On Technical Trading and Social Media Indicators for Cryptocurrency Price Classification Through Deep Learning

Marco Ortu[a] (marco.ortu@unica.it), Nicola Uras[a] (nicola.uras@unica.it), Claudio Conversano[a] (conversa@unica.it), Silvia Bartolucci[b] (s.bartolucci@ucl.ac.uk) , Giuseppe Destefanis[c] (giuseppe.destefanis@brunel.ac.uk)

[a] Dept. of Business School, University Of Cagliari, Viale Fra Ignazio 17, Cagliari (IT)

[b] Dept. of Computer Science, University College London, 66-72 Gower Street, WC1E 6EA London (UK)

[c] Dept. of Computer Science, Brunel University, Uxbridge, Middlesex UB8 3PH, London (UK)

Corresponding Author:

Marco Ortu

University Of Cagliari, Italy

Tel: (+39) 070-675-3316

Email: marco.ortu@unica.it

On Technical Trading and Social Media Indicators for Cryptocurrency Price Classification Through Deep Learning

Marco Ortu[a,*], Nicola Uras[a], Claudio Conversano[a], Silvia Bartolucci[b], Giuseppe Destefanis[c]

[a] *University Of Cagliari, Italy*
[b] *University College London, UK*
[c] *Brunel University, UK*

Abstract

Predicting the prices of cryptocurrencies is a notoriously challenging task due to high volatility and new mechanisms characterising the crypto markets. In this work, we focus on the two major cryptocurrencies for market capitalisation at the time of the study, Ethereum and Bitcoin, for the period 2017-2020. We present a comprehensive analysis of the predictability of price movements comparing four different deep learning algorithms (*Multi Layers Perceptron (MLP)*, *Convolutional Neural Network (CNN)*, *Long Short Term Memory (LSTM) neural network* and *Attention Long Short Term Memory (ALSTM)*). We use three classes of features, considering a combination of technical (e.g. opening and closing prices), trading (e.g. moving averages) and social (e.g. users' sentiment) indicators as input to our classification algorithm. We compare a *restricted model* composed of technical indicators only, and an *unrestricted model* including technical, trading and social media indicators. We found an increase in accuracy for the daily classification task from a range of 51-55% for the restricted model to 67-84% for the unrestricted one. This study demonstrates that including *both* trading and social media indicators yields a significant improvement in

*Corresponding author.
Email addresses: marco.ortu@unica.it (Marco Ortu), nicola.uras@unica.it (Nicola Uras), conversa@unica.it (Claudio Conversano), s.bartolucci@ucl.ac.uk (Silvia Bartolucci), giuseppe.destefanis@brunel.ac.uk (Giuseppe Destefanis)

the prediction and accuracy consistently across all algorithms.

Keywords: Cryptocurrencies, Text Analysis, Deep Learning, Social Media Indicators, Trading Indicators

---

1. Introduction

During the last decade, the global markets have witnessed an exponential growth in the number of cryptocurrencies traded and exchanged, reaching a market capitalization of hundreds of billions of US Dollars globally (reaching $\approx$ 1 trillion as of January 2021).

Recent surveys[1] also report a spike in demand and interest for the new crypto-assets from institutional investors, attracted by the novel features and the potential rise in value in the current financial turmoil, despite the risk associated with price volatility and market manipulation.

Boom and bust cycles, often induced by network effects and wider market adoption, make prices hard to predict with high accuracy. There is a large body of literature concerning this issue, proposing a number of quantitative approaches for cryptocurrency prices prediction (Karim et al. (2019); Lahmiri Salim (2019); Jing-Zhi H. (2018); Katsiampa (2017); Lahmiri et al. (2018)). In particular, Giudici & Polinesi (2019) applied hierarchically clustering on Bitcoin prices gathered from different exchanges to understand the dynamics of cryptoasset prices and, more specifically, how price information is transmitted across different Bitcoin exchanges, and between Bitcoin markets and traditional ones. Akyildirim et al. (2021) analysed the predictability of twelve cryptocurrencies at the daily- and minute-level frequencies using machine learning classification algorithms. The average classification accuracy of four tested algorithms was consistently above the 50% threshold for all cryptocurrencies at all timescales, showing the feasibility of prediction of trends in

[1] See https://www.fidelitydigitalassets.com/bin-public/060_www_fidelity_com/documents/FDAS/institutional-investors-digital-asset-survey.pdf

prices to a certain degree in cryptocurrency markets. The rapid fluctuations in volatility, autocorrelations and multi-scaling effects in cryptocurrencies have also been extensively studied Matta et al. (2015), also concerning their effect on Initial Coin Offerings (ICOs) (Hartmann et al. (2018, 2019)).

An important consideration that has gradually emerged from the literature is also the relevance of the "social aspect" and interactions in crypto trading. For example, the code underlying blockchain platforms is developed in an open source fashion on GitHub, recent additions to the crypto ecosystem are discussed on Reddit or specialised channels in Telegram, and Twitter offers a platform where often heated debates on the latest developments take place. By analysing interactions on social media, it has been shown that sentiment index can be used to predict bubbles in prices Chen & Hafner (2019) and that the sentiment extracted from topic discussions on Reddit correlates with prices Phillips & Gorse (2018). Open-source development also plays an important role in shaping the success and value of cryptocurrencies (Ortu (2015); Ortu et al. (2019); Marchesi et al. (2020)). In particular, a recent work by Bartolucci et al. (2020) – which this work is an extension of – showed the existence of a Granger causality between the sentiment and emotions time series extracted from developers' comments on GitHub and returns of cryptocurrencies. For the two major cryptocurrencies – Bitcoin and Ethereum – it has been also shown how including the developers' emotions time series in prediction algorithms could substantially improve the accuracy.

In this paper, we further extend previous investigations on price predictability using a deep learning approach and focusing on the two major cryptocurrencies by market capitalization, Bitcoin and Ethereum. We predict price movements by mapping the point-wise forecast of price into a classification problem: our target is a binary variable with two unique classes, upward and downward movements, which indicate prices rising or falling and compare the performances and outcome of the following deep learning algorithms: the Multi-Layer Perceptron (MLP), the Multivariate Attention Long Short Term Memory Fully Convolutional Network (MALSTM-FCN), the Convolutional Neural Network

(CNN) and the Long Short Term Memory neural network (LSTM).
We will use as input the following classes of (financial and social) indicators: (i) technical indicators, such as open and close price or volume traded, (ii) trading indicators, such as the momentum and moving averages calculated on the price, (iii) social media indicators, i.e. sentiment and emotions extracted from GitHub and Reddit comments.
For each deep learning algorithm, we consider a *restricted* and *unrestricted* data model at an hourly and daily frequency. The *restricted model* consists of data concerning technical variables for Bitcoin and Ethereum. In the *unrestricted model* we include, instead, the technical variables, trading and social media indicators from GitHub and Reddit.
A key contribution of this study is the implementation of fine-tuned, deep learning algorithms with higher classification performances compared to previous studies (Bartolucci et al. (2020); Uras & Ortu (2021)). A second important aspect is the construction of a model that includes a unique mix of social media features, obtained using state-of-the-art techniques for textual analysis and natural language processing, and trading indicators. Finally, our results demonstrate the relevance of social media indicators, suggesting that the stakeholders should pay an increased attention to crypto-related activities on social media to make informed decisions.

The paper is organised as follows. In Section 2 we describe in detail the hypotheses and the background of the study. In Section 3, we discuss the methodology of the experiments conducted. In Section 4, we present the main results and in Section 5 we discuss the implications of this study and outline future directions.

2. Theory, Hypothesis and Context

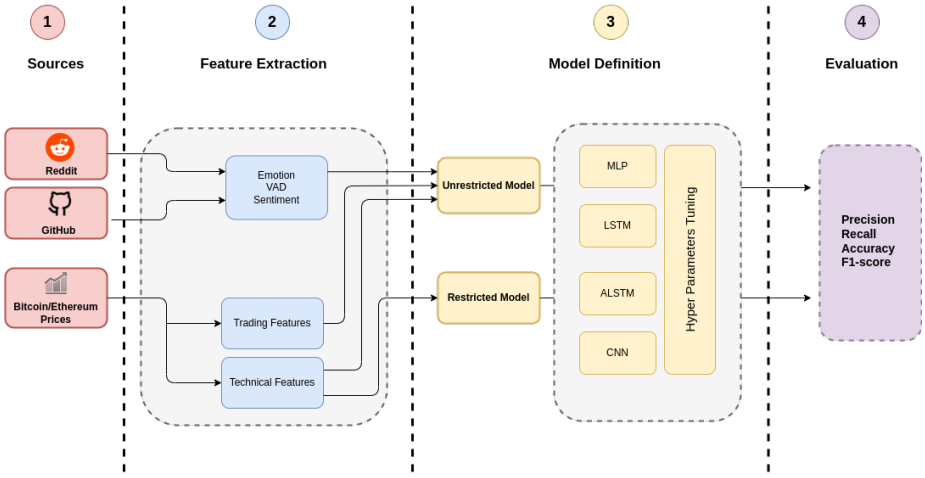
Stock markets and social media have deep links as witnessed for instance by the recent chaotic events involving GameStop (Gomez-Carrasco & Michelon (2017)). GameStop is a Texan video game company, which has unexpectedly

disrupted the financial markets for several trading sessions in January 2021, when the GME title registered +400% gain (on the 12th of January 2021 the value of a GME share was $19.95, while on the 29th of January it reached $325). The value of the stock skyrocketed after the action of a coordinated (through the forum *WallStreetBets*) group of online traders who targeted several hedge funds that decided to short-sell shares of the video game company (Jones et al. (2021)). Cryptocurrency markets are in many aspects similar to stock markets, and the links with social media are even stronger (Keskin Z. (2019); Phillips & Gorse (2018)). Building upon previous research (Bartolucci et al. (2020)), which highlighted the potential prediction power of social media features on cryptocurrencies markets, we hypothesize:

H1: Using a mix of trading and social media indicators leads to better cryptocurrencies price classification.

The research model is shown in Figure 1. Following the four steps represented in Figure 1, we constructed a model consisting of only technical indicators and a model consisting of technical, trading and social indicators.

Figure 1: Experimental design.



