FACULTY
OF INFORMATION
TECHNOLOGY
CTU IN PRAGUE

ASSIGNMENT OF BACHELOR’S THESIS

Title:	Market signal algorithm based on image recognition
Student:	Andrey Babushkin
Supervisor:	Ing. Stanislav Kuznetsov
Study Programme:	Informatics
Study Branch:	Knowledge Engineering
Department:	Department of Applied Mathematics
Validity:	Until the end of summer semester 2019/20

Instructions

The goal of the work is to create a new market signal based on the image recognition algorithm. The signal should predict a future movement of a price of a cryptocurrency pair and tell investors/traders if a SHORT/LONG position should be open.

1. Collect historical exchange data for some cryptocurrency pair from an exchange.
2. Analyze collected data, make preprocessing and make labels.
3. Using Data Mining algorithms for image recognition to build a model that predicts a future movement of a crypto pair price.
4. Evaluate the accuracy of predictions in the period and discuss how the quality of predictions can be improved.

References

Will be provided by the supervisor.

Ing. Karel Klouda, Ph.D. Head of Department	doc. RNDr. Ing. Marcel Jiřina, Ph.D. Dean

Prague January 22, 2019

FACULTY
OF INFORMATION
TECHNOLOGY
CTU IN PRAGUE

Bachelor’s thesis

Market signal algorithm based on image recognition

Andrey Babushkin

Department of Applied Mathematics
Supervisor: Ing. Stanislav Kuznetsov

May 16, 2019

Acknowledgements

I want to thank my supervisor, Ing. Stanislav Kuznetsov, for providing me with support and valuable feedback while creating this thesis. Also, I tremendously thank his colleague Milan who provided me with a rich list of literature in the trading area. I am really grateful that I was given a chance to enter market.

Also, I want to thank my mother, who has been supporting me morally for several years. Without her, it would be much more challenging to integrate into a foreign country. Her wise advice helped me overcome many difficulties that I met.

Many thanks go to my dear friends with whom we made a library our home. A special thank to Mykyta Boiko, who allowed me to use his funny photo in this thesis.

Declaration

I hereby declare that the presented thesis is my own work and that I have cited all sources of information in accordance with the Guideline for adhering to ethical principles when elaborating an academic final thesis.

I acknowledge that my thesis is subject to the rights and obligations stipulated by the Act No. 121/2000 Coll., the Copyright Act, as amended, in particular that the Czech Technical University in Prague has the right to conclude a license agreement on the utilization of this thesis as school work under the provisions of Article 60(1) of the Act.

In Prague on May 16, 2019 . . . .

Citation of this thesis

Babushkin, Andrey. *Market signal algorithm based on image recognition.* Bachelor's thesis. Czech Technical University in Prague, Faculty of Information Technology, 2019.

Abstrakt

Miliony transakcí jsou zpracovány na světových trzích. Obchodníci bojují o zisky prodejem a nákupem různých aktiv po celém světě. V této nekonečné válce za peníze vznikají tuny různých technik, cílem kterých je předpovědět cenu a pomoct obchodníkům učinit správná rozhodnutí.

Tato práce navrhuje nový přístup k analýze historických OHLCV dat a generování tržních signálů, které obchodníkům sdělují, jaké kroky by měly být učiněny právě teď. K zavedení nového modelu využíváme konvoluční neuronové sítě v kombinaci s plně propojenými neuronovými sítěmi. Dále diskutujeme techniku pro vytvoření tréninkové sady dat z vizuální reprezentace tržního indikátoru nazvaného Index relativní síly.

Navrhovaný model dosahuje 69% přesnosti z dat o kryptometrovém páru ETH/BTC, který, pokud vezmeme v úvahu celkovou volatilitu kryptomarket, je dobrou základnou pro budoucí řešení.

Klíčová slova kryptoměny, predikce ceny, indikátory technické analýzy, burzovní signály, deep learning, bitcoin, technická analýza, RSI

Abstract

Millions of transactions are processed in worldwide markets. Traders fight for profits by selling and buying different assets worldwide. In this endless war for money, tons of different techniques are being created, attempting to predict the price in advance and help traders make correct decisions.

This thesis proposes a novel approach to analyse historical data of the price and generate market signals that tell traders what action should be taken right now. We make use of convolutional neural networks in combination with fully-connected ones to introduce a new model. Moreover, we discuss a technique to create a training dataset from a visual representation of a market indicator called the Relative Strength Index.

The proposed model achieves 69% accuracy on data of the ETH/BTC cryptocurrency pair that, if taking into account the overall volatility of cryptomarkets, is a good baseline for future solutions.

Keywords cryptocurrencies, price prediction, market indicator, market signal, deep learning, bitcoin, technical analysis, RSI

Contents

List of Figures

List of Tables

Introduction

For many decades, millions of beginners and professional traders have been trying to find the Holy Grail, a mathematical expression describing the behaviour of a price of currency pairs and shares. And this is not surprising, the daily turnover of only the foreign exchange market, also known as Forex, averaged $5.1 trillion, according to the Bank of International Settlements [1]. If someone finds a way to predict the future price, they will be able to use this knowledge to earn millions of dollars.

Unfortunately, describing markets is not that simple, mainly since markets indices are highly volatile and behave mostly randomly. Furthermore, there are many factors that can influence the price of an index including the overall political situation, a possible economic crisis, the value of assets that countries or companies hold and mostly the random nature of the behaviour of human beings who play the central role in creating the value of assets by creating a demand for those.

However, it is worth mentioning that the analysis of markets is still possible, and even particular findings of patterns of the market behaviour can generate enormous profits. A suitable proof for this statement is an American hedge fund Renaissance Technologies LLC, the establisher of a vast and profitable portfolio called Medallion Fund. *„From 2001 through 2013, the fund's worst year was a 21 per cent gain, after subtracting fees. Medallion reaped a 98.2 per cent gain in 2008, the year the Standard & Poor's 500 Index lost 38.5 per cent"* [2].

Problem statement

There are a lot of different approaches on how to predict the price of an asset. One can perform an in-depth analysis of a market, also known as a *fundamental analysis*. This method can be quite useful, but it meets many difficulties and requires many resources. Firstly, a person who wants to generate profits using fundamental analysis needs to have in-depth knowledge in economics to understand all factors that can influence an asset's price including annual company reports, business strategies, marketing, business agreements, etc. Secondly, this type of analysis is time-consuming, and this can become a stumbling block for reacting to market changes promptly. Last but not least much information needed to effectively predict the future of an asset is not open to the public and being able to access insider data is either expensive or not possible at all without being a part of closed groups of people who are in close touch to a company's kitchen.

Another approach of a price prediction is using historical data from markets and learning common patterns that are followed by price changes either in an upward direction or downward. Most traders use different kinds of graphical representation of raw data; the visualisation helps to look at data from different perspectives and making decisions about upcoming trends by observing graphical patterns that are present in raw data. Figure 0.1 shows a simple example of how graphical representation can indicate forthcoming trends. The plot type that is used in this example is called *a candlestick chart* and will be discussed in details. Also, a popular indicator called Bollinger Bands is drawn over the candlestick chart. *An indicator* is a mathematical expression that is calculated from raw data, such as prices and market volumes, and are used by traders who employ technical analysis for predicting future price movements. From this example, we see that if a candle crosses the upper edge of Bollinger Bands, it can indicate that an asset is overbought and we can expect a forthcoming burst of sales, therefore, a price will probably go down.

With no doubt, the most effective method is combining technical and fundamental analyses altogether and making decisions based on the output from both of them. However, this thesis focuses on only one market signal, a small piece of thousands of events happening every second on every market. We will propose a state-of-the-art dataset for a deep learning neural network which includes computing an indicator on raw data, filtering it, post-processing a plot, and extracting several additional features. Additionally, we test several configurations of the model and suggest parameters that show the best accuracy in predictions.

Motivation

Before we start, let us discuss the motivation behind this work. Firstly, due to the relatively recent growth of popularity of cryptocurrencies market many promising cryptocurrency projects were born and the market has been developing in high velocity during past several years. Cryptocurrencies themselves propose a way of managing and transferring money, and during a decade the crypto became a whole new philosophy, some new liberal movement. Secondly, my research shows that in spite of thousands of existing signals and studies in

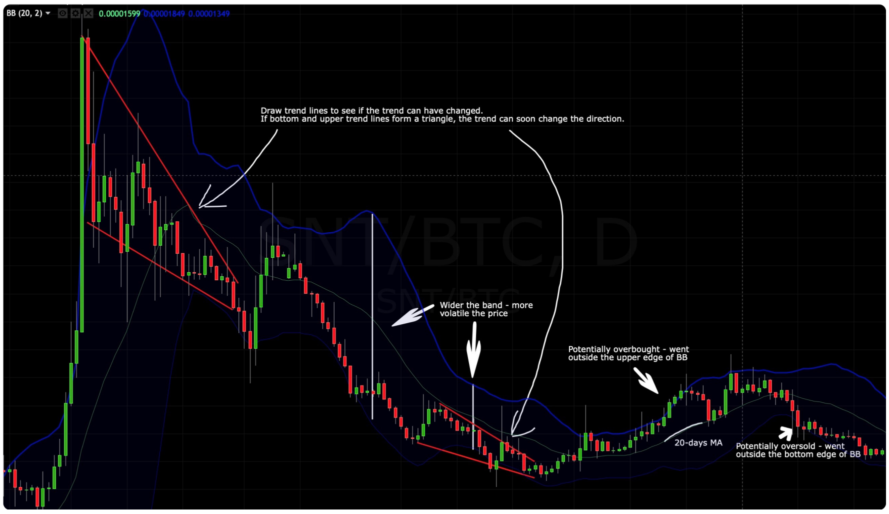


Figure 0.1: A candlestick chart of a cryptocurrency pair SNT/BTC with an additional indicator called Bollinger Bands. From this plot, one can observe current trends that are present on the market and guess future price movements. Bollinger Bands show how volatile market is and can indicate if an asset is overbought or oversold.

the area of technical analysis, there are not so many signals trying to imitate the behaviour of a daily trader. Traders look at candlestick charts and plots of a set of indicators and make a decision what action to take next based on the information they *see* and not a sequence of raw numbers. Last but not least, any market is a challenging race between millions of traders all around the world, and any new approach can potentially result in significant profits.

Related works

Although there is plenty of works related to market signals or trading itself, the vast majority of them is closed to the public. And it is not surprising, as a working algorithm that can automatically generate profits becomes outdated and stops working right after it becomes well known to traders. This effect is easy to explain: if we say to 1000 people to buy an asset at 17:00, half of them will buy it at 16:30 to catch an up-going trend after 17:00 and, therefore, it will break the whole proposed model.

In any case, there are some publicly available papers. For example, due to the rising popularity of deep learning algorithms, some studies are focused on deep learning itself, making it the heart of a study [3]. Such models can find patterns of a price movement and show high performance. However, they are unreliable due to rapid market changes and, as it was mentioned before, quite erratic behaviour of a market. Unfortunately, small patterns that a neural network can hit can be useless at the moment but pretty useful in general. Neural networks tend to forget information that came a long time ago, as described in [4], which is not wanted behaviour if we use them for market predictions.

