# PREDICTING THE BITCOIN TREND USING TECHNICAL INDICATORS FOR DEEP LEARNING ALGORITHMIC FEATURES

NURİ TUĞKAN İNCE

BOĞAZİÇİ UNVERSITY

2019

# PREDICTING THE BITCOIN TREND USING TECHNICAL INDICATORS FOR DEEP LEARNING ALGORITHMIC FEATURES

Thesis submitted to the

Institute for Graduate Studies in Social Sciences

in partial fulfilment of the requirements for the degree of

Master of Arts

in

Management Information Systems

by

Nuri Tuğkan İnce

Boğaziçi University

2019

Predicting the Bitcoin Trend

Using Technical Indicators for Deep Learning Algorithmic Features

The thesis of Nuri Tuğkan İnce

has been approved by:

Assist. Prof. Ahmet Onur Durahim
(Thesis Advisor)

Prof. Ceylan Onay Şahin

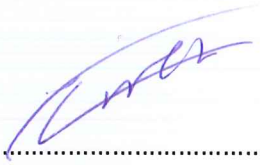
Prof. Ali Rana Atılgan
(External Member)

July 2019

DECLARATION OF ORIGINALITY

I, Nuri Tuğkan İnce, certify that

- I am the sole author of this thesis and that I have fully acknowledged and documented in my thesis all sources of ideas and words, including digital resources, which have been produced or published by another person or institution;
- this thesis contains no material that has been submitted or accepted for a degree or diploma in any other educational institution;
- this is a true copy of the thesis approved by my advisor and thesis committee at Boğaziçi University, including final revisions required by them.



Signature.....

Date .....15.08.2019.....

ABSTRACT

Predicting the Bitcoin Trend

Using Technical Indicators for Deep Learning Algorithmic Features

This research focused on utilization of technical indicators for predicting the trend of Bitcoin/USD price with different deep learning algorithms. The goal was to achieve better trend prediction accuracy results using technical indicators compared to using only close, open, high, low and volume (OHLCV) data for Bitcoin/USD parity. Through achieving this goal, three different deep learning algorithms, Deep Neural Networks (DNN), Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRU) were used because of the performance they exhibit in literature for financial stock prediction domain and their theoretical convenience. 156 technical indicators, mathematical transformations and financial patterns were used in feature set to test against OHLCV data of Bitcoin/USD. Experiments in this research showed that, utilization of technical indicators produced better accuracy results compared to OHLCV data for all three prediction models. For the imbalanced dataset distribution produced by a one-way transaction cost to decide buy, hold or sell operations, LSTM performed best among the models used in this research with achieving 56.33% accuracy score with reasonable individual class prediction rates whereas raw data could achieve 53.26%. In the scenario for which the one-way transaction cost is tuned to have a uniformly distributed dataset, GRU performed best with achieving 52.19% accuracy score whereas raw data could achieve 39.85%.

# ÖZET

## Teknik Göstergeleri Kullanarak
## Bitcoin Trendini Derin Öğrenme ile Tahmin Etmek

Bu araştırmada teknik göstergeleri farklı derin öğrenme yöntemleri ile kullanarak Bitcoin/Dolar paritesinin trendini tahmin etmek amaçlandı. Temel hedef, Bitcoin/Dolar paritesinin trendini tamin ederken, teknik göstergelerin kullanılması ile, yalın finansal veri olan açılış, kapanış, en yüksek, en düşük ve hacim (AKYDH) verilerinin kullanıldığı uygulamalardan daha iyi doğruluk başarım sonuçları elde etmek oldu. Bu doğrultuda, literatürde stok fiyatı tahmin etme alanında gösterdikleri başarımlardan ve teorik uyumluluklarndan ötürü, Derin Sinir Ağları (DSA), Uzun Kısa Vadeli Hafıza Ağları (UKVH) ve Kapılı Tekrarlayan Hücre (KTH) algoritmaları kullanıldı. AKYDH verilerine karşı test etmek amacı ile özellik kümesinde toplamda 156 adet teknik gösterge, matematiksel dönüşüm ve finansal kalıp kullanıldı. Bu araştırmadaki deneyler gösterdi ki, kullanılan bütün derin öğrenme modelleri için, teknik göstergeler kullanıldığında, bu göstergelerin kullanılmadığı, AKYDH verisinin kullanıldığı denemelerden daha başarılı sonuçlar elde edildi. Tek yönlü işlem komisyon oranı kullanılarak oluşturulan ve al, sat veya bekle sınıflarından oluşan, sınıf dağılımı açısından dengesiz veri kümesi için UKVH, kullanılan üç model arasında en başarılısı oldu. UKVH, bireysel sınıf tahminleme başarımlarının da kabul edilebilir ölçüde olduğu durumda %56.33'lük genel bir doğruluk tahminleme başarımı elde etti. AKYDH verisi ile bu başarım oranı en fazla %53.26 olabildi. Tek yönlü işlem komisyonu, daha dengeli dağılıma sahip bir veri kümesi elde etmek için değiştirildiğinde gözlemlenen en başarılı model %52.19 ile KTH oldu. Bu veri kümesi için AKYDH ancak %39.85'lik bir başarım elde edebildi.

TABLE OF CONTENTS

CHAPTER 1: INTRODUCTION ..... 1

1.1 Definition of the problem ..... 3

1.2 Research hypothesis ..... 3

1.3 Contributions ..... 3

CHAPTER 2: TECHNICAL BACKGROUND ON MACHINE LEARNING ..... 5

2.1 Machine learning problems ..... 6

2.2 Types of machine learning ..... 7

2.3 Machine learning classification algorithms ..... 10

CHAPTER 3: TECHNICAL BACKGROUND ON DEEP LEARNING ..... 15

3.1 Deep neural networks ..... 16

3.2 Recurrent neural networks ..... 17

CHAPTER 4: TECHNICAL BACKGROUND ON FINANCIAL ANALYSIS ..... 22

4.1 Fundamental analysis ..... 22

4.2 Technical analysis ..... 24

CHAPTER 5: TECHNICAL BACKGROUND ON TECHNICAL INDICATORS . 26

5.1 Trend indicators ..... 27

5.2 Momentum indicators ..... 30

5.3 Volatility indicators ..... 33

5.4 Volume indicators ..... 36

CHAPTER 6: LITERATURE REVIEW ..... 40

6.1 Machine learning for algorithmic trading ..... 40

6.2 Deep learning for algorithmic trading ..... 41

6.3 Technical indicator implementations for algorithmic trading ..... 44

6.4 Trend prediction in algorithmic trading ..... 45
CHAPTER 7: METHODOLOGY ..... 47
7.1 Goal of the research ..... 48
7.2 Input dataset ..... 48
7.3 Pre-processing of data ..... 50
7.4 Calculation of technical indicators ..... 50
7.5 Standardization ..... 53
7.6 Output space classification ..... 53
7.7 Model construction and implementation ..... 55
CHAPTER 8: EXPERIMENTS AND RESULTS ..... 60
8.1 Deep neural networks ..... 60
8.2 Long Short-Term Memory ..... 63
8.3 Gated Recurrent Units ..... 67
8.4 Results ..... 71
CHAPTER 9: CONCLUSIONS AND FUTURE WORK ..... 74
REFERENCES ..... 77

LIST OF TABLES

Table 1. List of Technical Indicators ..... 51

Table 2. List of Financial Patterns and Mathematical Functions..... 52

Table 3. Class Distribution for Transaction Cost Margin 0.1%..... 54

Table 4. Class Distribution for Transaction Cost Margin 0.7% ..... 54

Table 5. Hyperparameters of DNN Models ..... 61

Table 6. Performance Results of DNN Models ..... 61

Table 7. Confusion Matrix of DNN Models for Margin 0.001..... 62

Table 8. Confusion Matrix of DNN Models for Margin 0.007..... 62

Table 9. Hyperparameters of LSTM Models ..... 63

Table 10. Performance Results of LSTM Models ..... 64

Table 11. Confusion Matrix of LSTM Models for Margin 0.001..... 65

Table 12. Confusion Matrix of LSTM Models for Margin 0.007..... 65

Table 13. Additional Performance Metrics for LSTM Models with Best Accuracy 66

Table 14. Hyperparameters of GRU Models ..... 67

Table 15. Performance Results of GRU Models ..... 68

Table 16. Confusion Matrix of GRU Models for Margin 0.001..... 69

Table 17. Confusion Matrix of GRU Models for Margin 0.007..... 69

Table 18. Additional Performance Metrics for GRU Models with Best Accuracy .. 70

Table 19. Accuracy Performance Comparison of Models for Margin 0.007..... 71

Table 20. Accuracy Performance Comparison of Models for Margin 0.001..... 72

LIST OF FIGURES

Figure 1. Google Search trend for the term “Deep Learning” ..... 16

Figure 2. LSTM network composed of three cells. ..... 19

Figure 3. Example GRU cell ..... 21

Figure 4. SMA Line Example on financial chart ..... 28

Figure 5. EMA Line example on financial chart ..... 29

Figure 6. MACD Line example on financial chart ..... 30

Figure 7. RSI Line example on financial chart ..... 31

Figure 8. CCI Line example on financial chart ..... 32

Figure 9. CMO chart example on financial chart ..... 33

Figure 10. Bollinger Bands example on financial chart ..... 34

Figure 11. ATR Line example on financial chart ..... 35

Figure 12. Bollinger Bandwidth Line example on financial chart ..... 36

Figure 13. OBV Line example on financial chart ..... 37

Figure 14. ADL Line example on financial chart ..... 38

Figure 15. Chaikin Oscillator Line example on financial chart ..... 39

Figure 16. Average Bitcoin Time-Series Daily Price of Dataset ..... 49

Figure 17. Subset of the Input Data for the Experiments ..... 50

# CHAPTER 1

# INTRODUCTION

Over past decades, analysing a market and predicting its future behaviour based on data analysis have been one of the hot topics in both the academic and business world. Yet due to the uncertainties and number of stakeholders in a relatively chaotic domain, establishing a valid analysis and deriving a prediction has always been a challenge. In the financial domain, there are two types of analysis. The first one is known as the fundamental analysis which correlates the market behaviour with intuitive, political and psychological aspects. The second method of analysis, technical analysis, on the other hand, is a method of analysis that is mostly oriented around chart representations and statistical behaviour of prices in that particular market. Technical analysis takes advantage of the past data by evaluating them in certain formulas in order to establish an impression on the market and furthermore, predict its future behaviour.