We compared the price classification performance of these two models using four different deep learning algorithms, and we found better performance with

the model including *both* trading and social features. Consistently, across all four deep learning algorithms, we are able to show that the unrestricted model outperforms the restricted model. At hourly data frequency, the inclusion of trading and social media indicators alongside the classic technical indicators improves the accuracy of both Bitcoin and Ethereum price prediction. The accuracy increases from a range of 51-55% for the restricted model, to 67-84% for the unrestricted one. For the daily frequency resolution, in the case of Ethereum, the most accurate classification is achieved using the restricted model. For Bitcoin, instead, the highest performance is achieved for the unrestricted model including only social media indicators.

H2: Cryptocurrency markets' price classification is improved using social media indicators during sub-periods of financial distress.

We considered three different periods of cryptocurrency markets distress and we applied the *H1* model to these sub-periods comparing the distribution of *f1-score* of the four algorithms, separating the contribution of *trading indicators* from that of *social indicators*. We found that the use of social media indicators *significantly* improved the *daily* price classification, confirming that social media are the right place to look at for early signals of financial distress.

2.1. Context

In this study, we collect and analyse price data for the two main cryptocurrencies, Bitcoin and Ethereum, from January 2017 to January 2021. The last four years registered a tumultuous alternation of appreciation and depreciation of cryptocurrencies, with Bitcoin price at about 3000 USD in early 2017 and at about 60000 USD at the same time in 2021 (roughly 20 times more). The same trend holds for Ethereum, whose price was about 150 USD in early 2017 and went over 1500 USD in early 2021 (and at the time of writing is roughly over 2000 USD). We also collected for the period January 2017 - January 2021 data from the social media Reddit and the social coding platform GitHub, which hosts the development of the source code of the two cryptocurrencies. On one hand, Reddit hosts discussions of users who are investors or that discuss in gen-

eral about the technology itself. On the other hand, users in GitHub are the developers of the technology and the ones who are in charge of the main governance decisions for these platforms. These two social platforms aggregate users that for several reasons have a direct interest in the cryptocurrency markets and, therefore, act as stakeholders. Within the considered time period, we have three main events where the two cryptocurrencies went through speculation bubbles: from November 2018 to December 2018, from April 2019 to December 2019 and from October 2020 to February 2021. Several factors contributed to these speculation bubbles and, interestingly, some evidence of these factors can be directly found in the social platforms analysed.

3. Methodology

In the next sections, we discuss the technical, trading and social media indicators, providing some insights on the data collection and preparation processes. We also illustrate the deep learning algorithms and how they are designed for the price classification problem.

3.1. Technical and Trading Indicators

We considered all the available technical variables extracted from the *Crypto Data Download* web services[2], focusing on the data from the Bitfinex[3] exchange service. We considered the last 4-years period, spanning from 2017/01/01 to 2021/01/01, for a total of 35638 hourly observations. In our analysis, we separate the technical indicators into two main categories: pure technical and trading indicators. Technical indicators refer to "direct" market data such as opening and closing prices. Trading indicators refer to derived indicators, such as moving averages. The technical indicators are listed below.

- *Close*: the last price at which the cryptocurrency traded during the trading period.

[2]https://www.cryptodatadownload.com/data/bitfinex/
[3]https://www.bitfinex.com

- *Open*: the price at which the cryptocurrency first trades upon the opening of a trading period.
- *Low*: the lowest price at which the cryptocurrency trades over the course of a trading period.
- *High*: the highest price at which the cryptocurrency traded during the course of the trading period.
- *Volume*: the number of cryptocurrency trades completed.

From the knowledge of these technical indicators, it is possible to calculate the trading indicators. More precisely, we used the *StockStats* Python library to generate them. We used 36 different trading indicators as shown in Table 1, which are of 8 different types:

- Moving average (MA);
- Exponential moving average (EMA);
- Stochastic oscillator;
- Moving average convergence divergence (MACD);
- Bollinger bands;
- Relative strength index (RSI);
- Fibonacci retracement levels;
- Average directional index.

The lag values represent how previous values $(t-1, \ldots, t-n)$ are used as input. The *window size* indicates the number of previous values used to evaluate the indicator at time t, e.g. to calculate $ADXR_t$ at time t we use $ADX_{t-1}, ..., ADXR_{t-10}$, ten previous values.

We provide here the definitions of the five main trading indicators.

Table 1: Trading indicators with associated lags and window sizes. Lags represent how previous values at $(t-1, \ldots, t-n)$ are used as input. Window size represents the number of previous values used to compute the indicator at time t, e.g. to calculate $ADXR_t$ at time t we use $ADX_{t-1}, \ldots, ADXR_{t-10}$.

Trading Indicator	**Lag**	**Window size**
SMA: Simple Moving Average	-	10
WMA: Weighted Moving Average	-	10
RSI: Relative Strength Index	-	10
ROC: Price Rate Of Change	-	10
Mo: Momentum:	-	10
OBV: On Balance Volume	1	-
permutation (zero based)	1	-
log return	1	-
max in range	1	-
min in range	1	-
middle = (close + high + low) / 3	1	-
compare: le, ge, lt, gt, eq, ne	1	-
count: both backward(c) and forward(fc)	1	-
SMA: simple moving average	-	10
EMA: exponential moving average	-	10
MSTD: moving standard deviation	-	10
MVAR: moving variance	-	10
RSV: raw stochastic value	-	10
RSI: relative strength index	-	10
KDJ: Stochastic oscillator	-	10
Bolling: including upper band and lower band.	1	-
MACD: moving average convergence divergence	-	5
CR: price momentum index	1	-
WR: Williams Overbought/Oversold index	1	-
CCI: Commodity Channel Index	1	-
TR: true range	1	-
ATR: average true range	1	-
line cross check, cross up or cross down.	1	-
DMA: Different of Moving Average (10, 50)	1	-
DMI: Directional Moving Index, including	1	-
DI: Positive Directional Indicator	1	-
ADX: Average Directional Movement Index	-	5
ADXR: Smoothed Moving Average of ADX	-	10
TRIX: Triple Exponential Moving Average	-	10
TEMA: Another Triple Exponential Moving Average	-	10
VR: Volatility Volume Ratio	1	-

- Simple Moving Average (SMA): calculated as the arithmetic average of the cryptocurrency closing price over some period (known as *time period*).
- Weighted Moving Average (WMA): it is a moving average calculation that assigns higher weights to the most recent price data.
- Relative Strength Index (RSI): it is a momentum indicator that measures the magnitude of recent price changes. It is normally used to evaluate whether stocks or other assets are being overbought or oversold.
- Price Rate Of Change (ROC): it measures the percentage change in price between the current price and the price a certain number of periods ago.
- Momentum: it is the rate of acceleration of a security's price, i.e. the speed at which the price is changing. This measure is particularly useful to identify trends.
- On Balance Volume (OBV): it is a technical momentum indicator based on the traded volume of an asset to predict changes in stock price.

3.1.1. Summary statistics: Technical and Trading indicators

Table 2 shows the summary statistics for the technical indicators. In Figure 2 we also show the plot of the historical time series for the technical indicators. Table 3 shows the statistics of the trading indicators for the considered period of analysis. Technical and Trading indicators are then input as features for the price classification model.

Table 2: Summary statistics for the time series of technical indicators.

Cryptocurrency		High	Open	Low	Volume	Close
	mean	7972.769	7928.018	7879.276	176.319	7928.894
	std	5519.337	5471.592	5416.295	306.62	5472.983
	min	769.1	760.38	752	0	760.38
Bitcoin	**25%**	4161.6875	4137.995	4113.822	30.592	4138.475
	50%	7459.995	7428.09	7390.47	80.699	7428.41
	75%	9790.952	9751.84	9701.427	199.322	9752.37
	max	41999.99	41526.95	41000.24	8526.751	41526.95
	mean	313.202	310.856	308.253	1658.835	310.896
	std	248.731	246.069	242.971	6903.628	246.135
	min	8.17	8.15	8.15	0	8.15
Ethereum	**25%**	161.182	160.202	159.06	192.327	160.21
	50%	232.79	231.34	229.765	569.79	231.365
	75%	390.0575	388.0075	385.73	1632.640	388.025
	max	1440.54	1430.94	1411	903102.685	1431.4

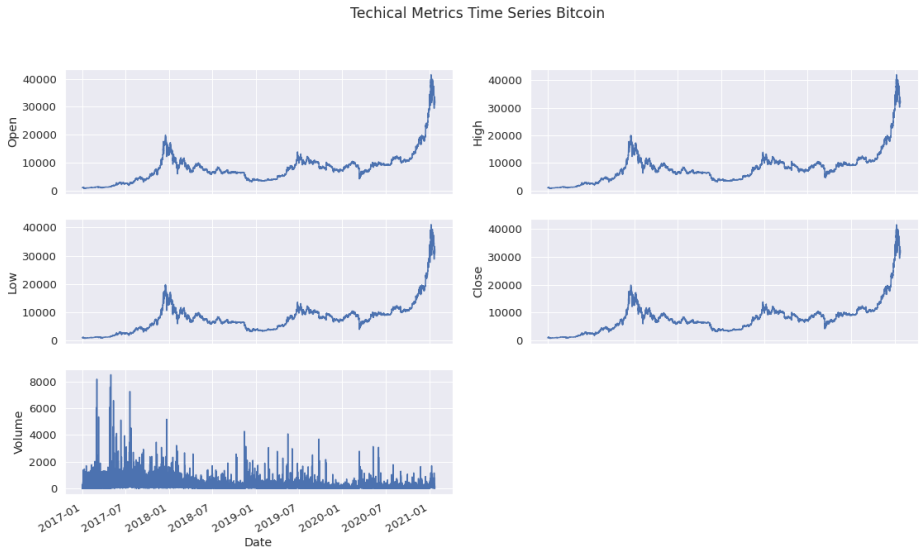
Table 3: Summary statistics for the time series of trading indicators.

