}^{x(j)} f(x(t))dx(t)$, in contrast of Polar coordinates definition that states $S_{i,j} = \int_{\phi(i)}^{\phi(j)} r[\phi(t)]^2 dx(\phi(t))$. As a result, in polar coordinates, values of S' are different for any time stamps $[i, i+k] = [j, j+k]$.

As a result, this transformation offers value differentiation at the very input of any computational model. The following are the step by step equations that need to be performed on any time series (Equation 3-1).

$$x'_l = \frac{(x_i - max(X)) + (x_i - min(X))}{max(X) - min(X)} \tag{3-2}$$

$$\phi = \arccos(x_l) \quad and \quad r = \frac{t_i}{N}, \quad t_i \in N \tag{3-3}$$

$$G = \begin{vmatrix} \cos(\phi_1 + \phi_1) & ... & \cos(\phi_1 + \phi_n) \\ \cos(\phi_n + \phi_1) & ... & \cos(\phi_n + \phi_n) \end{vmatrix} \tag{3-4}$$

$$G = X^T X^- \sqrt{I - X^2}^T \sqrt{I - X^2} \tag{3-5}$$

3.2.2. Results of Gramian Angular Field on HF FTS

Figure **3-3** shows the visualization of different FTS (transactions) after applying equations 3-2, 3-3, 3-4 and 3-5. This representation differentiates FTS peaks by colours intensity, helping to separate price variances within the original signal.

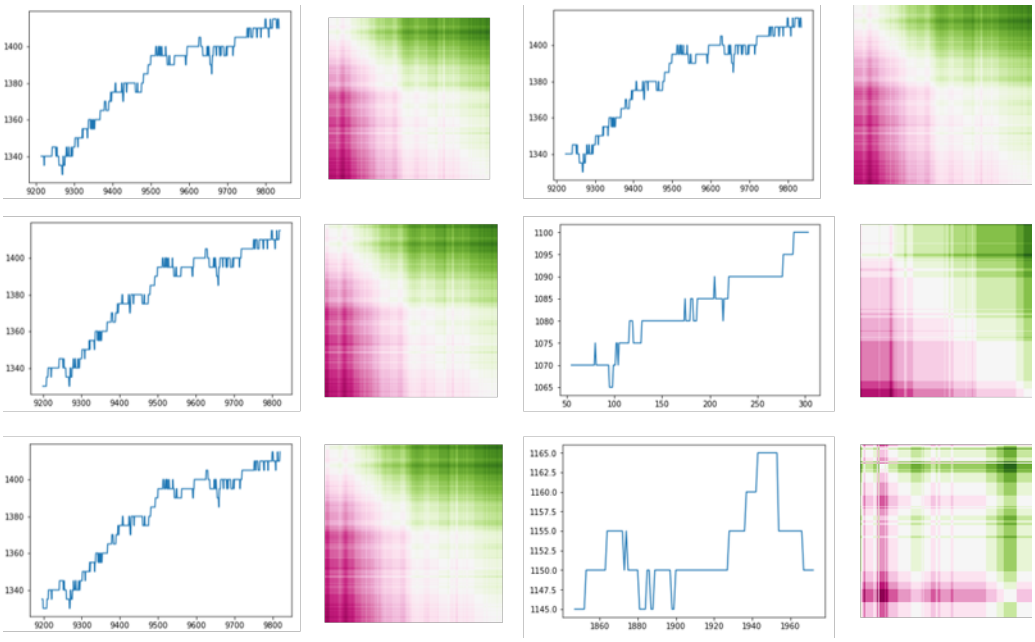

Figure 3-3.: Transaction Data as Image applying GAF. Up trend. Own elaboration based on [107].

Moreover, for FTS in up trending mode, higher prices are located upside on the image. Another interesting fact of this transformation is that color continuity signals liquidity. As the reader can notice, the last two images in the second column of Figure **3-3** present more squares, which are not present in other images of the same figure.

Figure **3-4** illustrates a representation of down trend transactions. As mentioned earlier, colors change positions. In this case, red color is at the right side of each image. Another interesting fact of this transformation can be seen in the second image of the second column. Flat periods are white colored.

Figure **3-5** show trendless transactions. As noticed earlier, white color is more common and grid patterns are more frequent for this kind of FTS. In consequence, GAF transformation also helps to identify liquidity levels, thanks to color continuity and direction separation due to peak colouring.

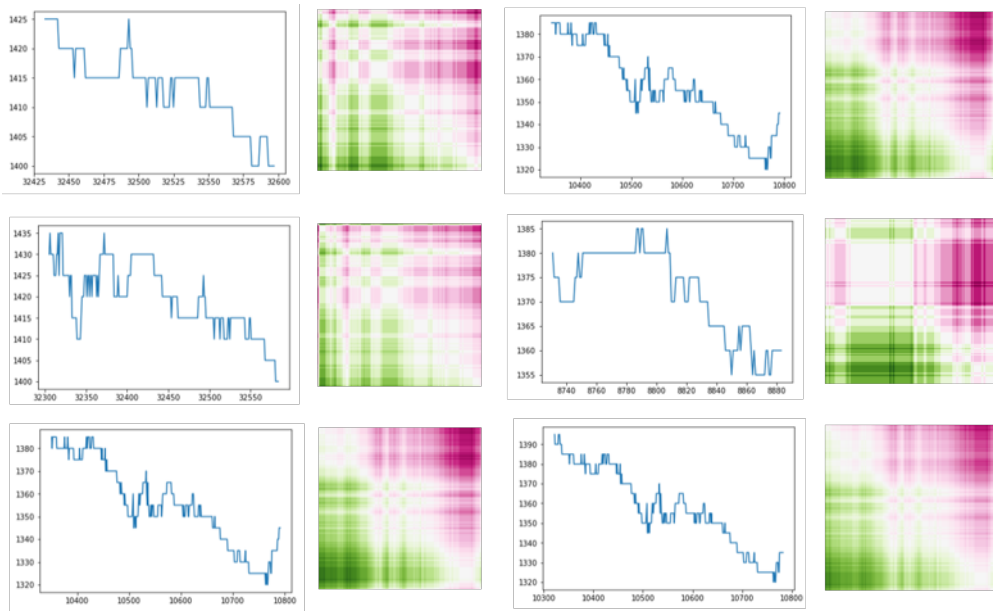

Figure 3-4.: Transaction Data as Image applying GAF. Down trend. Own elaboration based on [107].

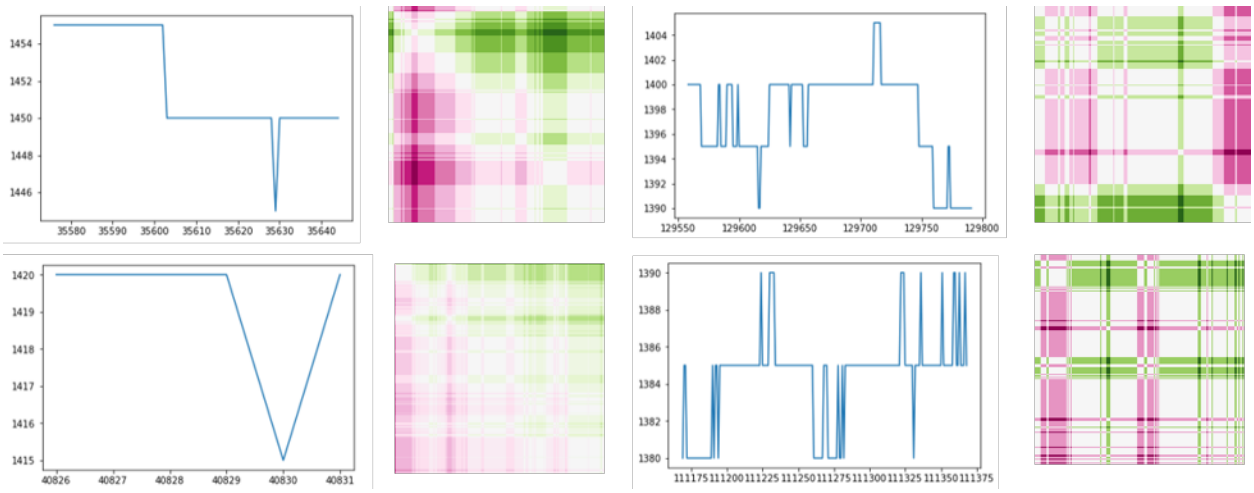

Figure 3-5.: Transaction Data as Image applying GAF. Trend less. Own elaboration based on [107].

As observed in Figures **3-3**, **3-4** and **3-5**, this representation informs:

- Original **FTS trend** information, depending on color distribution.
- **FTS trade-ability** information, depending of color gradient. This is a gain of information, because more transactions happening during a particular time have different gradient when compared to less transactions happeing for the same time period.

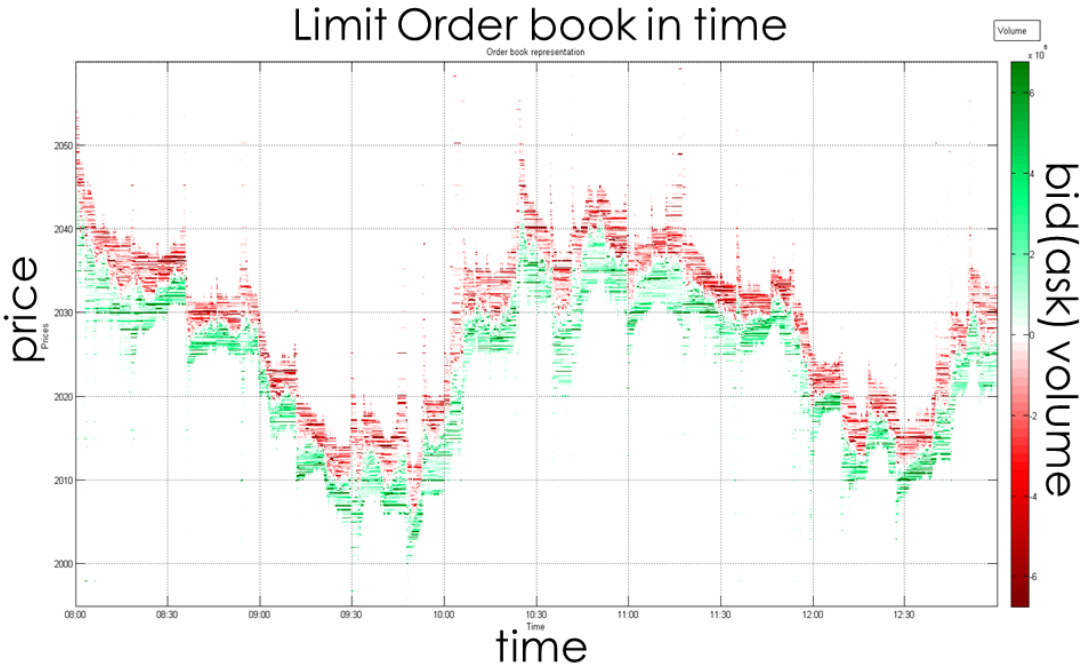

Figure 3-6.: Image like Representation of LOB Data based on quantities q at prices p, taken from [86].

3.3. Computational LOB Representation

As mentioned earlier LOB data include all trading intentions, which contain lots of information about price dynamics. Initially, LOB data can be represented as a matrix-like object (Figure **3-7**), based on information used in Figure **3-6**, where column labels are time stamps t, row labels are prices p and each cell contains the number of shares to bid/ask q. Due to order imbalances, the number of ask price (k) does not necessarily equal the number of bid prices j. Moreover, many cells in this matrix will have a 0 value, because it is necessary to fix matrix size for computational purposes. This basic representation of LOB was introduced in [86]. It only takes into account quantities q willing to trade.

3.3.1. Proposed 4-Channel Matrix LOB Representation

As mentioned earlier, information included in LOB is more than quantities. Therefore, it is possible to deviate or include additional variables to enrich such basic representation as presented below:

	t_0	t_1	...	t_n
$AskPrice_k$	$q_{a_{k,0}}$	$q_{a_{k,1}}$	...	$q_{a_{k,n}}$
$AskPrice_{k-1}$	$q_{a_{k-1,0}}$	$q_{a_{k-1,1}}$	...	$q_{a_{k-1,n}}$
⋮	⋮	⋮	⋱	⋮
$AskPrice_0$	$q_{a_{0,0}}$	$q_{a_{0,1}}$	...	$q_{a_{0,n}}$
$BidPrice_0$	$q_{b_{0,0}}$	$q_{b_{0,1}}$	...	$q_{b_{0,n}}$
⋮	⋮	⋮	⋱	⋮
$BidPrice_{j-1}$	$q_{b_{j-1,0}}$	$q_{b_{j-1,1}}$	...	$q_{b_{j-1,n}}$
$BidPrice_j$	$q_{b_{j,0}}$	$q_{b_{j,1}}$	...	$q_{b_{j,n}}$

Figure 3-7.: Matrix object representation. Own elaboration based on [86].

- The total quantity (volume) Q_px at one particular price.
- The maximum quantity (volume) at one particular price $maxVol = max(q_px)$. This information is important because it gives a sense of quantity (volume) distribution at each different price p to the representation.
- The total number of orders N at one particular price.

As a result, each cell for this new matrix contains a vector $v = [P_i, Q_i, N_i, maxVol_i]$, as shown in Figure **3-8**. Mathematically:

$$Q_{st} = \sum(q_{st}) \tag{3-6}$$

$$N = count_{p_{st}}(x_{st}) \tag{3-7}$$

$$maxVol = max(q_st) \tag{3-8}$$

It is important to notice that Figure **3-8** differentiates sides, that is, the first member of the vector v is 0 if and only if there is a bid order. Conversely, the second member of vector v has a 0 value if and only if there is an ask order. Therefore, equations 3-6, 3-7 and 3-8 change depending on the order side, as follows:

$$Q_{at} = \sum_k(q_{kt}) \quad or \quad Q_{jt} = \sum_j(q_{jt}) \tag{3-9}$$

$$N_{kt} = count(x_{kt}) \quad or \quad N_{jt} = count(x_{jt}) \tag{3-10}$$

$$maxVol_{kt} = max(q_{kt}) \quad or \quad maxVol_{jt} = max(q_{jt}) \tag{3-11}$$

	t_0	...	t_n
$AskPrice_k$	$[q_{a_{k,0}}, 0, n_{k,0}, max(q_{k,0})]$	...	$[q_{a_{k,n}}, 0, n_{k,n}, max(q_{k,n})]$
$AskPrice_{k-1}$	$[q_{a_{k-1,0}}, 0, n_{k-1,0}, max(q_{k-1,0})]$	...	$[q_{a_{k-1,n}}, 0, n_{k-1,n}, max(q_{k-1,n})]$
⋮	⋮	⋱	⋮
$AskPrice_0$	$[q_{a_{0,0}}, 0, n_{0,0}, max(q_{0,0})]$	...	$[q_{a_{0,n}}, 0, n_{0,n}, max(q_{0,n})]$
$BidPrice_0$	$[0, q_{b_{0,0}}, n_{0,0}, max(q_{0,0})]$	...	$[0, q_{b_{0,n}}, n_{0,n}, max(q_{0,n})]$
⋮	⋮	⋱	⋮
$BidPrice_{j-1}$	$[0, q_{b_{j-1,0}}, n_{j-1,0}, max(q_{j-1,0})]$	...	$[0, q_{b_{j-1,n}}, n_{j-1,n}, max(q_{j-1,n})]$
$BidPrice_j$	$[0, q_{b_{j,0}}, n_{j,0}, max(q_{j,0})]$	...	$[0, q_{b_{j,n}}, n_{j,n}, max(q_{j,n})]$

Figure 3-8.: 4-Channel Matrix Representation, Own elaboration.

As the reader can notice, differences between the basic and the extended representation are notable in Figures **3-7** and **3-8**. This fact should have a positive impact when applying ML to this proposed representation, because it includes more information.

3.3.2. Results of proposed LOB Representation

Having a 4-vector matrix is equivalent to have a 4-channel image. In order to complete the proposed representation, it is important to take into account **data normalization**. This is because for ML models is important to keep an adequate magnitude relationship. Moreover, computationally speaking an image manages channel colors between [0, 255] if RGB is used. Therefore, it is necessary to apply $(0-1]$ normalization for each information channel.

The $(0-1]$ normalization differs from the $[0-1]$ normalization due to the fact that some price lines may have no entries. This happens under real market conditions, in order words, market agents are willing neither to offer nor to bid on any possible price. As a result, some price lines will have null entries. To make this difference clear to the ML model the smallest not null entries will have a value slightly bigger than zero, while null entries will have a zero value.

After applying all the steps (equations 3-9, 3-10, 3-11), results look like those illustrated in Figure **3-9**. As the reader can notice, this proposed representation includes all LOB informa-

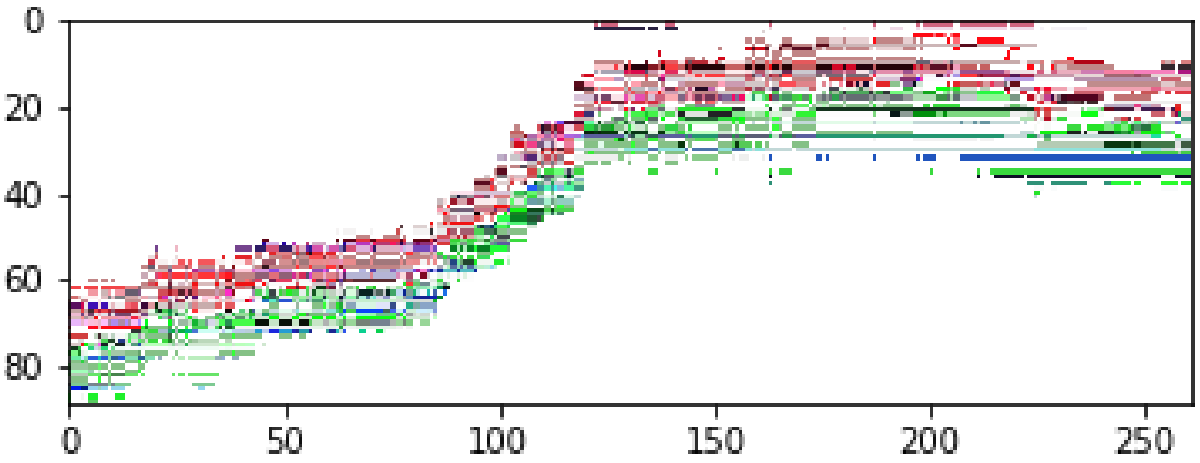

Figure 3-9.: LOB transformed to Image after applying the proposed representation. Own elaboration. See Annex C for multiple examples of LOB under the proposed representation

tion if compare to those made in [86] (Figure **3-6**). The proposed representation takes into account prices (p), total quantities (Q) at the same price, number of orders (N) at the same price and order distribution ($maxVol$) at the same price. As a result, it is more machine readable oriented. For visualization purposes, Figure **3-9** includes two different LOB (dot framed) and their channel decomposition rendered.