Structure

The study is divided into four parts. In Chapter 1, we describe theoretical essentials without which the work would not be possible. You will learn about markets and technical analysis, how the Relative Strenght Index and moving averages work, and what kinds of neural networks were used. Furthermore, you will learn about filtering techniques and using Fourier transform for predicting future prices. Chapter 2 describes our proposed model and dataset creation. We evaluate our model and discuss results in Chapter 3, while in Chapter 4, we briefly have a look at the implemented package called RSIVision.

Chapter 1

Theoretical background

In this chapter, we will dive in all theoretical background and technologies that are used in this work. If you are familiar with all techniques and definitions described below, feel free to skip to the next chapter. There are four main sections in this chapter:

1. *Trading essentials* – in this section we will introduce basic definitions and formulae used in the world of trading. You will learn what markets are, and how they work in general, how prices are generated, what ask and bid prices are, what are market indicators and signals and what mathematics they are based on. Also, we will learn how to create candlestick charts and how to read them. Furthermore, a brief introduction to the Fourier Transformation will be given. Last but not least, we will describe one of the most important indicators called Relative Strength Index, or simply RSI.

2. *Cryptocurrencies* – this section will guide you through the cryptocurrencies world and what principles and philosophy they are based on. Moreover, you will learn about two main cryptocurrencies, Bitcoin and Ethereum, what they are and what differences they have.

3. *Deep learning* – in this work we use a convolutional neural network, therefore, you will find here what a perceptron is, how a simple neural network is built, what are differences between a simple neural network and a convolutional one and how all of them are connected to the world of deep learning.

4. *Technologies* – in the last section, we will describe all technologies and libraries that were used to create a dataset and build a model based on neural networks.

1.1 Trading essentials

Let us talk about money. Markets in their direct meaning have been surrounding humanity for thousands of years. First money got in use even before the beginning of written history [5] and this fact is not surprising; people needed some universal thing to exchange the products they had for the products they needed. Barter was not a convenient way since the fact that someone wants to buy meat for carrots does not mean that a person who sells meat needs carrots. That is how money was invented, a universal means for exchanging goods between people.

As markets developed in different countries, many different currencies came to life and since every money had its value based on the economic power of a state. First international trades are recorded from 19th century BC [6]; therefore, there should have already existed a way to exchange one currency for another 40 centuries ago. Today, when there are 180 different fiat [1] currencies in the world, the foreign currency exchange markets, or Forex, became large with an annual turnover averaged $5.1 trillion.

We have already illustrated that traders always exchange one asset for another. On Forex, these assets are world fiat currencies. If a trader has the British Pound and wants to buy the US Dollar, he will open order on an exchange and wait until someone who has US Dollars and is ready to sell them for British Pounds fills the order. Therefore, traders work with *currency pairs* meaning that they want to exchange one asset for the other. Every currency has its *code*, or *symbol* – a short name uniquely identifying the currency. For example, the US Dollar has a code USD; the British Pound is GBP, Euro is EUR. Exchanges list many currency pairs that are available for trading. These pairs are noted as a concatenation of currency codes. For instance, a pair consisting of Euro and Dollar is noted EURUSD, or sometimes EUR/USD. Here, EUR is *the base currency*, while USD is *the quote currency*.

All market participants play one of two roles, either a buyer or a seller. Buyers generate a demand for an asset, while sellers create an offer. Those, who want to get assets, offer their price for which they are ready to buy it; this price is called *a bid price*. On the other hand, those who own the asset offer the amount for which they are ready to sell it; their price is called *an ask price*. The difference between these two prices is called *a spread* and it is always greater than zero since buyers offer lower prices and sellers give higher amounts. This confrontation between these two roles generate the current rate of an asset, and it never stays the same, since the number of sellers and buyers is always variable. Trading rules are simple – everyone wants to get higher profits, or, in other words, wants to buy for a lower price and sell for higher.

[1] *Fiat currency* is a term used to distinguish currencies that we are used to (like the US Dollar or Euro) from cryptocurrencies.

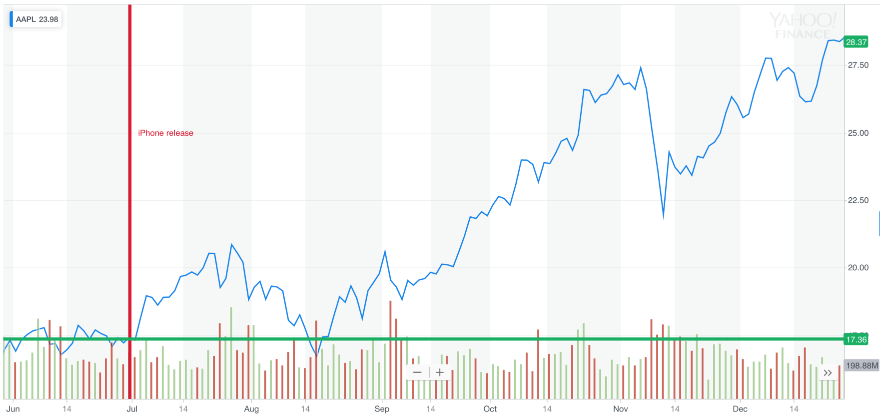


Figure 1.1: AAPL price chart from June 1, 2007 to January 1, 2008 [7]

A trader is a person who buys and sells an asset at the correct time. It means that being a successful trader requires an ability to look into the future and predict forthcoming trends of the price. The requirement sounds simple but is still very difficult to realise since markets do not follow simple, predictable rules, as mentioned in the introduction to this work. A trader should perform an analysis of a current situation on the market to correctly decide whether to buy or sell an asset at a given moment. There are two main types of analysis:

- *The fundamental analysis* includes the analysis of business assets, the political situation, the mood of the market in respect of buying the asset, business statements, annual reports etc. It brings us deep knowledge of what is happening on the market and gives us an ability to make decisions about the behaviour of the price of a given asset behaves shortly. Let me show an example of the fundamental analysis. The first iPhone was released on June 29, 2007. During next six month, the Apple Inc. stock price gained 63%, therefore, if we had bought 100 Apple's stocks on June 29, 2007, for $1 736 and sold them on January 1, 2008, for $2 837, the profit would have been $1 101. Figure 1.1 shows the line chart of the price.

- *The technical analysis* is different. It does not take into account news or official reports; the only source of knowledge for the technical analysis is historical data of the price itself.

In this thesis, we work only with a small part of technical analysis using a slightly modified market indicator that extracts valuable knowledge about trends of the price.

1.1.1 Technical analysis

Investopedia gave an excellent definition of technical analysis: *Technical analysis is a trading discipline employed to evaluate investments and identify trading opportunities by analysing mathematical trends gathered from trading activity, such as price movement and volume.* [8]. From the definition, it is clear that traders who use technical analysis for their trading activities work with historical data of the price. In other words, they use raw numbers and use several techniques to extract valuable knowledge from the past to use it for predicting the future. This type of analysis is not new. First outlines were created and published in 1688 by a Spanish merchant Joseph de la Vega [9]. Since then, technical analysis developed much and became an essential instrument for traders.

We already know about the main building blocks of a price – an offer and a demand. Also, we learned that the price was very dynamic and was constantly changing over time. An exchange, a platform which serves as an intermediary between traders, always records the cost of assets and forms the historical data which is later processed and analysed by traders. Usually, the information is not free and, moreover, quite expensive. However, the data from cryptocurrency exchanges are often easy to get since the market is not centralised and open to the public. A dataset consists of records of *candles* that will be discussed later in this section. After traders collect the data, they apply mathematical functions on price time series to transform the original data into a new space that is intended to indicate visually market trends. These mathematical functions are called *market indicators*. We will discuss one of such indicators called Relative Strength Index (RSI) later in this section.

Indicators are not meant to tell traders whether they should buy or sell an asset; indicators are only a transformation from one space to another. Previously, we saw that a trader needed to react to market changes on time to be able to generate profits. Markets can be either bullish or bearish. *Bulls* and *bears* are common terms in the trading world. Bulls always attack with their horns by bringing them upward. Bears, on the other hand, strike with their paws by swiping them downward. So, when prices are going down, we call it a bearish market. On the contrary, when prices are rising, the market is bullish.

A trader uses knowledge from one or more indicators to decide whether he should buy or sell an asset. An algorithm that predicts where the price will go next is called *a market signal*. Signals are used for the automated trading and trading bots, computer programs that perform market operations automatically relying on the information they get from one or more trading signals.

(a) A line plot (b) A candlestick chart

Figure 1.2: This figure compares two types of price charts. It can be noticed that although the charts look almost identical, the candlestick chart provides more information about the behaviour of the market over a time frame.

1.1.1.1 Candlestick charts

The price of assets is constantly changing; therefore, every moment, the data are being supplemented with new values. The series of numbers can be plotted in a straightforward manner using the line plot. However, these plots are not informative since traders need to have an overlook on the overall trends of the market. It is hard to interpret, for example, a one year trend using the line plot. Thus, there is a need to compress the data and extract meaningful information from a cut version of a dataset without losing the general information about price fluctuations. There are many ways how to transform the data, but the absolute standard of the price visualisation is *the candlestick chart.* Figure 1.2

Instead of connecting the dots with a straight line, traders split the dataset into pieces with a fixed length, *a time frame.* For every piece, several values are calculated:

1. *Open*: the first value met in the slice, or *the opening price.*

2. *High*: the maximum value met in the slice.

3. *Low*: the minimum value met in the slice.

4. *Close*: the last value met in the slice, or *the close price.*

These four numbers form *a candle.* Candles for which the closing price is higher than opening are drawn in white or green since the value increased on a time frame. On the other hand, if the close is less than open, the candle is red or black, since the value decreased on a given time frame. Figure 1.3 demonstrates a way how candles are drawn and read.

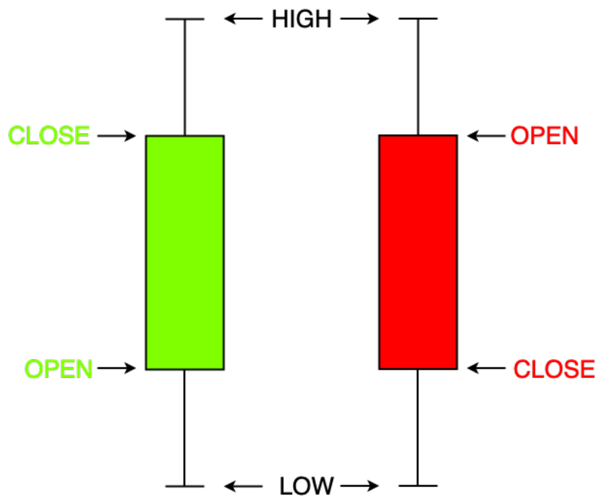


Figure 1.3: The values of Open, High, Low, Close are drawn as a candle.