Even though efficient market hypothesis argues that market prices are in the tendency of moving in a random walk manner, which results in an impossibility of prediction, technical analysts advocate that, historical prices in a market cumulatively represent the hidden information about that particular market, which provides the ability to predict the tendencies through observing past trends and prices (Patel, Shah, Thakkar, & Kotecha, 2014). Technical analysis approaches the market with the assumption of the tendency of repetition of the past behaviour or pattern (Tam, & Cuong, 2018). Therefore, a proper analysis of the statistical representation of a market behaviour should project the future behaviour of it in an empirical manner. The ability to predict future trends with past data through technical analysis

also shows parallelism and supports the Inefficient Market Hypothesis, which argues markets are not generally exhibiting random walk behaviour (Khare, Darekar, Gupta, & Attar, 2017). With the amount of data introduced to the world through financial domain, predicting a market's behaviour naturally became one of the hot topics of the machine learning area. Patel, Shah, Thakkar, and Kotecha (2015), references that, in order to achieve the best precision on predicting price behaviour, many different machine learning algorithms have been utilized in previous years such as Artificial Neural Networks (ANN), Fuzzy Logic, Genetic Algorithms (GA), Support Vector Regression and Support Vector Machine (SVM) along with the hybrid algorithms to realize the most accurate output.

Yet it was argued by Khare et al. (2017) that, among all other machine learning methods, usage of neural networks to establish a prediction algorithm is quite promising due to the structural aspects provided by neural networks that correlate with financial markets' behaviour by grouping these under three topics. First, Khare et al. (2017) indicate that neural networks have a considerable amount of potential in order to derive context from convoluted or estimated information which may be highly complex for human beings or other computational processes. Second, they mention that neural networks, compared to the linear models, are evaluated to be a preferable choice when it comes to performing predictions in non-linear domains. Finally, they emphasize the ability of neural networks to adapt easily to the changes observed in the behaviour of market data in the domain of analysis which can be regarded as one of the key characteristics of the financial domain.

## 1.1 Definition of the problem

Cryptocurrencies are relatively newer assets in the financial domain and the market itself is relatively new and unstable. As a result, it would be arguable that the inefficient market theory is valid for the cryptocurrency domain. So, the trends and movements in the cryptocurrency domain might actually be predictable with utilizing existing data and technical analysis.

However, for daily trend analysis, the dataset which can be used for machine learning algorithms may not be sufficient enough due to a comparably smaller amount of data. So, financial insights from the limited data should be derived as much as possible from the dataset in order to increase the prediction success rate.

In that respect, technical indicators can be utilized in order to derive additional useful features from the raw open, high, close, low and volume data of cryptocurrency assets. As they may show better correlation results with the trend output compared to raw data, in the end technical indicators are expected to bring significant advantages to the machine learning models to be generated in terms of predictive power.

## 1.2 Research hypothesis

Utilizing technical indicators as input features in deep learning approaches is expected to produce better results in terms of accuracy of the trend prediction for Bitcoin compared to raw financial asset data (open, close, low, high, volume)

## 1.3 Contributions

For this research, 156 technical indicators are used as input dataset for predicting bitcoin movements for a three-class domain which is composed of Buy, Hold and

Sell decisions. These classes are created with utilizing a one-way transaction cost margin.

As the first contribution, this research provides insight related with prediction accuracy performance comparison between models trained by using features generated with technical indicators, and models that are trained with standard open, high, low, close and volume (OHLCV) data for the Bitcoin domain.

Second, this research also measures three class classification performance of deep learning algorithms with the existence of one-way transaction cost for Bitcoin domain.

Finally, using two different one-way transaction cost margins to produce balanced and imbalanced output spaces, this research compares prediction performance of deep learning models for imbalanced and balanced class distributions for Bitcoin domain.

# CHAPTER 2

# TECHNICAL BACKGROUND ON MACHINE LEARNING

"Learning is the process of converting experience into expertise or knowledge." (Shalev-Shwartz & Ben-David, 2014, p.19). As indicated in the brief description by Shalev-Shwartz and Ben-David (2014), learning is a process and knowledge is derived from experiences.

Shalev-Shwartz and Ben-David (2014) enrich their definition with the example of Bait Shyness. They explained that rats start interacting with their food with smaller portions in order to get familiar with it, analyse it and determine the physiological effect of the food on their body in order to label that particular food in terms of safety. If the food is safe, then it is labelled as safe and mapped in their experience. Using that experience mapping rats in the future, can understand if a food is safe or not with relating the effects of the food they're having with their experiences in the past.

Similarly, Shalev-Shwartz and Ben-David (2014) explained that a machine can as well memorize e-mails which are labelled as spam and non-spam in order to establish relations and differentiate the types from each other in future encounters, which in the end makes a perfect example for "Machine Learning".

The term "Machine Learning" became more popular in the past decades and a conceptually well-known approach among many individuals. The methodologies which are brought with the machine learning concept brings a large number of capabilities of automation which are utilized to solve many problems in various research areas. Again, in order to solve the problems with machine learning, different

types of algorithms are used, and they brought insights to the research or problem area.

## 2.1 Machine learning problems

The spectrum of learning use cases and learning problems is quite large. Yet some of the templates of learning can be categorized under four sections

### 2.1.1 Binary classification

Smola and Vishwanathan (2008, p.9) describes this type of learning as “probably the most frequently studied problem in machine learning”. To briefly summarize, this learning is oriented around classifying the input data or feature set to a binary output space. Spam and non-spam e-mail example perfectly fits this type of learning.

### 2.1.2 Multiclass classification

Smola and Vishwanathan (2008) describe multiclass classification as the logical extension of binary classification. Similar to the binary classification, in this learning method, input data or feature set is utilized to classify data. Yet this learning method differs from binary classification with having more than two output classes to evaluate. Classifying financial time series data into one of buy, hold, sell decisions makes a good example of this type of learning.

### 2.1.3 Regression

The goal of this type of learning application is to estimate the actual value of the output variable instead of making a classification for input data (Smola &

Vishwanathan, 2008). Prediction attempts on prices of financial stock assets are good examples of regression type of learning.

### 2.1.4 Novelty/anomaly detection

This type of application is rather utilized to describe the issue of determining the odd or unusual observations by using past experiences (Smola & Vishwanathan, 2008). Smola and Vishwanathan (2008) explain that "unusual events occur rarely" which can also be referenced from the meaning of the word "unusual" itself. Applications that utilize machine learning for network anomaly detection are good examples for this practice (Shon & Moon, 2007).

## 2.2 Types of machine learning

Jordan and Mitchell (2015) indicates that there are three types of machine learning, "Supervised Learning", "Unsupervised Learning" and "Reinforcement Learning".

### 2.2.1 Supervised learning

LeCun, Bengio and Hinton (2015) describes supervised learning as the most common type of machine learning. In this type of learning, a training dataset, which is composed of different features, with corresponding correct responses so called "labels" is provided to the learning model (Marsland, 2015). Then model is expected to learn through generalization of the relation between input and output sets of the training data.

For example, a classification problem to classify spam and non-spam e-mails can be a subject of supervised learning. If we assume that we have a dataset which is composed of e-mails and their corresponding labels which indicate that they are

spam or non-spam, we can build a model and train it with that particular dataset to build a generalization with using the relation between input and output training dataset. Once the model is trained, for the further predictions, model can use the generalization function which is produced by training process.

Donalek (2011) claims that a proper training process, validation and testing the performance of the trained model is crucial for supervised learning. Donalek (2011) also argues that in order to be evaluated as effective, models trained by supervised learning need to be able to generalize the input data in order to make sure that model is able to provide an accurate output given a new data, for which a prior identical experience is not present.

One important challenge for supervised learning is “overfitting”. If trained poorly, a model might not be able to efficiently reflect the maneuvers in the input/output relation which can also be explained by the underfitting of the model. Yet on the other hand, if the model is trained aggressively with the training dataset, model might end up memorizing the relations in training data instead of generalizing to the underlying function which will result in relatively better performance on training dataset but performs relatively poor with the test set or new dataset, which is known as the “overfitting problem” (Donalek, 2011).

From the application domain perspective, supervised learning methods can be utilized for classification and regression problems (Smola, 2008).

### 2.2.2 Unsupervised learning

In unsupervised learning approach, on the counter of supervised learning, the model is unaware of the correct results during the training process (Donalek, 2011). Donalek (2011) adds that unsupervised learning methods can be used to cluster the

input data and group the input set into separate classes on the basis of their statistical properties.

Ayodele (2010) claims that unsupervised learning seems much harder compared to supervised learning. Ayodele (2010) argues that there are two approaches for unsupervised learning, first is introducing a reward function to the system as a success indicator, similar to reinforcement learning, and the second approach is clustering which aims to find similarities in the training data. Additionally, similar to the supervised learning methods, Ayodele (2010) indicates that unsupervised learning methods are also vulnerable to overfitting issues.

### 2.2.3 Reinforcement learning

Marsland (2015), positions this learning type between supervised and unsupervised learning. Because, for this learning, the data is labelled and algorithm is informed if the result is wrong. However, instead of feeding the model with how to correct the failure the algorithm is expected to iteratively try its options and learn how to correct its answer by itself (Marsland, 2015).

This approach is built around a reward strategy. For this type of learning, terminologically state means the memory. Through learning, the model tries out the input for possible actions in its current state and evaluates the results of that action. With giving feedback to itself based on that evaluation, it takes another action to maximize the numerical reward function which in the end is the indication of learning (Marsland, 2015).

Marsland (2015) simply exemplifies this type of learning with a child trying to stand up after many unsuccessful attempts. Throughout past decades, reinforcement learning methodologies have been subjects for many research areas

and machine learning problems like chemistry (Zhou, Li, & Zare, 2017), traffic lights optimization (Mousavi, Schukat and Howley, 2017), stock prediction (Lee, 2001), etc.

## 2.3 Machine learning classification algorithms

Since the problem, which is tried to be addressed with this research, is the classification of the movements in cryptocurrency prices, the learning type of the algorithm and hence the focus area is multiclass classification. In this section, some of the popular and relatively widely used multiclass classification methods will be briefly explained.

### 2.3.1 Decision tree

Kotsiantis (2007, p. 251) defines decision trees as "... trees that classify instances by sorting them based on feature values.". If we map decision trees to the actual trees, each node or leaf in the decision tree stands for a feature to be evaluated for a given instance to be classified and branches of the decision trees are utilized in comparing the values of those features of the instance against the model (Kotsiantis, 2007).

Moreover to their application area in prediction, decision trees are used for efficiently selecting features for a machine learning problem as well (Song & Lu, 2017). For example, XGBoost which is a scalable gradient tree boosting system has decision tree infrastructure (Chen & Guestrin, 2016). Chen and Guestrin (2016) states that XGBoost brings promising performance results in terms of ranking the feature importance compared to the best algorithm which is parallel gradient boosted regression trees (pGBRT) by the time of the performance comparison.