Cryptocurrency		SMA	WMA	RSI	ROCP	MOM	OBV
	mean	7924.974	7926.275	51.797	0.0013	8.685	126972.751
	std	5465.393	5467.642	14.484	0.027	298.311	33544.231
	min	767.801	766.912	2.426	-0.321	-5260.55	18811.069954
Bitcoin	**25%**	4135.225	4134.525	42.374	-0.0083	-53.64	110336.0947
	50%	7427.187	7427.521	51.877	0.001	4.97	126464.383
	75%	9753.094	9751.604	61.151	0.011	70.732	147814.358
	max	40996.6	41106.93	98.641	0.314	4069.26	213166.214
	mean	310.723	310.78	51.18	0.002	0.378	6.776e+05
	std	245.768	245.87	14.246	0.035	16.607	5.739e+05
	min	8.147	8.163	3.797	-0.317	-239.48	-4.993e+04
Ethereum	**25%**	160.202	160.252	42.063	-0.012	-2.8	9.864e+04
	50%	230.916	230.962	51.038	0.000934	0.09	5.185e+05
	75%	388.081	388.125	60.338	0.016	3.67	1.246e+06
	max	1404.89	1411.776	95.799	0.333	262.36	1.667e+06

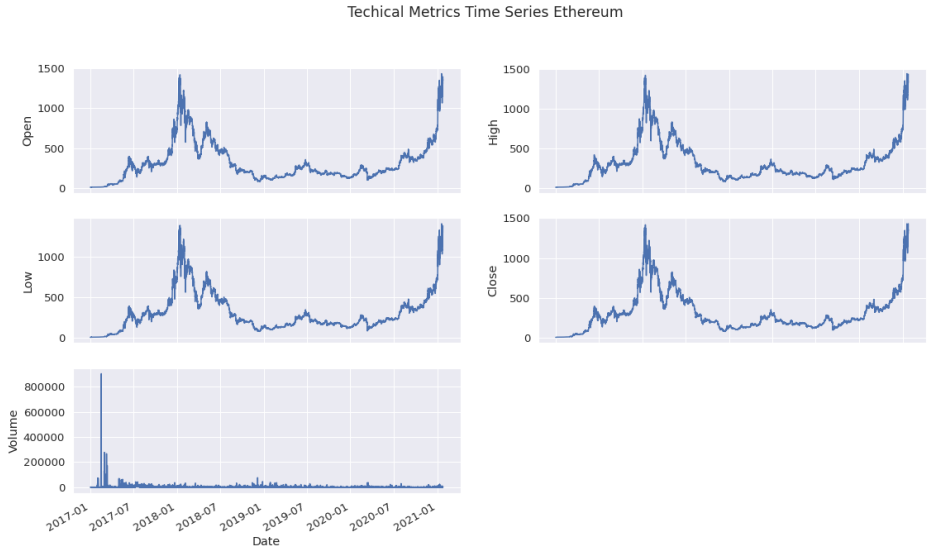
3.2. Social Media Indicators

In this section, we discuss how the time series of social media indicators are constructed from Ethereum and Bitcoin developers comments on GitHub and users' comments on Reddit, respectively.

Figure 2: Technical metrics for Bitcoin and Ethereum Time Series.



(a) Plot of the time series of Bitcoin's technical indicators.



(b) Plot of the time series of Ethereum's technical indicators.

The social media platform *Reddit* is an American social news aggregation, web content rating, and discussion website that reaches about 8 billion page views per month. Reddit is built over multiple subreddits, where each subreddit is dedicated to discussing a particular topic. In this paper, we analyse two subreddits for each cryptocurrency, one discussing technical aspects and one trading-related. The chosen subreddits are listed in Tab. 4. For each subreddit, we fetched all comments from January 2017 to January 2021.

Table 4: List of sub-Reddit channels considered in the analysis.

Cryptocurrency	***Technical Discussions***	***Trading Discussions***
Bitcoin	r/Bitcoin	r/BitcoinMarkets
Ethereum	r/Ethereum	r/EthTrader

An example of a user's comment extracted from Reddit r/Ethereum can be seen in Table 5. Quantitative measures of sentiment and emotions associated with the comments, as reported in this example, are computed using state-of-the-art textual analysis tools (further details are discussed in the following sections). The social media indicators computed for each comment are emotions as love (L), joy (J), anger (A), sadness (S), VAD (valence (Val), dominance (Dom), arousal (Ar)) and sentiment (Sent). Similar comments and associated social media indicators can be extracted for developers' comments from GitHub (Bartolucci et al. (2020)).

Concerning the activity on Github, as both the Bitcoin and Ethereum projects are open-source, we can extract all the interactions among contributors (Ortu et al. (2018)). Active contributors are continuously opening, commenting, and closing the so-called "issues". An issue is an element of the development process, which carries information about discovered bugs, suggestions on new functionalities to be implemented in the code, new features, or new functionalities being developed. An issue can be "commented", meaning that developers can start sub-discussions around it. They usually add comments to a given issue to

Table 5: Example of Reddit comment and corresponding emotions (love (L), joy (J), anger (A), sadness (S)), VAD (valence (Val), dominance (Dom), arousal (Ar)), politeness and sentiment (Pol and Sent respectively).

Comment	**L**	**J**	**A**	**S**	**Val**	**Dom**	**Ar**	**Sent**
All the tosspots focusing on Vitaliks wealth completely miss the point. If the crypto you are supporting has a purpose it will garner interest in the real world therefore the capital will flow to it. All is measured on the merit and proper fundamentals and not twitterbot pump and dumps...	0	0	0	0	2.13	1.98	2.26	-1

highlight the actions being undertaken or provide suggestions on the possible resolution.

In the next sections, we will briefly describe the algorithm used to extract emotions and sentiment features from Github's and Reddit's comments. We will provide summary statistics of the time series extracted that will be used as features in the deep learning algorithms for price classification.

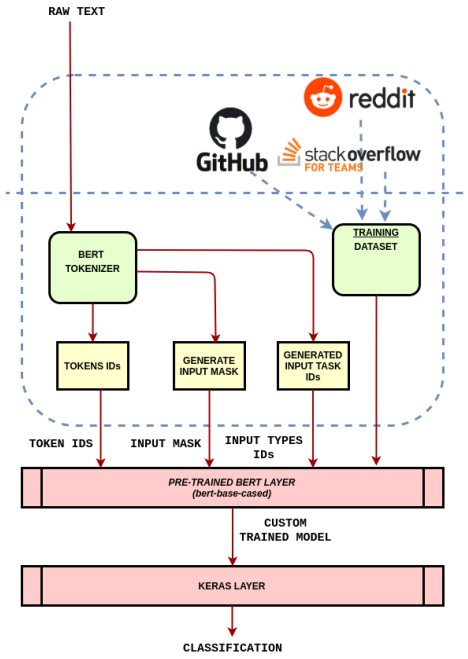
3.2.1. Extracting social media indicators

We extracted the social media indicators using deep, pre-trained, neural networks called Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al. (2018)). BERT and other Transformer encoder architectures have been successful in performing various tasks in natural language processing (NLP) and represent the evolution of Recurrent Neural Network (RNN) typically used in NLP.

Transformers are based on the *Attention Mechanism* where RNN units would encode the input up until timestamp t into one hidden vector h_t. The latter would then be passed to the next timestamp (or to the decoder in the case of a

sequence-to-sequence model). By using the attention mechanism, one no longer tries to encode the full source sentence into a fixed-length vector. Instead, one allows the decoder to attend to different parts of the source sentence at each step of the output generation. Importantly, we let the model learn what to attend to, based on the input sentence and what it has produced so far.

Figure 3: Scheme of the general bidirectional encoder representation from Transformer.



The Transformer architecture allows for the creation of NLP models trained on very large datasets, as we have done in this work. Training such models on large datasets is relatively feasible thanks to pre-trained language models, which can be, then, simply fine-tuned on the particular dataset. This means that the weights learnt by the extensively pre-trained models can be later reused for specific tasks by simply tailoring the weights to the specific dataset.

In this analysis, we used Tensorflow and Keras Python libraries with the Transformer package to leverage the power of these pre-trained *BERT-base-case* neural networks (NNs). Figure 3 shows the architectural design used to train

the three NNs classifiers that extract the social media indicators. The three gold datasets used to train our final models are GitHub, Stack Overflow and Reddit. We used the BERT tokenizer to preprocess raw text and the datasets as input for the pre-trained BERT layer (dashed blue square). This phase produces the *word-embedding* vector (Mikolov et al. (2013)) along with the tokens IDs. The output of the pre-trained BERT layer becomes the input of the last layer of our deep learning architecture, i.e., the `Keras` Layer.

In particular, we used a sentiment-labelled dataset consisting of 4423 posts mined from Stackoverflow users' comments to train the sentiment model for GitHub: comments on both platforms are written using the technical jargon language of software developers and engineers. We also used an emotion-labelled dataset of 4200 sentences from GitHub (Murgia et al. (2018)). This dataset is particularly suited for our analysis as the algorithm for emotion detection has been trained on developers' comments extracted from the Jira Issue Tracking System of the Apache Software Foundation, hence within the Software Engineering domain and context of the GitHub and Reddit comments analysed in this paper.

Valence, Arousal and Dominance (VAD) represent conceptualised affective dimensions that describe the interest, alertness and control an individual feels in response to a particular stimulus. In the context of software development, VAD measures may indicate the involvement of a developer in a project as well as their confidence and responsiveness in completing tasks. Warriner et al. (Warriner et al. (2013)) has created a reference lexicon containing 14,000 English words with VAD scores for Valence, Arousal, and Dominance, that can be used to train the classifier. In Mäntylä et al. (2016) they extracted the valence-arousal-dominance (VAD) metrics from 700,000 Jira issue reports containing over 2,000,000 comments and showed that issue reports of different type (e.g., feature request vs bug) had a fair variation of valence. In contrast, an increase in issue priority typically increased arousal.

Finally, sentiment is measured using the BERT classifier trained with (i) the public dataset used in similar studies (Calefato et al. (2018, 2017)) for

the GitHub data and (ii) a public gold dataset containing 33K labelled Reddit users' comments available from Kaggle, the largest and well-known web platform for sharing datasets [4] for Reddit data. The algorithm extracts the sentiment polarity expressed in short texts in three levels: positive (1), neutral (0) and negative (-1) sentiment.

Tables 6 and 7 show the performance of sentiment and emotion classification on the Github and Reddit dataset. The classifier can detect love, anger, joy and sadness with an f_1-score[5] close to 0.89 for all the emotions. Note that the accuracy reported is the global accuracy as it is evaluated for all classes, i.e., negative, neutral and positive sentiment.

In the next sections (see Sec. 3.1.1, 3.2.2), we discuss how we construct the time series of social and technical indicators and we provide the summary statistics.