The Open, High, Low, Close values are often provided with *the Volume*, the total amount of transactions observed over a specified time frame. These five numbers are called OHLCV (open, high, low, close, volume). The datasets from exchanges are often distributed as the sequence of candles over a specific time frame. Time frames that are commonly used are one minute, five minutes, 30 minutes, one hour, one day, one week, etc. It is worth mentioning that candles over broader time frames can be easily computed from candles over narrower ones using the same algorithm as for raw data.

1.1.1.2 Relative Strength Index

An RSI is a popular indicator developed for technical analysis by J. Welles Wilder Jr. in 1978. It is widely used because of its simplicity in interpretation. In this section, I refer to an original work [10] to explain the formula of RSI and discuss its meaning.

The Relative Strength Index, RSI, is a momentum indicator that measures the velocity of price movements. RSI is an oscillator, that is, its values fall into a band from 0 to 100 and oscillate around 50. When a value of RSI exceeds 70, the market is said to be in an overbought state; in other words, a forthcoming downtrend should be expected. On the other hand, when its value falls below 30, the market is in an oversold state; therefore, an uptrend is approaching.

Let t be a timeframe length used for RSI calculations. The original work suggests using 14-day timeframe. Calculations are based on the close prices for a given timeframe. The algorithm for calculating RSI is the following:

1. Calculate the average UP close, U_1^t, and the average DOWN close, D_1^t,

using the formulae:

$$U_1^t = \frac{\sum_{i \in \widehat{t-1}} max(0, close_{i+1} - close_i)}{t}$$

$$D_1^t = \frac{\sum_{i \in \widehat{t-1}} -min(0, close_{i+1} - close_i)}{t}$$

2. Calculate the first RSI value, RSI_1 using the formula:

$$RSI_1 = 100 - \frac{100}{1 + RS_1};$$

$$RS_1 = \frac{U_1^t}{D_1^t},$$

where RS is the Relative Strength.

3. To compute next RSI values, obtain the next average UP close, U_n^t, and the next average DOWN close, D_n^t, using the formulae:

$$U_n^t = \frac{(t-1) * U_{n-1}^t + max(0, close_{n+t} - close_{n+t-1})}{t};$$

$$D_n^t = \frac{(t-1) * D_{n-1}^t + max(0, close_{n+t-1} - close_{n+t})}{t}$$

4. Calculate next RSI values, RSI_n, using the same formula as above:

$$RSI_n = 100 - \frac{100}{1 + \frac{U_n^t}{D_n^t}}$$

By applying these formulae recursively for all price values in the dataset, we get RSI series. RSI values are plotted independently from a candlestick chart, and a line chart is usually placed under the candlestick chart. Also, a mentioned before band from 30 to 70 is plotted to indicate *failure swings* visually, the indicators of market reversals when the RSI value does not fall into the range. Figure 1.4 shows an example of an RSI chart with several hand-drawn notes of how traders use the visual patterns created by RSI values to decide whether to open order or not.

The Relative Strength Index is widely used with an originally suggested 14-day timeframe; however, 9-day and 25-day are also used.

Figure 1.4: An RSI chart with a 14-day timeframe drawn under a candlestick chart. Since the Relative Strength Index indicates a magnitude of a price change, strong ascending trends indicate that bulls are gaining confidence, and the price is going up. On the other hand, a descending trend means that bears are dominating over bulls, and the value of an asset is decreasing. When RSI intersects the upper edge of the $\langle 30, 70 \rangle$ band, we are talking about an overbought market, and it is an indicator of a forthcoming market reversal. The same rule is valid for the bottom edge; an oversold market indicates that bulls will prevail over bears soon.

1.1.1.3 Filters

Filtering is a set of techniques used in signal processing for removing unwanted features and noises from the input signal. Since price movements can be viewed as a signal, filters are widely used in technical analysis.

The Fourier transform described later in section 1.1.1.4 is only one example from many trading filters that are used for real trading. RSI, also, can be used as a filter depending on the interpretation of the results.

This section addresses a filter called the *exponential moving average* [11], EMA, also known as the *exponential weighted moving average*, EWMA, that is used in this thesis for smoothing the RSI signal before decomposing it into frequencies using the discrete Fourier transform. The exponential moving average is an extension to the simple moving average, filter. *Simple moving average*, SMA, is a moving window that calculates the means of price ticks with a fixed time frame.

Let the series of N close values from a OHLCV dataset be $\mathbf{x} = (x_n)_{n=0}^{N-1}$ and $MA^t = (ma_k^t)_{k=0}^{N-t}$ is the moving average of the price for a timeframe that equals t. The following equation applies:

$$ma_k^t = \frac{1}{t} \sum_{p=k}^{t+k-1} x_p$$

Exponential moving average also applies weighting factors that are exponentially changing. In technical analysis EMA is used to place a greater weight and significance on the most recent data points. Let the series of N close values from a OHLCV dataset be $\mathbf{x} = (x_n)_{n=0}^{N-1}$, $SMA^t = (sma_k^t)_{k=0}^{N-t}$ is a simple moving average and $EMA^t = (ema_k^t)_{k=0}^{N-t}$ is the exponential weighted moving average of the price for a timeframe that equals t. The following recursive equation applies:

$$ema_k^t = \begin{cases} sma_0^t, & k = 0 \\ x_{k+t-1} \cdot \beta + ema_{k-1} \cdot (1 - \beta), & k > 0 \end{cases}$$

where β is a weighting factor calculated by a simple equation:

$$\beta = \frac{2}{t+1}$$

SMA and EMA are essential filters used in trading. Firstly, both indicate a current trend. Secondly, different time frames represent *support and resistance levels* – the value tends to "bounce off" these lines. Figure 1.5 demonstrates an example of support levels.

The EMA has one crucial advantage over SMA; since weights of the recent values are higher, it is more sensitive to price movements; thus, reacts faster to market changes.

1.1.1.4 Fourier Transform

In this section, we will describe briefly what Fourier Transform is, how it works and, most importantly, how it can be used for trading activities. Here, we mostly refer to [12] and [13] to give a brief introduction to the Fourier analysis and discuss its use for trading activities.

The Fourier transform is widely used in audio signal processing since it decomposes the original signal into a sum of sines and cosines, or, in other words, the frequencies that make an audio sound as it sounds. This technique is useful for detecting and removing noise frequencies from the original signal.

OHLCV time series can be interpreted in the same way as for audio. We can apply the Discrete Fourier Transform on the price data and observe what frequencies compose the price signal. Furthermore, we can suppose that these frequencies will not change in a short period and compute several price values from the future.

The Fourier expansion is a mathematical transformation of a mathematical function or series; the function should be reasonably well-behaved – this

Figure 1.5: The candlestick chart of the ETH/BTC pair. SMA with a 14 hours timeframe is drawn in white. Notice how SMA indicates the support and resistance levels. For simplicity, an approximation of these levels are drawn in green and red, respectively.

definition, however, is out of the scope of this work. There are two types of Fourier expansion:

- *Fourier series* – a reasonably well-behaved *periodic* function can be written as a *discrete sum* of trigonometric or exponential functions.

- *Fourier transform* – a reasonably well-behaved function that is not periodic can be written as a *continuous integral* of trigonometric or exponential functions.

Since we work with markets that behave chaotically, the further explanation will be about the Fourier transform only. Furthermore, since the price data is not continuous and the candlestick datasets include the series of OHLCV values which are discrete, a discrete variant of Fourier transform will be discussed that is called Discrete Fourier Transform, DFT.

Here, we denote the series of N close values from a OHLCV dataset as $\mathbf{x} = (x_n)_{n=0}^{N-1}$. The Discrete Fourier Transform is then a vector $\tilde{\mathbf{x}}$ defined as:

$$\tilde{\mathbf{x}} = (\tilde{x}_k)_{k=0}^{N-1},$$

where $\tilde{x}_k$ is:

$$\tilde{x}_k = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} x_n e^{\frac{-2\pi ikn}{N}}$$

$\tilde{x}_k$ are complex numbers and represent discrete Fourier coefficients. The absolute value of $\tilde{x}_k$ represent the *amplitude*, or, in other words, the contribution amount of a frequency $\frac{k}{N} f_{sample}$ – where f_{sample} is a *sample rate*, a number of samples taken per a time period – to the overall signal. The amplitude is a great way to filter out those frequencies that do not contribute much to the output signal, thus, produce the noise.

The other relevant information that can be extracted from the coefficients is *a phase*. The phase is the angle from the positive real axis to the complex vector. The meaning of the phase is simple: it represents a shift of the sinusoid of a given frequency.

In this work, we use DFT for filtering and prediction purposes. After we decompose a signal into frequencies forming it, we select only the most important ones and compose a new signal using only these frequencies but for an extended number of samples, in other words, extending the length of the time axis and putting these frequencies on it. To compose an output, we need an inverse function of DFT. This function is called *the inverse discrete Fourier transform* and the equation applying is almost the same as for DFT:

$$x_n = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} \tilde{x}_k e^{\frac{2\pi ikn}{N}}$$

This way we can compose a smoothed version of a signal and use its plot as an input for a neural network. Figure 1.6 shows an example of the smoothed version of RSI with 24 predicted augmented values at the end.

1.2 Cryptocurrencies

Cryptocurrencies are a new thing in economics. They were born in November 2008 when a white paper called „Bitcoin: A Peer-to-Peer Electronic Cash System“ [14] was published to the public by an unknown author with a fictitious name – Satoshi Nakamoto. On January 2009 the first block on a Bitcoin blockchain was mined [2]. Many people claimed that they were Satoshi, but real authors of the whitepaper are still under a veil of mystery. Since 2009 Bitcoin gained at a price almost \$6000 in price at the time of writing and continues gaining popularity and trust all around the world. Moreover, today, there are already more than 1400 different cryptocurrencies, and this number keeps growing.

So, what are cryptocurrencies, and how are they used? Why do many crypto enthusiasts claim that cryptocurrencies are the future of payments and

[2] If you feel uncomfortable with these terms, they will be explained in details later on.

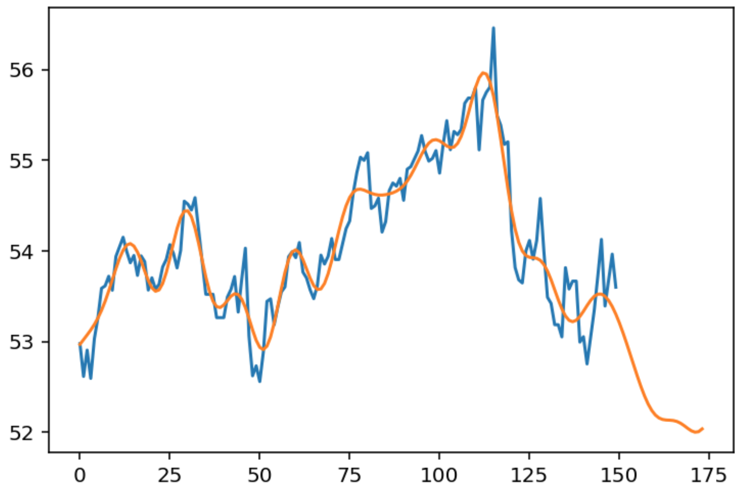


Figure 1.6: The line plot showing the original RSI in blue and a filtered version of it in orange. Filtering was performed by selecting 700 main frequencies using the Discrete Fourier Transform. Notice that the orange line continues further on the plot. These augmented values are predicted by prolonging the time axis and applying the inverse DFT. The data used for RSI calculations is the close prices of 1-day candlesticks of the cryptocurrency pair ETH/BTC listed on the GDAX exchange.

that they will replace traditional currencies that we use in our everyday lives? The answer is only one word – decentralisation.