Although they are easy to use and understand, capable of mitigating missing values in dataset and robust to outliers, they are quite vulnerable for overfitting and underfitting problems (Song & Lu, 2017). Additionally, decision trees might not be efficient in prediction of the input dataset has correlated features, which in this case makes this algorithm a weak candidate for time series stock prediction as the input data which is composed of open, close, low, high and volume, is naturally expected to be correlated.

### 2.3.2 Bayesian networks

Kotsiantis (2007) describes Bayesian networks as representing the graphical models of probability relationships among features. For Bayesian networks to function efficiently, the feature set needs to be conditionally independent.

Compared to decision trees, the most significant characteristic of Bayesian networks are the capability of utilizing different sources of data and existing domain knowledge. Related with Bayesian networks, it is argued that they tend to perform better when the input dataset is composed of the features which belong to different classes (Cheng & Greiner, 2001). This theoretical argument makes Bayesian networks weak algorithms in terms of predicting the trend of financial time series data as the input set is hardly composed of different classes of data.

Therefore, since the context of this research is oriented around making predictions on a relatively less known domain, with comparably high complexity. Utilization of Bayesian networks theoretically would be inefficient. Additionally, again on the counter of a fundamental requirement of having a conditionally independent feature set for Bayesian networks, financial time series data are composed of possibly dependent feature sets. As a result of the theoretical analysis,

Bayesian networks are not expected to be one of the popular algorithms for classifying financial time series data as a technical analysis approach.

### 2.3.3 Support vector machines

Vapnik's study (as cited in Kotsiantis, 2007) remarks that Support Vector Machines (SVM) are one of the newest algorithms for supervised learning methodologies. SVMs are oriented around the concept of a "margin", which is placed on both sides of a hyperplane and the hyperplane is actually a separator of two data classes. Being able to enlarge the margin results in increasing the capability of distinction between separate classes and eventually an increase in prediction accuracy (Kotsiantis, 2007).

Kotsiantis (2007) adds that if the input data is linearly separable and an optimum hyperplane(separator) can be obtained, the number of features in the training dataset has no effect on the model complexity of SVM. As a result, SVM algorithms are suitable candidates for challenging learning objectives that require learning among a large number of features. Singh, Thakur, and Sharma (2016) also supports that argument with stating that the number of features has no effect on the complexity of the domain and the algorithm is robust for high dimensional data.

In order to efficiently use this algorithm, the proper kernel function needs to be selected. This is one of the challenges through building the optimum model for SVM classification (Singh, Thakur, & Sharma 2016). As another challenge, SVM methods are suitable for binary classification tasks and in order to analyse and make predictions for multiclass classification problems, the multiclass output domain must be reduced to a set of multiple binary classification problems and this reduction might be somehow cumbersome (Kotsiantis, 2007).

### 2.3.4 Artificial neural networks

Artificial neural network algorithms are inspired by neurons in the human brain (Marsland, 2015). Artificial neural networks aim to optimize weights, which can be modelled as synapses in human brain, of neurons just like human brains and throughout the training of the algorithm, weights of the features are optimized in order to achieve best results in terms of prediction performance (Marsland, 2015).

Artificial neural networks are composed of input layers, hidden layers and output layers which are interconnected sequentially (Ata, 2015). Ata (2015) states that artificial neural networks can be utilized to solve complex real-life use cases efficiently. They can be used in areas such as pattern recognition, function approximation, simulation, prediction etc. (Ata, 2015).

Like selecting the kernel for an SVM algorithm, determining the size and number of hidden layers is a challenge for artificial neural networks as selecting a low number of hidden layers might result in low generalization performance and causes underfitting as well as selecting them in high numbers can cause overfitting issue (Marsland, 2015).

According to Kotsiantis (2007), SVM and neural network algorithms achieve better results through dealing with multidimensions and continuous features or input set. Kotsiantis (2007) adds that in order to obtain the best performance out of SVM and neural networks, large datasets need to be utilized. Additionally, ANNs can perform well if nonlinear relationships exist between input and output features (Kotsiantis, 2007). Again, as it will be explained in the literature review chapter in detail, since researches show that algorithms of deep learning such as Gated Recurrent Unit (GRU) and Long Short-Term Memory (LSTM) exhibit superior performance in terms of accuracy as compared to SVM, this research will focus on

deep learning algorithms to create trend prediction models for financial time series domain.

In the following chapter, the concepts deep learning and deep neural networks, which are a subset of artificial neural networks will be explained further as part of the research to provide an insight related to the concept which is going to be utilized for the research experiments.

# CHAPTER 3

# TECHNICAL BACKGROUND ON DEEP LEARNING

As described in the previous section, artificial neural networks are composed of interconnected neurons that produce real-valued activations. Neurons in hidden layers and output layers get activated through optimized weighted connections from previous layers and inner layers which get activated through sensors perceiving the environment (Schmidhuber, 2014). For the artificial neural networks, learning itself is oriented around finding out the weights that make the neural networks demonstrate the expected behaviour. Depending on the complexity of the problem and application domain of the artificial neural networks, the utilization of longer casual chains might be required where each stage transforms the aggregate activation of the neural network. Here the term deep learning represents deeper neural networks that are required to map and overcome problems in an artificial neural network domain (Schmidhuber, 2014).

According to Schmidhuber (2014), shallow artificial neural network models with a lesser number of stages are subjects of researches through the mid-nineteenth century. Yet backpropagation-based training of deeper neural networks started to be evaluated as a research topic by the end of the 1990s (Schmidhuber, 2014). Schmidhuber (2014) adds that the 1990s and 2000s also introduced enhancements and improvements for supervised deep learning algorithms. The popularity of deep learning methods however could be established after the 2000s and with the first decade of the second-millennium, utilization of deep learning algorithms in prediction problems resulted in many victories through the competition with other algorithms.

According to the Google Trends search analytics, the popularity of the term "Deep Learning" is increasing gradually as shown in Figure 1:

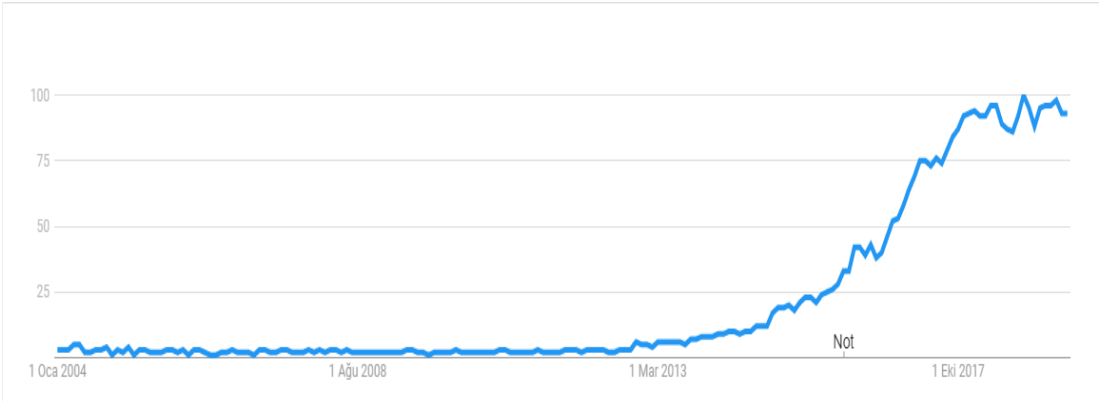



Figure 1. Google Search trend for the term "Deep Learning"

In the following sub-sections, due to their successful accuracy performances in financial time series trend prediction, some specific Deep Learning algorithms developed and used for prediction in the finance domain will be briefly explained. The scores they achieved throughout the past few years in terms of prediction accuracy performance will be mentioned in the literature review.

## 3.1 Deep neural networks

Deep neural networks (DNN) are no different than artificial neural networks. DNN as well are feed-forward artificial neural networks that are composed of at least two hidden layers instead of a single hidden layer (Hinton, Deng, Yu, Dahl, Mohamed, Jaitly, Senior, Vanhoucke, Nguyen, Sainath, & Kingsbury, 2012).

So, the working mechanism for this algorithm is still the same as ANNs. There is one input layer and an output layer, however, the hidden layers in the network are more than one.

Although DNNs are harder to train compared to ANNs with single hidden layer (Hinton et al., 2012), Hinton et al. (2012) argue that with the improvements in

machine learning and computer hardware, the training processes and methodologies of DNNs become significantly easy.

Hinton et al. (2012) mention that, with utilizing these new training methods for DNNs and with using multiple hidden layers, DNNs can achieve better performance scores compared to Gaussian Mixture Models (GMM) especially in some specific areas such as speech recognition problem, which needs understanding on highly complex non-linear relationships within the data. Hinton et al. (2012) add that DNNs outperforms methodologies such GMMs even sometimes with large margins.

On the other hand, Hinton et al. (2012) highlight that dealing with overfitting issues on DNNs and optimization of many hidden layers is a challenge for the implementation of these models.

## 3.2 Recurrent neural networks

Deep Neural Networks are from the family of feed forward neural networks. Recurrent neural networks (RNN) on the other hand, unlike feed-forward neural networks, take the advantage of remembering things from the past with adding feedback from previous steps (Du, Wang, & Wang, 2015).

To briefly explain, RNNs bring the capability of learning from prior input states through obtaining the output as part of the network. In other words, instead of processing the input in a single direction, RNNs can handle the information processing operation bi-directionally with generating internal loops in the neural network.

Schmidhuber (2014) argue that recurrent neural networks (RNN) won lots of contests and claims that RNNs (i.e. Long Short-Term Memories) are the deepest of

all neural networks. Their computational power is stronger compared to traditional feed-forward neural networks and in theory, they can produce and process memories of arbitrary sequences from input patterns (Schmidhuber, 2014). RNNs can learn with taking advantage of massively parallel computing which is quite crucial to be able to reduce computational cost with their capability of learning programs that mix sequential and parallel information processing in an efficient manner (Schmidhuber, 2014).

Due to their benchmark performances with their competitors, which will be explained in literature review chapter and their structural availability of automatic feature weight optimization which causes them to perform comparably better in domains with less human knowledge. In this research RNN algorithms will be considered in trend prediction.

In the following two sub-sections, Long Short-Term Memories and Gated Recurrent Units will be explained briefly as types of RNNs which are used to build prediction models in this research due to their wide usage in the financial time series prediction literature and their high predictive performance scores.

### 3.2.1 Long Short-Term Memory

Long Short-Term Memory architecture is introduced by Hochreiter and Schmidhuber (1997). The LSTM is composed of units called memory blocks, which contain memory cells with self-connections to store the temporal state as well as the gates to manage the information in the network (Sak, Senior & Beufays, 2014).