Table 6: Sentiment Classifier Evaluation.

	Cryptocurrency	**precision**	**recall**	**f1-score**
Bitcoin	**negative**	0.92	0.89	0.90
	neutral	0.97	0.98	0.98
	positive	0.95	0.95	0.95
	accuracy			0.95
	macro avg	0.95	0.94	0.94
	weighted avg	0.95	0.95	0.95
Ethereum	**negative**	0.98	0.85	0.91
	neutral	0.84	0.94	0.89
	positive	0.96	0.97	0.96
	accuracy			0.92
	macro avg	0.93	0.92	0.92
	weighted avg	0.93	0.92	0.92

[4] https://www.kaggle.com/cosmos98/twitter-and-reddit-sentimental-analysis-dataset

[5] The f_1-score tests the accuracy of a classifier. It is calculated as the harmonic mean of precision and recall.

Table 7: Emotion Classifier Evaluation For GitHub.

	precision	**recall**	**f1-score**
anger	0.83	0.77	0.80
sadness	0.89	0.89	0.89
joy	0.86	1.00	0.92
love	1.00	1.00	1.00
accuracy			0.89
macro avg	0.89	0.91	0.90
weighted avg	0.89	0.89	0.89

3.2.2. Summary statistics: Social Media Indicators on GitHub and Reddit

Our analysis focuses on three main classes of affect metrics: emotions (love, joy, anger, sadness), VAD metrics (valence, arousal, dominance) and Sentiment.

Once numerical values of the affect metrics are computed for all comments (as shown in the example in Table 5), we consider the comments timestamps (i.e. dates when the comments was posted) to build the corresponding social media indicators time series. The affect time series are constructed by aggregating sentiment and emotions of multiple comments for each hour and day, depending on the time resolution considered (hourly and daily). For a given social media emotion indicator, e.g., *anger*, and for a specific temporal resolution, we construct the time series by averaging the values of the affect metric over all comments posted on the same day.

In Table 8, we report in more details the summary statistics of the social indicators' time series for both cryptocurrencies. Table 9 shows the summary statistics for the social indicators of the two Bitcoin and two Ethereum subreddit channels considered.

Table 8: Summary statistics of GitHub affect metrics.

Cryptocurrency		sentiment	arousal	valence	dominance	joy	love	sadness	anger
	mean	0.141	2.273	3.321	3.365	0.0729	0.056	0.227	0.109
	std	0.774429	2.953897	4.324653	4.376877	0.293435	0.248373	0.566562	0.393479
	min	-11	0	0	0	0	0	0	0
Bitcoin	**25%**	0	0	0	0	0	0	0	0
	50%	0	1.27	1.85	1.87	0	0	0	0
	75%	0	3.29	4.8	4.87	0	0	0	0
	max	15	38.88	60.78	62.28	6	4	11	17
	mean	0.0842	0.7934	1.1405	1.147	0.0182	0.0761	0.0961	0.0356
	std	0.6754	1.7082	2.4653	2.477	0.1407	0.5284	0.3570	0.2052
	min	-4	0	0	0	0	0	0	0
Ethereum	**25%**	0	0	0	0	0	0	0	0
	50%	0	0	0	0	0	0	0	0
	75%	0	1.08	1.54	1.62	0	0	0	0
	max	31	35.19	52.5	54.35	3	31	6	4

3.3. Price Movement Classification

We, then, map the price regression problem into a classification task where we predict the direction of the price movement (up or down). The caveat here is that we only predict the direction of price movement and not its magnitude.

The target variable is a binary variable with two unique classes:

- **Upward movements**: This class, labelled with *up* and encoded with 1, represents the condition of increasing prices.
- **Downward movements**: This class, labelled *down* and encoded with 0, represents the condition of falling prices.

Table 10 shows the details about the instances of classes *down* and *up*, with $48,5\%$ and 51.5% respectively for Bitcoin and $49,8\%$ and $50,2\%$ for Ethereum with and hourly frequency. For daily frequency we have $44,8\%$ and 55.2% for Bitcoin and $48,5\%$ and $51,5\%$ for Ethereum of *down* and *up* class instances. For Bitcoin daily frequency we have a slightly unbalanced distribution towards *up* classes, in this case we will consider *f1-score* along with *accuracy* to better assess the model's performance.

Table 9: Summary Statistics of Reddit Social Media Indicators.

Subreddit		sentiment	arousal	valence	dominance	joy	love	sadness	anger
	mean	1.8582	8.6046	12.0466	11.7412	0.6492	0.2509	0.5579	2.2197
	std	4.3498	17.3895	24.4038	23.7624	1.7040	0.8278	1.4223	4.7780
	min	-9	0	0	0	0	0	0	0
r/Bitcoin	**25%**	0	0	0	0	0	0	0	0
	50%	0	1.09	1.54	1.51	0	0	0	0
	75%	2	10.25	14.22	13.88	1	0	0	2
	max	101	492.99	680.41	662.39	42	27	34	133
	mean	0.9264	4.0327	5.6617	5.5688	0.2419	0.1059	0.3383	1.0095
	std	2.7650	10.7106	14.9023	14.6243	0.8616	0.4553	1.0275	2.8993
	min	-9	0	0	0	0	0	0	0
r/Bitcoinmakets	**25%**	0	0	0	0	0	0	0	0
	50%	0	0	0	0	0	0	0	0
	75%	0	2.32	3.2925	3.26	0	0	0	0
	max	52	245.32	332.84	329.4	34	15	22	88
	mean	0.0842	0.7934	1.1405	1.147	0.0182	0.0761	0.0961	0.0356
	std	0.6754	1.7082	2.4653	2.477	0.1407	0.5284	0.3570	0.2052
	min	-4	0	0	0	0	0	0	0
r/Ethereum	**25%**	0	0	0	0	0	0	0	0
	50%	0	0	0	0	0	0	0	0
	75%	0	1.08	1.54	1.62	0	0	0	0
	max	31	35.19	52.5	54.35	3	31	6	4
	mean	0.8150	2.8479	4.0716	4.0046	0.2123	0.0855	0.2072	0.5983
	std	2.1528	6.1395	8.7652	8.6220	0.6807	0.3959	0.6641	1.577632
	min	-6	0	0	0	0	0	0	0
r/Ethtraders	**25%**	0	0	0	0	0	0	0	0
	50%	0	0	0	0	0	0	0	0
	75%	1	3.25	4.65	4.6	0	0	0	1
	max	62	138.39	207.38	191.95	28	23	13	37

Table 10: Class instances counts and percentages for Bitcoin and Ethereum at an hourly or daily frequency.

Frequency	Cryptocurrency	Class	Counts	Percentage
Hourly	Bitcoin	up	17246	48,5%
		down	18271	51,5%
	Ethereum	up	16844	49,8%
		down	16956	50,2%
Daily	Bitcoin	up	665	44,8%
		down	817	55,2%
	Ethereum	up	684	48,5%
		down	727	51,5%

3.3.1. Time Series Processing

Since we are analysing a supervised learning problem, we prepare our data to have a vector of x inputs and a y output with temporal dependence. In this case, the input vector x is called *regressor*. The x inputs include the model's predictors, i.e. one or several values from the past, the so-called *lagged* values. Inputs correspond to the values of the selected features discussed in the previous sections. The target variable y is a binary variable, which can be either 0 or 1. The 0 (*down*) instance represents downward price movements. A 0 instance at time t is obtained when the difference between the *close price* at time t and the *open price* at time $t+1$ is less than or equal to 0. The 1 (*up*) instance represents upward price movements. A 1 instance is obtained when the difference between the *close price* at time t and the *open price* at next time step $t+1$ is greater than 0. We considered two time series models:

- ***Restricted***: the input vector x consists of only technical indicators (open, close, high, low, volume).
- ***Unrestricted***: the input vector x consists of technical, trading and social indicators.

For both the *restricted* and *unrestricted* model we used 1 lagged value for each indicator. The purpose of considering two distinct models is to ascertain and quantify whether the addition of trading and social indicators to the *regressor* vector leads to an effective improvement in the Bitcoin and Ethereum price changes classification.

3.4. Deep Learning Algorithms

In the next section, we briefly discuss the deep learning algorithms used in our experiments. To conduct this analysis, we used the `Keras` framework for deep learning (Chollet et al. (2015)).

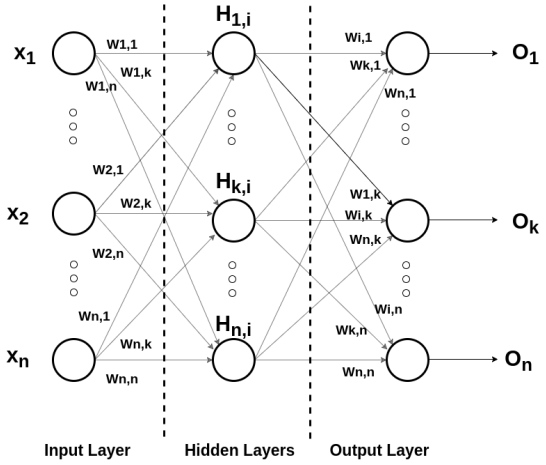
The selected architectures are particularly suitable for time series manipulation and forecasting (Brownlee (2018)). Although MLP is not a deep learning algorithm by definition, we included it for the following reasons. If we consider the two most common characteristics of DL architectures, i.e., the presence of multiple hidden layers and the use of non-linear activation functions, we can broadly classify the MLP algorithm as a deep feed-forward artificial neural network. Moreover, the MLP architecture represents one of the simplest neural networks and can be used as a baseline for comparisons. The other three architectures, namely LSTMs, CNNs and MALSTM-FCNs are also deep learning state-of-the-art tools for time series forecasting with increasing underlying architectural complexity. Testing the classification tasks on multiple DL algorithms allows us to ensure the robustness of our findings.

3.4.1. Multilayer Perceptron

A multilayer perceptron (MLP) Ivakhnenko et al. (1967) is a class of feed-forward artificial neural networks (ANNs), characterised by multiple layers of perceptrons and a typical activation function.

When composed of a single hidden layer, as shown in Figure 4, MLPs are called "vanilla" neural networks (in jargon and for practical use). In general, MLPs refer to neural network architectures with two or more hidden layers. A MLP comprises three main node categories: *input layer nodes*, *hidden layer*

Figure 4: Scheme of the Multilayer Perceptron architecture.



nodes and *output layer nodes*. All nodes of the neural network are perceptrons that use a nonlinear activation function, except for the input nodes.

