Insert Title Here

by

PARBHU Varun

Project Supervisor: Professor (Dr) BHURUTH MUDDUN OSK

Project submitted to the Department of Mathematics,
Faculty of Science, University of Mauritius,
as a partial fulfillment of the requirement
for the degree of
M.Sc. Mathematical and Scientific Computing (Part time)

December 2022

Contents

List of Figures

List of Tables

Acknowledgement

Placeholder

Abstract

Placeholder

Terms and Definitions

α	Learning Rate
η	Reduced Learning Rate
$\mathbf{I}$	Identity Matrix
$\odot$	Hadamard Product
NN	Neural Network

Chapter 1

Introduction

Introduction

Chapter 2

Sentiment Analysis

People's opinions, feelings and sentiments towards entities such products, services, other people, events, news, issues, topics, etc. can be in very large volume, complex and difficult to be understood and processed by machines and computers. Thus, sentiment analysis, also known as opinion mining, started to popularised along the rise of social media when large amount of digital text data were suddenly available for mining. Natural language processing (NLP) helps computers process and understand human based language to perform repetitive task. Sentiment analysis is a niche of NLP. It aims at quantifying the positivity, negativity and/or neutrality of implied or expressed in a given text.

Social media have been providing large platforms for people to share their opinion freely and expressed their views on any subject across various geographical and spatial boundaries. They have also allowed people to connect, influence and be influenced by such opinions and views. These interactions have been studied in the 1940s and 1950s among people in organizations by management science researchers. Since 2002, with social media, those studies have been performed at grand scales with the abundance of data. Thus, advanced sentiment analysis research have been performed in field political science, economics, finance and management science as they are heavily dependent on public opinions.

Asur and Huberman attempted to solve the revenue prediction problem using both the tweet volume and the tweet sentiment (Asur & Huberman 2010). Same kind of data along with polling results were also used, in (Bermingham & Smeaton 2011),

to train a linear regression model to predict election results. (Zhang, Fuehres & Gloor 2011) leveraged sentiments on Twitter to help predict the movement of stock market indices such as Dow Jones, S&P500 and NASDAQ. It was shown that large negative emotions and opinions caused Dow to go down the next day and lower negative emotions cause Dow to go up on the next day. (Bar-Haim, Dinur, Feldman, Fresko & Goldstein 2011) on one hand leverage sentiments on Twitter but on the other did not treat all Twitter authors equally. Only expert investors were used as features in training stock price movement predictors. Undeniably, modern social media (started early 2000s) have grown into a major influencer of human opinion and sentiments.

2.1 Rule-Based Methods

Conventional NLP techniques relied on a set of rules to process textual data to extract opinion, polarity, topic, and other information within the data. Tokenization, part-of-speech tagging, lemmatization and removal of stop words are some rules and technique used to processed data prior to analysis. Tokenization is the separation/breaking down of text data into words. The space between words in a document is commonly used as the delimiter of separation. Consider the example below:

"The price of Bitcoin wasn't great today!" (2.1)

Tokenizing the result above results in the list:

['The', 'price', 'of', 'Bitcoin', 'was', 'n't', 'great', 'today', '!'] (2.2)

In sentiment analysis, the tokenized sentence would be compared to predefined list polarized words (positive and negative). A naive way to get the polarity of the sentence would be to count the number of positive and negative words in the predefined polarized lists. The limitations are quite obvious, we are not taking into account the context in which the words are used, nor the preceding words. Part-of-speech

tagging refers to the classification of the token in a document based on predefined assignment. Tokens are classified as adjective, adverb, noun, verb, etc. (the full list can be obtained through the Universal POS tags). POS tagging is used to describe the syntactic structuring of a sentence.