In summary, FTS come in two main categories:

- LOB data
- Transaction data

LOB data is richer than transaction data since it contains all trade intentions from market agents. Conversely, transaction data only contains information regarding the trade or asset exchanged. Under this circumstance, expectations of a better-performing model based on LOB data are greater than using only transaction data [66].

Given the fact that this work uses multimodal inputs (LOB and transaction data), it is necessary to homogenize them while keeping individual characteristics.

On the one hand, it is necessary to represent transaction data. In general, ML techniques group data that are separable in multidimensional spaces. Such spaces are different from the

original one. To make inputs homogeneous, this work represents raw transaction data as images using GAF representation. The main reason to prefer GAF over other methodologies is that GAF transforms cartesian coordinates into polar ones, making a kernel based on angles. This fact is very important, because working distances over angles rather than traditional Euclidean spaces can add more precision to model prediction, particularly when FTS are very noisy. In fact, as shown in Figures **3-3**, **3-4** and **3-5**, important characteristics of FTS are well separated. Peaks separation and liquidity factors can help to improve predictions.

On the other hand, it is necessary to represent LOB data, making it homogeneous with transaction data representation. As a result, a LOB representation is proposed. This representation extends the one proposed in [86], by including all the available information in LOB data. Added information includes volume distribution $maxVol$, total volume Q and the number of orders N at each price p. Consequently, the computational model will learn from complete data when compared to the representation proposed in [86], as explained in the following chapter. This new representation may be harder for human understanding, but it is more machine readable, specially in very short periods of time, when humans cannot make decisions.

4. CNN Multimodal Trading Strategy

Machine Learning is another topic for this work. In consequence, this chapter presents a review on techniques used for FTS modelling, pointing out specially to ML and its applications to FTS prediction. Given the fact that ML techniques varies, it emphasizes in DL, particularly, Convolutional Neural Networks (CNN) which are the technique used for this work. In both cases, this chapter presents a comprehensive review of works using ML and DL for FTS modelling. At the end of the chapter, the reader finds the proposed CNN architecture for the price direction predictor that will feed the AT strategy. This proposed architecture feeds from multimodal data, represented using the methodology explained before.

4.1. Brief History of FTS Modelling

FTS modelling is an old discipline that did not include ML from the beginning. In the earliest 1800's, researches such as Legendre and Gauss started using Linear Regression, which is the first statistical technique used to model FTS [29]. The 20th century brought different formulations of well known models such as ARMA, ARIMA and GARCH [16], [36]. Moreover, stochastic techniques deviated from Browian Motion, Levy Process and Poisson Process were also formulated to model FTS. These techniques are currently used for FTS modelling, including stochastic techniques for LOB dynamics modelling [3], [19], [27], [71].

These techniques are very popular in Econometric, in fact, results from [69] indicate that statistical methods performs well when forecasting TS. However, many academic papers suggest that data driven techniques, i.e. machine-learning methods, are more successful when modelling FTS [2], [6], [7], [9], [10], [26], [47], [55], [79], [89], [96], [102], [115]. This happens because ML methods are capable of dealing better with non-linearity, being able of capturing long and short-term dependencies, as well as finding common patterns. In other words, ML methods are better at complexity handling, when compared to traditional methods for these kind of data such as Linear Regression, ARIMA, ARMA and GARCH not to mention all the assumptions made when using diffusion like models (Browian Motion, Levy and Poisson processes) [27]. Latest results from M4-Competition [70] show that hybrid approaches using ML techniques improve substantially results of forecasting TS in general.

4.2. Machine Learning Techniques Applied to Price Prediction

Literature is vast when referring to applications of ML to FTS in general and Stock Prices in particular. It includes a variety of techniques such as Clustering, Decision Trees, Support Vector Machines (SVM) and Artificial Neural Networks (ANN) [9], [10], [29], [53], [38], [39], [42], [50], [59], [78], [79], [87], [96], [100], [105], [106], [108], [111], [115]. However, most of the literature focuses on two techniques: SVM and ANN. This is due to the fact that both SVM and ANN have proven to be useful when dealing with non-linearity of the inputs [9], [100], [79].

On the one hand, SVM are related to statistical learning theory [58] by implementing a structural risk minimization principle [100]. They were introduced by Vapnik in 1996 and has been widely applied to financial time series forecasting since 2000 [53], [87], [100].

On the other hand, ANN date back to 1943 when McCulloch and Pits formulated them. They are among the most used techniques for FTS analysis [42], [53], [79], because ANN are good at describing FTS dynamics [100]. They have also been used to forecast many different financial instruments across many different geographies since 1988 [108], being prices and trading volume the most common inputs and two-layer the preferred architecture [9]. Despite outperforming other techniques when modelling FTS [42], ANN possess limitations, which include [42], [88], [115]:

- Adding layers to the ANN does not translate into better forecasting results and damage time efficiency of the model.
- Over-training causes over-fitting, i.e. the tendency of an ANN to memorize the data, which results in poor generalizations.

These limitations are caused by the fact that the minimization error procedure gets stuck in local minima avoiding the ANN to converge in an optimal solution, [18], [88] resulting in poor generalizations [77]. In order to better comprehend this fact, it is necessary to introduce the concept of Back Propagation (BP), a method used to back propagate errors, while the ANN is learning from data. The BP process is done using partial derivatives, which transfers errors from the output to the first layer, updating ANN weights at each layer. In ANN, BP fails because once a local minima is reached, error transfers backwards is near zero, therefore layers weights are not updated, causing the network to improve its learning.

Table 4-1.: Summary of academic reviews and surveys reporting applications of ML techniques to FTS.

Year	Techniques Mentioned	Market	Freq.	Ref.
2018	ANN, Trees, CNN	Futures	LFT	[50]
2016	Trees, Naive Bayes, Clustering	Currencies	LFT	[39]
2016	Clustering, ANN, SVM, Trees	Stocks, Currencies, Futures	LFT	[20]
2015	DBN (Dynamic Bayesian Nets)	Currencies	HFT	[84]
2014	ANN	Futures, Options	LFT	[42]
2012	ANN	Stocks	LFT	[96]
2012	ANN	Stocks	LFT	[79]
2010	ANN, HMM	Stocks	LFT	[53]
2009	Clustering, Trees, ANN, SVM	Currencies, Stocks, Futures	LFT	[9]
1997	ANN	Stocks	LFT	[59]

This was a recurrent problem for ANN, causing that most of the architectures used maximum two hidden layers, in order to avoid the local minima issue. It was only until 2006, when Hinton [45] proposes a novel way to train deep architectures, that overcomes the local minima stuck issue mentioned early. This was the beginning of Deep Learning (DL), and the following section will explain details of this advance.

4.3. Deep Learning

DL is an extension of ANN [88], and results suggest that DL efficiently overcomes the two major problems of ANN. Its early applications started in image processing in 2006 [4], [88], but have extended in recent years to other fields, including traffic modelling and prediction [48], electricity demand forecast [18], fly direction prediction [32], weather prediction [24], [23], [28], [68], as well as FTS modelling [5], [6], [7], [14], [22], [26], [31], [34], [35], [41], [43], [74], [73], [89], [90], [91], [93], [99], [102], [112], [117].

DL definition comes from Neuroscience [57], when seminal authors [75] proposed a novel explanation of how mammals' visual cortex processed data coming in through their visual system. Their proposal consisted in making sparse representations of input data, in order to have an appropriated representation of the data at the end. As a result, in computational terms, they proposed a network interpretation that consist of an output unit, determined by a combination of a feed-forward input, a recurrent term and a self-inhibitory term. In other words, any instance of data can be reconstructed as a different linear combination of

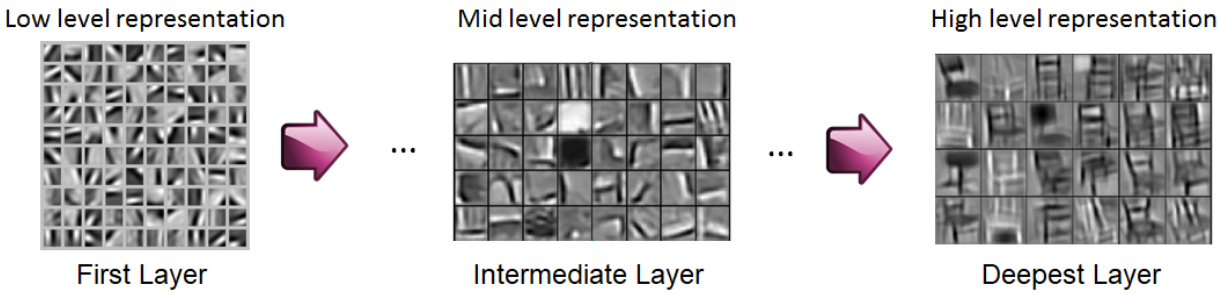

Figure 4-1.: Multilayered Visual Information Processing using DL. Adapted from [62].

the same components from sparse representations of the original data [57].

This development was practically feasible only until 2006, when seminal authors [45] proposed a novel Unsupervised Learning (UL) algorithm to train a deep architecture that consisted of Restricted Boltzmann Machines (RBM). This model was capable of building complex representations of data at deeper layers by capturing sparse representations from the previous ones. At that time, this algorithm won an image classification contest and it was seen as the introduction of DL to ML [57]. Figure **4-1** gives an illustration of how DL makes a complex representation from a simpler one, built in previous layers.

DL relevance relies in the way it represents and processes data. Traditional ML schemes include data pre-processing or feature extraction process. If poorly done, it might cause poor performance and if not done at all, data dimensionality can cause very inefficient time responses [4]. In contrast, mammals' brain do nothing of feature selection, instead the neocortex allows to propagate all the input data into a complex hierarchical structure, that learns to represent the data based on the regularities or sparse representation it has at each hierarchy or level [4]. At the first level, the input signal will have some fixed representations, but when it goes to a deeper level, the representation changes as a new input is added [57], as illustrated in Figure **4-1**.

Since Deep Networks have multiple hidden layers, they can make a better representation of the input data, due to the fact that each layer makes nonlinear transformation of data from the previous layer, resulting in learning of more complex representations [12], [61], [81].

Even tough its advantages for complex data modelling and learning, from the very beginning, researchers were aware of some challenges regarding DL architectures, including training times [11], [12], [28], sequential data modelling [114] and feature extraction to im-

prove learning [114]. Time and research interest in the subject have helped to overcome most of these challenges. In fact, decay in training times is a fact because of parallelism. DL frameworks, such as TensorFlow, include GPU implementations that help to reduce training times considerably.

Since DL tackles the gradient vanishing issue while training multi-layer networks it facilitates application of different neural network topologies, including those specifically designed to model sequential data, such as Recurrent Neural Networks (RNN) and Long Short-Term Memory (LSTM) Networks, in addition to other topologies such as Multilayer Perceptron (MLP), Deep Belief Networks (DBN) and Convolutional Neural Networks (CNN).

During the last ten years, DL has emerged as the prominent ML technique for both classification and regression problems. This is because DL has been able to overcome training issues of classical artificial neural networks, in relation to gradient vanishing problem. As a result, research in DL has increased in recent years, including studies and applications of DL for predicting FTS. Table **4-2** shows a summary of DL works specially designed for FTS modelling.

4.3.1. Deep Learning Applications to Price Prediction

DL pioneering applications are in the image classification domain [57]. From there, it has expanded to many other different domains. Reports of FTS modelling using DL started appearing in 2011, being DBN the primary architecture to use. The last two years have exhibited an interesting growth in reporting of FTS modelling applications using DL. Table **4-2** gives a compound application summary, including architectures and data used. Reported prediction results, ranging between 57 % to 72 %, suggest that DL improves prediction accuracy when compared to other ML techniques.

As the reader can notice, the explosion of DL applied to FTS occurred from 2017. DL classical topologies include MLP, DBN, RNN and CNN. It is worth to mention, that LSTM and GRU are sub-types of RNN, which are specially designed for sequential data modelling, in contrast of other more general purposes oriented such as DBN and MLP. CNN are mostly devoted for images as input. Previously of this work, the author researched on DBN, MLP and RNN topologies to predict FTS using HF data [5], [6], [7], [74].

It is important to quote that works in Table **4-2** use either prices or returns for inputs, except by [35], [73] and [74]. This happens because of availability of LOB data is difficult

Table 4-2.: Academic papers reporting applications of DL to FTS.

Year	Techniques Mentioned	Market	Freq.	Ref.
2011	DBN	Currency Prices	LFT	[22]
2012	DBN	Future Prices	LFT	[54]
2013	Stacked RBM	Stock Prices + indicators	LFT	[99]
2014	DBN	Stock Prices	LFT	[117]
2015	DBN	Currency Prices	LFT	[91]
2015	CNN	Stock News	LFT	[34]
2016	CNN	Stock Prices	LFT	[23]
2016	DBN	Stock Prices + LOB	HFT	[74]
2016	RL + RNN	Futures, Stock Prices	LFT (Minutes)	[31]
2016	MLP (DNN)	Stocks Prices	HFT	[5]
2017	CNN	Futures Prices	LFT	[41]
2017	CNN	Stock Prices	LFT	[93]
2017	DNN, RNN	Stock Prices	LFT	[6]
2017	DNN	Stocks Prices	LFT 5 Minute	[66]
2017	CNN	Stock LOB	HFT	[35]
2018	RNN, LSTM, GRU	Stocks Prices	HFT	[7]
2018	CNN	Stock Prices	LFT	[90]
2018	CNN	Stock Prices + LOB	HFT	[73]

and usually it is costly, instead of transaction data, which usually is widely available for free.

Raw data (LOB or transactions) could be transformed. There are several techniques used to deviated information from FTS, which include:

- Technical Analysis: It is a discipline that has developed a series of quantitative measures to determine technical indicators regarding prices. It includes moving averages and momentum indicators. Some authors use them to add information for ML modelling of FTS. Authors in [5], [6], [74] and [99] use some technical analysis tools to extend input information.

- Quantitative measures or transformations. Under this umbrella may classify data transformations, like for example Wavelets, Fourier Analysis and other analytical methods, used to change inputs spaces. Authors in [7], [41], [73] and [90] are examples of how to change input spaces using analytical methods.

Column **Freq** in Table **4-2** refers to data input frequency: High Frequency or Low Frequency.

Works labeled as LFT usually refers to days, unless other notation. HFT usually refers a predictions for very short time frames, usually a minute or less using data with millisecond granularity. Section below will give an explanation of the different DL topologies: DBN, MLP, RNN and CNN. An special emphasis will be done on CNN, technique used for this work.

4.4. Deep Learning Topologies

4.4.1. Deep Belief Networks (DBN)

A DBN is a probabilistic generative model composed of an input layer, an output layer and, in between, multiple stacked layers of Restricted Boltzmann Machine (RBM). Each RBM is composed of two layers, one visible, and another hidden, as illustrated in Figure **4-2**. Hinton proposed this model [44], where training happens in two steps: one called unsupervised pre-training and the second one called supervised fine tuning. In [45], Hinton presents an efficient approach to train this network in a greedy layer by layer manner. An RBM, which is bipartite graph, is the component of hidden layers. Bipartite graphs have two units: one is visible and an the other is hidden. An RBM is restricted because there are no connections between units of the same layer (visible, hidden) [11], [83].

Generally speaking, unsupervised pre-training allows the model to get better values for

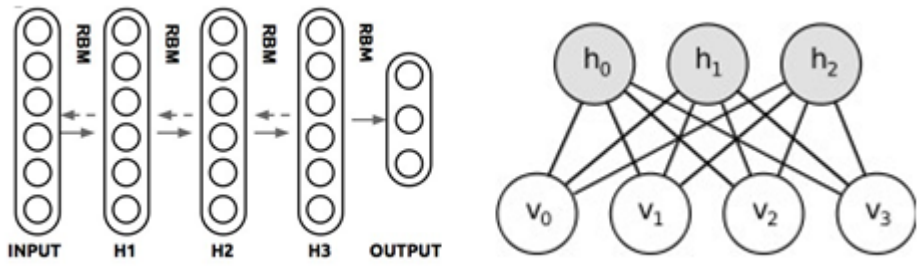

Figure 4-2.: DBN architecture. Source: [54].

weight matrix W, which results in finding a better local optima. The supervised fine tuning phase allows the model to adjust better the initial weights (W) found on the pre-training phase [54]. Table **4-2** shows three academic works reporting use of DBN for FTS modelling.

4.4.2. Multi-Layer Perceptron (MLP)

A Multi-layer Perceptron is an ANN with multiple hidden layers. This architecture dates back to the 1970's. However, it did not yield better results than a single hidden layer ANN, because of gradient vanishing problem [88]. Since DL tackles this issue, this architecture becomes the simplest to use within the DL architecture family. As in other DL architectures, MLP learns high-level representations by processing and refining large quantities of data, at each intermediate layer [5], [13], [88]. This network is characterized by having many neurons at each hidden layer. As a result, the number of parameters is very high, which results in more computational time for training, if compared to a CNN with the same size inputs [109]. Gradient Descent using BP is the learning algorithm for training this network. Table **4-2** shows two academic works reporting use of MLP for FTS modelling.

4.4.3. Recurrent Neural Networks (RNN)

RNN are ANN where connections include loops between output and input units at each layer. These loops are specially designed to keep information about short-term dependencies among data. This kind of architecture is specially designed to model sequential data, because including these loops allows the network to feed itself with temporal dynamics of the data. Figure **4-3** illustrates how this loop is used. As the reader can observe, the network keeps a memory of the previous calculation. There are different variations of RNN, including Long Short Term Memory (LSTM) and Gate Recurrent Units (GRU) [6], [88], [46]. They are used to model sequential data, like sentence prediction [17], as well as time series. Table **4-2** shows three academic works reporting use of RNN or its variations for FTS modelling.