In general, MLP neural networks are resilient to noise and can also support learning and inference when values are missing. Moreover, neural networks do not make strong assumptions about the mapping function and readily learn both linear and nonlinear relationships. An arbitrary number of input features can be specified, providing direct support for multidimensional forecasting. An arbitrary number of output values can also be specified, providing direct support for multi-step and even multivariate forecasting (Sutskever et al. (2014)). For these reasons, MLP neural networks may be particularly useful for time series forecasting (Brownlee (2018)).

In Fig. 4, the output of the i-th node (neuron) is indicated by O_i, and the weighted sum of the input connections is v_i. The most common activation functions are the following:

$$O(v_i) = \tanh(v_i) \quad \text{and} \quad O(v_i) = (1 + \mathrm{e}^{-v_i})^{-1} \ . \tag{1}$$

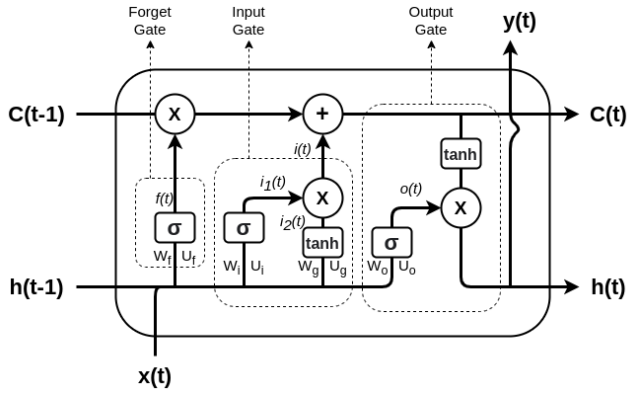
MLPs differ from linear perceptrons because of their multiple layers and nonlinear activation functions.

3.4.2. Long Short Term Memory

Long Short-Term Memory networks are a specialised version of Recurrent Neural Networks (RNNs), which are able to capture long-term dependencies in a sequence of data (Hochreiter & Schmidhuber (1997)). RNNs are a type of artificial neural networks with a particular topology specialised in the identification of patterns in different types of data sequences: natural language, DNA sequences, handwriting, word sequences, or numerical time series data streams from sensors and financial markets, to mention a few (Goldberg (2017); Quang & Xie (2016); Tsang et al. (2018)).

An LSTM neural network is organised in units called cells, performing transformations of the input sequence as shown in Fig. 5. An internal state variable is retained by an LSTM cell when forwarded from one cell to the next and is updated by the so-called Operation Gates (forget gate, input gate, output gate). All three gates have different and independent weights and biases, so that the network learns how much of the previous output and current input to retain and how much of the internal state to pass to the output. In an LSTM cell unit, the *cell state* brings information along the entire sequence and represents the memory of the network.

Figure 5: Scheme of an LSTM cell gate.



First, the input is passed to the *forget gate*, taking as input lagged values and deciding which fraction of the past information should be retained. The input

from the previous hidden state h_{t-1} and the current input x_t are transferred through the sigmoid function to the output gate:

$$f(t) = \sigma(x(t) * U_f + h(t-1) * W_f) \ . \tag{2}$$

An $f(t)$ output is close to 0 when a given piece of information can be forgotten, and 1 vice versa. The matrices W_f and U_f contain, respectively, the weights of the input and recurrent connections (the subscript f indicates the forget gate).

The second gate is the input gate. The previous hidden state and the current input are presented as inputs to a sigmoid activation function. To boost the network-tuning, they are also passed to the tanh function to compress values between -1 and 1. Then the output of the tanh and of the sigmoid are multiplied element by element (in Eq. (3) the symbol $*$ indicates the element by element multiplication of two matrices):

$$\begin{aligned} i_1(t) &= \sigma(x(t) * U_i + h(t-1) * W_i) \ , \\ i_2(t) &= \tanh(x(t) * U_g + h(t-1) * W_g) \ , \\ i(t) &= i_1(t) * i_2(t) \ . \end{aligned} \tag{3}$$

At this stage, the cell state is updated. Finally, the output gate specifies the value of the next hidden state including a certain amount of the information contained in the previous input. At this stage, the current input and the previous hidden state are summed up and forwarded to the sigmoid function: $C(t) = \sigma(f(t) * C(t-1) + i(t))$.

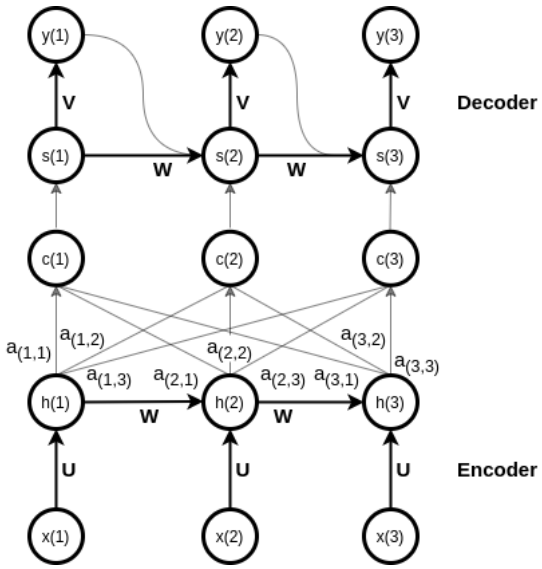
Our model consists of one stacked LSTM layer and a densely connected output layer with one neuron.

3.4.3. Attention Mechanism Neural Network

The attention mechanism is one of the key aspects of Deep Learning algorithms, specifically developed to improve the output on long input sequences (Niu et al. (2021)). Figure 6 shows the key idea behind the Attention Mechanism Neural Network (AMNN), which is to allow the decoder, during decoding, to access encoder's information selectively. This is achieved by creating new

context vectors for each decoder step, computed according to the previous hidden state as well as all encoder's hidden states, and assigning trainable weights to them. In this way, the attention technique assigns a higher priority to the most important inputs.

Figure 6: Attention Mechanism Neural Network scheme.



In the encoding step, the representation of each input sequence is determined as a function of the hidden state at the previous time step and the current input. The final hidden state $h(t)$ includes all encoded information from the previous hidden representations as well as the previous inputs.

Using the attention mechanism at each decoding step t, a new background vector $c(t)$ is computed. To compute $c(t)$, one calculates the so-called alignment scores $e(j,t)$:

$$e(j,t) = V_a * \tanh(U_a * s(t-1) + W_a * h(j)) \ , \tag{4}$$

where W_a, U_a and V_a are learning weights, also referred to as *attention weights*. The W_a weights are linked to the encoder's hidden states, the U_a weights to the decoder's hidden states, and the V_a weights determine the function that computes the alignment scores. The scores $e(j,t)$ are normalized at each time step

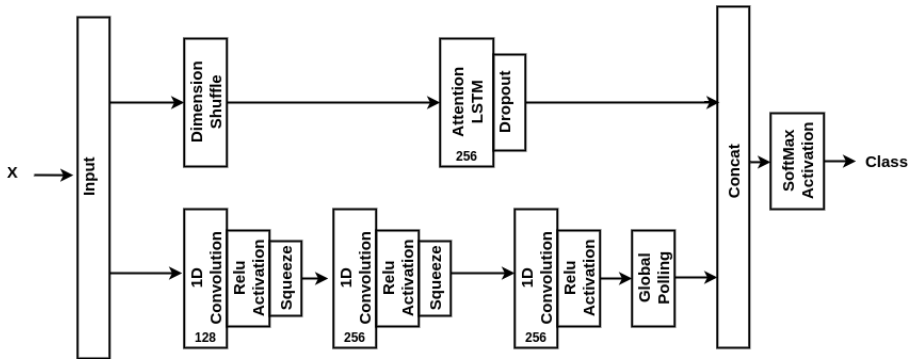
using the softmax function, obtaining the attention weights $\alpha(j,t)$ as follows:

$$\alpha(j,t) = \frac{\exp(e(j,t))}{\sum_{j=1}^{M} \exp(e(j,t))} \ . \tag{5}$$

The contextual vector $c(t)$ is now forwarded to the decoder to calculate the probability distribution of the next possible output. The softmax function is then used to calculate the output of the decoder.

Compared to LSTMs, the attention mechanism provides better results when processing long input sequences, thanks to the use of attention weights.

Figure 7: Attention LSTM cells to construct the MALSTM-FCN architecture (Karim et al. (2019)).



In this study, we specifically use a Multivariate Attention LSTM with Fully Convolutional Network (MALSTM-FCN) proposed and used in (Karim et al. (2019)) and (Karim et al. (2017)). Figure 7 shows the architecture for the MALSTM-FCN and specifies the number of neurons per layer. The input sequence is processed in parallel by a fully convolutional layer and an Attention LSTM layer, and it is concatenated and passed to the output layer via a softmax activation function for binary classification.

The fully convolutional block contains three temporal convolutional blocks of 128, 256 and 256 neurons respectively, used as feature extractors. Each convolutional layer is succeeded by batch normalisation, before the concatenation. The dimension shuffle transposes the temporal dimension of the input data, so that the LSTM is given the global temporal information of each variable at once.

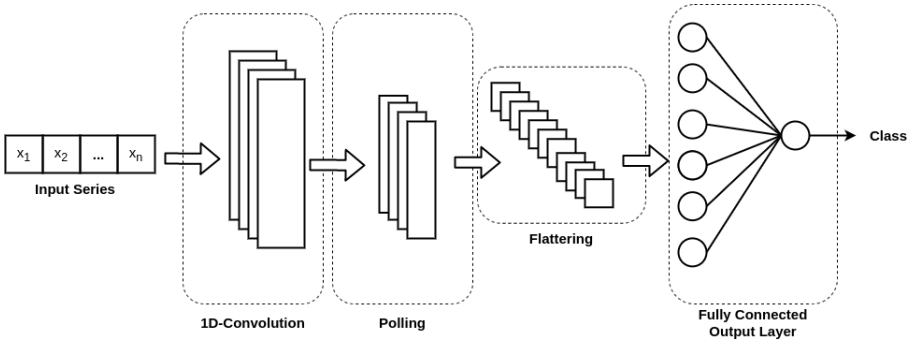
As a result, the dimension shuffle operation reduces the computational time for training and inference without losing accuracy in time series classification problems Karim et al. (2019).