Input triggers into the memory cell are managed by the input gate of an LSTM memory block and as naturally expected, the output gate is in charge of the output flow of cell activations to the further network (Sak et al., 2014). It means that

if an input produces a signal which is meaningful and valuable for network, it will be forwarded to the further layers with the coefficient it receives from activation function based on its value. Apart from the input to the cell itself, LSTM cells are fed with the memory from previous cells, in other words they are stateful among the network. Cells generate new memories with applying their own experiences on the input to the existing memory and forward the memory to the subsequent cells in the network. An example LSTM network is shown in Figure 2.

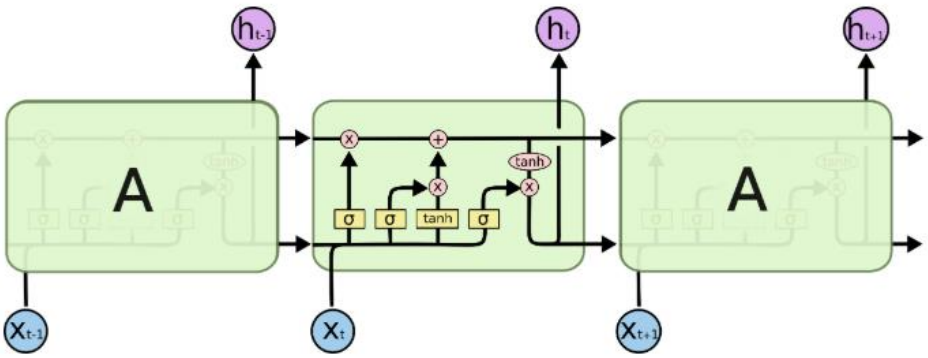



Figure 2. LSTM network composed of three cells. Reprinted from Understanding LSTM Networks, In Github, August 27, 2015, Retrieved July 12, 2019, from https://colah.github.io/posts/2015-08-Understanding-LSTMs/. Copyright 2015 by Christopher Olah. Reprinted with permission.

Gers, Schmidhuber, and Cummins (2000) add that, in order to fix a possible weakness in LSTM models, which is an impediment for processing continuous input stream that is not gone through segmentation into sub-sequences, forget gates are introduced to the architecture. These gates are utilized to adaptively reset the cell's memory with being capable of scaling the internal state of the cell before adding it as input to the cell using the recurring connection of it (Sak et al., 2014).

According to the Sak et al. (2014), despite the LSTM RNN's are already considered as deep learning architectures, having deeper layers in LSTM RNNs (Deep LSTM RNNs), brings more opportunities such as being able to use parameters better with distributing them over space through multiple layers. Using deeper

networks can be used to keep the memory size the same for LSTMs with introducing deeper layers which in the end results in inputs getting processed by a larger number of non-linear operations for each timestep.

The architecture has been used for many research applications and produced promising results in various domains including financial time series prediction, which will be referenced in the literature review section of this research.

### 3.2.2 Gated Recurrent Units

Gated Recurrent Units is a more recent RNN type algorithm introduced by Cho, van Merrinboer, Gulcehre, Bahdanau, Bougares, Schwenk, and Bengio (2014). For this algorithm, just like LSTMs, the units are capturing dependencies of different timescales adaptively (Cho et al., 2014). They're quite similar to LSTM algorithm except for the possession of separate memory cells.

Chung, Gulcehre, Cho, and Bengio (2014) argue that LSTM and GRU has similar approaches oriented around calculating the linear sum of newly computed state and the existing state. Chung et al. (2014) argues that the most significant similarity between LSTM and GRU is their additive approach for their state. According to Chung et al. (2014), the additive approach effectively creates shortcuts within network, which in the end allow the error to be back-propagated without unexpectedly disappearing.

However, as a difference, GRU cells also do not have forget gates, they instead have an update gate, which can be considered as combination of input and forget gates in LSTM cells (Chung, Gulcehre, Cho, & Bengio, 2014). An example GRU cell is shown in Figure 3.

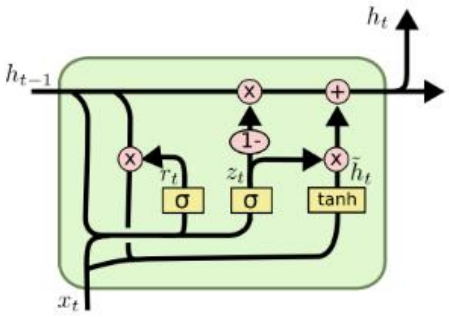



Figure 3. Example GRU cell. Reprinted from Understanding LSTM Networks, In Github, August 27, 2015, Retrieved July 12, 2019, from https://colah.github.io/posts/2015-08-Understanding-LSTMs/. Copyright 2015 by Christopher Olah. Reprinted with permission.

Moreover, GRU and LSTM differs in terms of having a mechanism to control the exposure of memory to the network. In LSTM units, the content of the memory and its exposure to the network are controlled by the output gates of the cell. However, there is no such control in GRUs and the memory always expose the whole content of the memory (Chung et al., 2014). This is a natural outcome of combining input and forget gates in a single update gate. This combination results in losing the capability of differentiating previous and current memory in the GRU cell.

Chung et al. (2014) also add that they differ due to the location of their input gate as well. LSTM is computing the new memory inside the cell without applying a separate control on the information gathered from the previous memories, whereas GRU controls the amount of information gathered from previous memories while calculating the new memory (Chung et al., 2014).

Chung et al. (2014) argue that the minor differences between LSTM and GRU cannot theoretically conclude or show that one algorithm is superior to the other. The only practical way to determine the winner in terms of performance could only be defined through empirical methods.

# CHAPTER 4

# TECHNICAL BACKGROUND ON FINANCIAL ANALYSIS

Financial analysis is an evaluation performed on a financial asset to determine its current state, performance or behaviour. Naturally, financial analysis methods are used to make short or long-term decisions in order to profit through the financial asset which is being analysed.

From a prediction domain perspective, there are two types of financial analysis. The first of them is fundamental analysis, the notion of which is utilizing domain and market-specific knowledge to make a profit from investment centred approaches. The second analysis approach is a technical analysis which on the other hand is oriented around utilizing data and charts related to that particular market and financial domain in order to make a profit from trade centred approaches.

In the following subsections, these analysis methods will be explained briefly along with their comparison.

## 4.1 Fundamental analysis

Fundamental analysis is a method to analyse a financial asset's value by utilizing the economic and financial factors. The approach basically uses each and every source of data to properly value the financial asset in the market.

The parameters which are subjects of fundamental analysis can include macroeconomic subjects such as government laws, industrial situations, co-operations between companies or unions, embargos.

Moreover to the macroeconomic factors, fundamental analysis is also interested in microeconomic factors that might have even the slightest effect on the

value of the subject which is being analysed such as management of a company, contracts or mergers and acquisitions decisions performed by the companies, revenues, profits or any kind of data that might have an effect on the negative or positive potential for that particular financial asset.

Through fundamental analysis, the fundamental analysts tend to make a profit with investment-based approaches which in the end is a long-term approach. Wafi, Hassan, and Mabrouk (2015) argue that, fundamental analysis methods are more suitable for making long term investment decisions. Wafi et al. (2015) add that, fundamental analysis methods produce lesser numbers of trade signals compared to technical analysis methods. This statement also coincides with fundamental analysis as well as the decisions based on fundamental analysis methodologies being long term.

The long-term investment decisions on the other hand, which are derived from fundamental analysis, should be taken carefully. Because, these methodologies produce less amount of trade signals (Wafi et al., 2015), therefore corrections for the actions taken in the past can take time and eventually cause the investor to lose money.

In order to make a reliable fundamental analysis, individuals need to have a well understanding related to the domain of the financial asset and due to its dependency on various sources, establishing a well-structured financial analysis might be quite challenging for average traders.

In order to make proper use of fundamental analysis, traders or investors can take the advantage of combining fundamental analysis with the aspects of technical analysis especially for shorter time horizons (Menkhoff, 2010).

## 4.2 Technical analysis

Technical analysis is the usage of market statistics, historical time-series price and graphical charts in order to efficiently predict price movements and proper positions in financial markets (Hsu, Taylor, and Wang, 2016). The main motivation for this approach is identifying previous market patterns, in other words learning from experiences which is quite similar to machine learning application approaches.

Even though the term “technical analysis” sounds like a complex approach, contrarily, the methodology itself is quite simple. To briefly explain, technical analysis is actually the study of prices with the utilization of financial charts.

Efficient Market Hypothesis argues that markets are extremely efficient in terms of adapting the information about individual stocks as well as the whole market (Malkiel, 2003). The technical analysis approach does not completely contradict with this hypothesis; however, technical analysts believe price movements even in an efficient market are not completely random and can be identified through re-occurring patterns (Hayes, 2019).

Technical analysts believe that market discounts everything, which indicates each and every fundamental factor for a financial asset is already priced and reflected into the asset itself (Teixeira & Oliveira, 2010). This approach actually weakens the necessity of evaluating all the fundamental factors while taking a financial position in the market as the price as well. Considering that the pricing in the market will be self-correcting and the technical analysis depends solely on price, the technical analysis methodologies should be self-correcting as well based on the adjustments in the market.

Moreover, technical analysis is oriented around repetition tendency of history (Siegel, Shim, Qureshi, and Brauchler, 2014). It is believed by technical analysts that

prices are moving with inspirations and patterns from history which results in a considerable amount of chance in terms of predictability. This assumption is quite similar to the very basic fundamentals of machine learning algorithms which argue that experience is a teacher and patterns can be learned. The source of the idea is human psychology which is expected to result in same or similar decisions in the same or similar market situations (Siegel et al., 2014).

Hsu et al. (2016), based on the results of their research, indicate that technical analysis methodologies has prediction power for both developed currencies and emerging currencies. Hsu et al. (2016) add that the prediction power for emerging markets is higher compared to the developed country currencies.

Moreover, Menkhoff and Taylor (2006) argue that the profitability through technical analysis is increased with the duration of financial patterns which are observable from technical analysis perspective. With this statement, it can be expected that machine learning algorithms, which are capable of handling long term dependencies should achieve better performance results compared to those which are not.

Due to auto-correcting behaviour of technical analysis, their fundamental similarities which shows parallelism for deep learning application areas, and lack of domain knowledge in cryptocurrency area, we decided to proceed with technical analysis with the help of technical indicators for this research.

# CHAPTER 5

# TECHNICAL BACKGROUND ON TECHNICAL INDICATORS

Technical indicators are mathematical and graphical tools for technical analysts which make use of mathematical and heuristic calculations utilizing the price and volume data of financial assets. There are multiple types of technical indicators focusing on different aspects of a financial asset such as price trends, volume, momentum, oscillators, moving averages, etc.