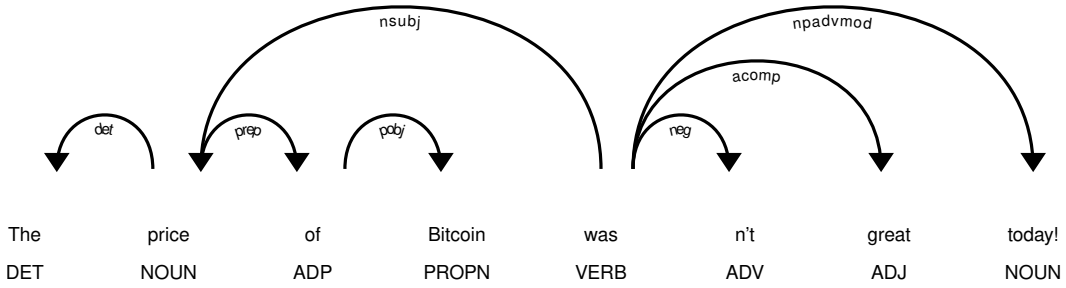

Figure 2.1: Part-of-speech tagging of sentence at 2.1 using spaCy

Lemmatization refers to the process of reducing words to their base form, also known as lemma. For example, the words below can be reduced as follows:

$$\text{great} \rightarrow \text{good}$$

$$\text{happiness} \rightarrow \text{happy}$$

$$\text{stemming} \rightarrow \text{stem}$$

It allows to map the meaning of multiple words at one time by reducing the former to its base. The assumption is that both form retain the same meaning syntactically. Moreover, the removal of stop words is also often perform to eliminate words that are neutral or bring no additional value within a sentence. Examples of such word would be "the", "a", "is", etc. Since rule-based models is performed individually over each word; the simple removal stop words reduced computation time of such models. However, such models are highly limited, computationally demanding and often inaccurate. Thus, with the rise of computation power available and machine learning, statistical and embedding based models were developed to tackle those limitations.

2.2 Word Embedding

Processing raw data from ruled based method is not ideal. A numerical representation of text data would be more appropriate for mathematical models to perform NLP. Word embedding is the representation of text data as vectors in a vector space. Words with similar meaning would be considered as very close to each other in such a space. Moreover, from linear algebra, we would expect that normalized vector of words with similar meaning to be close. Word embedding techniques are categorized among two types: frequency based embedding and prediction based embedding.

Frequency based embedding refers to the vectorization of words based on the frequency of occurrence of words in a document. The three types of frequency based embedding are: Count Vector, TF-IDF (Term Frequency-Inverse Term Frequency) Vector and Co-Occurence Matrix with a fixed context window. Such type of embedding results in very sparse vector of high dimensionality which are computationally challenging to manipulate.

Prediction based embedding refers to the prediction of a target word based on the context (neighboring words). Developed by (Mikolov, Chen, Corrado & Dean 2013), the word2vec was introduced. Word2vec is a word embedding method derived from two techniques: CBOW (Continuous Bag of Words) and Skip-Gram Model. The CBOW attempts to predict the probability of a word in the center from a given context while the Skip-Gram model is the opposite of the CBOW model; it tries to predict the context of words given a center word.

2.3 Transformer Model (Attention Mechanism)

Attention in neural network attempts to emulate the cognitive attention in the human brain. It focuses on only relevant part of the data while neglecting the rest. Developed by at team at Google Brain in 2017 (Vaswani et al. 2017), transformers are the current state-of-the-art attention techniques (instead of recurrence) for processing contextual data; text, audio, video, speech, etc. Similar to the RNN, LSTM and GRU, transformers process sequential data as well with the exception that transforms process all input at once thus allowing parallelization. Transformers

consist of two main parts: the encoder and the decoder. The encoder processes the input document, identifies the important text and creates an embedding for the text based on the importance of the latter to other words in the document. The decoder on the other hand tries to get the text back from the embedding created by the encoder.