3.4.4. Convolutional Neural Network

A Convolutional Neural Network (CNN) is a specific class of neural networks most commonly used for deep learning applications concerning image processing, image classification, natural language processing and financial time series analysis (Chen et al. (2016)).

The most critical part of the CNN architecture is the *convolutional* layer. This layer performs a *convolution*, i.e. a linear operation that involves a multiplication between a matrix of input data and a two-dimensional array of weights, known as a filter. These networks use the convolution operation in at least one of their layers.

Figure 8: Convolutional Neural Network architecture for time series forecasting.



Convolutional neural networks share a similar architecture with traditional neural networks, including an input and an output layer and multiple hidden layers. The main feature of a CNN is that its hidden layers typically consist of convolutional layers. Figure 8 depicts the general architecture of CNNs for time series analysis. We use a one-dimensional convolutional layer instead of the typical two-dimensional convolutional layer used in image processing tasks. This first layer is then normalised with a *polling layer* and later *flattened* so that

the output layer can process the whole time series at each step t. In this case, many one-dimensional convolution layers can be combined in a deep learning network.

Our model consists of two or more stacked 1-dimensional CNN layers, one densely connected layer with N neurons for *polling*, one densely connected layer with N neurons for *flattening*, and finally the densely connected output layer with one neuron.

3.5. Hyper-parameters tuning

The *hyper-parameters tuning* is a method for the optimisation of the hyper-parameters of a given algorithm. It is used to identify the optimal configuration of the hyper-parameters, within the given searching intervals, which would allow the algorithm to achieve the best performance, evaluated in terms of a specific prediction error. For each algorithm, the hyper-parameters to be optimised are selected, and for each hyper-parameter an appropriate searching interval is defined, including all values to be tested. The algorithm with the first chosen configuration of the hyper-parameters is, then, fitted on a specific portion of the dataset. The fitted model is tested on a portion of data that has not been previously used during the training phase. This testing procedure returns a specific value for the chosen prediction error.

The optimisation procedure via the `Grid Search` procedure (Lerman (1980)) ends when all possible combinations of hyper-parameter values have been tested. The hyper-parameter configuration yielding the best performance in terms of the selected prediction error is therefore chosen as the optimised configuration. Table 11 shows the hyper-parameters' searching intervals for each implemented algorithm. Since MALSTM-FCN is a deep neural network-specific architecture, the number of layers, neurons per layer and activation function of each layer are already pre-specified (as explained in Section 3.4.3).

To ensure the robustness of the hyper-parameter optimisation procedure, we use a model validation technique to assess how the performance achieved by a given model would generalise to an independent dataset. This validation

Table 11: Hyper-parameter searching intervals for different neural network architectures.

Algorithm	**Parameter**	**Searching Interval**
MLP	epochs	100, 250, 500, 1000
	hidden layers	1, 2, 3, 4, 5
	batch size	32, 64, 128, 256, 512
	optimizer	adam, Nadam, Adamax, RMSprop, SGD
	activation	relu, tanh, softmax
	neurons	16, 32, 64, 128, 256
LSTM	epochs	100, 250, 500, 1000
	hidden layers	1, 2, 3, 4, 5
	batch size	32, 64, 128, 256, 512
	optimizer	adam, Nadam, Adamax, RMSprop, SGD
	activation	relu, tanh
	neurons	16, 32, 64, 128, 256
MALSTM-FCN	epochs	100, 250, 500, 1000
	hidden layers	-
	batch size	32, 64, 128, 256, 512
	optimizer	adam, Nadam, Adamax, RMSprop, SGD
	activation	-
	neurons	-
CNN	epochs	100, 250, 500, 1000
	hidden layers	1, 2, 3, 4, 5
	batch size	32, 64, 128, 256, 512
	optimizer	adam, Nadam, Adamax, RMSprop, SGD
	activation	relu, tanh, softmax
	neurons	16, 32, 64, 128, 256

technique involves the partition of a data sample into a training set, used to fit the model, a validation set used to validate the fitted model, and a test set to assess the final optimised generalisation power of the model. In our analysis, we implemented the *Boostrap Method* (Efron & Tibshirani (1985)) with 37.8% of *out-of-bag samples* and 10000 iterations to validate the final hyper-parameters.

3.5.1. Hyper-Parameters For The Restricted Model

We briefly discuss here the fine-tuning of the hyper-parameters of the four deep learning algorithm mentioned in Section 3.5 considering the hourly frequency resolution. Table 12 shows the best results obtained for the different neural networks models, using the `Grid Search` technique in terms of the classification error metrics. The best identified parameters with the related results obtained for the *MALSTM-FNC* and *MLP* models are reported in Table 12.

The neural network that achieved the best accuracy is *MALSTM-FNC*, with an average accuracy of 53.7% and a standard deviation of 2.9%. Among the implemented machine learning models, the one that achieved the best f1-score is again *MALSTM-FNC*, with an average accuracy of 54% and a standard deviation of 2.01% (the *LSTM* obtained the same f1-score but we observe a higher variance).

3.5.2. Hyper-Parameters For The Unrestricted Model

Table 13 shows the best results obtained for the Neural Networks models, via the `Grid Search` technique with respect to the classification error metrics. The best identified parameters with the related results obtained for the *CNN* and *LSTM* models are reported in Table 13.

4. Results

In this section, we report the main results of the analysis. In particular, we discuss the outcome of the classification task for both the restricted and unrestricted model. These results are evaluated in terms of the standard classification error metrics: `accuracy`, `f1_score`, `precision` and `recall`.

Table 12: Restricted model - Neural networks optimal parameters.

Algorithm	**Parameter**	**Values**	**Accuracy** ($\mu \pm \sigma$)	**Prediction** ($\mu \pm \sigma$)	**Recall** ($\mu \pm \sigma$)	**f1-score** ($\mu \pm \sigma$)
MLP	epochs	250	0.537 ± 0.029	0.472 ± 0.143	0.511 ± 0.025	0.495 ± 0.027
	hidden layers	2				
	batch size	256,				
	optimizer	Nadam				
	activation	relu				
	neurons	128				
LSTM	epochs	250	0.535 ± 0.034	0.456 ± 0.200	0.485 ± 0.082	0.503 ± 0.285
	hidden layers	2				
	batch size	256,				
	optimizer	Adamx				
	activation	tanh				
	neurons	256				
MALSTM-FCN	epochs	250	0.542 ± 0.034	0.456 ± 0.200	0.485 ± 0.082	0.503 ± 0.201
	hidden layers	-				
	batch size	256,				
	optimizer	Adamx				
	activation	-				
	neurons	-				
CNN	epochs	250	0.435 ± 0.024	0.486 ± 0.210	0.485 ± 0.082	0.453 ± 0.265
	hidden layers	2				
	batch size	128,				
	optimizer	Nadam				
	activation	tanh				
	neurons	128				

Table 13: Unrestricted model - Neural networks optimal parameters.

Algorithm	**Parameter**	**Values**	**Accuracy** ($\mu \pm \sigma$)	**Prediction** ($\mu \pm \sigma$)	**Recall** ($\mu \pm \sigma$)	**f1-score** ($\mu \pm \sigma$)
MLP	epochs	500	0.81 ± 0.025	0.984 ± 0.180	0.541 ± 0.060	0.698 ± 0.908
	hidden layers	3				
	batch size	256				
	optimizer	Nadam				
	activation	relu				
	neurons	128				
LSTM	epochs	1000	0.86 ± 0.027	0.918 ± 0.033	0.873 ± 0.228	0.895 ± 0.175
	hidden layers	3				
	batch size	256				
	optimizer	Adamx				
	activation	tanh				
	neurons	256				
MALSTM-FCN	epochs	500	0.73 ± 0.027	0.75 ± 0.241	0.611 ± 0.175	0.673 ± 0.060
	hidden layers	-				
	batch size	256				
	optimizer	Adamx				
	activation	-				
	neurons	-				
CNN	epochs	1000	0.87 ± 0.027	0.782 ± 0.175	0.913 ± 0.060	0.842 ± 0.228
	hidden layers	2				
	batch size	256				
	optimizer	Nadam				
	activation	relu				
	neurons	256				

H1: Using a mix of trading and social media indicators leads to better cryptocurrencies price classification.

The results obtained for the unrestricted model highlight that the addition of trading and social media indicators to the model leads to an effective improvement in average accuracy. This result is consistent across all implemented algorithms. We can, therefore, rule out that this result is merely due to statistical fluctuations or that it may be an artefact of the particular classification algorithm implemented. The best result obtained with the unrestricted model is achieved using the *CNN* architecture, with a mean accuracy of 87% and a standard deviation of 2.7%.

Table 14 shows the results obtained using the four deep learning algorithms for the **hourly** frequency price movements classification task. This table presents the results for both the restricted (upper part) and unrestricted (lower part) model. First, it can be noted that for all four deep learning algorithms, the unrestricted model outperforms the restricted model in terms of accuracy, precision, recall and f1-score. The accuracy ranges from 51% for the restricted MLP to 84% for CNNs and LSTMs.

The fact that the result is consistent across all four classifiers, further confirms that this outcome is indeed due to the higher predictive power of the unrestricted model. For Bitcoin, the highest performances are obtained using the CNN architecture and for Ethereum by the LSTM.

We have also further explored the classification via the unrestricted model at hourly frequency considering two sub-models: a sub-model including technical and social indicators and the other including all the indicators (social, technical and trading). In this way, it is possible to disentangle the impact of social and trading indicators on the models' performance. We used a statistical t-test on the distributions of accuracy, prediction, recall and f1-score for the two unrestricted sub-modules finding that adding social indicators does not add a significant improvement to the unrestricted model. For this reason, in Table 14 we omitted the unrestricted model including social and technical indicators only.

Table 14: Accuracy, Precision, Recall, F1 score for Restricted and Unrestricted models for each Deep Learning Algorithm at Hourly Frequency.