1.2.1 Bitcoin

Bitcoin, a father of all cryptocurrencies and the first most popular cryptocurrency in the world with a record market capitalisation more than 111 billions US Dollars, is claimed as a currency that is not controlled by authorities like banks or government. There is no need to link digital wallets to a real identity; everyone can buy, sell and use Bitcoin for payments anonymously[3]. And that is the reason why Bitcoin became a number one payment method on the Dark Web. However, there are also many advantages for users who are not related to the dark corners of the World's economics or politics. With Bitcoin, a user will not pay additional fees for international money transfers as there is no such concept like country borders, Bitcoin stays an entirely

[3]Bitcoin is not meant to be completely anonymous, there are many ways to link Bitcoin transactions to real people and reveal a real identity of a Bitcoin wallet holder. If you are interested in entirely anonymous cryptocurrencies, you should consider yourself using Monero or Dash.

digital currency that is not connected to anything real. Money transfers take, on average, 20 minutes, which is much faster than money transfers between different banks. Furthermore, for a successful transfer, you should have only three things: a Bitcoin wallet, Bitcoins themselves and a wallet address of your recipient.

As mentioned above, Bitcoin is not connected to anything real in the world; it exists on a shared distributed public ledger, *a block chain*, that holds unencrypted date about all transactions with Bitcoin addresses of a sender and a recipient. However, there is no information how many Bitcoins these addresses hold; it can be only computed by connecting all transaction chains and based on the income and outcome values for the address the final available amount for spending gets known. Bitcoin is a peer-to-peer (P2P) network; therefore, a full copy of the blockchain is stored on every client node that runs Bitcoin client software.

A Bitcoin *wallet* is a name for an asymmetric cryptographic key pair, a public key and a private key. Public keys can be shared publicly and are used to receive money. Private keys, also known as *spend addresses*, are used for creating and signing transactions that are later sent to the blockchain, or, in other words, owning a private allows a user to spend Bitcoins.

As mentioned earlier, a full copy of the blockchain is stored on every machine that runs the Bitcoin client software. Since the cryptocurrency is a P2P network, it means that there should exist as many nodes as possible to keep the network alive and as much decentralised as possible. Thus, people who run these nodes should have the motivation to contribute to the blockchain. That is the reason why *mining* exists. Mining serves two essential purposes. By mining, nodes confirm transactions on the blockchain and include them in *blocks*. The first mining node that finds a correct hash for a block takes the sum of transaction fees that are included in the block. The fee amount is set manually when a user sends Bitcoin from one wallet to another; this fee is called *a mining fee*. Furthermore, mining is the only way to release further portions of the cryptocurrency into circulation; in other words, miners are minting new coins and put them in use. This consensus algorithm[4] that is used in the Bitcoin core is called *Proof of Work*, or PoW, as miners get paid for computations.

Bitcoin is listed on all cryptocurrency exchanges under codes BTC or XBT.

1.2.2 Ethereum

Ethereum is the second largest and most popular cryptocurrency in the world with a market capitalisation more than 18 billions US Dollars. If Bitcoin serves as a digital currency, Ethereum represents a decentralised platform which offers a technology called *smart contracts*. Smart contracts are like all

[4]Consensus algorithms are a collective name for a family of algorithms that are used for verifying the validity of transactions coming to a blockchain.

contracts we are used to seeing in the real world. For instance, if you want to buy a house, you enter into a contract with a party owning it. You do not want to transfer money before you get all required documents ready and you have all rights to the house well declared. Smart contracts work like small applications which offer a way to define such conditions programmatically without further possibilities of fraud or third-party interference.

Ethereum's blockchain was launched in 2015. First work was suggested in 2013 by Vitalik Buterin, the creator of Ethereum, and later supplemented and extended by Dr Gavin Wood in 2014 in a yellow paper called Ethereum: A Secure Decentralised Generalised Transaction Ledger [15].

Ethereum uses a different consensus algorithm for mining called *Proof of Stake*, or PoS. Here, the probability of validating the next block is higher for miners that hold more significant stakes, or, in other words, is determined by the number of coins a miner has. Bitcoin rewards all nodes with the sum of all fees for transactions included in a block, Ethereum, on the other hand, rewards miners with the sum of all network fees for transactions they have verified. These rewards in Ethereum are called *gas*. Because miners collect all fees from transactions and do not solve computationally difficult block hashes, the average time that is needed for a confirmation on the blockchain is much smaller.

Ethereum is listed on all cryptocurrency exchanges under a code ETH.

1.2.3 Cryptocurrency trading

Both cryptocurrencies described above have been leading the market for several years. We have chosen these two assets due to several reasons:

- They are considered the most influencing cryptocurrencies in the world since Bitcoin is a progenitor of all cryptocurrencies and Ethereum is the base currency for many other assets because it is meant to be a platform and not a digital coin.

- Due to the large capitalisation of these currencies, their price is not so volatile[5] than for smaller cryptocurrencies. That is very useful for detecting patterns in the price data that are statistically strong enough.

- Both of them have a long history of price records of their price. This fact is crucial for building a large enough dataset for a neural network.

- Every cryptocurrency exchange has the ETH/BTC pair available for trading. Since the price is not the same on different exchange platforms, it gives an ability to extend an overall dataset.

[5] A volatility measures a percentage change of a price during some period. A higher volatility means higher fluctuations in the price and higher investment risks.

An overall trading process for cryptocurrencies stays the same as for any other asset. The only difference that most cryptocurrencies are not supported by real commodities; therefore, their price is based only on investments volumes and the overall offer/demand levels. It leads to much bigger volatility levels than for fiat currencies like US Dollar or Euro and, therefore, higher and much quicker profits.

1.3 Deep Learning

The term *deep learning* appeared between scientists in 1986 after a work of Rina Dechter „Learning while searching in constraint-satisfaction problems“ [16]. During the last 20 years, deep learning became a popular and widely used concept due to the development of faster multi-core processors and efficient parallel algorithms.

Deep learning is a class of machine learning algorithms that uses a set of non-linear transformations of input data for the feature extraction. *Feature extraction* is the process of dimensionality reduction of the data that is used to extract essential features or classes from high-dimensionality inputs. Deep learning is called deep because the number of mathematical computations is much higher than for usual machine learning algorithms and requires many computational resources to teach a model. The term often refers to *neural networks* with several layers where the computation of *a gradient* of a mathematical equation is complex.

As an example, a famous „Iris dataset“ is a collection of measurements of iris flowers where features like sepal length and petal width – four in total – are collected. Each flower has a type – a class – to which the flower belongs. The dataset has three categories: Iris Setosa, Iris Versicolour and Iris Virginica. To accomplish the classification based on the input characteristics of flowers, we *teach* a model – which is a mathematical function – to reduce four dimensions into only one and, therefore, to get one class label as an output.

The provided example is a case of *supervised learning.* In supervised learning, data comes with a set of features and the output value that corresponds to the input. Technically speaking, we have a set of *precedents* – $(object, answer)$ pairs – and we suppose that an unknown correlation between them exists. For instance, the described above Iris dataset has flowers measurements as inputs and a flower kind as an output.

In general, we can divide the problems of supervised learning into two groups: *regression* problems and *classification* problems. In classification, the set of numbers is finite and usually represent class labels. Regression, on the other hand, works with answers represented by real numbers or vectors. The above example is a classic classification problem.

Unsupervised learning is a discipline that does not incorporate working with outputs. The goal of unsupervised learning algorithms is to find depen-

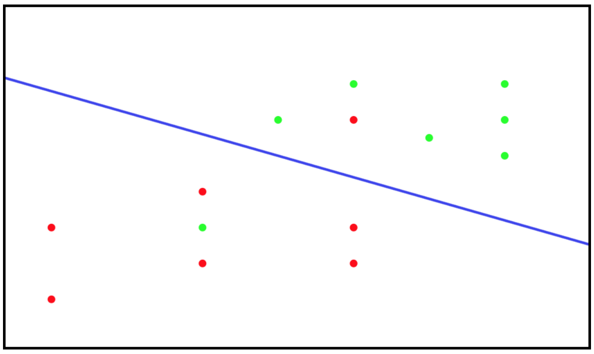


Figure 1.7: A plot of the 2-D input values. Red dots represent a class 0, green dots are of a class 1. Notice that the values are not perfectly separable due to the intersection of their clusters.

dencies and patterns in the data that *may* exist, or, in other words, to describe a dataset. For example, *clustering* is one of such tasks. Here, the goal of an algorithm is to separate the data into several clusters, the number of which is unknown in advance. As an example, imagine a nuclear reactor that has hundreds of sensors. If something goes wrong, we should detect the anomaly quickly and take actions to prevent a disaster. Each sensor produces measurements of a part of the reactor, and each can deliver value within an allowed range while the system, in general, works unproperly. Therefore, we should take into consideration measurements from sensors at once. The discipline that deals with detecting anomalies is called *anomaly detection*.

In this section, we will describe the concept of a perceptron and artificial neural networks, ANN, and learn an extended version of ANN called convolutional neural networks, CNN, that are used for image classification.

1.3.1 A perceptron

A perceptron algorithm was invented by Frank Rosenblatt in 1958 [17] and is an essential building block of every neural network used nowadays.

A single perceptron is a binary classifier function, $f(\mathbf{x})$, that maps an real-valued input vector $\mathbf{x}$ to a binary output. Suppose we have a learning dataset of length n $D = (\mathbf{X}, \mathbf{y})$ where $\mathbf{x}_i$ is a vector of features of a sample and y_i is a class label, either 0 or 1. The following equation applies:

$$f(x) = \begin{cases} 1, & \text{if } \mathbf{w} \cdot \mathbf{x} + b > 0, \\ 0, & \text{otherwise} \end{cases}$$

where $\mathbf{w}$ is a vector of real-valued weights, b is a real number, a bias and $\mathbf{w} \cdot \mathbf{x}$ is a dot product of two vectors defined as $\mathbf{w} \cdot \mathbf{x} = \sum_{i=0}^{m} w_i x_i$. As you may have noticed, the function is linear, thus, the input data is assumed to be linearly separable. There is a way to overcome this limit it it will be described shortly.

Taking into consideration the definition, we can see that we need to adjust the vector of weights and the bias for correct classification. It should be noticed that the data may not be ideally separable. Figure 1.7 shows an example of a dataset with two-dimensional input for which a perceptron is good enough but does not give the 100% accuracy. The learning algorithm for a perceptron is provided in the algorithm 1 block.