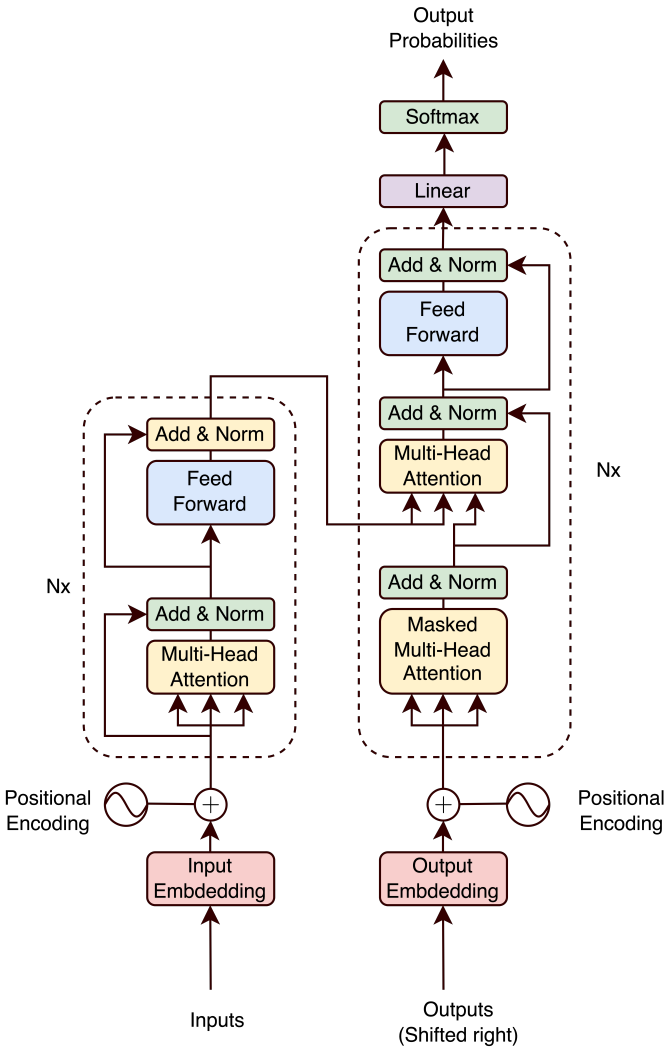

Figure 2.2: Transformer Layer (Vaswani et al. 2017)

The left part of the architecture is the encoder while the right side is the decoder. This structure allows the transformer model for parallelization computations instead of sequential (RNN, LSTM and GRU). Transformer models are now able to solve natural language processing problem with higher accuracy. We introduce two commonly used transformation models: BERT and RoBERTa.

Bidirectional Encoder Representations from Transformers (BERT) which was developed by Google (Devlin, Chang, Lee & Toutanova 2018). BERT uses only the encoder part of the transformer similar to the original transformer in (Vaswani et al. 2017).

BERT is able to create word representations that are dynamically influenced by the surrounding words, whereas word2vec has a fixed representation for each word independent of the context in which it occurs. Leading BERT to achieve multiple state-of-the-art NLP tasks when it was published, including at sentiment analysis (Chiorrini, Diamantini, Mircoli & Potena 2021) A number of pre-trained BERT models from unlabeled text are currently available (due to the large number of data and parameters required for training) and can be re-adapted using transfer learning.

RoBERTa stands for Robustly Optimized BERT Pretaining Approach was also developed by Google (Liu, Ott, Goyal, Du, Joshi, Chen, Levy, Lewis, Zettlemoyer & Stoyanov 2019). The approach found that BERT was currently under trained and could potentially do much better. RoBERTa uses an extensively large amount of hyperparameter, tuning and optimization. Additionally, much more data was also used during training. BERT used 16gb of data from CC-news while RoBERTa was trained on 160bg of data from CC-news (92gb), OpenWebText (38bg) and addition data from Stories (31gb) (Liu et al. 2019).

Chapter 3

Analysis of Bitcoin Price using Deep Learning Models

Bitcoin was the first cryptocurrency to be operational in January 2009 after the appearance of the mysterious paper titled "Bitcoin: A Peer-to-Peer Electronic Cash System", published in 2008 under the alias "Satoshi Nakamoto" (a person or group of people) (Nakamoto 2009). The paper described a peer-to-peer payment system using electronic currency (cryptocurrencies) that could be sent directly from one party to another without using a third party (often a financial institution) to validate the transaction. The main idea behind the paper was to emulate an online shared ledger on the peer-to-peer network to validate all transactions via a blockchain, eliminating the risk of forging the ledger. Hence, the rise of blockchain technology provides security, privacy and a distributed ledger applicable in IoT, distributed storage systems and many more. In exchange for maintaining the blockchain (run and validate), which is highly energy dependent on running electronic machines, "miners" are rewarded with cryptocurrency. Consequently, the value of such a type of currency is highly uncertain and dependent on many factors.