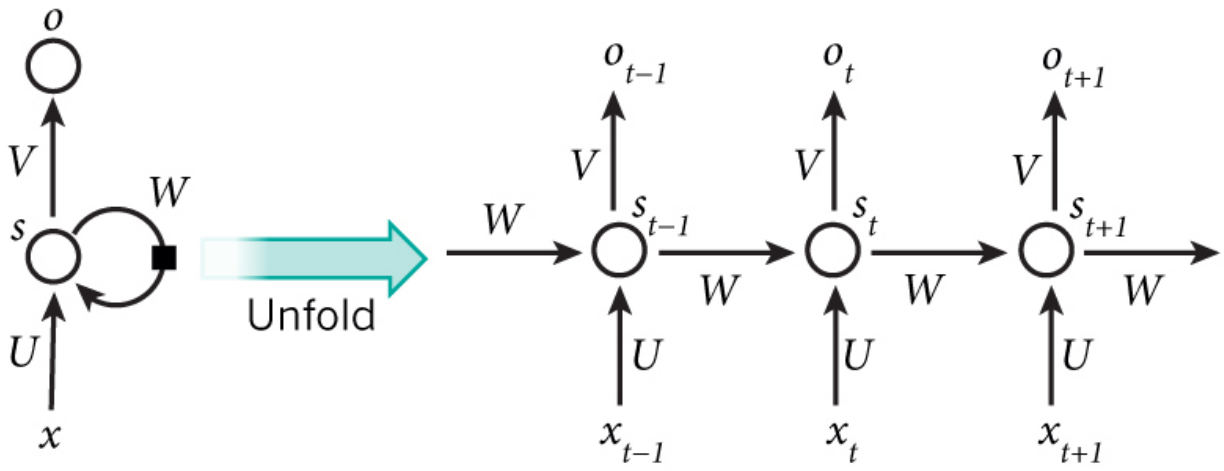

Figure 4-3.: A RNN, taken from: [17].

4.4.4. Convolutional Neural Networks

CNN are biologically inspired, trying to emulate what happens in mammals' visual cortex, where neural cells are specialized to distinguish particular features. Building blocks of a CNN architecture are in charge of doing this feature detection by activating or de-activating a set of neurons [61]. Since market agents' decisions are highly dependent on visual analysis of price changes and events in the LOB, it can be expected that an algorithm based on CNN can learn patterns in order to help trigger trading decisions. In fact, [86] showed that a visual dictionary could be constructed from LOB data and that dictionary had predicting capabilities. Following this path, a CNN could improve such predicting capabilities.

Different authors [22], [23], [35], [41], [47], [64], [63], [73], [93], [107], [116] have illustrated different possibilities of using CNN to either classify or predict time series and FTS in particular. Works of CNN applied to FTS are recent; however, applications of CNN to time series in general are older and more common. Reported results on previous researches show that CNN are good for FTS classification, though methods differ in how to represent raw data, in both cases uni-variate and multivariate.

A CNN has two main building blocks, a convolution layer and a pooling layer. These two in

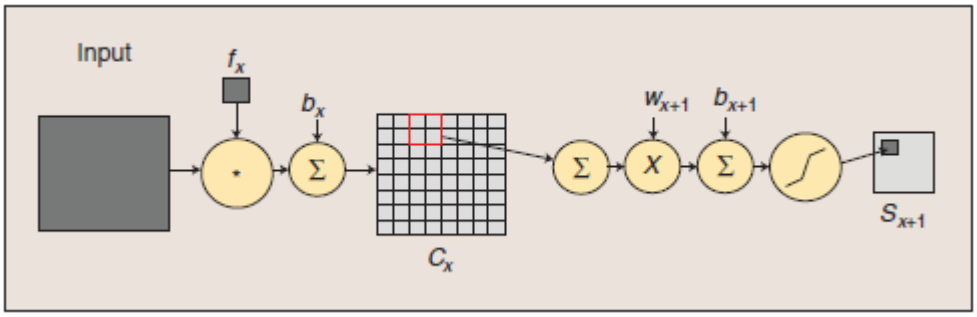

Figure 4-4.: Convolutional Process. Source [4].

conjunction with a dense layer, complete a CNN. In general, a CNN runs layer by layer in a forward pass, a tensor x goes through the different layers (Figure **4-4**). x^L is usually a C dimensional vector $x^L \in R^C$, encoding the posterior probability of x^1 from the $i-th$ class [110]:

$$x^l \rightarrow w^1 \rightarrow ... \rightarrow x^{L-1} \rightarrow w^{L-1} \rightarrow x^L \rightarrow w^L \rightarrow z \tag{4-1}$$

The last layer is associated with a cost function to measure discrepancy between the real

value of x^1 and the predicted one from the CNN.

$$z = \frac{1}{2}\|real - predicted\|^2 \tag{4-2}$$

Loss function equation (4-2) is for regression problems, cross entropy loss function is for classification [110]. Formally, a CNN prediction is:

$$arg_i \max x_i^L \tag{4-3}$$

A CNN learns through Stochastic Gradient Descent (SGD), which means that weights w are adjusted using partial derivatives from z all the way back up to w^1, updating weights in equation 4-1.

ReLU Layer

Every convolution layer includes the application of a function such as ReLU. Formally, for every element in input x^l,

$$y = x^{l+1} = \max 0, x_{i,j,d}^l \tag{4-4}$$

As a result, partial derivatives of equation (4-4) for the Back Propagation (BP) process are:

$$\frac{dy_{i,j,d}}{dx_{i,j,d}^l} = [x_{i,j,d}^l > 0] \tag{4-5}$$

$$[\frac{\partial z}{\partial x^l}]_{i,j,d} = [\frac{\partial z}{\partial y}]_{i,j,d}, \quad if \quad x_{i,j,d}^l > 0; \quad 0 \quad otherwise. \tag{4-6}$$

ReLU facilitates certain patterns activation within an input region by increasing CNN non-linearity. Other functions such as sigmoid or hyperbolic tangent are possible to use, however BP process reduces significantly the gradient, therefore after several layers, the gradient will be vanished, making learning more difficult [110].

Convolution Layer

This building block is in charge of applying the convolution operator to the input matrix. In other words, it applies a kernel to filter data input. Depending on the parameters used, it can reduce or maintain input dimensionality. The main reason to convolve is to identify edges. This means to identify or separate features that are used later to construct more complex representations in deeper layers [11], [109], [110]. Formally, an image I goes through a kernel K:

$$I : \{1, ..., n_1\} \times \{1, ..., n_2\} \to W \subseteq R, (h, w) \to I_{h,w} \tag{4-7}$$

$$K \in R^{2h_1+1 \times 2h_2+1} \tag{4-8}$$

$$(I \otimes K)_(r,s) := \sum_{u=-h_1]}^{h_1} \sum_{v=-h_2}^{h_2} K_{u,v} I_{r+u,s+v} \tag{4-9}$$

$$K = \begin{vmatrix} K_{-h_1,-h_2} & \cdots & K_{-h_1,h_2} \\ \vdots & K_{0,0} & \vdots \\ K_{h_1,-h_2} & \cdots & K_{h_1,h_2} \end{vmatrix} \tag{4-10}$$

Therefore, for an input image (4-7) of size $H^l \times W^l \times D^l$, and a kernel (4-8) of size $H \times W \times D^l \times D$, the convolution (4-1) size is:

$$(H^l - H + 1) \times (W^l - W + 1) \times D \tag{4-11}$$

Padding is a technique that allows equalling output size (convolution) with input size. Padded rows and columns usually have zero value. Moreover, the kernel K moves in **strides**. Moving the kernel through all possible positions on the input means a **stride** s is equal to one. However, if $s > 1$, every kernel movement skips $s - 1$ pixel location [110]. In general, mathematically speaking a convolutional layer, including a bias B, is:

$$(Y^{(l+1)}_{i,j,d}) = (B^{(l)}) + \sum_{j=0}^{H} \sum_{j=0}^{W} \sum_{d^l=0}^{D^l} (K^{(l)}_{i,j,d})_d (Y^{(l)}_{i,j,d}) \tag{4-12}$$

Pooling Layer

This building block is a local operator that takes convolution output and maps sub regions into a single number. The pooling operator can extract the max value of the mapped sub region **(max pooling)**, or the average value of the mapped sub region **(average pooling)**. In other words, it gets subsamples out of its input, usually a convolutional layer (equation 4-12) [110].

$$Max : y_{i^{l+1},j^{l+1},d} = \underset{(0 \le i \le H, 0 \le j \le W)}{\text{máx}} x^l_{i^{l+1} \times H+i, j^{l+1} \times W+j,d} \tag{4-13}$$

$$Average : y_{i^{l+1},j^{l+1},d} = \frac{1}{HW} \sum_{i,j} x^l_{i^{l+1} \times H+i, j^{l+1} \times W+j,d} \tag{4-14}$$

Usually, these two layers (equations 4-12 and 4-14) make one layer in the CNN topology. However, it is not necessary to have one convolution followed by one pooling layer. Therefore, it is possible to have different combinations of convolution and pooling layers in one CNN topology. Additionally, a CNN topology usually includes various layers of convolution and pooling, thus networks extract simpler features at the first layer, and by combining those, they can learn more complex features in deeper layers [11], [57].

Dense Layer

Finally, the deeper convolutional layer is connected to a dense layer (fully connected), from which the network obtains its outputs. As mentioned before, a CNN topology may have one or more dense layers. A CNN topology may include dropout layers as well.

From the computational point of view, convolving is far more efficient than working with traditional Feed Forward Networks (FFN). This is because of parameter reduction. Moreover, while working with images, traditional neural nets are more prone to over fitting because of a larger number of parameters [98], [110].

Receptive field is another important concept of CNN. Given the fact that CNN limits the connections between neurons by connecting each neuron to only a local region of the input volume, this local connectivity is known as receptive field and it is equivalent to the filter size, presented in equation 4-8.

4.5. Architecture Proposal

As mentioned earlier, the author has already explore different DL architectures to model HF data in FTS. It includes DBN ([74]), MLP ([5]) and RNN ([6] and [7]). Given the fact that LOB and transaction data are represented as images (See Chapter 3), it is natural to choose CNN, because they are widely used for image classification. Under these conditions, it is expected that a CNN yileds better results than another DL topology. This work is not about classification, it is about prediction of future price direction, therefore a CNN will be the building block for the machine learning predictor for this work.

As mentioned earlier, a CNN topology may include a different setup of convolutional and pooling layers. For this work, Figure **4-5** illustrates the proposed CNN topology used for modelling FTS data (LOB and transactions). Under this topology, the proposed CNN has two convolutional layers. A max-pooling layer follows each convolutional one. Finally, there are two fully connected layers, one of them with a dropout of 40 %. The proposed CNN classifies inputs into three possible categories: up, down and flat movements. The output signal serves as input for the trading system. The proposed CNN is a 3D CNN, because there are two different inputs, LOB data and transaction data. Both represented as 4-channel images using the transformations mentioned in chapter 3.

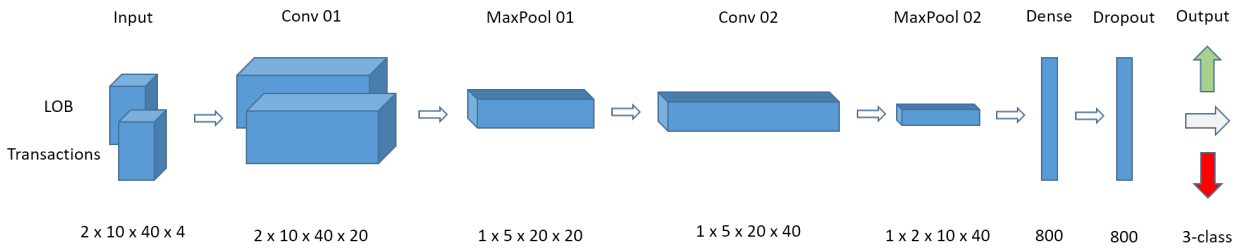

Figure 4-5.: Illustration of the proposed CNN Architecture. It is a 3-D CNN. Own elaboration.

It is important to remember that there is a trade off between training times and prediction accuracy. This is because the predictor will be feed into an AT strategy for very short times. As a result, several test were performed, changing the number of convolutional layers (convolutional operator + max pooling). Prediction accuracy improves as more layers are added, however increasing training times growths faster. As a result, the chosen architecture has two convolutional layers. Chapter 5 will show comparisons between the selected architecture and two well known CNN: LeNet and AlexNet.Such comparisons will show the benefit of the proposed architecture while trade off handling.

LeNet-5 is a CNN created by [60] and it was aimed to make hand-written number recognition. It consists of 7 layers (Input, Conv + Pool, Conv + Pool, Dense + Output). At that time, computing resources were scarce, creating a constraint for this technique. However, as computer resources got better in performance and cost, training this particular architecture is easy and it has become a baseline in image recognition contests.

AlexNet was created in 2012 and it became famous due to the fact that reduces the classification error in an Image Recognition Contest to 15.3 % by that time. Nowadays, classification error is much lower. Since AlexNet was the pioneer, it has become baseline architecture as LeNet. AlexNet took advantages of computer developments, particularly parallel processing through Graphics Processing Units (GPUs). It was created by [52]. It has more filters than LeNet as well as stocked convolution layers, as a result, it is deeper with more parameters.

At this point, it is worth to remember that training times are key, because CNN outputs serve as a trading signal for an autonomous high frequency trading system. As a result, it is necessary to balance model complexity in relation to its accuracy to generate quality signals within such very short time frames.

Proposed CNN Architecture Details

Table **4-3** shows the main aspects of the proposed CNN. It shows kernel sizes for different layers, as well as other layer-dependent settings.

Table 4-3.: Main characteristics of the proposed CNN.

Characteristic	**Description**
CNN type	3D CNN for LOB and Transaction data.
Input Size	$[2, 40, 10, 4]$.
Convolutional Layer 1	20 filters with a [2x2] kernel, Stride 1, ReLu.
Pooling Layer 1	Max pooling, Stride 1.
Convolutional Layer 2	40 filters with a [2x2] kernel, Stride 1, ReLu.
Pooling Layer 2	Max pooling, Stride 1.
Fully Connected Layer 1	Softmax.
Fully Connected Layer 2	Softmax, Dropout 40 %.
Learning Rate	0.0001
Batch Size	100

In summary, ML applications to FTS modelling are abundant. Complexity handling and non-linearity learning are the key aspects that differentiate and make more successful ML from other techniques when modelling FTS. Recent developments in Information and Communications, such as improvements in parallel processing and storage capacity have made possible to apply ML to different domains in general and to the financial industry in particular. Moreover, DL, which is a recent development within ML, enhances learning abilities of machines, thus modelling much better different problems in different domains.

Classifying and predicting FTS are one of such problems. Works devoted to FTS modelling using DL have increased considerably during recent years. These works have yield better results when compared to other ML techniques. Indeed, ALGOS research group members conduct testing of different DL models and topologies ([5], [6], [7], [74], [73]) for FTS. These works illustrate learning capabilities of DL models under different architectures and conditions, obtaining much better results than other ML techniques.

This work proposes a 3D CNN. This architecture includes two convolutional layers (convolution + max pooling) connected to a fully connected layer to classify FTS in order to generate trading signals for an AT strategy. Inputs to the model will be transaction and LOB data, after applying the representation proposed in the previous chapter. The next chapter

will show that results are very competitive, when compared to previous works with other topologies, as well as, when compared to well known CNN architectures such as LeNet and AlexNet. Moreover, it will show results when merging CNN signals with two AT strategies.

5. Experiments and Results

This chapter provides comprehensive details on experiments conducted and results achieved. Given the fact that it is necessary to present both FTS modelling results and implemented AT strategy results, it includes two separated sections for clarity. In general, both the prediction model and the AT strategy yielded positive results. Organization of this chapter is as follows:

- **Experimental Settings** provides a general outlook of the conducted experiments.
- **Results of the Proposed CNN Price Direction Predictor** presents results of the proposed CNN architecture as price direction predictor. It has the following subsections:
  - Experiments common considerations for both markets.
  - Results using US market data.
  - Results using Colombian market data.
  - Results comparison between the proposed CNN and other similar works, that use different DL architectures.
- **AT Strategy Results** contains information about experiments conducted for the AT strategy in two subsections:
  - Trading conditions for the proposed AT strategy.
  - Results for the aggressive AT strategy.
  - Results for the conservative AT strategy.

5.1. Experimental Settings

In general terms, experiments consist of raw data pre-processing, transforming and splitting. This is followed by model training and testing. After that, the model is invoked to perform out of sample testing. In other words, model training and testing is done with in-sample data. These data refer to the continuous data-set, that represents the FTS used for this purpose. Out of sample data refer to another data-set, which do not correspond to the same

time period of in-sample data-set. It is a completely unseen new data-set, that is used once the model is trained.

Figure **5-1** shows graphically the steps followed to conduct each experiment. As the reader

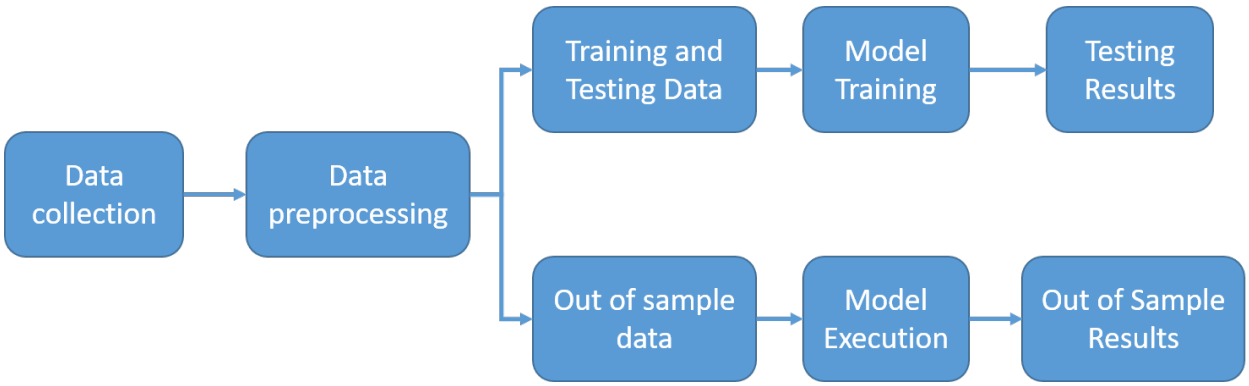

Figure 5-1.: General steps to conduct training and testing of the prediction model.

can notice, out of sample data run over a model already trained. This is done to verify whether or not the model generalizes well. Finally, the trained model generates trading signals for two AT strategies. One is for a less risk avert investor (aggressive strategy) and the other, for a more risk avert investor (conservative strategy).