Model	Algorithm	Cryptocurrency	Class	Accuracy	Precision	Recall	F1-score
Restricted	MLP	Bitcoin	down		0.57	0.28	0.38
			up		0.54	0.80	0.64
			average	0.55	0.56	0.55	0.51
		Ethereum	down		0.52	0.77	0.62
			up		0.55	0.29	0.38
			average	0.53	0.54	0.53	0.50
	MALSTM-FNC	Bitcoin	down		0.52	0.50	0.51
			up		0.55	0.57	0.56
			average	0.54	0.54	0.54	0.54
		Ethereum	down		0.52	0.80	0.63
			up		0.57	0.26	0.36
			average	0.53	0.54	0.53	0.50
	LSTM	Bitcoin	down		0.49	0.29	0.37
			up		0.53	0.73	0.61
			average	0.52	0.51	0.52	0.49
		Ethereum	down		0.51	0.70	0.59
			up		0.51	0.31	0.39
			average	0.51	0.51	0.51	0.49
	CNN	Bitcoin	down		0.52	0.65	0.57
			up		0.56	0.42	0.48
			average	0.53	0.54	0.53	0.53
		Ethereum	down		0.50	0.75	0.60
			up		0.56	0.31	0.40
			average	0.52	0.53	0.52	0.49
Unrestricted	MLP	Bitcoin	down		0.87	0.57	0.69
			up		0.70	0.92	0.79
			average	**0.75**	0.78	0.75	0.74
		Ethereum	down		0.80	0.79	0.80
			up		0.80	0.80	0.80
			average	**0.80**	0.80	0.80	0.80
	MALSTM-FNC	Bitcoin	down		0.97	0.32	0.48
			up		0.61	0.99	0.75
			average	**0.67**	0.78	0.67	0.62
		Ethereum	down		0.98	0.15	0.27
			up		0.54	1.00	0.70
			average	**0.58**	0.76	0.58	0.49
	LSTM	Bitcoin	down		0.79	0.90	0.84
			up		0.88	0.76	0.82
			average	**0.83**	0.84	0.83	0.83
		Ethereum	down		0.79	0.91	0.84
			up		0.90	0.76	0.83
			average	**0.84**	0.84	0.84	0.83
	CNN	Bitcoin	down		0.82	0.87	0.84
			up		0.87	0.82	0.85
			average	**0.84**	0.84	0.84	0.84
		Ethereum	down		0.72	0.97	0.83
			up		0.95	0.61	0.74
			average	**0.79**	0.83	0.79	0.78

Table 15: Accuracy, Precision, Recall, F1 score for Restricted and Unrestricted models for each Deep Learning Algorithm at Daily Frequency.

Model	Features	Algotithm	Cryptocurrency	Class	Accuracy	Precision	Recall	F1-score
Restricted	technical	MLP	Bitcoin	down		0.00	0.00	0.00
				up		0.59	0.96	0.73
				average	**0.58**	**0.36**	**0.58**	**0.44**
			Ethereum	down		0.96	1.00	0.98
				up		1.00	0.96	0.98
				average	**0.98**	**0.98**	**0.98**	**0.98**
		MALSTM-FNC	Bitcoin	down		0.51	0.47	0.49
				up		0.56	0.59	0.58
				average	**0.54**	**0.54**	**0.54**	**0.54**
			Ethereum	down		1.00	0.99	0.99
				up		0.99	1.00	0.99
				average	**0.99**	**0.99**	**0.99**	**0.99**
		LSTM	Bitcoin	down		0.00	0.00	0.00
				up		0.57	1.00	0.73
				average	**0.57**	**0.33**	**0.57**	**0.41**
			Ethereum	down		0.98	0.98	0.98
				up		0.99	0.99	0.99
				average	**0.99**	**0.99**	**0.99**	**0.99**
		CNN	Bitcoin	down		0.38	0.10	0.16
				up		0.60	0.89	0.72
				average	**0.58**	**0.51**	**0.58**	**0.50**
			Ethereum	down		0.88	1.00	0.94
				up		1.00	0.88	0.94
				average	**0.94**	**0.94**	**0.94**	**0.94**
Unrestricted	technical + social	MLP	Bitcoin	down		0.59	0.21	0.31
				up		0.61	0.90	0.72
				average	**0.60**	**0.60**	**0.60**	**0.55**
			Ethereum	down		0.79	0.95	0.87
				up		0.95	0.79	0.87
				average	0.87	0.88	0.87	0.87
		MALSTM-FNC	Bitcoin	down		0.41	0.41	0.41
				up		0.51	0.51	0.51
				average	**0.46**	**0.46**	**0.46**	**0.46**
			Ethereum	down		0.72	0.70	0.71
				up		0.77	0.78	0.77
				average	**0.75**	**0.75**	**0.75**	**0.75**
		LSTM	Bitcoin	down		0.44	0.10	0.17
				up		0.47	0.86	0.60
				average	**0.46**	**0.45**	**0.46**	**0.38**
			Ethereum	down		0.88	0.83	0.85
				up		0.87	0.91	0.89
				average	**0.87**	**0.87**	**0.87**	**0.87**
		CNN	Bitcoin	down		0.42	0.47	0.44
				up		0.56	0.52	0.54
				average	**0.50**	**0.50**	**0.50**	**0.50**
			Ethereum	down		0.77	0.83	0.80
				up		0.85	0.79	0.82
				average	**0.81**	**0.81**	**0.81**	**0.81**
	technical + social + trading	MLP	Bitcoin	down		0.59	0.20	0.30
				up		0.47	0.84	0.60
				average	**0.49**	**0.54**	**0.49**	**0.43**
			Ethereum	down		0.84	0.91	0.87
				up		0.91	0.84	0.87
				average	**0.87**	**0.88**	**0.87**	**0.87**
		MALSTM-FNC	Bitcoin	down		0.41	0.41	0.41
				up		0.62	0.62	0.62
				average	**0.54**	**0.54**	**0.54**	**0.54**
			Ethereum	down		0.79	0.88	0.83
				up		0.91	0.83	0.87
				average	**0.85**	**0.86**	**0.85**	**0.85**
		LSTM	Bitcoin	down		0.44	0.31	0.36
				up		0.43	0.58	0.49
				average	**0.44**	**0.44**	**0.44**	**0.43**
			Ethereum	down		0.92	0.87	0.89
				up		0.86	0.91	0.88
				average	**0.89**	**0.89**	**0.89**	**0.89**
		CNN	Bitcoin	down		0.52	0.55	0.54
				up		0.62	0.59	0.60
				average	**0.57**	**0.57**	**0.57**	**0.57**
			Ethereum	down		0.92	0.85	0.89
				up		0.87	0.93	0.90
				average	**0.89**	**0.90**	**0.89**	**0.89**

Table 15 shows the results obtained by the four deep learning algorithms price movements' classification at the **daily** frequency. This table presents results for both the restricted (upper part) and unrestricted (lower part) model. The unrestricted model is further divided in *technical-social* and *technical-social-trading* sub-models to better highlight the contribution of social and trading indicators to the model separately.

The MALSTM-CNF achieves the best classification performance for Ethereum with 99% of accuracy using the restricted model composed of only technical indicators. For Bitcoin, the best results are achieved by MLP with an f1-score of 55% and accuracy of 60% with the unrestricted model with **only social media indicators** and technical indicators. In this case, we consider f1-score and accuracy for Bitcoin because of the presence of a slightly unbalanced class distribution as described in Section 3.3.

For the daily frequency classification, we can see that, in general, technical indicators alone performs best in the classification of next day price movement. The more indicators we add to the model, the more the performance decreases. Another general result is that the accuracy, precision, recall and f1-score for the daily classification of Ethereum price movements are far better than those of Bitcoin. The results for price classification are in line with other recent studies (Akyildirim et al. (2020)) for both the hourly and daily classification, with a significant improvement when considering the hourly unrestricted model. The social media indicators turn out to be particularly relevant at the daily frequency for the Bitcoin case. This result is in agreement with the recent findings on the impact social media sentiment on cryptocurrency markets (Bartolucci et al. (2020)): the effects of social media on markets show a long lag, which is not captured nor relevant at an hourly frequency.

H2: Cryptocurrency markets' price classification are improved using social media indicators during sub-periods of financial distress.

We split the price classification problem in three sub-periods of known crypto markets distress for both Bitcoin and Ethereum. We constructed the differences in f1-score distributions for the four algorithms between *technical-trading* indi-

cators versus *technical-social* indicators.

For the *H2* hypothesis we constructed the following test, using the `Wilcoxon Signed Rank Test` for validation at 5% level of significance:

- H_0 *: the median difference of f1-scores is zero.*
- H_1 *: the median difference of f1-scores is negative.*

Table 16 shows the selected sub-periods of financial distress and the observed p-values along with the test statistics. For the selected sub-periods we reject the null hypothesis at 5% level of significance, i.e., we observe an improvement in f1 score using technical and social indicators. Although the introduction social media indicators did not induce a significant improvement when considering the whole period of analysis, we can clearly see a significant improvement when we restrict to known periods of financial distress in the crypto markets.

Table 16: Wilcoxon Signed Rank Test results for f1-score distribution differences of *technical-trading* versus *technical-social* indicators.

Cryptocurrency	From	To	W-Statistic	p-value
Bitcoin	2018/06	2018-12	36.0	0.005
	2019/04	2019/12	1.0	0.008
	2020/11	2021/02	1.0	0.0058
Ethereum	2018/09	2018/12	9.0	0.0058
	2019/04	2019/12	32.0	0.027
	2017/11	2018/06	30.0	0.027

5. Discussion and conclusion

Several attempts have been made in the most recent literature to model and predict the erratic behaviour of prices or other market indicators of the major cryptocurrencies. Notwithstanding massive efforts devoted to this goal by many research groups, the analysis of cryptocurrency markets still remains one of the

most debated and elusive tasks. Several aspects make grappling with this issue so complicated. For instance, due to its relatively young age, the cryptocurrency market is very dynamic and fast-paced. The emergence of new cryptocurrencies is a routine event, resulting in unexpected and frequent changes in the makeup of the market itself. Moreover, the high price volatility of cryptocurrencies and their 'virtual' nature are at the same time a blessing for investors and traders, and a curse for any serious theoretical and empirical modelling, with huge practical implications. The study of such a young market, whose price behaviour is still largely unexplored, has fundamental repercussions not only in the scientific arena but also for investors and main players and stakeholders in the crypto-market landscape.