Considering that the input data for these mathematical calculations are price and volume data of the security itself, it can be said that they are just further meaning derived from the raw historical data in that particular domain without the interference of human interactions or emotions.

Since the technical indicators are one of the main instruments of technical analysis, utilization of these instruments as part of technical analysis for an emerging currency such as Bitcoin can be expected to increase the prediction performance (Hsu et al., 2016).

Furthermore, Dempster, Payne, Romahi and Thompson (2001) argue that technical indicators can be leveraged to derive useful information from historical time-series data of Foreign eXchange (FX) assets, which eventually can have positive effect on the prediction performance of different machine learning algorithms.

Moreover, Zarrabi, Snaith, and Coakley (2016) argue that the effects of the technical indicators in terms of FX prediction performance can change in time due to the changing behaviours in FX markets. In this case, the ability of auto-correcting the weights of neural networks, hence somehow the importance of the input features,

which forms the basis of this research, coincides with requirement of adaptation on changes in importance of technical indicators for a FX trading domain.

In their article, Muchiri, Pintelon, Gelders, and Martin (2011) categorize technical indicators in terms of their application areas and behaviour into two, as leading and lagging. Muchiri et al. (2011) explains that leading technical indicators can produce trading signals before the trend has started or it is just about to start.

However, lagging types of technical indicators are those, which follow the actions in the already ongoing trend instead of early signalling and leading those movements in price (Muchiri et al., 2011). Even if they seem with less use in terms of prediction, they're quite useful in validating trends and smoothing values in a relatively more unstable input domain.

From a different perspective, technical indicators can be separated into four types (Yazdi & Lashkari, 2012). In the following sub-sections, these types are going to be explained briefly with formulated and graphed exemplifications.

## 5.1 Trend indicators

They are utilized to measure a trends' direction and strength mostly with leveraging price averaging techniques. Increases in the average for a certain period naturally indicates a bullish trend and on the contrary, decreases in the average for a certain period indicates a bearish trend for a financial asset.

Despite trend indicators being follower instead of being leader indicators, their ability of reflecting the underlying trend for a financial asset can be used a supporting tool while making buy, sell or hold decisions (Ni & Yin, 2009).

In the following sub-sections three popular trend indicator examples, which are used for this research as well, will be explained further.

### 5.1.1 Standard Moving Average

Standard or Simple Moving Average (SMA) is the average close price value of a financial asset for a certain period.

SMA is calculated with equation (1) (Droke, 2001), and example graph of SMA indicator is shown in Figure 4.

$$SMA = (SUM(N))/N \quad (1)$$


Figure 4. SMA Line Example on financial chart

### 5.1.2 Exponential Moving Average

The technical indicator is, as understandable from its name, a moving average indicator. It aims to decrease the amount of lag with assigning relatively more weight to the recent prices.

Exponential Moving Average (EMA) is calculated with equations (2) and (3) (Droke, 2001), and example graph of EMA is shown in Figure 5.

$$Multiplier = \frac{2}{(Period+1)} \quad (2)$$

$$EMA = \big(Close - EMA(Previous\ Day)\big)\ x\ Multiplier + EMA(Previous\ Day) \quad (3)$$


Figure 5. EMA Line example on financial chart

### 5.1.3 Moving Average Convergence Divergence

Appel (1979) introduced this technical indicator in financial technical analysis literature. Moving Average Convergence Divergence (MACD) draws to trend following indicators and crossovers of these trends show buy and sell signals.

Even though this indicator is composed of combinations of EMAs which are considered as lagging technical indicators, crossovers between two lines of MACD are leading and signalling buy or sell operations as explained above.

Appel (1979) calculates MACD with equations (4), (5) and (6). Example graph of MACD indicator is shown in Figure 6.

$$MACD\ Line = ShorPeriod\ EMA - LongPeriod\ EMA \quad (4)$$

$$Signal\ Line = SignalPeriod\ EMA\ of\ MACD\ Line \quad (5)$$

$$MACD\ Histogram = MACD\ Line - Signal\ Line \quad (6)$$


Figure 6. MACD Line example on financial chart

## 5.2 Momentum indicators

Those indicators are calculated by checking current close prices against their previous values. They can be used to analyse the pace of the price movement as well as volume and the divergence between the actual price and this kind of indicator can generate a signal of change in upcoming prices for the financial asset.

In the following sub-sections three popular momentum indicator examples, which are used for this research as well, will be explained further.

### 5.2.1 Relative Strength Index

Relative Strength Index (RSI) indicator is found by Wilder (1978). RSI is a momentum indicator that is used to determine and predict the speed and change in price movements (Wilder, 1978).

For a financial asset, calculated RSI oscillates between 0 and 100. Levels higher than 70 in this spectrum are evaluated as sell signals while levels under 30 are evaluated as buy signals due to overselling indication.

Wilder (1978) calculates RSI with equations (7) and (8). Example graph of RSI indicator is shown in Figure 7.

$$RS = \frac{Average\ of\ 14\ day's\ closes\ up}{Average\ of\ 14\ day's\ closes\ down} \tag{7}$$

$$RSI = 100 - \frac{100}{1+RS} \tag{8}$$


Figure 7. RSI Line example on financial chart

### 5.2.2 Commodity Channel Index (CCI)

Commodity Channel Index (CCI) was found and developed by Lambert (1983). It can be used to determine if an asset is overbought or oversold as CCI will be relatively high when prices are above the level of expectations and will be relatively low when the prices are below the level of expectations.

Lambert (1983) calculates CCI with equations (9) and (10). Example graph of CCI indicator is shown in Figure 8.

$$Typical\ Prica(TP) = \frac{(High+Low+Close)}{3} \tag{9}$$

$$CCI = \frac{TP - SMA_{TP}(N)}{Constant\ X\ Mean\ Deviation} \tag{10}$$


Figure 8. CCI Line example on financial chart

### 5.2.3 Chande Momentum Oscillator (CMO)

Chande Momentum Oscillator is a momentum oscillator found by Chande and Kroll (1994). The mechanism of the technical indicator is quite similar to relative strength index (RSI). The indicator oscillates between +100 and -100 where values above +50 indicate overbought and values below -50 indicate oversold.

Chande and Kroll (1994) calculates CMO with equations (11), (12) and (13). Example graph of CMO indicator is shown in Figure 9.

$$CMO = 100\ x\ \frac{sH - sL}{sH + sL} \tag{11}$$

$$sH : The\ sum\ of\ higher\ closes\ over\ N\ periods \quad (12)$$

$$sL : The\ sum\ of\ lower\ closes\ over\ N\ periods \quad (13)$$


Figure 9. CMO chart example on financial chart

## 5.3 Volatility indicators

The domain of these indicators is volatility which aims to measure the rate and depth of price movements. These indicators generally utilize high and low data for a financial asset and are handy in terms of defining ranges for buying and selling operations in a market (Pan, 2003).

In the following sub-sections three popular volatility indicator examples, which are used for this research as well, will be explained further.

### 5.3.1 Bollinger Bands

The indicator is given this name after his inventor Bollinger (2002). The bands for this indicator are volatility bands trying to demonstrate and limit the possible ranges of prices in a certain period.

The indicator is composed of three bands, upper, lower and middle band and among these three, the middle band is just a simple moving average with a certain period for that particular financial asset (Bollinger, 2002).

Bollinger (2002) calculates Bollinger Bands with equations (14), (15) and (16). Example graph of Bollinger Bands is shown in Figure 10.

$$Middle\ Band = N\ day\ simple\ moving\ average \tag{14}$$

$$Upper\ Band = Middle\ Band + ((Standard\ deviation\ of\ price)\ x\ 2) \tag{15}$$

$$Lower\ Band = Middle\ Band - ((Standard\ deviation\ of\ price)\ x\ 2) \tag{16}$$


Figure 10. Bollinger Bands example on financial chart

### 5.3.2 Average True Range (ATR)

Average True Range is an indicator developed by Wilder (1978) which is oriented around measuring the volatility. It is argued that a volatility formula that utilizes the only high-low range is not sufficient enough to capture volatility from gap or limit moves, hence the ATR is introduced.

Wilder (1978) calculates ATR with equations (17), (18), (19), (20) and (21). Example graph of ATR is shown in Figure 11.

$$True\ Range(TR)\ is\ the\ gratest\ of\ Method\ 1, Method\ 2\ and\ Method\ 3 \quad (17)$$

$$Method\ 1 = Current\ High - \ Current\ Low \quad (18)$$

$$Method\ 2 = |Current\ High - Previous\ Close| \quad (19)$$

$$Method\ 3 = |Current\ Low - Previous\ Close| \quad (20)$$

$$ATR(i) = \frac{(ATR(i-1)\ x\ (N-1))+TR(i)}{N} \quad (21)$$


Figure 11. ATR Line example on financial chart

### 5.3.3 Bollinger Bandwidth

Bollinger Bandwidth is an indicator that is derived from the Bollinger band technical indicator (Bollinger, 2002). Falling values in bandwidth indicator show decreasing volatility values and as expected the rising values in bandwidth values show increasing volatility.

Bollinger (2002) calculates Bollinger Bandwidth with equation (22). Example graph of Bollinger Bandwidth is shown in Figure 12.

$$Bollinger\ Bandwidth = \frac{Upper\ Band - Lower\ Band}{Middle\ Band} \quad (22)$$


Figure 12. Bollinger Bandwidth Line example on financial chart

## 5.4 Volume indicators

The subject of these indicators is volume data of the financial assets and through using volume data, they aim to measure the strength of a trend or validate a directional position.

In the following sub-sections three popular volume indicator examples, which are used for this research as well, will be explained further.

### 5.4.1 On Balance Volume

On Balance Volume(OBV) is developed by Granville (1963). Technical analysts use this indicator to search for divergences between OBV and price to understand price movements and trends.

Granville (1983) calculates OBV with equations (23), (24) and (25). Example graph of OBV is shown in Figure 13.

If the close price is above the prior close price,

$$OBV(i) = OBV(i-1) + Volume(i) \tag{23}$$

If the close price is below the prior close price,

$$OBV(i) = OBV(i-1) - Volume(i) \tag{24}$$

If the close price is equal to the prior close price,

$$OBV(i) = OBV(i-1) \tag{25}$$


Figure 13. OBV Line example on financial chart

### 5.4.2 Accumulation Distribution Line (ADL)

The volume-based indicator is developed by a famous trader Marc Chaikin (Utthammajai & Leesutthipornchai, 2015). Technical analysts utilize this indicator to analyse the underlying trend for a financial asset or to predict reversals through a good understanding of divergences from the asset's price.