Experiments include data for two different markets: US market and CO market. In both cases, particular considerations are presented separately, because of their differences: US market is highly liquid with a wide range to instruments to choose from; in contrast, CO market has low liquidity, with a small set of tradable stocks. These facts implies different considerations while pre-processing data from each market. In consequence, results for the ML predictor (i.e. the proposed CNN) are presented separately for each market, respectively.

5.2. Results of the Proposed CNN Price Direction Predictor

5.2.1. Common Considerations

Experiment standardization in Finance is difficult, mainly due to facts like data-sets variety, different time-frames, markets diversity, architecture possibilities, information sources, among others [9], [37], [47], [49], [53], [78], [79], [94], [101]. Therefore, this work conducts different experiments while keeping standard the following aspects:

- Data sources and their representation: LOB and transaction data represented as explained in chapter 3.
- CNN architecture. It is the one proposed in Chapter 4. In order to illustrate the difference having both LOB and tick data as input, in contrast of only one of these data sources, two experiments are presented. Experiment with LOB and tick data are referred as **3D-CNN**, whereas experiments with only LOB data are referred as **2D-CNN**.
- Pre-processing details, as presented below in this chapter for each market (**US** and **CO**).In general terms, pre-processing was the most time consuming of all the experiment. This is due to the fact that raw data (LOB plus transaction) for both markets should be converted to an image like representation, following steps presented in chapter 3. This task is time consuming because it means to generate many images implying lots of disk read-write operations.
- Data-set split with 90 % and 10 % for training and testing, respectively. Given the fact that there is a good quantity of data, this data split facilitates more data for training while offers a good quantity for testing purposes.
- Number of instruments included in the studies.
- Same class labelling, as well as similar class balancing.
- Number of days for out of sample testing.
- Directional Accuracy (**DA**) as performance measure. Mathematically, it is defined as $\frac{1}{N}\sum_t 1_{sign(X_t - X_{t-1})} = signF_t - X_{t-1}$, where X_t is the real (observed) value and F_t is the predicted value.
- Additional performance metrics for the model, as follows:
  - MSE, $Precision$, $Recall$, $F1Score$, Mean Class Accuracy (MCA). Tensor flow metrics is the module used for calculations.
- Infrastructure used. It was an Intel computer with two CPUs, 2 GPUs, each one with 2500 processors, 16 GB in RAM, SO Linux Centos and TensorFlow as DL framework.

Table **5-1** gives a brief summary of all the experiments conducted. As the reader can notice, Experiments labeled $2D$ means that it uses only one source of data as input (LOB). Conversely, $3D$ quotes experiments that uses both LOB and transaction data.

Table 5-1.: Summary of the experiments conducted for both markets: US and CO.

Experiment	Data input used
Proposed 2D CNN	LOB data only.
Proposed 3D CNN	US LOB and transaction data.
3D Lenet CNN	LeNet architecture with LOB and transaction data.
3D AlexNet CNN	AlexNet architecture with LOB and transaction data.

Additionally, there were some changes, mainly because of market differences (US market vs COlombian market), which are summarized as follows:

- Number of samples, US market is more liquid, deep and developed than CO market. As a result, there are more samples for US market.
- Number of lines (prices) of LOB. For US market data collection includes information of EDGX venue, operated by BatsTrading.com. LOB only contains 5 price lines for each side. In contrast, raw CO market data includes all LOB entries. This work considers 10 at each side for CO market.
- Time granularity. For US market data, there is a book every 5 seconds. For CO market data, there is a book every 30 seconds.
- Sliding Window. For all experiments 10 LOB were taken in account. As a result, Sliding Window for US market is 50 seconds, while for CO market it is 300 seconds.
- Labelling threshold varies, because of differences in spread sizes (price discovery process) between the two markets.

Complete data description and pre-processing details are presented for each market (US and CO) separately in sections below. This is because of market differences, as mentioned earlier.

5.2.2. Results using US Market Data

Table **5-2** summarizes the results using US market data for each experiment. As mentioned earlier, results reflect a very good performance. The section below shows characteristics of data used for different purposes: training, testing and out of sample results.

Data Description

Data includes information about 11 equity instruments traded on NYSE. This collection includes well-known companies such as Facebook (FB), Microsoft (MSFT), American In-

Table 5-2.: Results summary for experiments using testing **US data**.

Experiment	DA	MCA	Precision	Recall	F1 Score
Proposed 2D CNN	59.37 %	58.15 %	65.89 %	67.81 %	66.84 %
Proposed 3D CNN	**74.15 %**	**74.90 %**	76.62 %	78.1 %	77.38 %
Proposed 3D LeNet CNN	75.11 %	75.15 %	78.28 %	77.72 %	78.00 %
Proposed 3D AlexNetCNN	76.39 %	76.29 %	81.13 %	76.77 %	78.89 %

ternational Group (AIG) and Citibank (C). Moreover, it includes foreign stocks, known as ADR (American Depository Receipts). This category contains names such as Deutsche Bank (DB), Royal Bank of Scotland (RBS), Grupo Aval (AVAL), Avianca (AVH), Bancolombia (CIB) and Ecopetrol (EC). Additionally, it includes data from an Exchange Traded Fund (ETF) SP500 (SPY).

This collection includes companies from different sectors, different countries, as well as different liquidity levels. SPY, FB and C are the most liquid on this collection and the Colombian ADR the least. This wide range of instrument characteristics obeys to the need of training a single model that generalizes trending patterns under different market scenarios. Data includes information from 2016-04-01 to 2016-10-31. 47 venues make the whole US market, that is 47 locations are in charge of registering and matching orders. Data used come from only 1 venue, EDGX, operated by BatsTrading. Information was collected and facilitated by DataDrivenMarket. In total, 4.5 million files containing raw data were processed to generate 770,000 images, 385,000 each (LOB and Transaction data), following the methodologies mentioned in chapter three for both LOB and transaction data.

Preprocessing Considerations

- Liquidity was the main aspect to take into account in order to preprocess data. As a result, daily data of instruments that exhibit low liquidity at particular dates were sampled out. The threshold was the number of LOB files collected in a particular day. Therefore, any instrument considered should have more than 1,500 LOB files for a single day. In order to avoid look ahead bias, trading strategy should be implemented for highly liquid stocks.

- Data normalization was done individually for each channel of figure **3-8** for each stock. This guarantees two things: first, information homogeneity regardless instrument nominal values (prices, quantities, number of quotes at same price) and second, keeping magnitude measures among individual channels. Therefore, inputs are already stan-

dardized and meaningful as a collection across different assets.

- Sliding window: 50 seconds is the threshold for this parameter. As a result, all the recorded books in this time frame are joined as matrix object (see Figure **3-8**) and converted to image.

- Data labelling includes three categories: (a) Up trend; (b) Down trend; (c) Trendless. Labels depend on the price percentage change within a sliding window. The first and the last transactions are used to calculate such change. Any change bigger than 0,025 % means up trend; changes below −0,025 % are down trend, otherwise it is considered trendless.

- Class distribution: Under the conditions mentioned previously, Up and Down Trend classes are balanced, 28 % each, and 44 % of the labels correspond to the trendless class.

Training, Testing and Out of Sample Results

Table **5-3** shows a summary for different metrics achieved at training, testing and out of sample testing phases. It is worth mentioning that out of sample testing means running the trained model using data unseen by it. Performance of the proposed architecture is very competitive, when compared to well-known CNN architectures. Improvements of the classification results using LeNet and AlexNet come with a high cost in training times. Table **5-4** illustrates a trade-off between DA and training times.

As the reader can observe, a slight improvement of 1 or 2 percentile points means training time increases by 85 % and 221 %, respectively. The proposed architecture offers a good balance between DA, which is very competitive, while handling a shorter time during training. This fact complies with the premise of the work: adding intelligence for a short term trading system, while facilitating model retraining. In consequence, for real industry implementations, the proposed architecture offer a great advantage. Moreover, Table **5-5** displays confusion matrix metrics for different experiments.

5.2.3. Results using Colombian Market Data

Table **5-6** summarizes the results using CO market data for each experiment. Two major experiments, quoted A and B, were done. **Experiment A included 134,696 images**, whereas **experiment B included 179,828 images**. The aim is to show how model results differ depending on training data availability. In [7] instruments with more data exhibited a

Table 5-3.: Results complement for experiments using **US data**.

Experiment	Loss	MSE	Testing DA	Out of sample DA
Proposed 2D CNN	1.0707	0.9303	59.37 %	49.65 %
Proposed 3D CNN	**0.56116**	**0.6328**	**74.15 %**	**63.45 %**
3D LeNet CNN	0.47869	0.5393	75.11 %	64.17 %
3D AlexNetCNN	0.5222	0.5948	76.39 %	66.04 %

Table 5-4.: Trade-off training times vs DA for experiments using **US Data**.

Experiment	Training Minutes	Testing DA	Trade-off
Proposed 2D CNN	8	59.37 %	-42.8 %, -14.78 points
Proposed 3D CNN	**14**	**74.15 %**	-,-
3D LeNet CNN	26	75.11 %	+85.7 %, +0.96 points
3D AlexNet CNN	45	76.39 %	+221.4 %, +2.24 points

Table 5-5.: Confusion matrix summary for experiments using **US data**.

Experiment	TP	FP	TN	FN
Proposed 2D CNN	6,710	3,473	6,710	3,185
Proposed 3D CNN	**8,513**	**2,597**	8,513	**2,381**
3D LeNet CNN	8,390	2,328	8,390	2,405
3D AlexNet CNN	8,624	2,001	8,624	2,603

better DA, therefore, experiments quoted A and B respectively, can give additional evidence to this fact.

Data Description

Data include information about 10 equity instruments traded on BVC (Colombian Stock Exchange). This collection includes well major Colombian companies from different industries, including energy and gas, banking, retail, airlines and insurance. Names of companies are: Ecopetrol (ECOPETROL), Avianca (PFAVH), Cementos Argos (CEMARGOS), Interconexión Eléctrica (ISA), Bancolombia (PFBCOLOM), Grupo Aval (PFAVAL), Almácenes Exito (EXITO), Grupo Sura (GRUPOSURA), Pacific Rubiales (PREC)[1] and Canacol Energy (CNEC). It is important to mention that four of these companies trade on NYSE under the ADR figure.

[1]Delisted on April 18^{th}, 2016

Table 5-6.: Result summary for experiments using testing **CO data**.

Experiment	**DA**	**MCA**	**Precision**	**Recall**	**F1 Score**
Proposed 2D CNN A	58.23 %	58.58 %	74.64 %	60.13 %	66.60 %
Proposed 3D CNN A	*65.31 %*	*65.72 %*	*79.65 %*	*77.62 %*	*78.62 %*
Proposed 2D CNN B	58.83 %	59.04 %	76.6 %	59.45 %	66.97 %
Proposed 3D CNN B	**70.31 %**	**70.56 %**	**79.93 %**	**84.55 %**	**82.18 %**
3D LeNet CNN B	70.38 %	70.73 %	75.64 %	86.54 %	80.72 %
3D AlexNetCNN B	71.78 %	72.03 %	79.46 %	85.00 %	82.14 %

As the US market experiment, this wide range of instruments offers a variety of characteristics that helps to generalize trending patterns under different market scenarios. Data for Experiment A include information from 2016-02-16 to 2017-12-28; while data for experiment B go from 2016-02-16 to 2018-07-31. Information was collected and facilitated by DataDrivenMarket. In total, 0.4 million files containing raw data were processed to generate 179,828 images, 89,914 each (LOB and Transaction data), following the methodologies mentioned in chapter three.

Preprocessing Considerations

- As mentioned earlier, liquidity was the main aspect to take into account in order to preprocess data. For CO market data, this fact is more restrictive. The reason is that CO market is very small and low liquid, therefore fewer instruments may qualify to be considered for short time frames trading. The threshold was the number of LOB files collected in a particular day. Therefore, for any instrument to be considered, it should have more than 100 LOB files for a single day. As the reader can realize, this is an remarkable difference when compared to US market.

- Data normalization was done individually for each channel of Figure **3-8** for each stock. This guarantees two things: first, information homogeneity regardless instrument nominal values (prices, quantities, number of quotes at same price) and second, keeping magnitude measures among individual channels. Therefore, inputs are already standardized and meaningful as a collection across different assets.

- Sliding Window: 300 seconds is the threshold for this parameter. As a result, all recorded books in this time frame are joined as matrix object (see Figure **3-8**) and converted to image. This is another difference when compare to preprocessing considerations for

US market data.

- Data labelling includes three categories: (a) Up trend; (b) Down trend; (c) Trendless. It depends on the price percentage change within a Sliding Window. The first and the last transactions are used to calculate such change. Any change bigger than 0,18 % means up trend; changes below −0,18 % are down trend, otherwise it is considered trendless.

- Class distribution: Under the aforementioned conditions, Up and Down Trend classes are balanced, 32 % each, and 36 % of the samples correspond to the trendless class.

Training, Testing and Out of Sample Results

Table **5-7** shows a summary for different metrics, achieved at training, testing and out of sample testing phases. It is important to restate that out of sample testing means running the trained model using data unseen by it. Performance of the proposed architecture is very competitive, when compared to well known CNN architectures. Improvements of classification results using LeNet and AlexNet come with a high cost in training times, as happened with US data. Table **5-8** illustrates this fact. A slight improvement of 0.3 or 1.5 percentile points means training time increases by 60 % and 150 % respectively.

For this case, increases in training times are not as high as for US data. This happens because there is less quantity of data fro CO data. Nevertheless, the proposed architecture offers a better balance between DA and training times. Table **5-9** displays confusion matrix metrics for different experiments.

Table 5-7.: Results complement for experiments using **CO data**.

Experiment	**Loss**	**MSE**	**Testing DA**	**Out of sample DA**
Proposed 2D CNN B	1.0813	0.9536	58.83 %	47.56 %
Proposed 3D CNN B	**0.7376**	**0.6498**	**70.31 %**	**59.17 %**
3D LeNet CNN B	0.7308	0.6481	70.38 %	59.32 %
3D AlexNet CNN B	0.7275	0.6368	71.78 %	60.11 %

5.2.4. Comparison against other DL Architectures with similar works.

As mentioned earlier, within the FTS domain, ML techniques lack experiment standardization as occurs in other domains. However, this work presents a comparison of the proposed

Table 5-8.: Trade-off training times vs DA for experiments using **CO data**.

Experiment	**Training Minutes**	**Testing DA**	**Trade-off**
Proposed 2D CNN B	5	58.83 %	-50 %, -11.48 points
Proposed 3D CNN B	**10**	**70.31 %**	-,-
3D LeNet CNN B	18	70.38 %	+80 %, +0.07 points
3D AlexNet CNN B	25	71.78 %	+150 %, +1.47 points

Table 5-9.: Confusion matrix summary for experiments using **CO data**. Source: TensorFlow execution.

Experiment	**TP**	**FP**	**TN**	**FN**
Proposed 2D CNN A	3,520	1,196	3,520	2,334
Proposed 3D CNN A	*3,867*	*988*	*3,867*	*1,115*
Proposed 2D CNN B	4,015	1,221	4,015	2,739
Proposed 3D CNN B	**4,938**	**1,240**	**4,938**	**902**
3D LeNet CNN B	5,054	1,228	5,054	786
3D AlexNet CNN B	5,064	1,183	5,064	876

Table 5-10.: Comparison across DL architectures.

Experiment	**Data used**	**Models**	**Testing DA**
Proposed 3D CNN	**LOB + tick from 11 US Stocks**	1	74,15 %
MLP [5]	Tick from 2 US Stocks	1	66 %
RNN, LSTM, GRU [7]	Tick from 19 US Stocks	19	66 % – 72 %
CNN [35]	LOB + tick from 1 UK Stock	1	67 %
DBN [74]	LOB + tick from 1 US Stock	1	57 %

architecture against other DL works that have been found in literature reviewed. From Table **4-2** the closest experiment to compare are [5], [7], [35] and [74]. Such works have as a common ground to include HF data, either tick by tick, LOB or both. Data sources include US as well as UK instruments and they use DL as a modelling technique for price direction prediction. Table **5-10** presents details of each work and makes a comparison with the proposed architecture that this work introduces based on DA.

As observed the proposed architecture overpass others architectures such DBN, MLP and RNN family. This may have different explanations, as follows:

- DL underlining origin: It is based on how mammals visual cortex works [75], [76], as a

result it works better for image processing.

- LOB characteristics: As mentioned in chapter three a LOB is a set of lists in time. Therefore a visual like approach may work better for this multivariate FTS. In fact, real human traders make their decisions after doing a visual processing of LOB data.
- Change in time to frequency space of the inputs. The proposed representation gives the ability to change domains of the inputs. In fact, many analytical tools of TS include use of Fourier series or Wavelets, that offers such change in order to facilitate pattern learning. Moreover, an image makes inputs time invariant, which may contribute to improve pattern learning.

The closest research to compare this work is [35]. As observed, both uses a CNN as architecture, as well as LOB and transaction data for inputs. The difference between these two works is the way data inputs are represented. [35] does not make input space transformation, whereas this works proposed a novel representation for input data.

5.3. AT Strategy Results

This section presents results for two different strategies: aggressive and conservative. Differences between these two alternatives relies on a risk aversion constrain.

Figure **5-2** shows the AT Strategy pipeline. The trained model moves to a simulated pro-

Figure 5-2.: AT strategy pipeline.

duction environment where out of sample data goes through the model. Its outputs are feed up into an AT Strategy.

Authors in [5] and [6] introduce an AT using HF data. This strategy executes every minute depending on a given signal from a DNN. The strategy includes a form of transaction cost, expressed in spread widening, but a stop loss neither a take profit. It is a very simple trading strategy that proved to be highly profitable. As mentioned in [15], [20], [49], [67], [72] and [103], an AT strategy should consider aspects such as market impact, liquidity, profitability, risk aversion and other real market conditions in order to materialize a good

trading signal generator into a profitable trading system. In other words, it should include real market characteristics in order to be considered for a real implementation.

5.3.1. Trading Conditions for the Proposed AT Strategy

As mentioned earlier, the AT strategy proposed has two different approaches: one aggressive, another conservative. Table **5-11** presents the summary for these approaches.

Table 5-11.: Summary of trading conditions for AT strategy.