From basically nothing in January 2009 to reaching the highest price (known to date) of $67,566.83 on November 8, 2021, Bitcoin has been highly volatile compared to other traditional forms of FIAT payment. During that time, there have been substantial price changes over short periods. During 2017 the value of a single Bitcoin increased by 2000, going from $863 on January 9, 2017, to a high of $17,550

on December 11, 2017. By eight weeks later, on February 5, 2018, the price of a single Bitcoin had been more than halved, with a value of $7,9643 (Abraham, Higdon, Nelson & Ibarra 2018). Due to its volatility and never seen behaviour like traditional currencies, Bitcoin (and cryptocurrencies in general) are extremely difficult to predict.

Building on the work by Abraham on Cryptocurrency Price Prediction Using Tweet Volumes and Sentiment Analysis (Abraham et al. 2018), we gathered data from Twitter, Google Trends and Yahoo Finance about Bitcoin. That is, Tweets and Tweet Volume from Twitter, Search Value Index (SVI) from Google Trend, and Open, High, Low and Close value of Bitcoin hourly. We first explore the raw data gathered for any observable trend. Then, we perform feature engineering by cleaning the data, generating sentiment scores of Tweets using the state-of-art model roBERTa, identifying any potential influencers using K-Means neighbouring, performing hourly aggregations and finally scaling the data. We then modelled the resulting dataset in two different ways. First, we used an LSTM NN model to predict the Bitcoin closing prices for the next 24hr and, secondly, a Feedforward NN to predict any uptick of prices in the next hour.

3.1 Data Collection

In order to tackle the problem of prediction of Bitcoin prices and upticks, we gathered a large amount of data from different sources which we believe might contribute information to predict the price. We firstly consider the sentiment analysis of Tweets from Twitter as our first input. We then get the Search Value Index (SVI) of Bitcoin from Google Trends. Finally, we retrieved Bitcoin market prices from Yahoo Finance. All data are scrapped between 01 June 2022 and ending 13 July 2022.

Twitter

In order to get Tweets related to Bitcoin from Twitter, we leverage Tweepy - an open-source Python Library for accessing the Twitter API. The API has different access levels: Essential, Elevated and Academic Research. We requested Academic

Research access through the Twitter Developers Portal to get the appropriate API key and API token. Once the access was provided, we used the query features from the documentation to scrape Tweets with either the word "bitcoin" or hashtag bitcoin (#bitcoin). For each query, we collected the following information: username, user description, location, friends count, followers count, total tweets count, retweet count, hashtags in the tweets and the tweet.

```
query = 'bitcoin OR #bitcoin  lang:en -is:retweet'
```

Listing 3.1: Tweepy query for bitcoin, #bitcoin and excluding retweets

The request cap for academic access is 100 per 15 minutes. We set up our scrapping strategy for 10 consecutive days at a rate of 65 Tweets a minute to match the request cap. This strategy allowed us to scrap 814,818 tweets with the query mentioned above. Additionally, we used the same access to get the tweet volume pertaining to the query in listing 3.1 and predefined start time and end time.

```
query_params = {'query': query ,'granularity': 'hour', '
start_time': start_time, 'end_time': end_time}
```

Listing 3.2: Tweepy query to get tweet count withing each hour between predefined start time and end time

Google Trends

Google is a massive search engine dealing with a vast amount of data daily, including the volume of searches of people based on keywords. Google trend data is a reflection of the volume of searches that people do. However, the data is extensive to be consumed directly due to the number of global users. The trend data is anonymised (personally identifiable information removed), categorised (topic associated with each search query) and aggregated (grouped). The trend data provided by Google are indexed to 100, where 100 is the maximum search interest for the time and location selected. The combination of the pre-processing of raw volume searches provides a measure of interest in a particular topic across different regions or globally. We used the pytrends package - an unofficial open-source Python Library for accessing the Google trend data. We pass the least ambiguous keyword,