ADL is calculated with equations (26), (27) and (28) (Utthammajai & Leesutthipornchai, 2015). Example graph of ADL is shown in Figure 14.

$$Money\ Flow\ Multiplier\ (MFM) = \frac{(Close-Low)-(High-Close)}{High-Low} \quad (26)$$

$$Money\ Flow\ Volume(MFV) = MFM\ x\ Volume \quad (27)$$

$$ADL = ADL(i-1) + \text{MFV(i)} \quad (28)$$


Figure 14. ADL Line example on financial chart

### 5.4.3 Chaikin Oscillator

The indicator is found by Marc Chaikin as well as ADL in order to measure the momentum of the ADL using MACD (Utthammajai & Leesutthipornchai, 2015). To generate buy and sell signals based on bullish and bearish divergences, the oscillator basically creates signals below and above the zero lines.

Chaikin Oscillator is calculated with equation (29) (Chen, 2019). Example graph of Chaikin Oscillator is shown in Figure 15.

$$Chaikin\ Oscillator = 3DayEMA(ADL) - 10DayEMA(ADL) \quad (29)$$


Figure 15. Chaikin Oscillator Line example on financial chart

For our research, due to their popularity and availability, along with the given examples for the types of technical indicators above, 156 mathematical inputs will be utilized as input feature set as well as the raw price and volume data in order to predict the direction of cryptocurrency with the best accuracy performance.

# CHAPTER 6

# LITERATURE REVIEW

In this section, the related work is divided and explained under four sub-sections. In the first subsection, different algorithms used in algorithmic trading researches including the traditional machine learning algorithms (i.e. SVM, ANN, Logistic Regression) are exemplified and exhibited. The second subsection composes performance analysis of deep learning algorithms such as LSTM, RNN and DNN derived from the researches. In the third subsection, the usage of technical indicators in algorithmic trading is exemplified from different researches and the results are presented. The final subsection puts forth trend prediction accuracies of deep learning algorithms in the latest researches.

## 6.1 Machine learning for algorithmic trading

Various approaches have been utilized to obtain the best results in terms of accuracy and error rates while predicting the behaviour of a market. Varon and Soroka (2016) used reinforcement learning with Q-Learning algorithms with raw price data to simulate and calculate the annualized return for S&P 500 which resulted in a 12.5% yearly return.

Li, Xie, Wang, Cai, Cao, Wang, Min, and Deng (2016) also utilized machine learning algorithms in order to predict Profit and Loss (PNR) and Sharpe ratio in Support Vector Machine, Back Propagation Neural Network (BP-NN), Bayesian Extreme Learning Machine (B-ELM) and Kernel Extreme Learning Machine (K-ELM) algorithms. They observed the fact that SVM and K-ELM algorithms produce more accurate results and are faster as compared to others. Confirming the

conclusions shown in the work of Li et al. (2016), Wang, Zhang, Rao, Li and Zhang (2017) used the same set of machine learning algorithms to predict the Hong Kong Exchange stock market prices for 2001 and came to the same conclusion that SVM and K-ELM are more accurate compared to others.

In a different research problem, Barboza, Kimura, and Altman (2017) used machine learning algorithms in order to predict bankruptcy of a set of companies located in America and Canada. They showed that relatively newer machine learning algorithms like SVM and Random Forest outperformed traditional approaches such as Multivariate Discriminant Analysis (MDA), Logistic Regression (LR) and ANN with a ratio of around 15%. Using the machine learning methods in a niche domain, Foruzan, Scott and Lin (2015) compared SVM, ANN and Ant Colony Optimization Artificial Neural Networks (ACO-ANN) for electricity price prediction and on the contrary of the some of the previous work, they found SVM to be less successful in terms of error rate production compared to ANN and ACO-ANN. However, they still argue that SVM has the fastest training time among these three.

## 6.2 Deep learning for algorithmic trading

Lecun, Bengio, and Hinton (2015) explain that, with introducing multiple levels of abstraction, Deep Learning enables models, which are composed of multiple layers, to learn representations of data. Deep learning algorithms are composed of multi layered artificial neural networks. These artificial neural networks mimic the neurons in the human brain in order to produce a learning web and learn through a non-linear structure. The differentiating factor of deep learning is the ability of the algorithm to learn through data using a general-purpose learning procedure (LeCun, Bengio & Hinton, 2015). Recurrent Neural Networks, Long Short-Term Memory and

Convolutional Neural Networks (CNN) can be the algorithms that can exemplify deep learning methods.

Alessandretti, ElBahrawy, Aiello, and Baronchelli, (2018) compared the performance of gradient boosting trees with Long Short Term Memory algorithms and argued that for short time windows like 5-10 days, decision trees are performing better. Yet on the other hand LSTM outperformed decision trees in creating more accurate predictions for longer time periods due to its ability to capture long term dependencies. Selvin, Vinayakumar, Gopalakrishnan, Menon, and Soman (2017) applied three different deep learning algorithms on historical minute-based times series data of 1721 NSE listed companies to predict stock price. Among the three algorithms, Convolutional Neural Networks produced better results in terms of error rate compared to LSTM and Recurrent Neural Networks while all of these machine learnings outperformed ARIMA. RNN is a type of neural network which is oriented around forming directed circles by connections between computational units and they utilize their internal memory to be able to process arbitrary sequence of inputs. RNNs are generated with implementing the same set of weights recursively over a graph-like structure and they commonly use hidden layers (Selvin et al., 2017). Although LSTM is also an RNN algorithm, Selvin et al. (2017, p.1645) explain the specialty of LSTM which differs from common RNN algorithms as: “In the case of LSTM architecture, the usual hidden layers are replaced with LSTM cells. The cells are composed of various gates that can control the input flow. An LSTM cell consists of input gate, cell state, forget gate, and output gate. It also consists of sigmoid layer, tanh layer and point-wise multiplication operation. “. They stated that the reason for comparably successful output of CNN is due to its structural independence of data outside of the analysis window. Considering that the analysis is performed in a

minute-based dataset within a highly dynamical system, Selvin et al. (2017) argue that LSTM and RNN might be insufficient in terms of adjusting the dynamical changes in a dataset which causes them, in the end, to be less performant compared to CNN. Chen, Chen, Huang, Huang and Chen (2016) used deep convolutional neural networks (DCNN) with raw data to analyse price prediction accuracy with using three different 2D conversion approaches, Gramian Angular Field (GAF), Moving Average Mapping (MAM) and Double Moving Average Mapping (DMAM). They achieved the best results for the Taiwan stock index with the dataset of the year 2011 and argued that the introduction and presence of artificial operations (automated algorithmic trading) in stock exchange causes a prediction to produce lower accuracy levels. Jiang and Liang (2017) utilized deep reinforcement learning methods to develop a CNN agent to obtain the best portfolio value by making trade decisions with achieving 16.3% final portfolio return. Yet the Passive Aggressive Mean Reversion approach achieved 21.8% final portfolio value with using the same dataset. They also showed that Online Newton Step has a lower risk in trade operations with relatively high Sharpe ratio, despite the final portfolio value of Online Newton Step model could not even pass 3% which again shows higher risk means higher gains.

Persio and Honchar (2016) utilized Multilayer Perceptron, Wavelet-CNN, CNN and RNN in order to predict the binary trend S&P500 stock exchange. They achieved the best result with combining weights of algorithms Wavelet CNN (W-CNN), CNN and LSTM as 0.49, 0.32, 0.19 respectively. The combination of these three methods for raw data produced accuracy around 57% for their model. Fischer and Krauss (2018) also argued that LSTM algorithms can be effectively used in order to get returns from investments by receiving a 0.23% return for 240-day trade

execution on their experiment with their model. Deng, Bao, Kong, Ren and Dai (2016) compared deep learning methods CNN, LSTM, and RNN in terms of trading decisions. According to the results of the research, LSTM achieved the highest accuracy scores compared to CNN, RNN and the method introduced by researchers Fuzzy Deep Direct Reinforcement (FDDR). On the other hand, they argue that these three models make buy & sell decisions more frequently than FDDR and other traditional models. As a natural result, if trading fees are increased the return values obtained by these three algorithms would be much lower.

### 6.3 Technical indicator implementations for algorithmic trading

Technical indicators have also been used in numerous researches in order to obtain better results in terms of price or trend prediction in a financial domain. Chen and Hao (2017) used Support Vector Machine K-Nearest Neighbor (SVM-KNN) algorithms with a bunch of technical indicators in order to compare their performance with feature weighting using dataset derived from the Chinese Stock Market. With receiving better root mean square error (RMSE) scores, they concluded that using feature weighting on technical indicators performs better as compared to the equally weighted feature set for SVM-KNN. With using SVM, ANN and random forest algorithms, Khaidem, Saha and Dey (2016) achieved 85% to 95% accuracy especially for long term trend predictions by including 6 technical indicators in their feature set. Ratto, Merello, Oneto, Ma, Malandri, and Cambria (2018) used 8 best out of 10 technical indicators of their research scope as a feature set for their SVM algorithm in order to predict the trend for most capitalized stocks of NASDAQ. They obtained around 62% accuracy with their model.

Li and Tam (2017) utilized RNN methods LSTM and SVM on the Chinese Stock Market in order to predict the trend of the stocks in the market by feeding them with 51 different technical indicators. They have also applied pre-processing operations on their data set such as Principle Component Analysis (PCA), Independent Component Analysis (ICA) and Partial Least Squares (PLS). According to their analysis, for raw learning models, SVM performs slightly better than LSTM. Yet on the other hand with enhanced models, PCA-LSTM and PLS-LSTM are the best performers in terms of accuracy. Singh and Srivastava (2017) also implemented more than 30 technical indicators while comparing the performances of Deep Neural Networks, RNN and Radial Basis Function Neural Network (RBFNN) for Google Stock Multimedia in terms of price prediction success. The reason why the authors preferred using a single stock for their analysis is to eliminate the effect of different characteristics of different companies in a market. According to their analysis, RNN performed poorly compared to others and DNN is analysed to be the most successful algorithm among three in terms of Hit Rate and the correlation coefficient between the actual and predicted return.

### 6.4 Trend prediction in algorithmic trading with deep learning

Karmiani, Kazi, Nambisan, Shah and Kable (2019) achieved around 1.5% better accuracy results with utilizing LSTM compared to SVM while predicting the trend of popular stocks in the market (i.e. Apple, Amazon, Google, Hp). Song (2018) compared trend predicting capabilities of SVM, LSTM, and GRU. He analysed that LSTM and GRU produce pretty close, yet better results than SVM algorithms by predicting 20 well-known public companies' stocks traded at both NASDAQ and NYSE. Alonso, Estrada, and Moulin (2018) showed with their experiment that the

trend prediction accuracy performance of LSTM is superior to SVM and ANN. For most of the stocks in their training set, LSTM achieved a better hit/accuracy rate compared to SVM and NN.