Aspect	**Aggressive Strategy**	**Conservative Strategy**
Trading against LOB data:	Trades against best quotes.	Trades against best quotes.
Take Profit policy:	a. 1 %, b. 0.8 %	a. 1 %, b. 0.8 %
Stop Loss policy:	1 %	1 %
Transaction Cost [2]:	a. 0.2^3 %, b. 0.1^4 %	a. 0.2 %, b. 0.1 %
Liquidity provision:	$Liquidity >= USD15,000$	$Liquidity >= USD15,000$
Trading day Closing Positions:	Yes, any open before 5 minutes market close	Yes, any open before 5 minutes market close
Pre trade analysis:	Do nothing if flat signal, trade any other	Do nothing if flat signal, trade any other if no open positions

Details of the trading conditions are the following:

- Exit strategy by defining an exit point, both Take Profit (TP) and Stop Loss (SL) policies. Any open position five minutes before market closes will be closed at market quotes (bid / ask) for a (sell / buy) position.
- Trading execution against LOB. AT strategy in [5] and [6] traded against transaction data, therefore this AT includes a real market constrain.
- AT as liquidity taker. This is because it trades against best quotes by opening sells and buy orders, depending on a trading signal given by the classification model (**Proposed 3D CNN**).
- Monitoring of market conditions in order to execute trades. This deviated in two strategies. The first one is aggressive, which trades on any up or down trading signal generated. It means that the AT strategy increases stakes in already open positions. The AT strategy does nothing on flat signals. The other strategy, a conservative one,

only keeps one open position, therefore, it does not increase stakes in already open positions, reducing the number of entries to the market.

- Liquidity checking before entering / exiting trades, in order to handle market impact.
- It trades on several assets (8 out of 10 trained) for a 41 trading days time frame.

In general, the proposed AT strategy takes more market constrains into consideration when compared to the AT strategy in [5] and [6]. As a result, it is closer to real market conditions. Moreover, it includes two major risk constrains, creating two variants: one riskier or aggressive strategy and the other one a more conservative approach for risk management. Table **5-11** contrasts such considerations for these two scenarios.

5.3.2. Aggressive Strategy Results

This variant of the AT strategy trades every signal generated by the proposed 3D CNN model that complies with liquidity constrains. As a result, there could be more than one open position at once on the same direction. Figure **5-3** illustrates the general steps of this strategy. This is a riskier approach, since lack of liquidity may affect exit strategies, or changes in market conditions may cause larger losses, while trying to close open positions. Table **5-12** shows a summary of trading results for the aggressive approach. Table **5-13** shows

Table 5-12.: Results summary for the aggressive strategy.

Trading Days	41
Trades Done	4,498
Buy Trades Done	1,935
Sell Trades Done	2,563
Best Performer	ECOPETROL
Worst Performer	PFAVH

detailed information about the best performer stock: ECOPETROL. Conversely, Table **5-14** illustrates what happens to the worst performer stock. Finally, Table **5-15** exhibits results for the whole portfolio (8 stocks). Figure **5-4** shows graphically the portfolio performance.

5.3.3. Conservative Strategy Results

Figure **5-5** includes an open position constrain. As a result, this variant o f the AT strategy trades some signals generated by the proposed 3D CNN model that complies with liquidity constrains. Basically, the conservative approach will open a position if and only if there

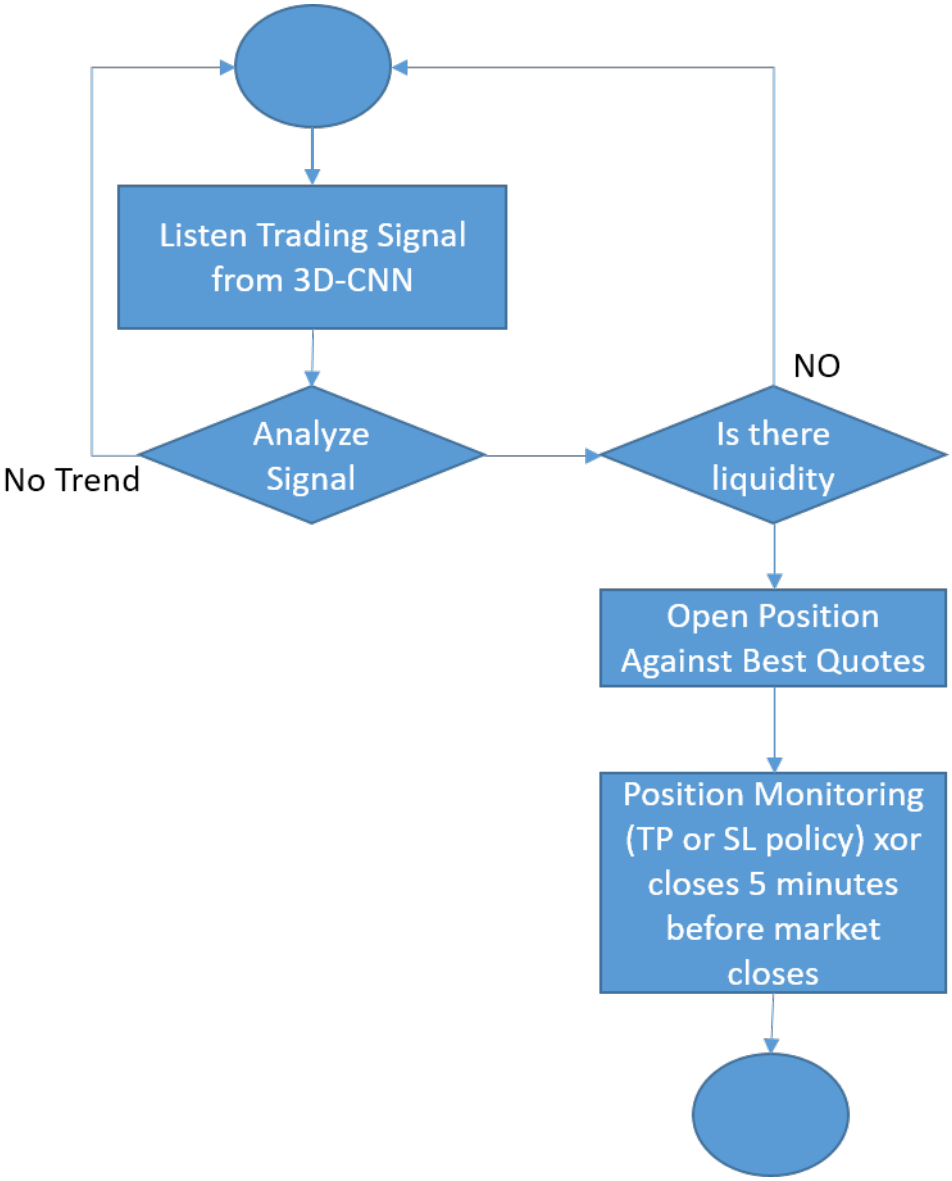

Figure 5-3.: Aggressive AT strategy proposed for this work. Own elaboration.

Table 5-13.: Detailed results for the best performer: ECOPETROL, aggressive strategy.

Transaction Cost	**TP;SL**	**Result**
0,2 %	1,0 %	35,97 %
0,2 %	0,8 %; 1,0 %	21,04 %
0,1 %	1,0 %	42,14 %
0,1 %	0,8 %; 1,0 %	30,98 %

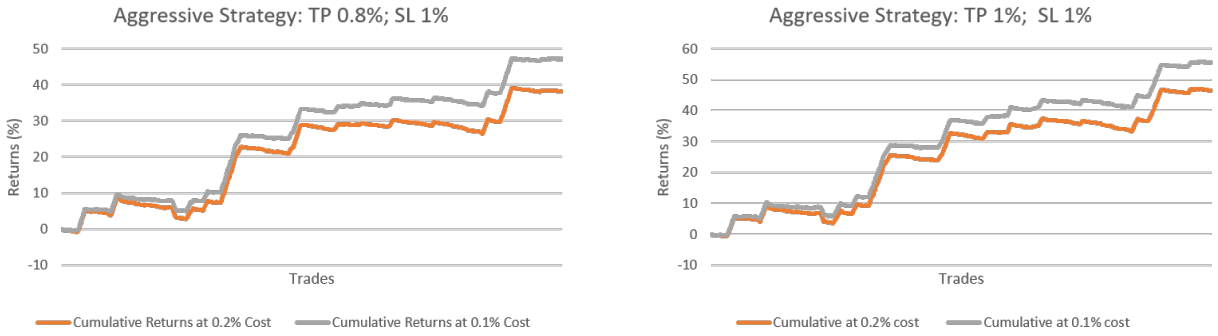

Figure 5-4.: Aggressive ATS results. Own elaboration.

Table 5-14.: Detailed results for the worst performer: PFAVH, aggressive strategy.

Transaction Cost	TP;SL	Result
0,2 %	1,0 %	−9,41 %
0,2 %	0,8 %; 1,0 %	−8,50 %
0,1 %	1,0 %	−8,21 %
0,1 %	0,8 %; 1,0 %	−7,30 %

Table 5-15.: Portfolio detailed results, aggressive strategy.

Transaction Cost	TP;SL	Result
0,2 %	1,0 %	46,54 %
0,2 %	0,8 %; 1,0 %	38,17 %
0,1 %	1,0 %	55,53 %
0,1 %	0,8 %; 1,0 %	47,17 %

are no open positions, otherwise it will do nothing. This condition puts huge constrains in trading, but avoids causing vast losses if market conditions change suddenly. Table **5-16** shows a summary of trading conditions for the conservative approach.

Table **5-17** shows detailed information about the best performer stock: ECOPETROL. Conversely, Table **5-18** illustrates what happens to the worst performer stock. Finally, Table **5-19** exhibits results for the whole portfolio (8 stocks). Figure **5-6** shows graphically the portfolio performance.

In summary, results for different experiments exhibit good performance for the CNN predic-

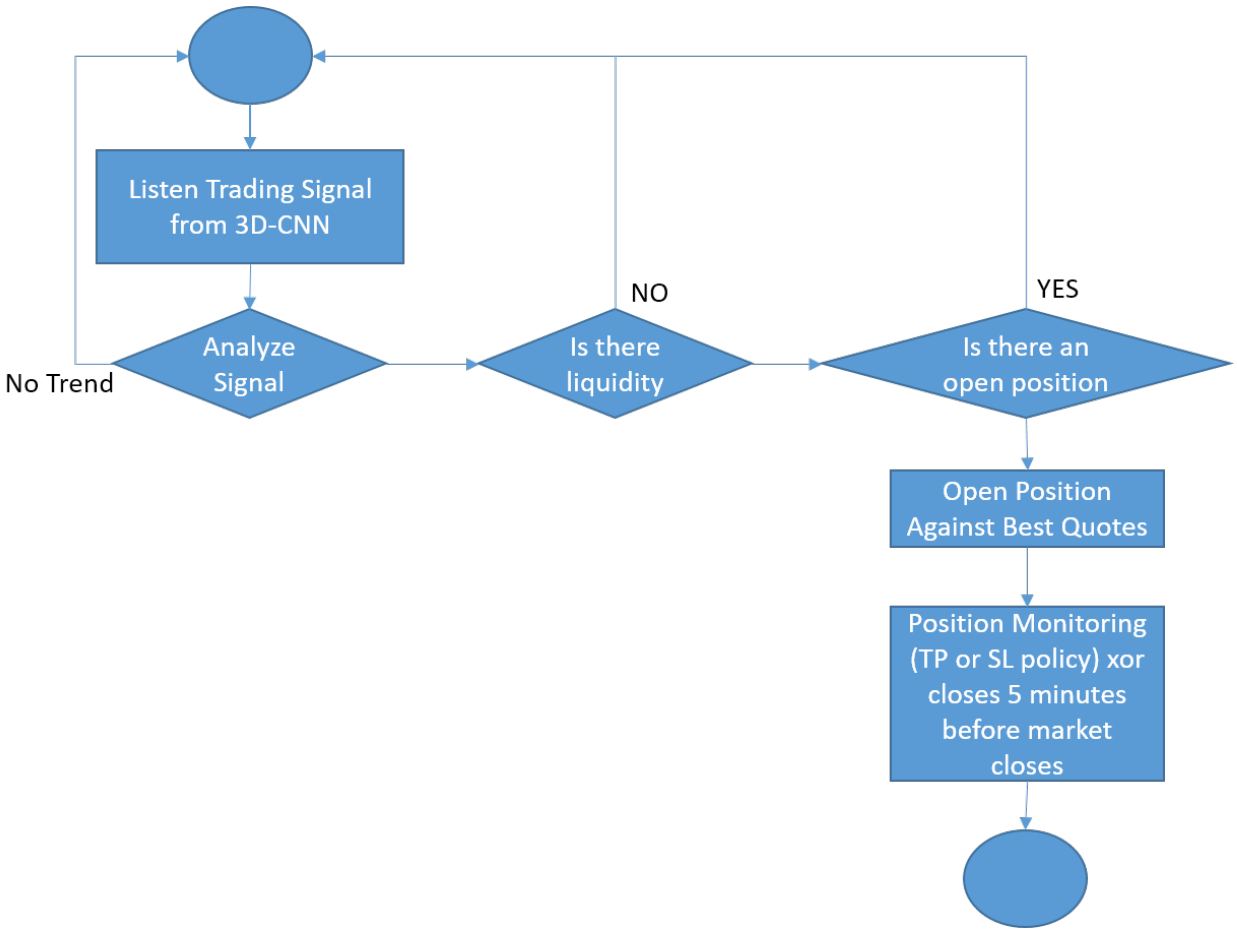

Figure 5-5.: Conservative AT strategy proposed for this work. Own elaboration.

Table 5-16.: Results Summary for the conservative strategy.

Trading Days	41
Trades Done	196
Buy Trades Done	89
Sell Trades Done	107
Best Performer	ECOPETROL
Worst Performer	ISA

tor as well as the AT strategy proposed. On one hand, the proposed 3D-CNN architecture yields very competitive results when predicting price direction for both cases: US and CO market. Moreover, if compared to other well known CNN architectures, the proposed 3D-CNN exhibit a notorious time efficiency advantage with a slightly reduction in prediction accuracy. This trade-off can be very attractive for real implementations, since there are very

Table 5-17.: Detailed results for the best performer: ECOPETROL, conservative strategy.

Transaction Cost	TP;SL	Result
0,2 %	1,0 %	0,56 %
0,2 %	0,8 %; 1,0 %	0,20 %
0,1 %	1,0 %	0,70 %
0,1 %	0,8 %; 1,0 %	0,36 %

Table 5-18.: Detailed results for the worst performer: ISA, conservative strategy.

Transaction Cost	TP;SL	Result
0,2 %	1,0 %	−0,36 %
0,2 %	0,8 %; 1,0 %	−0,42 %
0,1 %	1,0 %	−0,30 %
0,1 %	0,8 %; 1,0 %	−0,35 %

Table 5-19.: Portfolio detailed results, conservative strategy.

Transaction Cost	TP;SL	Result
0,2 %	1,0 %	−0,17 %
0,2 %	0,8 %; 1,0 %	−0,41 %
0,1 %	1,0 %	0,18 %
0,1 %	0,8 %; 1,0 %	−0,20 %

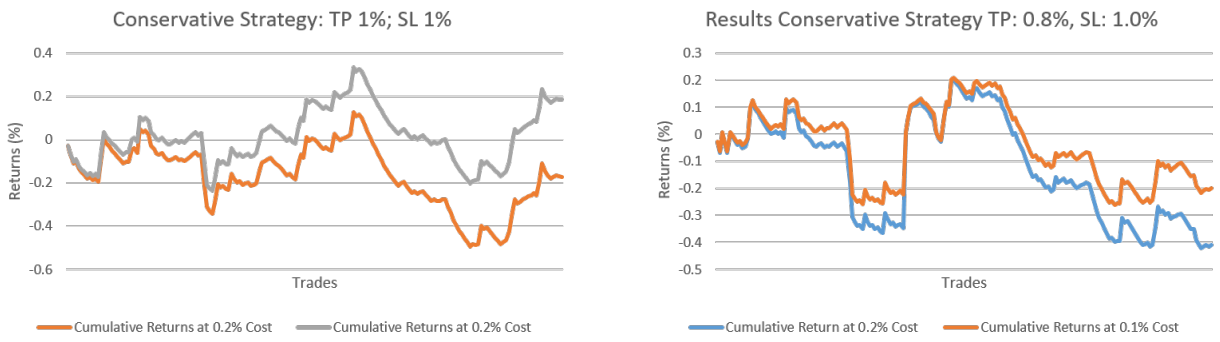

Figure 5-6.: Conservative ATS results. Own elaboration.

short time frames and training times become key under such conditions. Furthermore, the proposed approach presents much better prediction accuracy when compare to other DL architectures.

On the other hand, the proposed AT strategy includes real market constrains, while generating trades from signals of the proposed 3D-CNN predictor. Having said that, results from AT strategy, aggressive and conservative, allow to reaffirm the positive impact that AT brings to markets in terms of liquidity, while generating psotive returns for potential inverstors.

6. Conclusions and Recommendations

6.1. Conclusions

- Model General Performance and Multimodal Data Representation.
  CNN worked well for FTS price direction prediction. DA results are very competitive, in fact, better than other approaches tested before, as seen in Table **5-10**. This allows concluding that CNN identifies and learns patterns. Moreover, data representation is very important because it facilitates learning. The referred works in Table **5-10** illustrated how data representation helps to compensates information asymmetries. For example, authors in [5] and [7] include model training with far more millions of transactions than the ones used for this work. Or authors in [35] use raw LOB data, including updated and cancelled orders. However, this work shows better results working with similar instruments or even less liquid ones. This can be due to the fact that GAF representation for transaction data separates peaks and gives information about more liquid periods of the chosen instruments, as well as that, the proposed LOB representation includes all available information of the LOB.

- More Data Improve Results.
  Results from [7] showed that data driven approaches for modelling FTS improve as more data are available. That work showed a better performance (72 %) for the most liquid stock, i.e. the one with more transactions and more information. Conversely, stocks with low liquidity (less information) exhibited lower DA results. This observation was confirmed in this work among experiments conducted. Overall, results for US market were much better than those for CO market. As previously mentioned, US raw data exceed by approximately 11 times CO data. Moreover, experiments conducted within the CO market, that differ in the number of input images exhibit same behavior: more data means better results. Therefore, information availability is a key driver for these data driven techniques and as more data is available, better results can be achieved.

- Multimodality.
  Multimodal entries [8] help enriching model learning capabilities. This work uses LOB data and Transaction data, resulting in a more complete information of market dynamics. Results on tables **5-2** and **5-6** present such fact when the reader compares 2D CNN versus 3D CNN results. In consequence, multimodal entry representation [8] helps to accomplish such task. On the one hand, this is due to the conversion to frequency domain that an image-like representation implies, and on the other, the representation in polar coordinates, implies an angular measure rather than an euclidean one. This is important because in multidimensional spaces, cosine distance-based metrics may work better than euclidean ones [107].