"bitcoin", into the library and retrieve the hourly search index for the said keyword.

```
kw_list = ['bitcoin']
search_df = pytrends.get_historical_interest(kw_list, year_start
    =2022,month_start=6, day_start=1, hour_start=0, year_end=2022,
    month_end=7, day_end=16, hour_end=0,cat=0, geo='', gprop='')
```

Listing 3.3: pytrend query to get google trend index withing each hour between predefined start time and end time

Yahoo Finance

In order to get Bitcoin prices, we used the yfinance package - an unofficial open-source Python Library for accessing market data on cryptocurrencies, regular currencies, stocks and bonds, fundamental and options data, and market analysis and news. We used the yfinance package to scrape the open, high, low and close prices and the volume of bitcoin traded hourly.

```
BTC_Ticker = yf.Ticker("BTC-USD")
BTC_Data_long = BTC_Ticker.history(start="2022-06-01", end="
    2022-07-16",interval= '1h')
```

Listing 3.4: yfinance query to get bitcoin price and traded volume within each hour between predefined start time and end time

We briefly explore the raw data scrapped from the different sources to get a better understanding of the latter. The tweets from Twitter had several issues. A large number contained URLs, symbols, excessive punctuations and were in different languages (even if the query was restricted to English tweets only).

Figure 3.1: Example of raw tweet scraped using the tweepy package through Python

The number of tweets at the end of each hour was continuous and had no major issues from the API response.

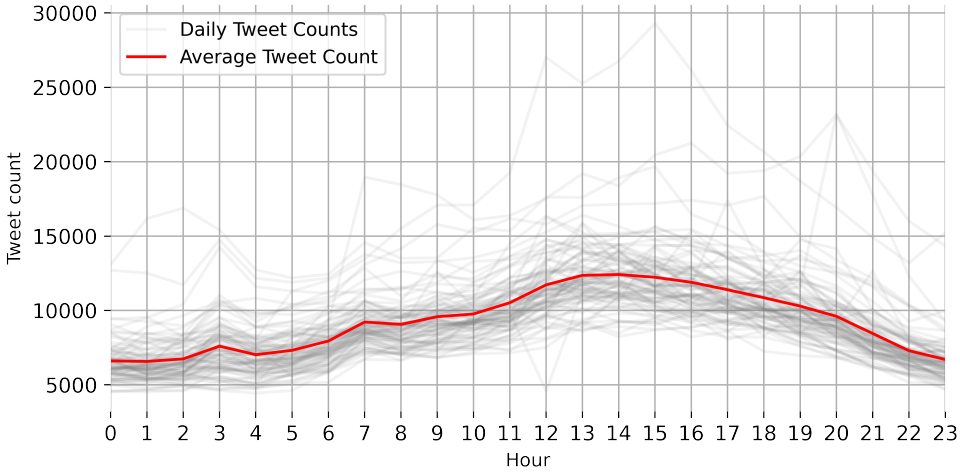

Figure 3.2: Distribution of tweet count over each day at the end of every hour over 24 hours

The data for the open, high, low, and close prices scrapped from Yahoo Finance was clean and had no issues. However, we did see that not all closing hours have the associated volume. Thus, volume was not included in the final dataset.

Figure 3.3: Bitcoin closing price over every hour for everyday

3.2 Feature Engineering

Feature engineering is the manipulation of our dataset to provide extra features to input in our model to improve accuracy. Manipulation includes addition, deletion, combination and transformation of the dataset. Practical feature engineering is dependent on the business problem and our objective. Common types of feature engineering include scaling and transformation, fitting missing values, feature coding, feature construction and feature extraction.