Algorithm 1: Perceptron learning algorithm

Algorithm `train`(D, η)
    Input: dataset $D = (\mathbf{X}, \mathbf{y})$, a learning rate η
    Output: a vector of weights $\mathbf{w}$ where $\mathbf{w}_0$ is a bias
    set all weights of $\mathbf{w}$ to zero;
    for *every sample* $(\mathbf{x}_i, y_i)$ *in* D **do**
        // $\mathbf{x}_0$ represents a bias;
        append 1 to the beginning of $\mathbf{x}_i$;
        $p \leftarrow$ `predict`($\mathbf{x}_i$, $\mathbf{w}$);
        // update weights;
        if $p \neq y_i$ **then**
            if $y_i = 0$ **then**
                $\mathbf{w} = \mathbf{w} - \eta\mathbf{x}_i$;
            else
                $\mathbf{w} = \mathbf{w} + \eta\mathbf{x}_i$;
            end
        end
    end
    return $\mathbf{w}$;

Procedure `predict`($\mathbf{x}$, $\mathbf{w}$)
    $p \leftarrow \mathbf{w} \cdot \mathbf{x}_i$;
    if $p > 0$ **then**
        return 1;
    else
        return 0;
    end

Perceptrons are good not only for linearly separable data. A perceptron, as mentioned before, is a building block of neural networks; we can combine several perceptrons and train weights of each to get more flexible functions. For example, the XOR function produces values that cannot be separated linearly, and we need only four perceptrons to accomplish the task. Figure 1.8 illustrates the solution.

1.3.2 Artificial neural networks

The XOR problem solved by linear perceptrons brings us to the next concept, *an artificial neural network*, ANN. A neural network is a mathematical

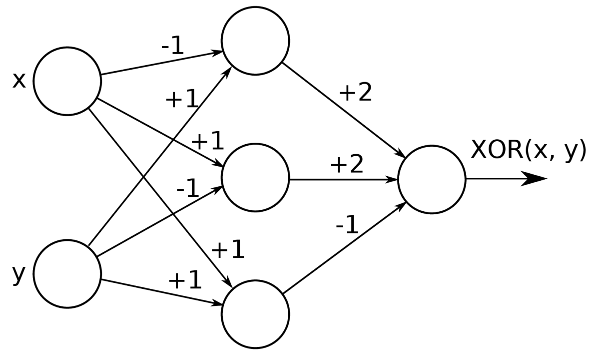


Figure 1.8: XOR is not a linear function, thus, a single perceptron cannot be used to successfully predict two classes. However, the combination of four linear perceptrons can accomplish the task.

model that is built in the image and likeness of a biological brain. Every brain has billions of neurons, simple computational units, that are interconnected with each other through synapses. A neuron gets electrical signals from other neurons through dendrites, and if the electric charge in a cell exceeds a threshold, an action potential occurs that is transmitted through an axon to other neurons. Of course, reality is much more complicated, but this simplified explanation demonstrates the basics that were borrowed from nature to create neural networks.

In this and the following section, we mostly refer to [18]. An artificial neural network is a system of connected and interacting between each other artificial neurons. As real neurons, artificial ones get responses from other neurons, apply a mathematical transformation to the input vector and output one real value that is transmitted to neurons following next in a chain. All these small computational units are grouped into *layers* that are lined up. The first layer of neurons is *the input layer*, which represents the input vector. The last one is *the output layer* that consists of one or more neurons and represents a predicted value or a class. All layers between input and output layers are called *hidden*. Figure 1.9 demonstrates a neural network with one hidden layer. It can be noticed that a neural network consists of several interconnected perceptrons and an overall picture of ANN looks the same as for the solution of the XOR problem. However, there is an important difference between a simple perceptron and that one used in ANNs. The matter is *an activation function*.

Activation functions play an essential role in the architecture of neural networks. Since a linear combination of linear functions is also a linear function, we need a way to get rid of such linearity to be able to fit any data. For these purposes, a linear combination of inputs to a neuron is transformed into a non-linear impulse using an activation function. The activation func-

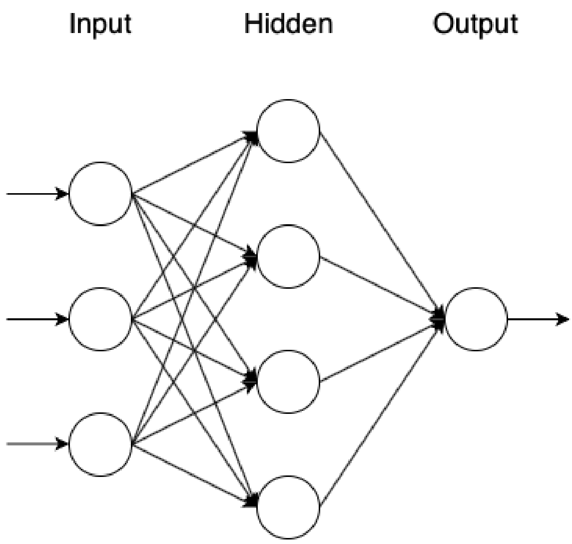


Figure 1.9: An fully-connected artificial neural network with three input neurons, one hidden layer with four nodes and one output.

tion should have a derivative to be used in neural networks since a gradient descend method is used for learning. This section describes three activation functions that are used in the thesis: a sigmoid, a rectified linear unit, and a softmax.

A sigmoid is a function defined as:

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

A sigmoid forms a curve of monotonically increasing values from 0 to 1 and intersects the y-axis in the 0.5 point. The derivative of a sigmoid is easily computed and equals to:

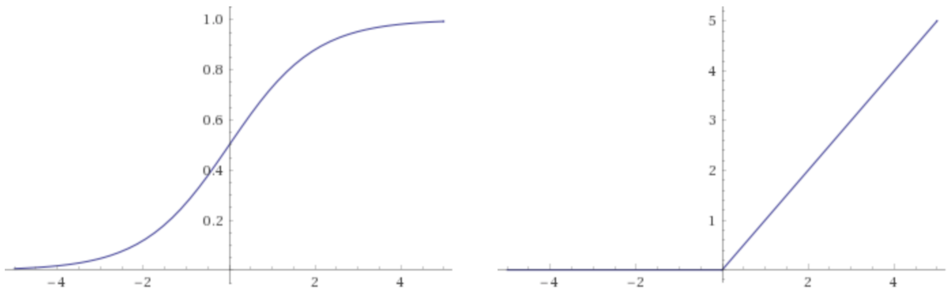
$$\sigma'(x) = \frac{e^{-x}}{(1 + e^{-x})^2}$$

The function is widely used in neural networks for transforming linear inputs. However, it has its drawbacks. For values that are far from 0, the sigmoid tends to converge to either 0 or 1 creating the vanishing gradient problem[6].

The other widely used function is a rectified linear unit, ReLU. In simple words, all this function does is just dropping negative values. The following equations applies:

$$f(x) = max(0, x)$$

[6] The vanishing gradient problem is the term used to describe a situation when a gradient of a loss function of a neural network tends to approach zero using certain activation functions making the network hard to train.



(a) The sigmoid function

(b) A rectified linear unit

Figure 1.10: This figure compares two activation functions used in this work.

$$f'(x) = \begin{cases} 1, & \text{if } x > 0, \\ 0, & \text{otherwise.} \end{cases}$$

The function avoids the vanishing gradient problem, but can result in a weight update that will make a neuron never activate on any data point. In other words, ReLU can cause a dead neuron.

The last widely popular function in ANNs used for output layers is a softmax. It transforms an input vector z into a vector of the same dimensionality where all coordinates are in the interval from 0 to 1, and the sum of coordinates is equal to 1. The softmax is used for classification tasks where every coordinate represents the probability of a class. The function is defined as:

$$\sigma(z)_i = \frac{e^{z_i}}{\sum_{k=0}^{K-1} e^{z_k}}$$

Now, we are ready to move to the next important concept, convolutional neural networks.

1.3.3 Convolutional neural networks

Convolutional neural networks is an extension to standard ANNs that finds its use in image processing and pattern recognition. A standard artificial neural network works with inputs that can be represented with a vector. However, images are not vectors and flattening matrices of pixels can result in losing the information about visual patterns appearing on an image. Therefore, we should have a way to pass an image to a neural network „as is". But, first of all, we should learn the basics of how images are represented in computer memory.

An image has its width and height measured in pixels. Thus, every image is represented by a matrix of *pixels*, the smallest units, each having one colour. Every image belongs to *a colour space*, a way to describe a pixel value. Standard colour space for most pictures is RGB, in which a pixel is

(a) RGB (b) Grayscale (c) A binary image

Figure 1.11: A photo of my best friend, Mykyta Boiko. The left picture in the RGB color space, the centeral is a converted version to grayscale. The last one is the binary image that was generated using a simple thresholding method with a threshold set to 90.

represented by three values, each from 0 to 255, and indicating the amount of the Red hue, Green and Blue in final colour. Therefore, in RGB, there are $255^3 = 16,581,375$ colours in total. However, sometimes there is no need to process color and all we need is the information about contours of depicted objects and geometrical patterns, colours, in this case, are redundant. Thus, we can transform three RGB values into one and get a grayscale image. *Grayscale* is another colour space in which each pixel consists of only one number from 0 to 255 – in other words, from black to white – that represents a shade of grey. The transformation that is used to convert RGB to grayscale is out of the scope of this work.

But we can go even further. Imagine that we want to find contours of objects on an image. A contour is a simple line; there is no need to draw it in colour or using the shades. The line exists or not. In this case, we talk about *binary images*. Pixels here can have only two values, either 0 or 1, true or false. One of many methods to generate a binary image from grayscale is a simple thresholding[7]. In this thesis, we use this specific type of images and reasons of this choice will be explained in 2.1.

Convolutional neural networks use a matrix operation called *a convolution* (hence the name). This operation involves defining *a convolutional kernel*, a much smaller in size matrix than an input matrix. Mathematically, the image convolution is defined as:

$$g(x,y) = \omega * f(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} \omega(s,t) f(x-s, y-t),$$

[7]Thresholding is a technique to convert pictures from grayscale to a binary colour space. Every pixel that has a value above a defined threshold becomes 1; others are set to 0.