- Information Completeness.
  The proposed LOB representation that includes all the available information facilitates learning. Results from the original representation in [86] differ widely from those presented in this work, regardless of the different techniques used. The proposed image-like representation allows including more data in different channels. This has an enormous potential with 3D CNN, because it is possible to build many different images of the LOB, that include other features such as time, or other derived variables like those mentioned by [5] and [74].

- Training Times.
  In general, applications of CNN to FTS modelling may offer another advantage related to training times of a model. The reason is that convolutional operator and pooling layers impose a big reduction in the number of parameters and therefore the number of operations. This fact is critical for real AT applications in the HF domain, where speed is very important. Moreover, the proposed 3D CNN architecture simplicity present considerable advantages in training times when compared to other well-known CNN architectures like LeNet and AlexNet. Real AT applications require model retraining periodically. For very short time frames, 25 or 45 minutes may be very long periods, therefore the proposed architecture reduces training times significantly without damaging DA results.(See Tables **5-4** and **5-8**).

- Model Singularity.
  Other consideration that is worth to mention is the ability of generating one single model for multiple instruments. From literature reviewed, none previous work had build

a single model for different instruments. This fact helps to simplify the problem of working for one model for one particular asset, which results in higher operating costs while conducting AT.

- AT Strategy.
  Having a good classifier/predictor is key for an AT System (Figure **2-1**), but transforming it into a good trading strategy is another problem. Several aspects should be taken into account, including market impact, pre trade analysis, exit strategies, risk aversion constrains, technological constrains such as latency, algorithm complexities, among others [15], [49], [37], [72], [94] [101], [103]. This work includes an AT strategy that embraces most of these issues, including pre trade analysis, liquidity constrains, exit strategies, transaction costs and risk aversion policies.

  Results exhibited in Tables **5-12** and **5-15** illustrate that it is possible to build profitable strategies. Such strategies are in line with the typical financial trade-off between profits and risk. As expected the aggressive approach yields much better returns than conservative one, as evidenced in Tables **5-15** and **5-19**.

- Market Constrains.
  This work also induces the impact of HFT in financial markets [15], [49]. The proposed CNN price direction predictor generates almost 5,000 trading signals for a 41-day time window. Under aggressive conditions, all of them were executed and closed. For a small and low liquid market such as the Colombian one, this exercise reveals the potential of HFT for liquidity generation, which implies a better price discovery. On average, there were 15 trading signals for stock in the portfolio, i.e. 120 per day. This quantity of signals for a single market agent is enormous, when usually whitout automatic agents there are no more than 1,500 to 2,000 transactions per day in all the Colombian equity market. In average, an automated trading agent was capable of generating near 10 % of the transactions for the period of analysis.

6.2. Recommendations

There are many different lines of work associated to this proposed model in order to continue building it and improving it:

- Data Representation and Multimodality [8].
  It is possible to include more information in this model from the same data. Experimenting and modelling with additional derived variables may enhance learning capabilities. For example, transaction volume data (quantities exchanged) could be used as an additional FTS. Moreover, including other sources of information, such as financial news, is another possibility to increase multimodality and enrich inputs to the model.

- Combinational Approaches.
  It is possible to mix other architectures such as LSTM or RNN with the proposed one to analyze the trade-off between accuracy and training times. Under this umbrella, it is possible to explore additional representations of raw data and to test different architecture mixes that can take advantage of such representations. Literature reviewed and previous works have shown that RNN, LSTM and GRU works better than DBN and MLP, therefore a combinational apprach that includes CNN and LSTM, RNN or GRU can have better results regarding prediciton accuracy.

- Algorithmic Trading Strategies.
  Building AT strategies is difficult and there are several constrains. This work presented two extremes approaches, however there are a huge number of variations that can be researched. Genetic Algorithms can serve as a tool for parameters searching under this possible extension. It includes parameter searching in different spaces (Market Impact, Liquidity, Exit Strategies, Risk Aversion Constrains), among others.

- Model Retraining.
  As previously mentioned, real AT applications are retrained periodically. Model developing that helps to predict model retraining periods are another possible line of research. On the reviewed literature, it was an issue mentioned that requires more attention and formality [94].

- Model Adaption.
  ML models learn from events presented on training data. But what happens about variations and unknown conditions that are present in the real world? Open World Through Tactics Introduction [97] is a technique used to improve ML models for autonomous vehicles that may be applied to this problem. A research on this topic could be another extension for this work.

A. Annex: Glossary of terms

The following definitions are taken from [15], [92] and [103].

- Ask price: the price that a seller is willing to receive for selling an asset.
- Bid Price: the price that a buyer is willing to pay for buying an asset
- Bid-ask spread: the difference in price between the highest price that a buyer is willing to buy an asset and the lowest price the seller is willing to sell it.
- HTF: European Securities and Markets Authority (ESMA) defines it as trading activities that employ sophisticated algorithms to interpret signals from the markets and implementation of trading strategies that generate high frequency orders transmitted with low latency to the markets in very short time frames. Positions are closed at the end of the trading day and involve the own capital of the brokerage firm, not of its clients. The Securities and Exchange Commission of the US (SEC) defines HTF as an activity made by professional traders in which they generate a large number of trades on a daily basis, using sophisticated algorithms for routing and executing orders in very short time frames, without carrying in positions for the next day.
- In-sample: Dataset used for initial parameter estimation and model selection.[1]
- Limit order: an order to buy / sell a specific quantity of a financial asset to a specific price.
- Liquidity: the ability to buy or sell an asset without greatly affecting its price.
- Look ahead bias: It refers to the use of data for simulation purposes that would not been available during the testing period.[2]
- Order book: the collected limit order to buy or sell an asset.

[1] See https://ec.europa.eu/eurostat/statistics-explained/index.php/Glossary:In-sample_vs._out-of-sample_forecasts

[2] See. https://www.investopedia.com/terms/l/lookaheadbias.asp

- Out of sample: Dataset used to evaluate model forecast performance. Empirical evidence is more trustworthy on out of sample dataset rather than in-sample dataset, because it reflects better the information available to the forecaster in real-time".[3]
- Market efficiency: the concept that market prices reflect the true underlying value of the asset.
- Market order: order to buy / sell a specific amount of a financial asset at the best available price in the order book.
- Market transparency: the ability to see market information. Post trade transparency is the ability to see trade prices and quantities, and pre-trade transparency refers to the ability to see quotes.
- Price discovery: market process whereby new information impounds into asset prices. By doing this, market agents find out best prices (bid/ask) to asset exchange.
- Price efficiency: when an asset price reflects the true underlying value of an asset.
- Transaction cost: the costs traders incur to buy or sell an asset.
- Tick data: Equivalent to transaction data, in other words traded prices for a particular asset. For this work, tick by tick data means that transaction data include all of the transactions registered for an asset.
- Volatility: variability of an asset price over time.

[3]See https://ec.europa.eu/eurostat/statistics-explained/index.php/Glossary:In-sample_vs._out-of-sample_forecasts

B. Annex: Python Code

B.1. Raw LOB to Img

```
import pandas as pd
from os.path import join
from os import listdir
import numpy as np
import json
from collections import defaultdict
import matplotlib.pyplot as plt

def readMetaData(tkr):
    #read Max and Mins for normalizations
    df=pd.read_csv(join(outDir,tkr,'metadata.csv'))
    mins=df.iloc[0].values
    maxs=df.iloc[1].values
    return mins,maxs

def writeMetaData(date,times,delta):
    #main method to read LOB raw data
    i=0
    volsDF=pd.DataFrame()
    puntasDF=pd.DataFrame()
    for t in times:
        data=pd.read_csv(join(inDir,ticker,date,t))
        data=data.sort_values(by=['price'])
        grouped=data.groupby('price')
        tmp=grouped['volume','puntas'].agg(sum)
        tmp['side']=grouped['side'].agg(max)
        tmp=tmp.reset_index()
```

```
        idx=tmp[tmp['side']=='C'].index[-1]
        idxMax=tmp.index[-1]
        if idxMax>idx+puntas:
            if idx>=puntas:
                idxs=[x for x in range(idx-puntas+1,idx+puntas+1)]
            else:
                idxs=[x for x in range(0,idx+puntas+1)]
        else:
            if idx>=puntas:
                idxs=[x for x in range(idx-puntas+1,idxMax)]
            else:
                idxs=[x for x in range(0,idxMax)]
        tmp=tmp[tmp.index.isin(idxs)]
        normVols,normPuntas=normalize(tmp)
        normVols.columns=['h' + t[:-4]]
        normPuntas.columns=['h' + t[:-4]]
        if len(volsDF)== 0:
            volsDF=normVols
            puntasDF=normPuntas
        else:
            volsDF=volsDF.join(normVols,how='outer')
            puntasDF=puntasDF.join(normPuntas,how='outer')
    return volsDF,puntasDF

def normalize(dataIn):
    #normalization using volumes,
        #N (number of lines at the same price), maxVol
    df=dataIn.set_index('price')
    side=df['side']
    df=df[['volume','puntas','maxVol']]
    xMins=mins[1:]
    xMaxs=maxs[1:]
    vals=(df-xMins)/(xMaxs-xMins)
    green=side=='V'
    if len(vals[green])>0:
        vals['volume'][green]=-1*vals['volume'][green]
    return pd.DataFrame(vals['volume']),
```

```
pd.DataFrame(vals['puntas']), pd.DataFrame(vals['maxVol'])

folders=listdir(join(inDir,ticker))
keyDates=[]
puntas=10
mins,maxs = readMetaData()
for folder in folders:
    try:
        files=listdir(join(inDir,ticker,folder))
    except:
        continue
    if len(files)>=100:
        dfVols,dfPuntas,dfMaxVol=writeMetaData(folder,files,4)
        dfVols=dfVols.fillna(0)
        dfPuntas=dfPuntas.fillna(0)
        dfMaxVols=dfMaxVols.fillna(0)
        hours=dfVols.columns
        prices=dfVols.index
        flatDF=dfVols.values.flatten()
        flatPuntas=dfPuntas.values.flatten()
        flatMaxVols=dfMaxVols.values.flatten()
        flat=np.ravel(np.column_stack((flatDF,flatPuntas,flatMaxVols)))
        result=[]
        for i in range(0,len(flat),3):
            if flat[i]>0:
                tmp=np.multiply(flat[i],np.array([0,1,0,0]))
            else:
                tmp=np.multiply(-1*flat[i],np.array([1,0,0,0]))
            tmp=np.append(tmp,1-flat[i+1])
            tmp=np.append(tmp,1-flat[i+2])
            result=np.ravel(np.append(result,tmp))
        M=result.reshape(len(prices),len(hours),4)
        plt.imsave(join(outDir,ticker,str(folder)+'.png'),M)
        pd.DataFrame(prices).to_csv(join(outDir,ticker,
'p'+str(folder)+'.csv'),index=False)
        pd.DataFrame(hours).to_csv(join(outDir,ticker,
'h'+str(folder)+'.csv'),index=False)
```

B.2. Raw Transactions to Img

```
import pandas as pd
from os.path import join
from os import listdir
import numpy as np
import re
import math
from scipy.misc import imsave

def normalize(xIn):
    maxP=metaData['price'][1]
    minP=metaData['price'][0]
    return ((xIn - maxP)+ (xIn-minP))/(maxP-minP)

for ticker in tickers:
    print (ticker)
    allFiles=listdir(join(inDir,ticker))
    tFiles=[x for x in allFiles if re.search(tFilesRx,x)]
    tickDF=pd.read_csv(join(tradeDir,ticker+'.csv'), low_memory=False)
    metaData=pd.read_csv(join(inDir,ticker,'metadata.csv'))
    for f in tFiles:
        times=pd.read_csv(join(inDir,ticker,f))
        times=times.drop_duplicates(subset=['0'])
        date=int(f[1:-4])
        times=list(set(times['0']))
        times=sorted(times)
        N=len(times)
        tickLen=[]
        for i in range(deltaB,N,1):
            upToTime=f[1:-4]+str(times[i-1][1:])
            good=(tickDF['time']>=date) & (tickDF['time']<int(upToTime))
            tmp=tickDF[good]
            if (len(tmp)>0) & (N-i>=deltaB):
                tickLen.append(len(tmp))
                date=int(f[1:-4]+str(times[i-deltaB][1:]))
                dataNorm=normalize(tmp['price'].values)
```

```
                dataCos=
np.array([sum(dataNorm[j:j+1]) for j in range(len(tmp))])
                dataSin=np.sqrt(1 - dataCos**2)
                dataSin=np.matrix(dataSin)
                dataCos=np.matrix(dataCos)
                matrixImg=dataSin.T*dataCos - dataCos.T*dataSin
                imgName=str(date)+'.png'
                plt.imsave(join(inDir,ticker,imgName),
matrixImg,cmap=plt.cm.PiYG)
```

B.3. Training

```
import tensorflow as tf
from os.path import join
from os import listdir, makedirs
import re
import pandas as pd
from PIL import Image
from skimage.color import rgba2rgb,rgb2gray

def cnn_model_fn(features, labels, mode):
    """Model function for CNN."""
    # Input Layer
    #print('Keys for features dict: ', features.keys())
    #print('Shape Labels: ', labels.shape)
    #print('============')
    input_layer = tf.reshape(features["x"],
[-1, 2, 10, 40,4],name='input')
    #input_layer = tf.reshape(features["x"], [-1, 2, 20, 20, 4])
    print('Input laye shape: ', input_layer.shape)
    # Convolutional Layer #1
    conv1 = tf.layers.conv3d(input_layer,filters=20,
                             kernel_size=[1,2,2],
                             padding="same",
                             activation=tf.nn.relu,name='Conv-01')
    print('Layer 1 output shape: ',conv1.shape)
    # Pooling Layer #1
```

```
    pool1 = tf.layers.max_pooling3d(inputs=conv1, pool_size=[2,2,2],
strides=[1,2,2],name='Max-Pool-01')
    print('Pool1 shape: ', pool1.shape)

    # Convolutional Layer #2 and Pooling Layer #2
    conv2 = tf.layers.conv3d(pool1,filters=40,
                             kernel_size=[2,2,2],
                             padding="same",
                             activation=tf.nn.relu,name='Conv-02')
    print('Layer 2 output shape: ', conv2.shape)

    pool2 = tf.layers.max_pooling3d(inputs=conv2, pool_size=[1,2,2],
strides=[1,2,2],name='Max-Pool-02')
    print('Pool2 shape: ', pool2.shape)

    # Dense Layer
    pool2_flat = tf.reshape(pool2, [-1, np.prod(pool2.get_shape().as_list()[1:])
    print('Pool2_flat shape: ',pool2_flat.shape)
    dense = tf.layers.dense(inputs=pool2_flat,
units=pool2_flat.shape[1]+1,
activation=tf.nn.relu,name='Dense')
    dropout = tf.layers.dropout(
      inputs=dense, rate=0.4,
training=mode == tf.estimator.ModeKeys.TRAIN,
name='Dropout')
    print('Dense shape: ', dense.shape)
    # Logits Layer
    logits = tf.layers.dense(inputs=dropout, units=num_classes)

    predictions = {
        # Generate predictions (for PREDICT and EVAL mode)
        "classes": tf.argmax(input=logits, axis=1),
        # Add `softmax_tensor` to the graph. It is used for PREDICT and by the
        # `logging_hook`.
        "probabilities": tf.nn.softmax(logits, name="softmax_tensor")
    }
```

```
    if mode == tf.estimator.ModeKeys.PREDICT:
        return tf.estimator.EstimatorSpec(mode=mode,
 predictions=predictions,
                                          export_outputs={'predict':
tf.estimator.export.PredictOutput(predictions)})

    # Calculate Loss (for both TRAIN and EVAL modes)
    onehot_labels = tf.one_hot(indices=
tf.cast(labels, tf.int32), depth=num_classes)
    loss = tf.losses.softmax_cross_entropy(
      onehot_labels=onehot_labels, logits=logits)

    # Configure the Training Op (for TRAIN mode)
    if mode == tf.estimator.ModeKeys.TRAIN:
        optimizer = tf.train.GradientDescentOptimizer
(learning_rate=0.0001)
        train_op = optimizer.minimize(
            loss=loss,
            global_step=tf.train.get_global_step())
        return tf.estimator.EstimatorSpec(mode=mode,
loss=loss, train_op=train_op)

    # Add evaluation metrics (for EVAL mode)
    eval_metric_ops = {
        "accuracy": tf.metrics.accuracy(labels=labels
, predictions=predictions["classes"]),
        "mean_class_acc":tf.metrics.mean_per_class_accuracy
(labels=labels,
predictions=predictions["classes"],num_classes=3),
        "mse":tf.metrics.mean_squared_error(labels=labels,
predictions=predictions["classes"]),
        "true_positives":tf.metrics.true_positives(labels=labels,
predictions=predictions["classes"]),
        "false_positives":tf.metrics.false_positives(labels=labels,
predictions=predictions["classes"]),
        "false_negatives":tf.metrics.false_negatives(labels=labels,
predictions=predictions["classes"])
```

```
    }
    return tf.estimator.EstimatorSpec(
      mode=mode, loss=loss, eval_metric_ops=eval_metric_ops)

def readData2():
    y_eval=[]
    y_train=[]
    x_eval=[[],[]]
    x_train=[[],[]]
    for i in range(num_classes):
        for ticker in tickers:
            pngFiles=listdir(join(data_folder,str(bookSize)+'lob',
ticker,str(i+1)))
            for png in pngFiles:
                try:
                    xTicks=np.asarray(Image.open(join
(img_folder,ticker,png)).resize((10,40),Image.ANTIALIAS))
                except:
                    continue
                x=np.asarray(Image.open(join(data_folder,
str(bookSize)+'lob',ticker,str(i+1),png)).resize((10,40),Image.ANTIALIAS))
                x_train[0].append(x)
                x_train[1].append(xTicks)
                y_train.append(i)
    #suffle data
    x=np.array(x_train)
    y=np.array(y_train)
    tmp=np.random.permutation(y.shape[0])
    eval_size=int(0.1*len(y))
    train_size=len(y)-eval_size
    x=x[:,tmp,:,:,:]
    y=y[tmp]
    x_train=x[:,0:train_size,:,:,:]
    y_train=y[0:train_size]
    x_eval=x[:,train_size:,:,:,:]
    y_eval=y[train_size:]
    return np.transpose(x_train,[1,0,2,3,4]),
```

```
np.transpose(x_eval,[1,0,2,3,4]),y_train,y_eval

def main(unused_argv):
    # Load training and eval data
    train_data = X_train.astype(np.float32) # Returns np.array
    eval_data = X_eval.astype(np.float32)
    eval_labels=Y_eval

    # Create the Estimator
    cnnModel = tf.estimator.Estimator(model_fn=cnn_model_fn,
 model_dir=modelPath)

    # Set up logging for predictions
    tensors_to_log = {"probabilities": "softmax_tensor"
                      }
    logging_hook = tf.train.LoggingTensorHook(tensors=tensors_to_log,
every_n_iter=10000,at_end=True)

    # Train the model
    train_input_fn = tf.estimator.inputs.numpy_input_fn
(x={"x": train_data},y=Y_train,
batch_size=100,num_epochs=None,shuffle=True)
    cnnModel.train(input_fn=train_input_fn,
steps=100000,hooks=[logging_hook])

    # Evaluate the model and print results
    eval_input_fn = tf.estimator.inputs.numpy_input_fn
(x={"x": eval_data},
y=eval_labels,num_epochs=1,shuffle=False)
    eval_results = cnnModel.evaluate(input_fn=eval_input_fn)
    print(eval_results)
    feature_spec={'x':tf.placeholder(dtype=tf.float32,
 shape=[None,2,10,40,4])}
    input_receiver_fn=
tf.estimator.export.build_raw_serving_input_receiver_fn(feature_spec)
    cnnModel.export_savedmodel
(export_dir_base=savePath,
```

```
serving_input_receiver_fn=input_receiver_fn)
    print('-----')
    predict_results=cnnModel.predict
(input_fn=
input_fn('20180516100009.png','PFAVAL'))
    for idx,prediction in enumerate(predict_results):
        print(idx)
        for key in prediction:
            print('....{}: {}'.format(key,prediction[key]))
```

C. Annex: LOB Images Using The Proposed Representation

This annex shows various images of LOB data using the proposed representation, explained in Chapter 3.