Sentiment Analysis on Tweet Data

In order to be able to quantify the tweets as features in our model, we transform the tweets into numerical values by performing sentiment analysis on them. However, as seen in the tweet example in listing (3.1), the raw tweet would be difficult for the existing natural language processing model to transform them. Using regular expression syntax in Python, we cleaned the tweets first.

```
def pre_process_tweet (text):
#Remove all links starting with http...
text = re.sub('https?:\/\/.*[\r\n]*','',text)
#Remove RT
text = re.sub('^RT[\s]+','',text)
#Remove @[User]
text = re.sub('@[^ ]+', '', text)
#Remove Punctuation first
text = text.translate(str.maketrans('','', string.punctuation))
#Convert text to lower case
text = text.lower()
#Remove new line
text = re.sub('\n',' ',text)
#Replace emojis with description
text = demoji.replace_with_desc(text,sep=' ')
#Reducing whitespaces to one everywhere
text = re.sub('\s+',' ',text)
return text
```

Listing 3.5: Python function to clean Tweets using RegEx (regular expression)

Applying the python function defined in listing 3.5 to the raw tweet in 3.1 leads to the following string:

> 'mushroom the psychonaut ape division mushroom too weird to live and too rare to die framed picture a bold collection of psychonaut apes by internationally recognised artist woahjonny nft nfts ethereum eth eth crypto solana bnb avax tezos tron bitcoin btc' (3.1)

We apply the pre-process tweet function across the 814,818 tweets that were scraped. Thus, making the tweets easier to process in sentiment analysis models. Once the tweets are clean, we use the RoBERTa model defined in Section 2.3 to perform sentiment analysis. We used the version *cardiffnlp/twitter-roberta-base-sentiment* which is made available Hugging Face. 'Twitter-roBERTa-base' for Sentiment Analysis is a pretrained on approximately 58 million tweets and fine-tuned for sentiment analysis (Barbieri, Camacho-Collados, Neves & Anke 2020). The output of the model are is the percentage of a tweet being negative, neural and positive.

```
1     encoded_text = tokenizer(preprocessed_tweet, return_tensors='pt
      ')
2     output = model(**encoded_text)
3     scores = output[0][0].detach().numpy()
4     scores = softmax(scores)
5     scores_dict = {'roberta_neg':scores[0], 'roberta_neu':scores
      [1], 'roberta_pos':scores[2]}
6     print(scores_dict)
7
8  >> {'roberta_neg':0.298603, 'roberta_neu':0.5974009, 'roberta_pos'
      :0.10399604}
```

Listing 3.6: Applying the twitter-roberta-base-sentiment model to the preprocessed tweet in (3.1)

We apply the same function through the whole preprocessed dataset to get the scraped tweets' scores. Tweets that were too long or had unknown characters that the model could not process were dropped (less than 0.5% of the tweet dataset).

Hourly Averaging

Processing the point data of all the scrapped tweets can be overly demanding in terms of computational power. This will result to an overfitting and not being able the model the general trend of a single day. Additionally, we do not have point data for price and google trends search index. To overcome this problem, we aggregated the sentiment scores (POS, NEU, NEG) over each hour across the time period. The resulting dataset was 1032×3 (24 hours for 43 days).

Data Encoding

3.3 Modelling Bitcoin Price using LSTM NN

Model Architecture and Hyperparameters

3.4 Modelling Bitcoin Upticks using Feedforward NN

Model Architecture and Hyperparameters

Chapter 4

Conclusion and Future Works

Bibliography

Abraham, J., Higdon, D. W., Nelson, J. & Ibarra, J. (2018), Cryptocurrency price prediction using tweet volumes and sentiment analysis.

Asur, S. & Huberman, B. (2010), 'Predicting the future with social media', *Proceedings - 2010 IEEE/WIC/ACM International Conference on Web Intelligence, WI 2010* **1**.