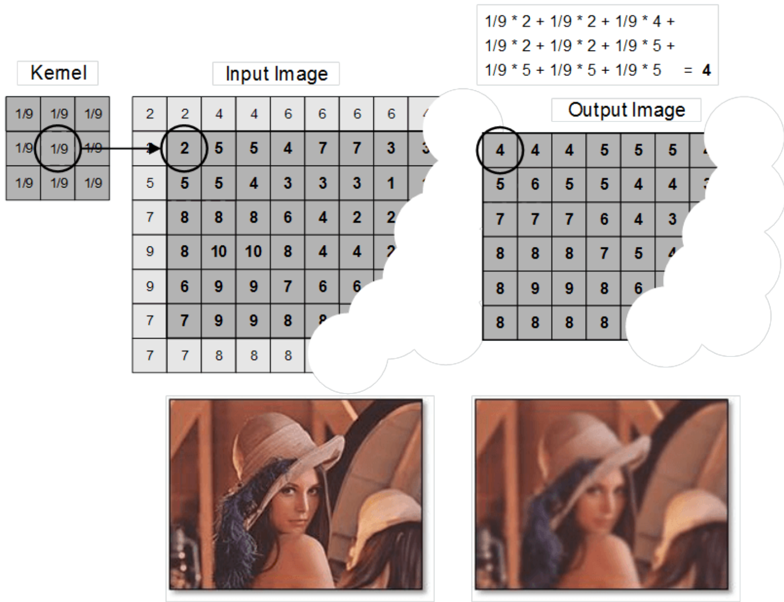


Figure 1.12: A convolution kernel is multiplied with a window from an input image of the same size element-wise, and all results are summed up getting a new value of a pixel. The kernel is moved along both image axes to get values for all pixels. To save input width and height, we add pixel rows around the image copying the original border pixels.

where $g(x, y)$ is the output image, $f(x, y)$ is the original image and ω is the convolutional kernel. Every element of the kernel is considered by $-a \leq s \leq a$ and $-b \leq t \leq b$ [19]. A great visual example of the image convolution taken from [20] is given in figure 1.12.

Now we are ready to meet convolutional neural networks. In a usual fully-connected ANN every neuron is connected to all neurons of a previous layer, and every connection has its weight. In a CNN, a convolution operation a weight matrix of limited size is moved over a layer that is being processed, forming an activation impulse for a neuron on the next layer at the same position. It is essential to understand that the same matrix is used for a whole layer; this matrix represents a convolutional kernel. While the kernel can be interpreted as a visual representation of the existence of some feature on an image, e.g. a line with a specific width, hidden layers show a presence of features on previous layers. So, hidden layers in convolutional neural networks are called *feature maps*. One kernel cannot represent all possible features. Thus, a set of kernels is used. These matrices are generated during a standard learning process with the backpropagation algorithm, and it makes one layer include several feature maps. The activation function that is widely used in convolutional models is ReLU.

Feature maps of each layer have lower dimensionality than maps from

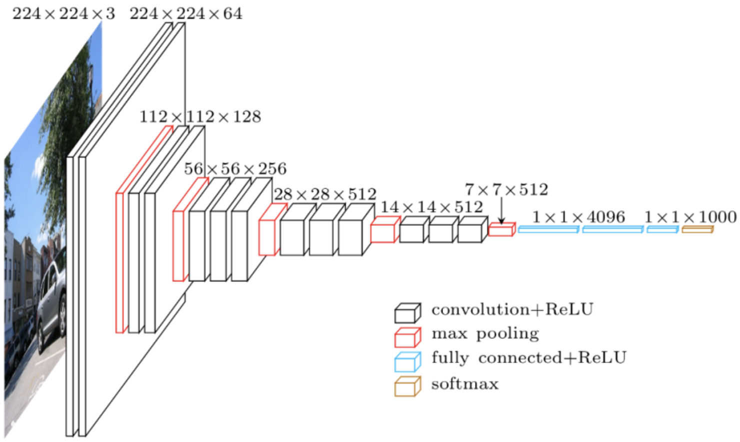


Figure 1.13: A example scheme how a convolutional neural network can look like.

a previous layer. It is achieved by a technique called *pooling*, or *sampling*. Sampling is a maximum or an average of several neighbouring neurons from a previous layer. *Maxpooling* is a short name for sampling with a $max()$ function.

After creating the last feature maps, we need to transform their output into class labels or a real number depending on a defined task. To achieve this, we *flatten* output feature maps and get a normal vector of numbers that can be passed to a fully-connected ANN. A great scheme of a convolutional neural network architecture is given in figure 1.13 taken from [21].

1.4 Used technologies

In this section, we will describe all the technologies and instruments that were used in this thesis. Firstly, we will introduce several great web resources with the help of which we could be able to create all supporting charts. Secondly, we will describe several Python libraries that allowed to integrate technical analysis into the ultimate RSIVision package. Last but not least, we will say a few words about the technological backend for creating a neural model described in detail in section 2.1.

Every trader works with charts of real-time data. It is vital to keep a working environment comfortable to work in since market players need to react to changes in assets' values quickly. Our personal choice for creating beautiful and meaningful plots of the price data and indicators is a web platform *TradingView.com* [22]. Almost all charts that can be found throughout this work are made on this web service. It provides a tremendous simple graphical interface and the ability to create charts for any asset. Besides, it supports most of the technical indicators, including the RSI.

For building the signal that we call RSIVision, we use a Python programming language due to its flexibility and many additional libraries available that increased the velocity of the research. In the next section, all used Python libraries are discussed.

1.4.1 Python libraries

As mentioned before, we chose Python [23] for the implementation of the signal. Python code is more readable than most of the other languages, and it is useful for creating prototypes. We use version 3.7 with several libraries that will be discussed in this section.

The first essential library for purposes of technical analysis has an amusing name *TA-Lib*, Technical Analysis Library [24]. It started in 1999 and became widely used by many applications. TA-Lib includes built-in functions for calculating indicators of any taste, and this is the main reason why we chose it – it allows us to extend our model in the future without a need to change anything in the code. Furthermore, the library is incredibly fast. It is initially written in C/C++, but there are a lot of wrappers for many programming languages, including Python [25] that we love to use.

The second deeply integrated library into RSIVision is *Pandas* [26]. It provides developers and data scientists with the essential, high-performance instruments that are a need for data analysis and data processing. In RSIVision, Pandas is used for parsing raw datasets from exchanges and processing the data before we use it as inputs to neural networks. Moreover, it has built-in functions for the use with OHLCV datasets which, in our case, saves us many additional lines of code.

When a dataset is parsed and is ready to be used for teaching a neural network, we need to build the model itself. For the purposes of this thesis we use *Keras* library with a *TensorFlow* backend. TensorFlow [27] is an open-source Machine Learning framework developed by Google. Its main idea is a representation of computational operations using data flow graphs. In other words, all mathematical operations are transformed to graphs where nodes are mathematical operations and edges are tensors[8]. TensorFlow has a lot of advantages over different frameworks like Theano or CNTK, but the biggest one is the ability to parallelise computations because of the independence of nodes in a data flow graph. Since neural networks' gradient is a derivative of many composite functions and since the chain rule of computing derivatives of composite functions applies, computations of neural network's weights are easily parallelised. Therefore, TensorFlow is a great tool for creating deep learning networks and efficiently teaching them.

[8]Tensors are used here as a term describing multidimensional data vectors.

We mentioned Keras [28]. This library is used as a front-end for TensorFlow (however, not only TensorFlow is supported) and contains many high-level functions that are often used as building blocks for neural networks. Keras has built-in implementation for neural networks layers, activation functions or loss functions. The library simplifies many deep learning activities and allows us to write more efficient programs with less code.

Chapter 2

Building a model

We the previous chapter we learned the basics of markets and technical analysis, we discovered convolutional neural networks, and now we are ready to combine gained knowledge to implement a new market signal that is called RSIVision. In this chapter, we propose a state-of-art method to create a learning dataset for a convolutional neural network and describe an architecture of the CNN itself. Here, we follow the Cross-Industry Standard Process for Data Mining [29],[30], CRISP-DM, that includes six steps:

1. *Business understanding*: determining business objectives, assess the situation, determine data mining goals, a product project plan.

2. *Data Understanding*: includes collecting data, describing it and verifying its quality.

3. *Data Preparation*: selecting data and cleaning it, formatting and generating a dataset.

4. *Modeling*: choosing modelling techniques, building a model.

5. *Evaluation*: evaluating results, reviewing the model and determining next steps.

6. *Deployment*: the creation of deployment and maintenance plans.

These steps are not fixed, CRISP-DM is instead a blueprint for organising a working process of Data Scientists. However, we will follow these steps and describe them.

The first step in CRISP-DM is business understanding. We have already discussed in the introduction to this thesis the key points why analysing markets is essential and can generate enormous profits. The generated signal can be used for both personal purposes and as a product for selling to other traders.

Unix Timestamp	Date	Symbol	Open	High	Low	Close	Volume
1530302400000	2018-06-29 20:00:00	ETHBTC	0.06947	0.0701	0.06941	0.0701	46.21962605
1530298800000	2018-06-29 19:00:00	ETHBTC	0.06976	0.06976	0.06934	0.06947	13.802563
1530295200000	2018-06-29 18:00:00	ETHBTC	0.06992	0.06992	0.06976	0.06976	18.31447
1530291600000	2018-06-29 17:00:00	ETHBTC	0.06992	0.06993	0.06992	0.06992	16.12607758
1530288000000	2018-06-29 16:00:00	ETHBTC	0.07011	0.07013	0.06977	0.06992	21.579165

Table 2.1: Five OHLCV rows from a collected dataset from the Binance exchange.

This chapter goes through steps from 2 to 4, that is, data understanding and preparation and modelling, and next two sections are devoted to evaluating the model and finding optimal internal parameters to achieve the best accuracy and the deployment of a final model.

2.1 Data preprocessing

To generate a dataset that can be used for teaching a neural network, we need much data. Moreover, the dataset for using to predict markets prices of assets should be quiet big due to the variability and randomness of the market behaviour. Finally, when we talk about cryptocurrencies, the difficulty of correct predictions significantly increases because no one controls these markets; thus, they are very volatile and prone to manipulations from big players.

Fortunately, since cryptocurrency exchanges are more open to the public than traditional fiat ones, most exchanges have Application Programming Interfaces, API, for downloading historical data. For this work, we chose Binance as a data provider. Binance is a global cryptocurrency exchange platform that has many advantages like no commission for neither withdrawals nor deposits and a simple registration process with higher operational limits without the ID verification. Besides, Binance has a RESTful – Representational State Transfer, REST – web service [31] that allows downloading historical data of any cryptocurrency pair for any period. We chose an ETH/BTC pair candlestick dataset with a one-hour time frame. A downloaded dataset consists of more than 8,000 OHLCV values from 2017 to 2019. Table 2.1 illustrates how a final dataset looks like.

The most important part of the thesis is how this raw dataset is processed. The following steps apply:

1. Calculate the RSI for a raw dataset using Close prices and a time frame t^{RSI}.

2. Calculate a filtered version of the RSI, RSI^{EWMA}, using the EWMA filter with a time frame t^{EWMA}.

3. Apply the DFT on RSI^{EWMA} and extract only h harmonics (frequencies) to get RSI^{DFT}.

4. Move a sliding window with length t^w on RSI^{DFT} and create a vector of RSI^{DFT} slices, W.

5. From every w_i of W extract the following additional features:

   - The area between the curve and a line connecting its first and last points, $area_i$.
   - A flag c_i if the line intersects 50.0, the value around which the RSI oscillates.
   - A tangent of the line, tg_i.

6. Detrend every w_i sample to keep only the shape and patterns and create a binary plot from the detrended version, P_i.