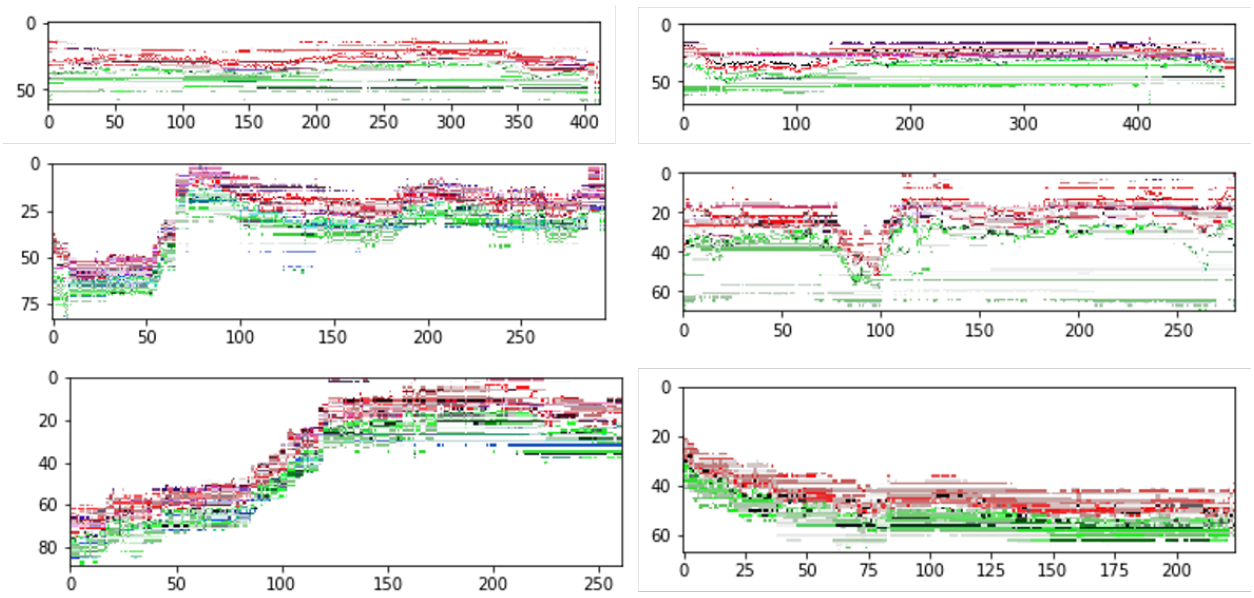

Figure C-1.: Multiple LOB data under proposed representation

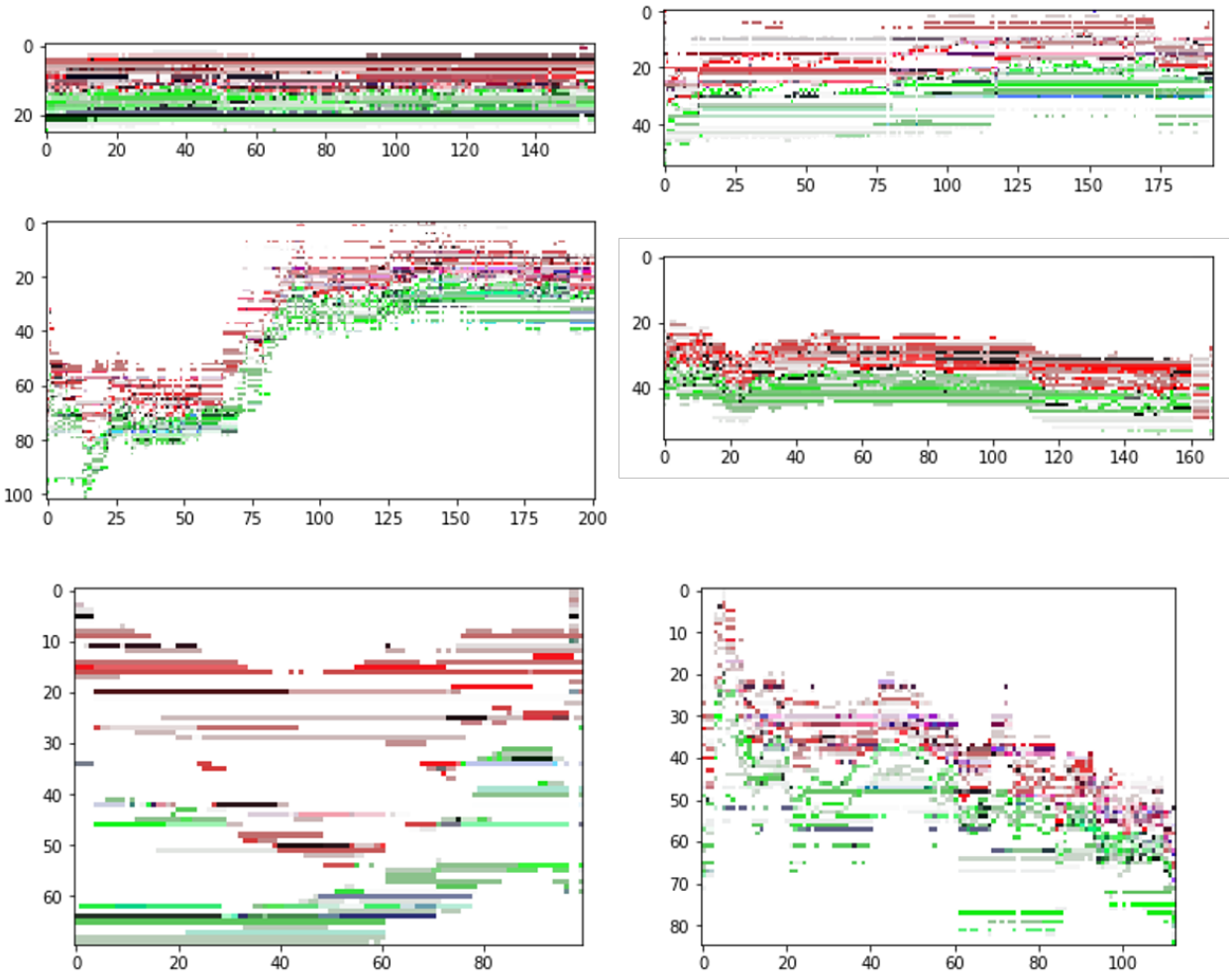

Figure C-2.: Multiple LOB data under proposed representation

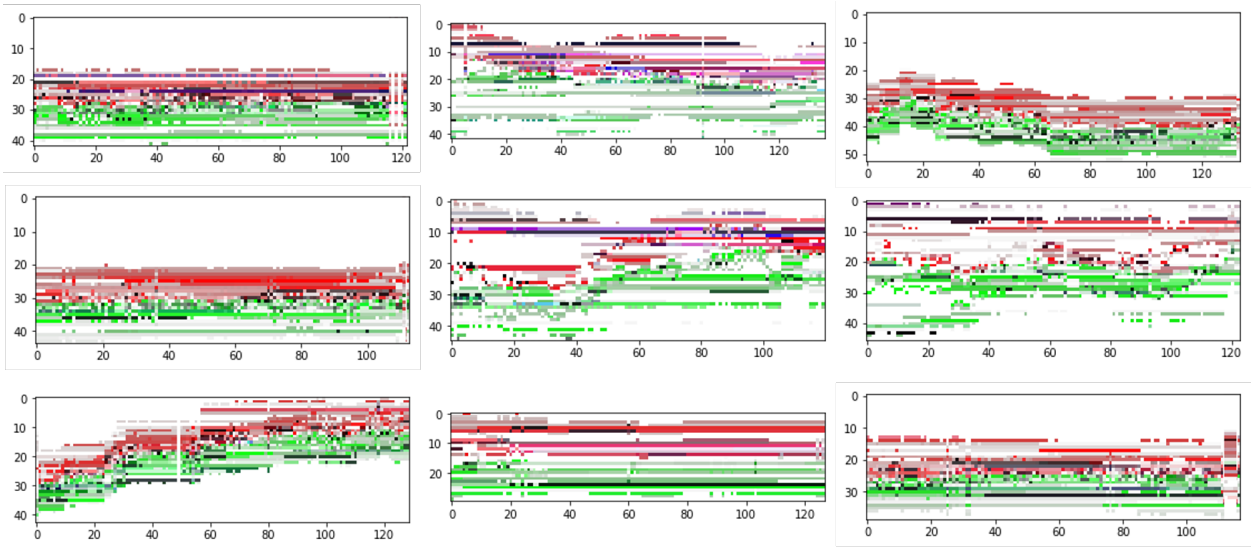

Figure C-3.: LOB data under proposed representation

Bibliografía

[1] Andersen, Ac ; Mikelsen, Stian: A Novel Algorithmic Trading Framework Applying Evolution and Machine Learning for Portfolio Optimization. (2012), S. 198

[2] Antoniadis, Anestis ; Poggi, Jean M. ; Brossat, Xavier: Modeling and stochastic learning for forecasting in high dimensions. In: *Lecture Notes in Statistics* 217 (2015), S. 183–192. – ISBN 9783319187310

[3] Applebaum, David: Lévy processes-from probability to finance and quantum groups. In: *Notices of the American Mathematical Society* 51 (2004), Nr. 11, S. 1336–1347. – ISSN 00029920

[4] Arel, Itamar ; Rose, Derek C. ; Karnowski, Thomas P.: Deep Machine Learning — A New Frontier in Artificial Intelligence Research. In: *IEEE Computational Inteligence Magazine* 5 (2010), Nr. November, S. 13–18

[5] Arévalo, A. ; Nino, J. ; Hernández, G. ; Sandoval, J.: *High-frequency trading strategy based on deep neural networks*. Bd. 9773. 2016. – ISBN 9783319422961

[6] Arévalo, Andrés ; Nino, Hernandez G. ; Sandoval, León Diego ; Aragón, Arbey: Algorithmic Trading Using Deep Neural Networks on High Frequency Data. In: *Communications in Computer and Information Science* Bd. 742. 2017. – ISBN 9783319669625, S. 144–155

[7] Arévalo, Andrés ; Nino, Jaime ; León, Diego ; Hernandez, German ; Sandoval, Javier: Deep Learning and Wavelets for High-Frequency Price Forecasting. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 10861 LNCS (2018), S. 385–399. – ISBN 9783319937007

[8] Arevalo-Ovalle, J.E.: *Multimodal Representation Learning with Neural Networks*, Universidad Nacional de Colombia, Dissertation, 2018

[9] Atsalakis, George S. ; Valavanis, Kimon P.: Surveying stock market forecasting techniques - Part II: Soft computing methods. In: *Expert Systems with Applications* 36 (2009), Nr. 3, S. 5932–5941. – ISBN 0957–4174

[10] Ballings, Michel ; Van Den Poel, Dirk ; Hespeels, Nathalie ; Gryp, Ruben: Evaluating multiple classifiers for stock price direction prediction. In: *Expert Systems with Applications* (2015). – ISBN 0957–4174

[11] Bengio, Yoshua: *Learning Deep Architectures for AI.* 2009. – 1–127 S. – ISBN 2200000006

[12] Bengio, Yoshua ; LeCun, Yan: Scaling learning algorithms towards AI. In: *Large-Sclae kernel machines.* MIT Press, 2007

[13] Bengio, Yoshua ; Thibodeau-Laufer, Éric ; Alain, Guillaume ; Yosinski, Jason: Deep Generative Stochastic Networks Trainable by Backprop. In: *International Conference on Machine Learning* (2014). ISBN 9781634393973

[14] Bhutani, Ankit ; Sharma, Harshvardhan: CS671: Predicting Stock Movement from News. (2013), S. 1–7

[15] Bouveret, A. ; Guillaumie, C. ; Roqueiro, C. ; Winkler, C. ; Nauhaus, S.: High-frequency trading activity in EU equity markets. ESMA Report on Trends, Risks and Vulnerabilities. 2014 (1). – Forschungsbericht. – 41–47 S

[16] Brigo, D. ; Mercurio, F.: *Interest Rate Models – Theory and Practice.* 2012

[17] Britz, Denny. *Recurrent Neural Networks Tutorial, Part 1 – Introduction to RNNs.* 2017

[18] Busseti, Enzo ; Osband, Ian ; Wong, Scott: Deep Learning for Time Series Modeling. 2012. – Forschungsbericht. – 5 S

[19] Cai, Ning ; Kou, S. G.: Option Pricing Under a Mixed-Exponential Jump Diffusion Model. In: *Management Science* 57 (2011), Nr. 11, S. 2067–2081. – ISBN 00251909

[20] Cavalcante, Rodolfo C. ; Brasileiro, Rodrigo C. ; Souza, Victor L F. ; Nobrega, Jarley P. ; Oliveira, Adriano L I.: Computational Intelligence and Financial Markets: A Survey and Future Directions. In: *Expert Systems with Applications* 55 (2016). – ISSN 09574174

[21] Chaboud, Alain ; Chiquoine, E. ; Vega, C.: Rise of the Machines : Algorithmic Trading in the Foreign Exchange Market. In: *Journal of Finance* 69 (2014), Nr. 5, S. 2045–2084

[22] Chao, Jing ; Shen, Furao ; Zhao, Jinxi: Forecasting exchange rate with deep belief networks. In: *The 2011 International Joint Conference on Neural Networks* (2011), S. 1259–1266. – ISBN 978–1–4244–9636–5

[23] Chen, Jou-fan ; Chen, Wei-lun ; Huang, Chun-ping: Financial Time-series Data Analysis using Deep Convolutional Neural Networks. (2016), S. 99–104. ISBN 9781509035557

[24] Chen, Junfei ; Jin, Qiongji ; Chao, Jing: Design of deep belief networks for short-term prediction of drought index using data in the huaihe river basin. In: *Mathematical Problems in Engineering* 2012 (2012). – ISSN 1024123X

[25] Cheng, Wei ; Liu, Shan C. ; Jiao, He Y. ; Qiu, Wan H.: How does limit order book information affect trading strategy and market quality: Simulations of an agent-based stock market. In: *Proceedings - International Conference on Management and Service Science, MASS 2009* (2009). ISBN 9781424446391

[26] Chong, Eunsuk ; Han, Chulwoo ; Park, Frank C.: Deep learning networks for stock market analysis and prediction: Methodology, data representations, and case studies. In: *Expert Systems with Applications* 83 (2017). – ISSN 09574174

[27] Cont, Rama ; Stoikov, Sasha ; Talreja, Rishi: A Stochastic Model for Order Book Dynamics. In: *Operations Research* 58 (2008), Nr. 3, S. 23. – ISBN 0030364X

[28] Dalto, Mladen: Deep neural networks for time series prediction with applications in ultra-short-term wind forecasting / University of Zagreb. – Forschungsbericht. – 9 S

[29] De Gooijer, Jan G. ; Hyndman, Rob J.: 25 Years of Time Series Forecasting. In: *International Journal of Forecasting* 22 (2006), Nr. 3, S. 443–473. – ISBN 0169–2070

[30] Deng, Li: A tutorial survey of architectures, algorithms, and applications for deep learning. In: *APSIPA Transactions on Signal and Information Processing* 3 (2014), 1, S. e2. – ISBN 2048–7703

[31] Deng, Yue ; Bao, Feng ; Kong, Youyong ; Ren, Zhiquan ; Dai, Qionghai. *Deep Direct Reinforcement Learning for Financial Signal Representation and Trading.* 2016

[32] Desell, Travis ; Clachar, Sophine ; Higgins, James ; Wild, Brandon: *Evolving Deep Recurrent Neural Networks using Ant Colony Optimization*. Bd. 4446. 2007. – 154–165 S. – ISBN 978–3–540–71614–3

[33] Dijk, van M.: The social value of finance. In: *Research in Management Series*. Rotterdam : Erasmus Research Institute of Management, 2014, S. 2009–2011

[34] Ding, Xiao ; Zhang, Yue ; Liu, Ting ; Duan, Junwen: Deep learning for event-driven stock prediction. In: *IJCAI International Joint Conference on Artificial Intelligence* 2015-Janua (2015), Nr. Ijcai, S. 2327–2333. – ISBN 9781577357384

[35] Doering, Jonathan ; Fairbank, Michael ; Markose, Sheri: Convolutional neural networks applied to high-frequency market microstructure forecasting. In: *2017 9th Computer Science and Electronic Engineering (CEEC)* (2017), S. 31–36. ISBN 978–1–5386–3007–5

[36] Engle, Robert: Autoregressive Conditional Heteroscedasticity with Estimates of the Variance of the United Kingdom Inflation. In: *Econometrica* 50 (1982), Nr. 4, S. 987–1008