Bar-Haim, R., Dinur, E., Feldman, R., Fresko, M. & Goldstein, G. (2011), Identifying and following expert investors in stock microblogs, *in* 'Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing', Association for Computational Linguistics, Edinburgh, Scotland, UK., pp. 1310–1319.
URL: *https://aclanthology.org/D11-1121*

Barbieri, F., Camacho-Collados, J., Neves, L. & Anke, L. E. (2020), 'Tweeteval: Unified benchmark and comparative evaluation for tweet classification', *CoRR* **abs/2010.12421**.
URL: *https://arxiv.org/abs/2010.12421*

Bengio, Y., Simard, P. & Frasconi, P. (1994), 'Learning long-term dependencies with gradient descent is difficult', *IEEE Transactions on Neural Networks* **5**(2), 157–166.

Bermingham, A. & Smeaton, A. (2011), On using Twitter to monitor political sentiment and predict election results, *in* 'Proceedings of the Workshop on Sentiment Analysis where AI meets Psychology (SAAIP 2011)', Asian Federation of Nat-

ural Language Processing, Chiang Mai, Thailand, pp. 2–10.
URL: *https://aclanthology.org/W11-3702*

Chiorrini, A., Diamantini, C., Mircoli, A. & Potena, D. (2021), Emotion and sentiment analysis of tweets using bert.

Devlin, J., Chang, M., Lee, K. & Toutanova, K. (2018), 'BERT: pre-training of deep bidirectional transformers for language understanding', *CoRR* **abs/1810.04805**.
URL: *http://arxiv.org/abs/1810.04805*

Hinton, G. & Tieleman, T. (2012), 'Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude.', COURSERA: Neural Networks for Machine Learning, 4, 26-30.

Hochreiter, S. & Schmidhuber, J. (1997), 'Long Short-Term Memory', *Neural Computation* **9**(8), 1735–1780.
URL: *https://doi.org/10.1162/neco.1997.9.8.1735*

Kingma, D. P. & Ba, J. (2014), 'Adam: A method for stochastic optimization'.
URL: *https://arxiv.org/abs/1412.6980*

Kolen, J. F. & Kremer, S. C. (2001), *Gradient Flow in Recurrent Nets: The Difficulty of Learning LongTerm Dependencies*, pp. 237–243.

LeCun, Y., Bengio, Y. & Hinton, G. (2015), 'Deep learning', *Nature* **521**(7553), 436–444.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L. & Stoyanov, V. (2019), 'Roberta: A robustly optimized BERT pretraining approach', *CoRR* **abs/1907.11692**.
URL: *http://arxiv.org/abs/1907.11692*

McCulloch, W. S. & Pitts, W. (1943), 'A logical calculus of the ideas immanent in nervous activity', *The bulletin of mathematical biophysics* **5**(4), 115–133.

Mikolov, T., Chen, K., Corrado, G. & Dean, J. (2013), 'Efficient estimation of word representations in vector space'.
URL: *https://arxiv.org/abs/1301.3781*

Nakamoto, S. (2009), 'Bitcoin: A peer-to-peer electronic cash system', *Cryptography Mailing list at https://metzdowd.com* .

Robbins, H. & Monro, S. (1951), 'A Stochastic Approximation Method', *The Annals of Mathematical Statistics* **22**(3), 400 – 407.
URL: *https://doi.org/10.1214/aoms/1177729586*

Rosenblatt, F. (1958), 'The perceptron: a probabilistic model for information storage and organization in the brain.', *Psychol Rev* **65**(6), 386–408.

Rumelhart, D. E., Hinton, G. E. & Williams, R. J. (1986), 'Learning representations by back-propagating errors', *Nature* **323**(6088), 533–536.

Schuster, M. & Paliwal, K. (1997), 'Bidirectional recurrent neural networks', *IEEE Transactions on Signal Processing* **45**(11), 2673–2681.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. & Polosukhin, I. (2017), 'Attention is all you need'.
URL: *https://arxiv.org/abs/1706.03762*

Zhang, X., Fuehres, H. & Gloor, P. A. (2011), 'Predicting stock market indicators through twitter "i hope it is not as bad as i fear"', *Procedia - Social and Behavioral Sciences* **26**, 55–62. The 2nd Collaborative Innovation Networks Conference - COINs2010.
URL: *https://www.sciencedirect.com/science/article/pii/S1877042811023895*