Arora and Parimala (2019), argued based on the results they achieved that, LSTM performs better than SVM and Back Propagation for predicting the trend of a stock. McNally, Roche, & Caton (2018) received around 52.7% accuracy in terms of trend prediction with a binary prediction approach on the bitcoin domain using LSTM algorithms. The accuracy of the models for RNN and ARIMA analysed to be slightly and relatively lower.

# CHAPTER 7

# METHODOLOGY

In the previous chapters, necessary background information related to the machine learning algorithms, financial analysis, and technical indicators is given together with the literature review regarding the algorithmic trading and machine learning approaches employed in finance domain. Due to the theoretical strengths as well as their successful achievements in the literature, DNN, LSTM, and GRU algorithms are selected as the algorithms to be utilized to build trend prediction models in this research.

As for the financial market, the cryptocurrency market is selected. Three reasons can be given for this selection. First, in terms of being financial subjects, cryptocurrencies are a relatively newer market compared to traditional currencies, stocks, equities, etc. Furthermore, the domain expertise and knowledge are considerably low compared to other markets which makes it a better candidate for the application of deep learning algorithms. Finally, the market itself is not mature yet, not regulated and controlled (Böhme, Christin, Edelman, & Moore, 2015), quite open to unstable actions and even vulnerable for manipulations. Therefore, efficient market hypothesis could hardly be argued for this particular market. Even the situation seems risky from a trader perspective, the state of the market also brings opportunities for higher profits for those who could leverage it.

In this chapter, after explaining the goal of the research, the methodology followed along with the data and features that are used in this research are going to be explained further.

## 7.1 Goal of the research

Normally, the expectation from deep learning algorithms is to optimize the weights for the neuron connections through training in order to obtain the best prediction result for the output values. The relations between the input and output data should automatically be calculated and embedded in the neural network.

However, deep learning algorithms perform better with large amounts of data and the relations between input and output space become stronger and more meaningful when the input space is saturated in terms of amount of available data. Yet for daily stock analysis, researchers practically cannot work on data more than a couple of thousands due to data availability.

In daily trend prediction case, this research argues that, even if deep learning algorithms should be capable of deriving the meaning from input data itself without human interference, introducing practically more meaningful data from that input data itself, such as technical indicators, to the deep learning algorithms may have positive impact on the prediction accuracy performance due to limited amount of available daily input data in the financial time series prediction domain.

That is why, this research will aim to introduce technical indicators as generated input features to be utilized to obtain better trend prediction accuracy scores than an input dataset which is composed of only OHLCV data.

## 7.2 Input dataset

From the cryptocurrency domain, a dataset for Bitcoin, which is considered as a gold reference for the cryptocurrency market, historical data is collected from open API tickers of cryptocurrency markets to use through the experiments of the research. The dataset has 3186 records as shown in Figure 16.

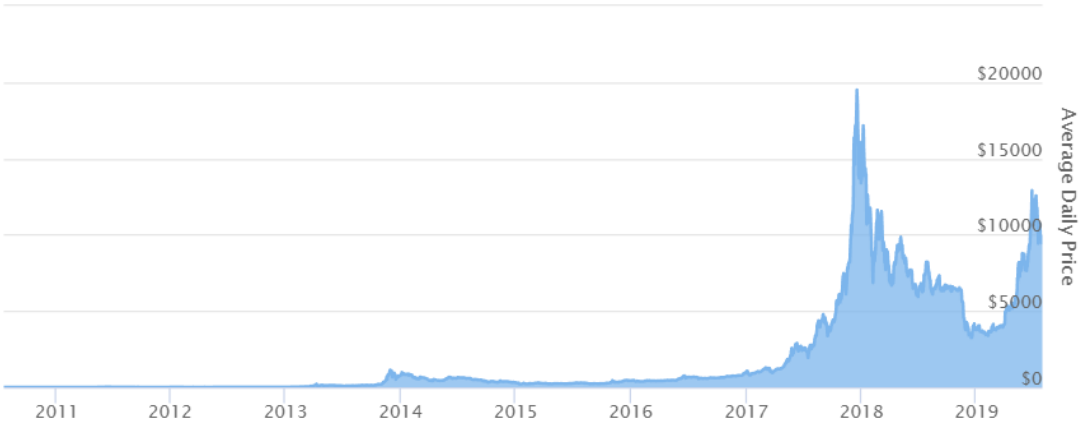



Figure 16. Average Bitcoin Time-Series Daily Price of Dataset

However, as seen in Figure 16, there is an observable imbalance in terms of scale of the price for Bitcoin between its early years and its recent years. Before year 2016, it can be observed that the price of the currency has no significant movements as compared to the movements in the asset's price in recent years.

Supporting, during the experiments, it is observed that the mercurial behaviours in the market, especially for the data rows which belong to early years, and the numerically differentiating scale for the price data was preventing the model to establish a proper generalization. Because in the row which belongs to the oldest record in the dataset, the price is 0.1 dollars whereas it varies between 20000 dollars and 5000 dollars in the last year.

So, in order to prevent training, the model with misleading data, models are trained with the data starting from 1st of January 2016 to 11th of April 2019, as it can be seen in Figure 17, which is composed of 1196 time steps.


Figure 17. Subset of the Input Data for the Experiments

## 7.3 Pre-processing of data

Pre-processing operations are applied on the collected data to transform data into a machine-readable format with moderating JSON responses, replacing incorrect comma characters, inverting the values based on their date to order them properly in time domain and making necessary corrections to establish a finite dataset for standardization process.

Analysing the input dataset, it is seen that there are problematic rows which has infinite or Nan values belong to the dates before 19th August of 2013. So, eliminating that portion of the data not only prevents training the model with misleading data but also eliminating non-computable values from the input dataset.

## 7.4 Calculation of technical indicators and forming input space

In our research, the technical indicators, mathematical functions and financial patterns listed in Table 1 and Table 2 are either used from TA-Lib (https://www.ta-lib.org) or implemented in python and used.

Table 1. List of Technical Indicators

| | | | |
|---|---|---|---|
| Double Exponential Moving Average | Simple Moving Average | Exponential Moving Average | Triple Exponential Moving Average |
| Moving Average Convergence Divergence | Weighted Moving Average | Percentage Price Oscillator | Directional Movement Index |
| Hilbert Period | Hilbert Phase | Hilbert Phasor | Hilbert Sinewave |
| Hilbert Trend | Rate of Change | Stochastic Fast K | Stochastic Fast D |
| Average Directional Index | Average Directional Index Rating | Commodity Channel Index | Chande Momentum Oscillator |
| Absolute Price Oscillator | Aroon Up | Aroon Down | Aroon Oscillator |
| Balance of Power | Money Flow Index | Momentum | Hilbert Instantaneous |
| Minus Directional Indicator | Minus Directional Movement | Plus Directional Indicator | Plus Directional Movement |
| Rate of Change Percentage | Normalized Average True Range | Stochastic Relative Strength Index | Triangular Moving Average |
| Trix | Ultimate Oscillator | Williams R% | Bollinger Bands |
| Kaufman | Midpoint | Midpoint Price | Stochastic K |
| Average True Range | True Range | Stochastic Slow | Stochastic D |
| Chaikin Accumulation Distribution Oscillator | Chaikin Accumulation Distribution Line | On Balance Volume | Relative Strength Index |
| Typical Price | Median Price | Average Price | Weighted Close |
| Beta | Pearson Correlation | Linear Regression | Variance |
| Linear Regression Intercept | Linear Regression Slope | Time Series Forecast | Standard Deviation |
| Linear Regression Angle | Arithmetic Add | Arithmetic Divide | Highest in Period |
| Index of Highest | Lowest in Period | Index of Lowest | Bollinger Bandwidth |
| Arithmetic Multiply | Arithmetic Subtraction | Summation | T3 |

Table 2. List of Financial Patterns and Mathematical Functions

| | | | |
|---|---|---|---|
| Trigonometric Acos | Trigonometric Asin | Trigonometric Cos | Trigonometric Sin |
| Trigonometric Atan | Vector Ceil | Vector Floor | Vector Natural Log |
| Vector 10 Log | Vector Square Root | Trigonometric Tan | Trigonometric Tanh |
| Two Crows | Three Black Crows | Three Line Strike | Abondoned Baby |
| Three Inside Up/Down | Advance Block | Belt-Hold | Breakaway |
| Three Outside Up/Down | Closing Marubozu | Counterattack | Dark Cloud Cover |
| Three Stars in the South | Doji | Doji Star | Dragonfly Doji |
| Three Advancing White Soldiers | Concealing Baby Swallow | Up/Down Gap Side by Side White Line | Modified Hikkake Pattern |
| Engulfing Pattern | Evening Doji Star | Evening Star | Gravestone Doji |
| Harami Cross Pattern | Harami Pattern | Hanging Man | Hammer |
| High-Wave Candle | Hikkake Pattern | Homing Pigeon | Identical Three Crows |
| Kicking Bull – Bear | Kicking | Inverted Hammer | In Neck Pattern |
| Ladder Bottom | Long Legged Doji | Long Line Candle | Marubozu |
| Matching Low | Mat Hold | Morning Doji Star | On Neck Pattern |
| Piercing Pattern | Rickshaw Man | Separating Lines | Shooting Star |
| Rising Falling Three Methods | Upside Gap Two Crows | Unique 3 River | Upside Downside Gap Three Methods |
| Short Line Candle | Spinning Top | Stalled Pattern | Stick Sandwich |
| Tristar Pattern | Thrusting Pattern | Takuri | Tasuki Gap |

While selecting periods for technical indicators, default values of open-source libraries for technical indicators are applied. If no recommendation or application is found related to the technical indicator, short period of 9 and long period of 14 are used in the research.

## 7.5 Standardization

Standardization is used over normalization as per the recommendation of the research of McNally et al. (2016). Data standardization is the operation of rescaling attributes to a form in which their mean value is zero and their standard deviation is one (Raschka, 2015). Standardization is established within the range of (0-1) with a standard scaler used from sklearn library of python (Hackeling, 2014) in order to maintain the data on the same scale to eliminate peaky differences within the feature set.

## 7.6 Output space classification

While labelling the output space for the supervised learning algorithms, a margin representing the one-way transaction cost is selected from transaction cost information of Binance (https://www.binance.com). This margin is selected to represent the transaction cost in order to take financially logical decisions by minimizing the money lost in transactions. The motivation of the trades will be increasing the amount of Bitcoin for the trader and trading decisions are made according to the equations (30), (31), (32) and (33).

$$Movement = (Close_{i+1} - Close_i)/Close_i \quad (30)$$

$$Buy: Movement > Margin \quad (31)$$

$$Sell: Movement < -Margin \quad (32)$$

$$Hold: -Margin \leq Movement \leq Margin \quad (33)$$

For the experiments, transaction cost margin is selected as 0.001 referencing from the one-way transaction cost table retrieved on 5th of May 2019 from Binance (https://www.binance.com/en/fee/schedule). The details and results of the tests will be explained in the experiments section.