[37] Fraenkle, Joe ; Rachev, Svetlozar: Review : algorithmic trading. In: *Investment Management and Financial Innovations* 6 (2009), Nr. 1, S. 7–20

[38] Gallo, C ; Letizia, C ; Stasio, G: Artificial Neural Networks in Financial Modelling. (2006)

[39] Gerlein, Eduardo A. ; McGinnity, Martin ; Belatreche, Ammar ; Coleman, Sonya: Evaluating machine learning classification for financial trading: An empirical approach. In: *Expert Systems with Applications* (2016). – ISSN 09574174

[40] Gould, Martin D. ; Porter, Mason a. ; Williams, Stacy ; McDonald, Mark ; Fenn, Daniel J. ; Howison, Sam D.: Limit Order Books. In: *Quantitative Finance* 13 (2010), Nr. 11, S. 42. – ISSN 1469–7688

[41] Gunduz, Hakan ; Yaslan, Yusuf ; Cataltepe, Zehra: Intraday prediction of Borsa Istanbul using convolutional neural networks and feature correlations. In: *Knowledge-Based Systems* 137 (2017), S. 138–148. – ISBN 0950–7051

[42] Hamid, Shaikh A. ; Habib, Abraham: FINANCIAL FORECASTING WITH NEURAL NETWORKS. In: *Academy of Accounting and Financial Studies Journal* 18 (2014), Nr. 4, S. 37–56

[43] Heaton, J. B. ; Polson, N. G. ; Witte, J. H. *Deep learning for finance: deep portfolios.* 2017

[44] Hinton, Geoffrey: A Practical Guide to Training Restricted Boltzmann Machines. In: *Computer* 9 (2010), S. 1. – ISBN 978–3–642–35288–1

[45] Hinton, Geoffrey E. ; Osindero, Simon ; Teh, Yee-Whye: Communicated by Yann Le Cun A Fast Learning Algorithm for Deep Belief Nets. In: *Neural Computation* 18 (2006), S. 1527–1554

[46] Hochreiter, Sepp ; Schmidhuber, Jürgen: Long short-term memory. In: *Neural computation* 9 (1997), Nr. 8, S. 1735–80. – ISBN 08997667 (ISSN)

[47] Hu, Yong ; Liu, Kang ; Zhang, Xiangzhou ; Su, Lijun ; Ngai, E.W.T. ; Liu, Mei: Application of evolutionary computation for rule discovery in stock algorithmic trading: A literature review. In: *Applied Soft Computing* 36 (2015), 11, S. 534–551. – ISSN 1568–4946

[48] Huang, Wenhao ; Song, Guojie ; Hong, Haikun ; Xie, Kunqing: Deep Architecture for Traffic Flow Prediction: Deep Belief Networks With Multitask Learning. In: *IEEE Transactions on Intelligent Transportation Systems* 15 (2014), Nr. 5, S. 1–11. – ISBN 1524–9050 VO – 15

[49] K. Sudhakar, S. N. ; Mohan, Y. R.: A Survey on Computer Automated Trading in Indian Stock Markets. In: *International Journal of Mechanical and Production Engineering Research and Development (IJMPERD)* 8 (2018), Nr. 1, S. 531–540. – ISSN 22498001

[50] Katarya, Rahul ; Mahajan, Anmol: A survey of neural network techniques in market trend analysis. In: *Proceedings of the International Conference on Intelligent Sustainable Systems, ICISS 2017* (2018), Nr. Iciss, S. 873–877. ISBN 9781538619599

[51] Kozhan, Roman ; Salmon, Mark: The information content of a limit order book: The case of an FX market. In: *Journal of Financial Markets* (2012). – ISBN 1386–4181

[52] Krizhevsky, Alex ; Sutskever, Ilya ; Hinton, Geoffrey E.: ImageNet Classification with Deep Convolutional Neural Networks. In: *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1.* USA : Curran Associates Inc., 2012 (NIPS'12), S. 1097–1105

[53] Krollner, Bjoern ; Vanstone, Bruce ; Finnie, Gavin: Financial time series forecasting with machine learning techniques : A survey. In: *European Symposium on Artificial Neural Networks ESANN2010*. Bruges, Belgium, 2010. – ISBN 2930307102

[54] Lai, Anthony ; Li, M K. ; Pong, Foon W.: Forecasting Trade Direction and Size of Future Contracts Using Deep Belief Network / Standford ML Class 2012. 2012. – Forschungsbericht

[55] Längkvist, Martin ; Karlsson, Lars ; Loutfi, Amy: A review of unsupervised feature learning and deep learning for time-series modelling. In: *Pattern Recognition Letters* 42 (2014), Nr. 1. – ISBN 0167–8655

[56] Larsen, Fredrik: Automatic stock market trading based on Technical Analysis / Norwegian University of Science and Technology. 2007 (February). – Forschungsbericht. – 92 S

[57] Laserson, Jonathan: From Neural Networks to Deep Learning. In: *XRDS: Crossroads, The ACM Magazine for Students* 18 (2011), Nr. 1, S. 29. – ISSN 15284972

[58] Law, Martin. *A simple introduction to support vector machines*. 2005

[59] Lawrence, Ramon: Using Neural Networks to Forecast Stock Market Prices. In: *Methods* (1997), S. 1–21

[60] Lecun, Y. ; Bottou, L. ; Bengio, Y. ; Haffner, P.: Gradient-based learning applied to document recognition. In: *Proceedings of the IEEE* 86 (1998), Nov, Nr. 11, S. 2278–2324. – ISSN 0018–9219

[61] Lecun, Yann ; Bengio, Yoshua ; Hinton, Geoffrey: Deep learning. In: *Nature* 521 (2015), Nr. 1. – ISBN 9780521835688

[62] Lee, Honglak: Deep Machine Learning : Panel Presentation. (2013)

[63] Lee, Honglak ; Grosse, Roger ; Ranganath, Rajesh ; Ng, Andrew Y.: Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In: *Proceedings of the 26th Annual International Conference on Machine Learning ICML 09* 2008 (2009), S. 1–8. – ISBN 9781605585161

[64] Lee, Honglak ; Pham, Peter T. ; Largman, Yan ; Ng, Andrew Y.: Unsupervised feature learning for audio classification using convolutional deep belief networks. In: *Nips* 9 (2009), S. 1096–1104

[65] León, Diego ; Aragón, Arbey ; Sandoval, Javier ; Hernández, Germán ; Arévalo, Andrés ; Nino, Jaime: Clustering algorithms for Risk-Adjusted Portfolio Construction. In: *Procedia Computer Science* 108 (2017), Nr. October, S. 1334–1343. – ISSN 18770509

[66] Levendovszky, János ; Kia, Farhad: Prediction based – high frequency trading on financial time series. In: *Periodica Polytechnica Electrical Engineering and Computer Science* 56 (2012), Nr. 1, S. 29–34

[67] Linton, Oliver B. ; Mahmoodzadeh, Soheil: Implications of High-Frequency Trading for Security Markets. In: *Ssrn* (2018). – ISSN 1941–1383

[68] Liu, James N K. ; Hu, Yanxing ; You, Jane J. ; Chan, Pak W.: Deep Neural Network Based Feature Representation for Weather Forecasting, 2014

[69] Makridakis, S. ; Hibon, M.: The M3-Competition: results, conclusions and implications. In: *International Journal of Forecasting* 16 (2000), S. 451–476

[70] Makridakis, S. ; Spiliotis, E. ; Assimakopoulos, V.: The M4 Competition: Results, findings, conclusion and way forward. In: *International Journal of Forecasting* 34 (2018), S. 802–808. – ISSN 0169–2070

[71] Mordecki, Ernesto. *Procesos de Levy en matematica financiera y acturial.* 2000

[72] Moriyasu, Hiroshi ; Wee, Marvin ; Yu, Jing: The role of algorithmic trading in stock liquidity and commonality in electronic limit order markets. In: *Pacific Basin Finance Journal* 49 (2018), Nr. October 2017, S. 103–128. – ISSN 0927538X

[73] Nino, Jaime ; Arévalo, Andrés ; Leon, Diego ; Hernandez, German ; Sandoval, Javier: Price Prediction with CNN and Limit Order Book Data. In: *Applied Computer Science in Engineering* Bd. 1. 2018, S. 124–135

[74] Nino, Jaime H. ; Hernández, Germán: Price Direction Prediction on High Frequency Data Using Deep Belief Networks. In: *Communications in Computer and Informations Science* 657 (2016), S. 74–83

[75] Ohlshausen, B ; Field, D J.: Natural Image statistics and efficient coding. In: *Network: Computation in Neural Systems* 7 (1996), Nr. January, S. 333–339

[76] Ohlsson, Stellan. *Stellan Ohlsson: Deep Learning: How the Mind Overrides Experience.* 2012

[77] Ortega, Luis F.: A Neuro-wavelet Method for the Forecasting of Financial Time Series. In: *Proceedings of the World Congress on Engineering and Computer Science* Bd. I. San Franncisco, 2012. – ISBN 9789881925169

[78] Paliwal, Mukta ; Kumar, Usha A.: Neural networks and statistical techniques: A review of applications. In: *Expert Systems with Applications* 36 (2009), 1, Nr. 1, S. 2–17. – ISSN 0957–4174

[79] Preethi, G. ; Santhi, B.: Stock market forecasting techniques: A survey. In: *Journal of Theoretical and Applied Information Technology* 46 (2012), Nr. 1, S. 24–30. – ISSN 19928645

[80] Ranaldo, Angelo: Order aggressiveness in limit order book markets. In: *Journal of Financial Markets* 7 (2004), Nr. 1, S. 53–74. – ISBN 1386–4181

[81] Ranzato, Marc ; Boureau, Y-Lan ; Chopra, Sumit ; LeCun, Yan: A Unified Energy-Based Framework for Unsupervised Learning. In: *Proc. Conference on AI and Statistics (AI-Stats)*, 2007

[82] Russell, Jeffrey R. ; Kim, Taejin: A new model for limit order book dynamics. In: *Volatility and Time Series Econometrics* i (2010). ISBN 9780199549498

[83] Salakhutdinov, Ruslan ; Hinton, Geoffrey: Deep Boltzmann Machines. In: *Artificial Intelligence* 5 (2009), Nr. 3, S. 448–455. – ISSN 15324435

[84] Sandoval, Javier: *High Frequency Exchange rate prediction using dynamic bayesian networks over the limit order book information.*, Universidad Nacional de Colombia, Dissertation, 2015

[85] Sandoval, Javier ; Hernández, Germán: Computational Visual Analysis of the Order Book Dynamics for Creating High-frequency Foreign Exchange Trading Strategies. In: *Procedia Computer Science* 51 (2015), S. 1593–1602. – ISSN 18770509

[86] Sandoval, Javier ; Nino, Jaime ; Hernandez, German ; Cruz, Andrea: Detecting Informative Patterns in Financial Market Trends Based on Visual Analysis. In: *Procedia Computer Science* 80 (2016), S. 752–761. – ISSN 18770509

[87] Sapankevych, Nicholas ; Sankar, Ravi: Time Series Prediction Using Support Vector Machine: A Survey. In: *IEEE Computational Inteligence Magazine* (2009), Nr. May, S. 24–38

[88] Schmidhuber, J: Deep Learning in Neural Networks: An Overview. In: *arXiv preprint arXiv:1404.7828* 61 (2014), S. 1–66. – ISSN 08936080

[89] Selvin, Sreelekshmy ; Vinayakumar, R ; Gopalakrishnan, E. A. ; Menon, Vijay K. ; Soman, K. P.: Stock price prediction using LSTM, RNN and CNN-sliding window model. In: *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)* (2017), S. 1643–1647. ISBN 978–1–5090–6367–3

[90] Sezer, Omer B. ; Ozbayoglu, Ahmet M.: Algorithmic financial trading with deep convolutional neural networks: Time series to image conversion approach. In: *Applied Soft Computing Journal* 70 (2018), S. 525–538. – ISSN 15684946

[91] Shen, Furao ; Chao, Jing ; Zhao, Jinxi: Forecasting exchange rate using deep belief networks and conjugate gradient method. In: *Neurocomputing* 167 (2015). – ISSN 18728286

[92] Shiller, R ; Press, Princenton U. (Hrsg.): *Finance and the Good Society*. 1. Princenton : Princenton University Press, 2012. – 304 S. – ISBN 978–0–691–15488

[93] Singh, Ritika ; Srivastava, Shashi: Stock prediction using deep learning. In: *Multimedia Tools and Applications* 76 (2017), Nr. 18, S. 18569–18584. – ISBN 13807501

[94] Stasinakis, Charalampos ; Sermpinis, Georgios: FINANCIAL FORECASTING AND TRADING STRATEGIES : A SURVEY *. In: Dunis, Christian (Hrsg.) ; Spiros, Likothanassis (Hrsg.) ; Karathanasopoulos, Andreas (Hrsg.) ; Sermpinis, Georgios (Hrsg.) ; Theofilatos, Konstantinos (Hrsg.): *Computational Intelligence Techniques and Trading and Investment*. 2014. – ISBN 9780415636803, Kapitel 2

[95] Stiglitz, Joseph E.: Financial markets and development. In: *Oxford Review of Economic Policy* 5 (1989), Nr. 4, S. 55–68. – ISBN 0266–903X

[96] Sureshkumar, Kk ; Elango, Nm: Performance Analysis of Stock Price Prediction using Artificial Neural Network. In: *Global Journal of Computer ...* 12 (2012), Nr. 1

[97] Sylnice, J.R. ; Alferez, G.H.: Dynamic Evolution of Simulated Autonomous Cars in the Open World Through Tactics. In: *Proceedings of the Futures Technologies Conference FTC 2018* 880 (2019). – ISBN 978–3–030–02685–1

[98] Szegedy, Christian ; Liu, Wei ; Jia, Yangqing ; Sermanet, Pierre ; Reed, Scott ; Anguelov, Dragomir ; Erhan, Dumitru ; Vanhoucke, Vincent ; Rabinovich, Andrew: Going deeper with convolutions. In: *Proceedings of the IEEE Computer*

Society Conference on Computer Vision and Pattern Recognition Bd. 07-12-June-2015, 2015. – ISBN 9781467369640

[99] Takeuchi, Lawrence ; Lee, Yya: Applying Deep Learning to Enhance Momentum Trading Strategies in Stocks. In: *Cs229.Stanford.Edu* (2013), Nr. December 2013. ISBN 7845005002

[100] Tay, Francis E H. ; Cao, Lijuan: Application of support vector machines in financial time series forecasting. In: *Omega* 29 (2001), S. 309–317. – ISBN 0305–0483

[101] Treleaven, Philip ; Galas, Michal ; Lalchand, Vidhi: Algorithmic trading review. In: *Communications of the ACM* 56 (2013), Nr. 11, S. 76–85. – ISSN 00010782

[102] Tsantekidis, Avraam ; Passalis, Nikolaos ; Tefas, Anastasios ; Kanniainen, Juho ; Gabbouj, Moncef ; Iosifidis, Alexandros: Forecasting Stock Prices from the Limit Order Book Using Convolutional Neural Networks. In: *2017 IEEE 19th Conference on Business Informatics (CBI)* (2017), S. 7–12. ISBN 978–1–5386–3035–8

[103] UK's Gov Office: The Future of Computer Trading in Financial Markets An International Perspective. 2012. – Forschungsbericht. – 184 S

[104] Vvedenskaya, N ; Suhov, Y ; Belitsky, V: A non-linear model of limit order book dynamics. In: *2011 IEEE International Symposium on Information Theory Proceedings* (2011), S. 1260–1262. – ISBN 978–1–4577–0596–0

[105] Wang, Donglin ; Li, Yajie: A novel nonlinear RBF neural network ensemble model for financial time series forecasting. In: *Third International Workshop on Advanced Computational Intelligence* (2010), S. 86–90. ISBN 978–1–4244–6334–3

[106] Wang, Yanshan ; Choi, Ic: Market Index and Stock Price Direction Prediction using Machine Learning Techniques: An empirical study on the KOSPI and HSI. In: *arXiv preprint arXiv:1309.7119* 00 (2013), S. 1–13

[107] Wang, Zhiguang ; Oates, Tim: Encoding Time Series as Images for Visual Inspection and Classification Using Tiled Convolutional Neural Networks. In: *Workshops at the Twenty-Ninth AAAI Conference on Artificial Intelligence* (2015), Nr. January 2015. ISBN 9781577357254

[108] White, Halbert: Economic prediction using neural networks: The case of IBM daily stock returns. In: *Neural Networks, 1988., IEEE International Conference on* (1988), S. 451 – 458. ISBN 0–7803–0999–5

[109] Wiatowski, Thomas ; Bölcskei, Helmut: A Mathematical Theory of Deep Convolutional Neural Networks for Feature Extraction. (2015), S. 1–22. – ISSN 0018–9448

[110] Wu, Jianxin: Introduction to Convolutional Neural Networks. 2016. – Forschungsbericht. – 1–28 S. – ISBN 9783642286605

[111] Yao, Jingtao ; Tan, Chew L. ; Poh, Hean-Lee: Neural Networks for Technical Analysis: a Study on Klci. In: *International Journal of Theoretical and Applied Finance* 02 (1999), Nr. 2, S. 221–241. – ISSN 0219–0249

[112] Yeh, Shu-hao ; Wang, Chuan-ju: Corporate Default Prediction via Deep Learning. In: *Intermational Institute of Forecasting* (2014)

[113] Yiwen, Yu: The limit order book information and the order submission strategy: A model explanation. In: *Proceedings - ICSSSM'06: 2006 International Conference on Service Systems and Service Management* 1 (2007), Nr. 200510463025, S. 687–691. ISBN 1424404517

[114] Yu, Dong ; Deng, Li: Deep Learning and Its Applications to Signal and Information Processing [Exploratory DSP]. In: *Signal Processing Magazine, IEEE* 28 (2011), Nr. January, S. 145–154. – ISBN 1053–5888 VO – 28

[115] Zekic, Marijana: Neural network applications in stock market predictions-a methodology analysis. In: *proceedings of the 9th International Conference on ...* (1998), S. 1–11

[116] Zheng, Yi ; Liu, Qi ; Chen, Enhong ; Ge, Yong ; Zhao, J. L.: Time series classification using multi-channels deep convolutional neural networks. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 8485 LNCS (2014), S. 298–310. – ISBN 9783319080093

[117] Zhu, Chengzhang ; Yin, Jianping ; Li, Qian: A Stock Decision Support System based on DBNs. In: *Journal of Computational Information Systems* 2 (2014), S. 883–893. – ISSN 15539105