7. Compute a percentage difference between the close price on the start of a sliding window and the end, $diff_i$.

8. Set an output label o_i to BUY if $diff_i \leq -act_{thr}$, $SELL$ if $act_{thr} \leq diff_i$, and $HOLD$ otherwise. act_{thr} is a threshold for an action; a low percentage difference indicates that opening an order can result in losing money since a trader pays for every order a small fee to an exchange platform.

9. Create a final dataset $DS = (\{P_i, w_i, area_i, c_i, tg_i\}, o_i)$.

We should describe and explain the steps in this algorithm. Firstly, we smooth the RSI using the exponential moving average. This step is essential since it filters out random fluctuations and the noise that we do not need in patterns plots. Then we use the DFT to filter out the noise frequencies and leave only smooth curves. Next, we extract additional features to extend the search space a neural network gets as inputs. This step is not necessary; however, it can dramatically increase the overall precision of predictions. In Data Science, this step is called *a feature extraction.* When we have the RSI prepared, we detrend[9] the data to narrow the search space for a neural network and concentrate its attention only on a curve shape and not its trend. After getting clean patterns, we create line plots of this data and store plots as binary images. Using them here has the same purpose as with detrending – narrowing the search space. Patterns can be represented with a solid line, like sketches, and there is no need to forward additional information to a neural network like shadows of the line since the plots are generated automatically.

The dataset contains sets of extracted features, plotted patterns, raw RSI values and outputs represented by 3 class labels: SELL (or SHORT),

[9] Detrending here is a subtracting a linear approximation of a curve from the curve itself.

HOLD (or FLAT), BUY (or LONG). An output label represents an action for a signal that a trader should take. Words in round brackets are used here as synonyms for SELL/HOLD/BUY actions. The overall sequence of steps is not arbitrary; the algorithm is based on different trading techniques that are, however, out of the scope of this work.

A created dataset DS can be passed to a neural network model that will be described in the next section.

2.2 Modeling

We propose a neural network with the following structure:

1. The first neural network is a CNN that takes as inputs binary images. They pass several layers of feature maps where the dimensionality is reduced using max-pooling. The last layer of feature maps is flattened into a vector. This vector is the first input to a predictor neural network.

2. The second neural network is an appendix to the first and contains an additional features processor that we manually extracted. This network is small since the number of inputs is limited. Its structure follows the fully-connected design pattern. The last layer generates a vector that is the second input to a predictor neural network.

3. The third network is a network that proceeds raw indicator data and extracts important hidden features from it. This network is also small enough since a sliding window size is not a big number. It is also a fully-connected ANN. A vector of values from the last layer is the third input to a predictor neural network.

4. A predictor neural network, the one that generates an output, is a fully-connected artificial neural network that takes as inputs the output nodes from three networks described above. The input vector to this ANN has high dimensionality since if a CNN that has M feature maps with a $(W \times H)$ shape flattening the last layer results in creating a vector with $W \cdot H \cdot M$ values.

The architecture of a neural network is based on two main ideas:

1. Patterns are descriptive, but they are not sufficient since the information about their location is not present on the line plots. Remember an RSI plot 1.4 with indications of market overbought/oversold states. A peak on a chart can arrear anywhere, but only peaks above 70 indicate that a market can become bearish soon. Therefore, values that form a pattern are as important as a pattern itself.

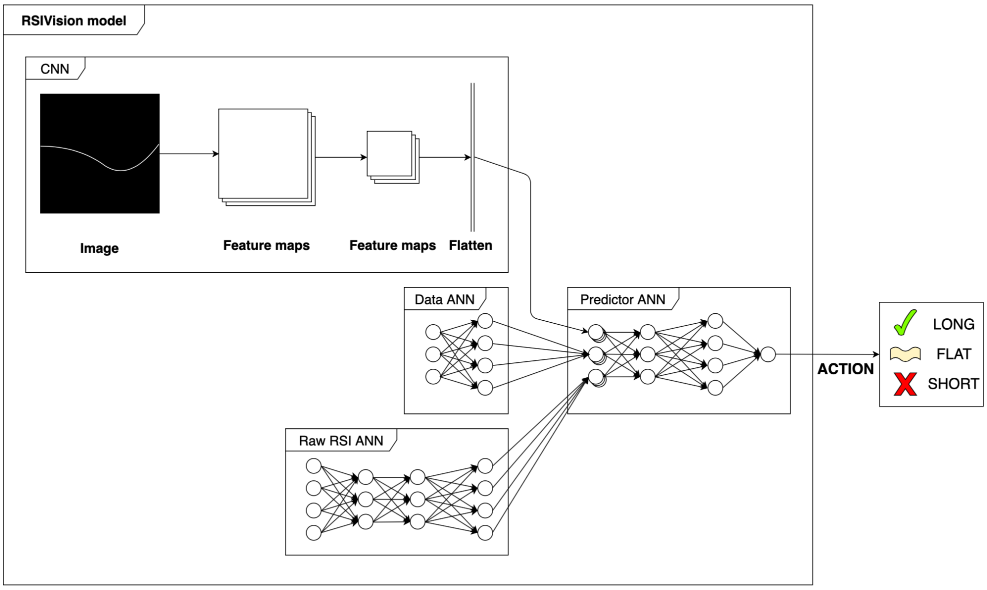


Figure 2.1: An architecture of neural networks of the RSIVision package. A dataset described in 2.1 is passed as inputs of ANNs.

2. An output from a neural network should include all three types of data: images, additional features and raw values. However, they should not interfere with each other since we deal with different types of data and different network structures for images and vectors.

Figure 2.1 visually represents the internal architecture of all four neural networks.

2.2.1 Predicting the future

A signal is useless if it cannot say in advance whether the price is going to increase or decrease shortly. We should be able to tell right away what colour the next candle is going to be. As you may remember, in section 1.1.1.4 we talked about prolonging the time axis to the future and putting the frequencies from DFT on it. That is the way used to generate the next sample in a dataset for generating a signal for the next candle.

To generate a signal for the next candle, we take the following steps:

1. Take the dataset $D = (s_i)_{i=0}^{N-1}$ and a neural model $f(s)$ already trained on it.

2. Using Fourier analysis, generate the next value of the RSI, RSI_N, without dropping any frequency.

3. Using Fourier analysis, generate the next value of a filtered version of the RSI, RSI_N^{EWMA}.

4. Extract all additional features described in 2.1 from a window w_N with the same size, t_w, on values $\{RSI^{EWMA}_{N-t_w+1},\ \ldots,\ RSI^{EWMA}_{N-1},\ RSI^{EWMA}_{N}\}$.

5. Create a binary plot for a detrended version of the sequence with a predicted value:

$$P_N = plot_{bin}(\{RSI^{EWMA}_{N-t_w+1},\ \ldots,\ RSI^{EWMA}_{N-1},\ RSI^{EWMA}_{N}\}).$$

6. Feed the input sample $s_N = \{P_N, w_N, area_N, c_N, tg_N\}$ to $f(s)$ and generate the output label, o_N.

7. o_N is the signal for the next candle.

Chapter **3**

Evaluation

In this chapter, we will compare the precision of predictions of the proposed model using sets of different parameters. Since teaching deep learning models requires much computational power, we use only 150 epochs, or rounds, to train our neural network. It is not much, and we cannot say with confidence if neural networks can achieve better results if we give them more time to learn. However, it is enough to compare different parameters and find the best set of them for use.

There are many variable parameters of the model that can be changed to achieve better predictions:

- The time frame for the RSI, t^{RSI}.
- The time frame for the EWMA filter, t^{EWMA}.
- How many harmonics, h, for the use with Fourier transform.
- The length of a moving window, t^{w}.
- Plots' height and width.
- How many layers and nodes in them for every neural network in our model.
- A loss function used for teaching neural networks[10].

Table 3.1 demonstrates a few selected evaluation results. Since the table with all tests is large, only five best results were selected. Plot sizes were chosen to be 20 pixels high and 50 pixels wide due to the enormous compression of the model size and, therefore, a huge speed-up in the learning process. The time frames t^{RSI} were chosen according to technical analysis standards as described in 1.1.1.2 – 9, 14 and 25 candles.

[10] Loss functions and different optimisation algorithms were not covered in this work, but if you are interested, you can learn the majority of them from [32], [33] and [34].

t^{RSI}	t^{EWMA}	h	t^{w}	CNN	Data ANN	Raw RSI ANN	Predictor ANN	Accuracy
25	**200**	**300**	**24**	**16,5,r–8,3,r**	**2,r**	**13,r**	**30,r–15,s**	**69.25%**
14	200	1500	27	16,5,r–8,3,r	2,r	8,s	40,r–15,r	67.13%
25	200	700	24	16,7,r–8,5,r	2,r	13,s	100,s–30,r	67.02%
9	100	200	27	32,5,r–16,3,r	1,r	5,r	80,r–30,r	66.88%
25	200	1500	27	16,5,r–8,3,r	2,r	15,r	40,r–15,r	66.10%

Table 3.1: This table shows five best results from got from many different configurations of the RSIVision package. t^{RSI} is the RSI time frame, t^{EWMA} is the EWMA time frame, h stands for harmonics left after applying the Fourier filtering, t^{w} is the sliding window time frame. CNN structure cells have a format {feature maps, kernel size, an activation function}, other ANNs have a format: {a number of nodes, an activation function}. „–“ separates several layers. „r“ stands for the ReLU activation function, „s“ – the sigmoid.

From the results of the evaluation tests, we can conclude several statements:

1. The best t^{RSI} value is 25 appearing three times in the table.
2. The best t^{EWMA} value is 200. That means that smoothing the curve much produces better results.
3. The previous statement is supported by the number of harmonics left; the best result was achieved with only a few frequencies.
4. The sliding window time frame should stay from 24 to 27.
5. ReLU, in our case, provides better results than a sigmoid.
6. Too complex neural networks tend to overfit[11] the data. A simple rule of selecting how many nodes should be present in a layer is being confirmed – hidden layers should contain about half of the nodes from a previous layer.

The best achieved result is only 69%, however, we consider it a good result for a proposed model since we work with cryptocurrency markets that are not easily predictable and do not always follow logical rules.

[11] Overfitting is a term denoting a situation when an algorithm learns random noisy patterns in the training data and, thus, an overall accuracy decreases.

Chapter **4**

Deployment

In this chapter, we will discuss the deployment process of the RSIVision package. The package was developed with an idea to extend it in the future; thus, RSIVision was built as a library with a set of classes and functions that allow a trader to modify the behaviour of implemented algorithms, change the indicator and window sizes for all parameters that were described before.

The package itself was built with a *setup.py* script that allows installing it to the system. We used a Python version 3.7, all dependencies will be automatically installed after running the script. There is, however, a requirement to be able to install RSIVision successfully. TA-Lib is distributed as a C/C++ library and should be downloaded and installed manually from the official web page [24]. The package was tested on macOS Mojave Version 10.14.4.

The configuration of the package is simple since all options that can be changed are stored in one file with the *.yml* extension. This thesis comes with the source code of RSIVision and a *run.py* script that implements a command line interface for interacting with the library.