In this paper, we aimed to assess whether the addition of social and trading indicators to the "classic" technical variables would lead to practical improvements in the classification of price changes of cryptocurrencies considering hourly and daily frequencies. This goal was achieved implementing and benchmarking a wide array of deep learning techniques, such as *Multi-Layer Perceptron* (MLP), *Multivariate Attention Long Short Term Memory Fully Convolutional Network* (MALSTM-FCN), *Convolutional Neural Network* (CNN) and *Long Short Term Memory* (LTMS) neural networks. We considered in our analysis the two main cryptocurrencies, Bitcoin and Ethereum, and we analysed two models: a restricted model, considering only technical indicators, and an unrestricted model that includes social and trading indicators.

In the restricted analysis, the model that achieved the best performance, in terms of accuracy, precision, recall, and f1-score, is MALSTM-FCN with an average f1-score of 54% for Bitcoin, and the CNN for Ethereum with hourly frequency. For the unrestricted case, the best result is achieved by the LSTM neural network for both Bitcoin and Ethereum with an average accuracy of 83% and 84% respectively. The most important finding for the hourly frequency classification for the unrestricted model is that the addition of trading and social indicators to the model leads to an effective improvement in the average accuracy, precision, recall, and f1-score. We have verified that this finding is not the

result of a statistical fluctuation, since all the implemented models yielded the same achievements. For the same reason, we can exclude that the results depend on the particular implemented algorithm. Finally, for the daily classification, the best classification performance – in line with recent research (Akyildirim et al. (2020))– has been achieved by MALSTM-CNF for Ethereum with 99% of accuracy when using the restricted model including only technical indicators. The only caveat here is that we predict the *direction* of price movement and not its *magnitude*. For Bitcoin, the best results are achieved by MLPs with f1-score of 55% and accuracy of 60% with the unrestricted model including social media indicators and technical indicators: in this case, we consider f1-score and accuracy for Bitcoin because of the slightly unbalanced class distribution. For the daily frequency classification, we can see that in general technical indicators alone perform better in the classification of next day price movements. The more indicators we add to the model, the more the performance decreases. Another general result is that the accuracy, precision, recall, and f1-score for daily classification of Ethereum price movements are far better than those for Bitcoin.

Our results show that with a specific design and fine-tuning of deep learning architecture, it is possible to achieve high performance in the classification of price changes of cryptocurrencies. We have also shown the importance and effect of social media indicators in the context of cryptocurrency price prediction, providing further evidence of the strong links between markets and social communities in the crypto ecosystem.

Competing Interests The authors declare no competing interests.

Author contributions

Marco Ortu: Conceptualization, Methodology, Writing-Reviewing and Editing. **Nicola Uras:** Software, Writing-Reviewing and Editing, Data Curation. **Claudio Conversano:** Writing-Reviewing and Editing. **Silvia Bartolucci:** Conceptualization, Writing-Reviewing and Editing. **Giuseppe Destefanis:** Conceptualization, Writing-Reviewing and Editing.

References

Akyildirim, E., Goncu, A., & Sensoy, A. (2020). Prediction of cryptocurrency returns using machine learning. *Annals of Operations Research*, (pp. 1–34).

Akyildirim, E., Goncu, A., & Sensoy, A. (2021). Prediction of cryptocurrency returns using machine learning. *Annals of Operations Research*, *297*, 3–36.

Bartolucci, S., Destefanis, G., Ortu, M., Uras, N., Marchesi, M., & Tonelli, R. (2020). The butterfly "affect": Impact of development practices on cryptocurrency prices. *EPJ Data Science*, *9*, 21.

Brownlee, J. (2018). *Deep learning for time series forecasting: predict the future with MLPs, CNNs and LSTMs in Python*. Machine Learning Mastery.

Calefato, F., Lanubile, F., Maiorano, F., & Novielli, N. (2017). Sentiment polarity detection for software development. *Empirical Software Engineering*, (pp. 1–31).

Calefato, F., Lanubile, F., Maiorano, F., & Novielli, N. (2018). [journal first] sentiment polarity detection for software development. In *2018 IEEE/ACM 40th International Conference on Software Engineering (ICSE)* (pp. 128–128). IEEE.

Chen, C. Y.-H., & Hafner, C. M. (2019). Sentiment-induced bubbles in the cryptocurrency market. *Journal of Risk and Financial Management*, *12*, 53.

Chen, J.-F., Chen, W.-L., Huang, C.-P., Huang, S.-H., & Chen, A.-P. (2016). Financial time-series data analysis using deep convolutional neural networks. In *2016 7th International conference on cloud computing and big data (CCBD)* (pp. 87–92). IEEE.

Chollet, F. et al. (2015). Keras, github repository `https://github.com/keras-team/keras`.

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, .

Efron, B., & Tibshirani, R. (1985). The bootstrap method for assessing statistical accuracy. *Behaviormetrika*, *12*, 1–35.

Giudici, P., & Polinesi, G. (2019). Crypto price discovery through correlation networks. *Annals of Operations Research*, (pp. 1–15).

Goldberg, Y. (2017). Neural network methods for natural language processing. *Synthesis lectures on human language technologies*, *10*, 1–309.

Gomez-Carrasco, P., & Michelon, G. (2017). The power of stakeholders' voice: The effects of social media activism on stock markets. *Business Strategy and the Environment*, *26*, 855–872.

Hartmann, F., Grottolo, G., Wang, X., & Lunesu, M. I. (2019). Alternative fundraising: success factors for blockchain-based vs. conventional crowdfunding. In *2019 IEEE international workshop on blockchain oriented software engineering (IWBOSE)* (pp. 38–43). IEEE.

Hartmann, F., Wang, X., & Lunesu, M. I. (2018). Evaluation of initial cryptoasset offerings: the state of the practice. In *2018 International Workshop on Blockchain Oriented Software Engineering (IWBOSE)* (pp. 33–39). IEEE.

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

Ivakhnenko, A. G., Ivakhnenko, A. G., Lapa, V. G., & Lapa, V. G. (1967). *Cybernetics and forecasting techniques* volume 8. American Elsevier Publishing Company.

Jing-Zhi H., J. N., William H. (2018). Predicting bitcoin returns using high-dimensional technical indicators. *The Journal of Finance and Data Science*, . URL: http://www.sciencedirect.com/science/article/pii/S2405918818300928. doi:https://doi.org/10.1016/j.jfds.2018.10.001.

Jones, C. M., Reed, A. V., & Waller, W. (2021). When brokerages restrict retail investors, does the game stop? *Does the Game Stop*, .

Karim, F., Majumdar, S., Darabi, H., & Chen, S. (2017). Lstm fully convolutional networks for time series classification. *IEEE access*, *6*, 1662–1669.

Karim, F., Majumdar, S., Darabi, H., & Harford, S. (2019). Multivariate lstm-fcns for time series classification. *Neural Networks*, *116*, 237–245.

Katsiampa, P. (2017). Volatility estimation for bitcoin: A comparison of garch models. *Economics Letters*, *158*, 3–6.

Keskin Z., A. T. (2019). Information-theoretic measures for non-linear causality detection: application to social media sentiment and cryptocurrency prices. *arXiv:1906.05740*, .

Lahmiri, S., Bekiros, S., & Salvi, A. (2018). Long-range memory, distributional variation and randomness of bitcoin volatility. *Chaos, Solitons & Fractals*, *107*, 43–48.

Lahmiri Salim, B. S. (2019). Cryptocurrency forecasting with deep learning chaotic neural networks. *Chaos, Solitons and Fractals*, *118*, 35 – 40.

Lerman, P. (1980). Fitting segmented regression models by grid search. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, *29*, 77–84.

Mäntylä, M., Adams, B., Destefanis, G., Graziotin, D., & Ortu, M. (2016). Mining valence, arousal, and dominance: possibilities for detecting burnout and productivity? In *Proceedings of the 13th international conference on mining software repositories* (pp. 247–258).

Marchesi, L., Marchesi, M., Destefanis, G., Barabino, G., & Tigano, D. (2020). Design patterns for gas optimization in ethereum. In *2020 IEEE International Workshop on Blockchain Oriented Software Engineering (IWBOSE)* (pp. 9–15). IEEE.

Matta, M., Lunesu, I., & Marchesi, M. (2015). Bitcoin spread prediction using social and web search media. In *UMAP workshops* (pp. 1–10).

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, .

Murgia, A., Ortu, M., Tourani, P., Adams, B., & Demeyer, S. (2018). An exploratory qualitative and quantitative analysis of emotions in issue report comments of open source systems. *Empirical Software Engineering*, *23*, 521–564.

Niu, Z., Zhong, G., & Yu, H. (2021). A review on the attention mechanism of deep learning. *Neurocomputing*, *452*, 48–62.

Ortu, M. (2015). Mining software repositories: measuring effectiveness and affectiveness in software systems., .

Ortu, M., Hall, T., Marchesi, M., Tonelli, R., Bowes, D., & Destefanis, G. (2018). Mining communication patterns in software development: A github analysis. In *Proceedings of the 14th International Conference on Predictive Models and Data Analytics in Software Engineering* (pp. 70–79).

Ortu, M., Orrú, M., & Destefanis, G. (2019). On comparing software quality metrics of traditional vs blockchain-oriented software: An empirical study. In *2019 IEEE International Workshop on Blockchain Oriented Software Engineering (IWBOSE)* (pp. 32–37). IEEE.

Phillips, R. C., & Gorse, D. (2018). Mutual-excitation of cryptocurrency market returns and social media topics. In *Proceedings of the 4th International Conference on Frontiers of Educational Technologies* (pp. 80–86). ACM.

Quang, D., & Xie, X. (2016). Danq: a hybrid convolutional and recurrent deep neural network for quantifying the function of dna sequences. *Nucleic acids research*, *44*, e107–e107.

Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to sequence learning with neural networks. In *Advances in neural information processing systems* (pp. 3104–3112).

Tsang, G., Deng, J., & Xie, X. (2018). Recurrent neural networks for financial time-series modelling. In *2018 24th International Conference on Pattern Recognition (ICPR)* (pp. 892–897). IEEE.

Uras, N., & Ortu, M. (2021). Investigation of blockchain cryptocurrencies' price movements through deep learning: A comparative analysis. In *2021 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER)* (pp. 715–722). IEEE.

Warriner, A. B., Kuperman, V., & Brysbaert, M. (2013). Norms of valence, arousal, and dominance for 13,915 english lemmas. *Behavior research methods*, *45*, 1191–1207.