Conclusion and Future Research

The goal of this thesis was to create a new market signal based on image recognition having OHLCV data of cryptocurrency pairs from an exchange. The data was not labelled, and for being able to train the model, we created a dataset that makes use of graphical representation of one of the most leading market indicators, the Relative Strength Index. We used several filtering techniques to smooth the RSI curve that base on weighted moving averages and Fourier transform. Also, we extracted several features from the data that helped us improve the overall precision of the model.

The model that we have created consists of four neural networks: three fully-connected and one convolutional that reads charts and extracts visual patterns. They are connected to one bigger neural network and predict the next unknown candle from the future, that is, creating a market signal.

The whole proposed solution was implemented as a Python package called RSIVision. The source code and an example of the running script are attached to the work.

Market predictions is not a trivial task, and there are no guarantees that any working solution today will not become irrelevant in just a couple of days. In spite of that, our model could achieve a 69% accuracy in predictions today. We take it as an excellent result since it indicates that neural networks can generate accurate forecasts, and there is something in the data that indicates an upcoming trend.

However, there is much space for future investigations and extension to the solution. First of all, we can change the indicator to another oscillator and combine several models, that is, creating an ensemble. Secondly, we can use more sophisticated feature extraction techniques to extend the dataset and enlarge the search space for neural networks. Thirdly, we can try to combine predictions for different time frames, and, therefore, have more information about the price trend.

Despite the amount of work that we should conduct to conclude with certainty whether the proposed model can generate profits, the results already look promising. However, the solution is far from professional market signals and should not be used for real trading since it can result in losing money.

The world of trading is cruel, but we hope that we could gain the preponderance over the majority of investors that lose money.

Bibliography

1. BANK FOR INTERNATIONAL SETTLEMENTS. *Foreign exchange turnover in April 2016* [online]. 2016. Available also from: `https://www.bis.org/publ/rpfx16fx.pdf`. Technical report. Monetary and Economic Department.
2. RUBIN, Richard; COLLINS, Margaret. How an Exclusive Hedge Fund Turbocharged Its Retirement Plan. *Bloomberg.com* [online]. 2015 [visited on 2019-05-14]. Available from: `https://www.bloomberg.com/news/articles/2015-06-16/how-an-exclusive-hedge-fund-turbocharged-retirement-plan`.
3. GANESH, Prakhar; RAKHEJA, Puneet. *Deep Neural Networks in High Frequency Trading*. 2018. Available from eprint: `arXiv:1809.01506`.
4. MCCLOSKEY, Michael; COHEN, Neal J. Catastrophic Interference in Connectionist Networks: The Sequential Learning Problem. In: BOWER, Gordon H. (ed.). Academic Press, 1989, vol. 24, pp. 109–165. Psychology of Learning and Motivation. ISSN 0079-7421. Available from DOI: `https://doi.org/10.1016/S0079-7421(08)60536-8`.
5. H., R. G.; KEYNES, J.M.; KEYNES, J. M. A Treatise on Money. Vol. I. The Pure Theory of Money. *Journal of the Royal Statistical Society*. 1931, pp. 13. ISBN 9781139520645. ISSN 09528385. Available from DOI: `10.2307/2341941`.
6. STEARNS, Peter N. *The Encyclopedia of World History: Ancient, Medieval, and Modern, Chronologically Arranged*. 6th ed. Ed. by LANGER, William Leonard. Houghton Mifflin Harcourt, 2001. ISBN 978-0395652374.
7. YAHOO! INC. *AAPL Interactive Stock Chart — Apple Inc. Stock - Yahoo Finance* [online] [visited on 2019-05-14]. Available from: `https://finance.yahoo.com/chart/AAPL/`.

8. HAYES, Adam. *Technical Analysis* [Investopedia] [online]. 2019 [visited on 2019-05-12]. Available from: `https://www.investopedia.com/terms/t/technicalanalysis.asp`.
9. PENSO DE LA VEGA, Josef. *Confusion des confusiones, 1688.* Boston: Baker Library, Harvard Graduate School of Business Administration, 1957. Confusion de confuisiones.English. Available also from: `http://hdl.handle.net/2027/uc1.32106019504239`.
10. WILDER, J W. *New Concepts in Technical Trading Systems.* Trend Research, 1978. ISBN 9780894590276. Available also from: `https://books.google.cz/books?id=WesJAQAAMAAJ`.
11. THAM, M. T. *DEALING WITH MEASUREMENT NOISE - Averaging Filter* [School of Chemical Engineering and Advanced Materials] [online]. 2010 [visited on 2019-05-14]. Available from: `https://web.archive.org/web/20100329135531/http://lorien.ncl.ac.uk/ming/filter/filewma.htm`.
12. MORIN, David. Fourier Analysis. In: *Waves.* 2009, chap. Fourier Analysis. Available also from: `http://www.people.fas.harvard.edu/~djmorin/waves/Fourier.pdf`.
13. EHLERS, John. Fourier Transform For Traders. 2007, vol. 25, no. 1, pp. 24.
14. NAKAMOTO, Satoshi. *Bitcoin: A Peer-to-Peer Electronic Cash System.* 2008. ISBN 978-972-757-716-3. ISSN 09254560. Available from DOI: `10.1007/s10838-008-9062-0`. Technical report.
15. WOOD, Gavin. Ethereum: A secure decentralised generalised transaction ledger [online]. 2014. Available also from: `http://gavwood.com/paper.pdf`.
16. DECHTER, Rina. Learning While Searching in Constraint-Satisfaction-Problems. In: 1986, pp. 178–185.
17. LOISEAU, Jean-Christophe B. *Rosenblatt's perceptron, the very first neural network* [Towards Data Science] [online]. 2019 [visited on 2019-05-12]. Available from: `https://towardsdatascience.com/rosenblatts-perceptron-the-very-first-neural-network-37a3ec09038a`.
18. GOODFELLOW, Ian; BENGIO, Yoshua; COURVILLE, Aaron. *Deep Learning.* MIT Press, 2016. `http://www.deeplearningbook.org`.
19. AHN, Song Ho. *Convolution* [songho.ca] [online]. 2013 [visited on 2019-05-13]. Available from: `http://www.songho.ca/dsp/convolution/convolution.html#convolution_2d`.

20. VOICA, Alex. *Heterogeneous compute case study: image convolution filtering* [Imagination] [online]. 2015 [visited on 2019-05-13]. Available from: `https://www.imgtec.com/blog/heterogeneous-compute-case-study-image-convolution-filtering/`.

21. GEBREHIWOT, Asmamaw; HASHEMI-BENI, Leila; THOMPSON, Gary; KORDJAMSHIDI, Parisa; LANGAN, Thomas. Deep Convolutional Neural Network for Flood Extent Mapping Using Unmanned Aerial Vehicles Data. *Sensors*. 2019, vol. 19, no. 7, pp. 1486. Available from DOI: `10.3390/s19071486`.

22. *Free Stock Charts, Stock Quotes and Trade Ideas* [TradingView] [online] [visited on 2019-05-15]. Available from: `https://www.tradingview.com/`.

23. *Welcome to Python.org* [Python.org] [online] [visited on 2019-05-15]. Available from: `https://www.python.org/`.

24. *TA-Lib : Technical Analysis Library - Home* [TA-Lib : Technical Analysis Library] [online] [visited on 2019-05-15]. Available from: `https://www.ta-lib.org/`.

25. BENEDIKTSSON, John. *Python wrapper for TA-Lib (http://ta-lib.org/). Contribute to mrjbq7/ta-lib development by creating an account on GitHub* [online]. 2019 [visited on 2019-05-15]. Available from: `https://github.com/mrjbq7/ta-lib`. original-date: 2012-03-23T17:14:15Z.

26. *Flexible and powerful data analysis / manipulation library for Python, providing labeled data structures similar to R data.frame objects, statistical functions, and much more: pandas-dev/pandas* [online]. pandas, 2019 [visited on 2019-05-15]. Available from: `https://github.com/pandas-dev/pandas`. original-date: 2010-08-24T01:37:33Z.

27. *An Open Source Machine Learning Framework for Everyone: tensorflow/tensorflow* [online]. tensorflow, 2019 [visited on 2019-05-15]. Available from: `https://github.com/tensorflow/tensorflow`. original-date: 2015-11-07T01:19:20Z.

28. CHOLLET, François. *Deep Learning for humans. Contribute to keras-team/keras development by creating an account on GitHub* [online]. Keras, 2019 [visited on 2019-05-15]. Available from: `https://github.com/keras-team/keras`. original-date: 2015-03-28T00:35:42Z.

29. SHEARER, C. The CRISP-DM model: the new blueprint for data mining. *J Data Warehouse*. 2000, vol. 5, pp. 13–22.

30. KOTOCHIGOV, Konstantin. *CRISP-DM: a verified methodology for Data Scientists* [online]. 2017 [visited on 2019-05-15]. Available from: `https://habr.com/ru/company/lanit/blog/328858/`.

31. *Official Documentation for the Binance APIs and Streams* [online]. Binance Exchange, 2019 [visited on 2019-05-14]. Available from: https://github.com/binance-exchange/binance-official-api-docs. original-date: 2017-12-04T03:55:21Z.

32. HAO, Zhang. *Loss Functions in Neural Networks* [Isaac Changhau] [online]. 2017 [visited on 2019-05-15]. Available from: https://isaacchanghau.github.io/post/loss_functions/.

33. GÓMEZ, Raúl. *Understanding Categorical Cross-Entropy Loss, Binary Cross-Entropy Loss, Softmax Loss, Logistic Loss, Focal Loss and all those confusing names* [Raúl Gómez blog] [online]. 2018 [visited on 2019-05-15]. Available from: https://gombru.github.io/2018/05/23/cross_entropy_loss/.

34. WALIA, Anish Singh. *Types of Optimization Algorithms used in Neural Networks and Ways to Optimize Gradient Descent* [Towards Data Science] [online]. 2017 [visited on 2019-05-15]. Available from: https://towardsdatascience.com/types-of-optimization-algorithms-used-in-neural-networks-and-ways-to-optimize-gradient-95ae5d39529f.

Appendix **A**

Acronyms

API Application Programming Interface

ANN Artificial Neural Network

BTC Bitcoin

CNN Convolutional Neural Network

CRISP-DM Cross-Industry Standard Process for Data Mining

DFT Discrete Fourier Transform

EMA Exponential Moving Average

EWMA Exponential Weighted Moving Average

ETH Ethereum

OHLCV Open, High, Low, Close, Volume

P2P Peer-to-Peer

PoS Proof of Stake

PoW Proof of Work

ReLU Rectified Linear Unit

REST Representational State Transfer

RGB Red, Green, Blue

RS Relative Strength

RSI Relative Strength Index

SMA Simple Moving Average

Appendix B

Contents of enclosed CD

```
readme.txt ........................ the file with CD contents description
resources ...................... the directory with models and datasets
src ...................................... the directory of source codes
  impl ...................................... implementation sources
  thesis .............. the directory of LaTeX source codes of the thesis
text .......................................... the thesis text directory
  thesis.pdf ........................... the thesis text in PDF format
```