Using the transaction cost margin 0.001 resulted in the output classification distribution shown in Table 3. It is seen from Table 3 that the classes in the output space are not uniformly distributed. Buy decisions occupy the largest portion with 53.8% and sell decisions are 41.8% of the output space, whereas hold decisions are only 4.4% of the output space.

Table 3. Class Distribution for Transaction Cost Margin 0.1%

| | |
|---|---|
| Buy | 642 |
| Sell | 499 |
| Hold | 53 |

It is clearly seen from the table that there is an imbalanced distribution between different classes which makes evaluation process of the trained model harder compared to evenly distributed output space. To extend the scope of the research and analyse the model performance for evenly distributed output spaces, model is tested with margin value 0.007 as well which results in the distribution shown in Table 4.

Table 4. Class Distribution for Transaction Cost Margin 0.7%

| | |
|---|---|
| Buy | 470 |
| Sell | 342 |
| Hold | 382 |

Comparing Table 3 and Table 4, which demonstrates the class distribution for different one-way transaction cost margin values, it is clearly seen that margin 0.007 produces a relatively more evenly distributed output space with having the portion for least occurring class “Sell” as 28.6%, whereas this value is around 4.5% for 0.001 margin value for “Hold” class.

## 7.7 Model construction and implementation

For this research, three different learning algorithms, which are DNN, LSTM, and GRU, are implemented in order to test the trend prediction accuracy performance for the bitcoin domain. In the following sub-sections, some of the hyperparameter selection strategies will be explained further with references from previous researches.

### 7.7.1 Number of hidden layers

Heaton (2005) states that the majority of the non-linear functions can be approximated by utilizing only one hidden layer. Supporting, McNally et al. (2016) argue the same statement as well.

Through the experiment phase for this research, it is analysed that introducing additional layers to DNN, LSTM, GRU algorithms did not have a significant effect on the performance of the models except for one test setup. Experiments showed that GRU algorithm for classification margin 0.001 required two additional hidden layers to train more efficiently.

Introducing additional layers when it is not necessary resulted even negatively for DNN, LSTM and GRU algorithms which caused models to predict always with the oversampled class. Even though that approach produced better

performance results for certain experiments, since model should be capable of making predictions for different classes, number of hidden layers is selected as two to ensure the learning model is capable of making predictions for different classes and is still a deep learning model where two hidden layers are required.

### 7.7.2 Batch size

McNally et al. (2016) argue that, despite its effect on training speed, using different values for batch size while training and testing the model does not have a significant effect on the prediction accuracy performance.

Supporting this statement, our experiments as well show that using different values for batch size does not have a significant effect on both loss function and accuracy performance for all three of the models. Values from 1 to 100 are used as batch size through hyperparameter selection tests with increasing the size five by five. However, the only observable effect of this parameter is analysed to be on the training speed.

Masters and Luschi (2018) on the other hand argue that deep learning models achieve the best accuracy score with batch sizes between 2 and 32. So in order to keep the model performance reliable and establish faster training times, batch size is selected as 32.

### 7.7.3 Dropout rate

Implementing dropout layers are quite important in order to prevent early overfitting of the model, enable longer training epochs and establish a better generalization. As suggested by McNally et al. (2016), for the models used in this research, dropout layers are added right after hidden layers.

In the case of our research, dropout layers are added after each hidden layer. Different configurations for dropout layers in experiments show that utilizing different dropout layer coefficient does not have a significant effect on training and test accuracies for saturated models. So, the value of 0.5 is selected similar to McNally et al. (2016) for their dropout layer coefficient for RNNs.

### 7.7.4 Optimizer

Three different optimizers are used during experiments. The first experimented optimizer was Adam optimizer. Kingma and Ba (2017) argue that Adam optimizer collectively has the advantages of both AdaGrad and RMSProp optimizers and performs sufficiently well for a wide spectrum of machine learning problems.

The second optimizer which is tested for this research is Nadam optimizer which incorporates Nesterov momentum into Adam optimizer. Research of Dozat (2016) shows that the loss function with the utilization of Nadam optimizer is expected to have lower values compared to Adam and RMSprop optimizer for both train and validation datasets.

Third and final optimizer tested for this research is RMSProp optimizer which is used and recommended in the research of McNally et al. (2016)

Experiments showed that RMSprop optimizer performed significantly better in terms of reducing training and validation loss values for both of the models with all variations of hyperparameters. As a result, RMSprop optimizer is used for the DNN, LSTM and GRU models of this research.

### 7.7.5 Temporal length

Temporal length is the window size of the data which is fed to the recurrent neural network algorithms. This parameter is the historical dependency, or in other words, the memory size of recurrent neural networks.

The temporal lengths from 10 days to 100 days are tested with increments of 10 for LSTM and GRU algorithms in this research. Experiments show that both LSTM and GRU algorithms performed better with longer temporal lengths which is an indication of their capability of handling long term dependencies.

Considering that in this research a fixed size of 1196 data rows is used, increase in the temporal length results in less number of samples to be used in testing.

### 7.7.6 Early stop configuration to avoid overfitting

As recommended by McNally et al. (2016), an early stopper is configured for the models which are implemented in the research. The early stopping parameter is selected as validation loss and the patience level is set to 5 (McNally et al., 2016). This ensures that the model does not overfit over training periods.

In the next chapter, the experiments will be put forth and explained further for DNN, LSTM and GRU models with referencing hyperparameters which are not explained in this chapter such as number of epochs and number of hidden nodes as the changes in those hyperparameters are observed to be significant for the performances of the models used in this research.

### 7.7.7 Number of neurons for hidden layers

The number of neurons in the hidden layers are optimized empirically. Decreasing validation loss parameter through epochs indicates that model keeps learning. While selecting the number of hidden neurons, starting from the size 10, neuron sizes are increased by 2 between 10 and 100 to achieve the lowest validation accuracy loss score along with a reasonable validation accuracy. While optimizing this parameter, early stopper explained in Section 7.7.6 is utilized in order to make sure that model is not overfitting.

### 7.7.8 Model validation and testing

In model validation and testing, application of k-fold cross validation is not suitable as the data itself is time series and not stateless. Mittal and Goel (2011) also argue that testing on past values with a model trained on future values does not have any meaning. For the model validation, simplified cross validation methodology (Ghiassi & Saidane, 2005) is applied in which a validation set of 20% from the training data is used. This ensures that the model is not underfitted and overfitting with monitoring and early stopping the training process with call-back functions to validation loss or validation accuracy.

The same model validation approach is used by Guresen, Kayakutlu, and Daim (2011) and McNally et al. (2016) as well. Ultimately for this research, 60% of the data is used for training, 20% of data used for validation and the remaining 20% of the data is used for testing.

# CHAPTER 8

# EXPERIMENTS AND RESULTS

In this chapter, the experiments performed with using DNN, LSTM and GRU algorithms will be explained further for different models.

For the margin value 0.001, models are picked according to the accuracy score. However, they needed further analysis for individual class prediction performances due to the class distribution imbalance explained in section 7.6. For margin 0.007, model is picked according to the accuracy value and the class distribution for margin value 0.007 is quite close to be called as uniformly distributed.

## 8.1 Deep neural networks (DNN)

Experiments with deep neural networks showed that models performed efficiently with utilization of two hidden layers. With the empirical optimization explained in methodology section, the configuration shown in Table 4 is used for the DNN model in experiments.

Using the configuration shown in Table 5, the results shown in Table 6 are achieved. The results show that utilization of technical indicators achieved around 4% better accuracy for DNN model compared to using open, high, low, close and volume data for the margin 0.001 for which the output space distribution is imbalanced. Yet it is clearly seen from Table 6 that utilizing technical indicators resulted in better score performances in all of the listed metrics compared to OHLCV data.

Table 5. Hyperparameters of DNN Models

| Margin 0.001 | | | | Margin 0.007 | | | |
|---|---|---|---|---|---|---|---|
| Open, High, Close, Low, Volume (OHLCV) | | Technical Indicators (TI) | | Open, High, Close, Low, Volume (OHLCV) | | Technical Indicators (TI) | |
| Number of Hidden Nodes | 134 | Number of Hidden Nodes | 32 | Number of Hidden Nodes | 22 | Number of Hidden Nodes | 30 |
| Number of Epochs | 9 | Number of Epochs | 9 | Number of Epochs | 12 | Number of Epochs | 11 |

Table 6. Performance Results of DNN Models

| Metrics | Margin 0.001 | | Margin 0.007 | |
|---|---|---|---|---|
| | OHLCV | Technical Indicators | OHLCV | Technical Indicators |
| Test Accuracy | 0.5021 | 0.5401 | 0.3924 | 0.4430 |
| Precision Macro | 0.3510 | 0.3871 | 0.2870 | 0.4715 |
| Precision Weighted | 0.4989 | 0.5490 | 0.3289 | 0.4958 |
| Recall Macro | 0.3414 | 0.3701 | 0.3700 | 0.4615 |
| F1 Macro | 0.2592 | 0.3266 | 0.2976 | 0.4431 |
| F1 Weighted | 0.3926 | 0.4716 | 0.3296 | 0.4453 |

Analysing the experiment for margin value 0.007, for which the output space is almost evenly distributed, it is seen that using technical indicators produced better results for all of the listed metrics. Utilization of technical indicators achieved around 5% better accuracy performance compared to OHLCV data. Additionally, the performance difference for the other metrics are even more significant than 5%.

However, in order to understand the model performance for margin 0.001 better, confusion matrixes need to be used to understand model's capability of predicting different classes in dataset. Because the accuracy itself is not sufficient for analysing the performance of a model in cases of imbalanced distributions in classes. Confusion matrix shown in Table 7 is produced for OHLCV and Technical Indicators dataset for further analysis.

Analysing the results in Table 7, it is seen that even though the accuracy results seems promising, DNN models fail to learn under-sampled classes both for technical indicators and OHLCV data. For both dataset model could not even recognize the least sampled class “Hold” and never attempted to make a prediction for it.

Table 7. Confusion Matrix of DNN Models for Margin 0.001

| OHLCV | | | | | Technical Indicators | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Actual | Buy | 109 | 0 | 8 | Actual | Buy | 105 | 0 | 12 |
| | Hold | 12 | 0 | 0 | | Hold | 11 | 0 | 1 |
| | Sell | 98 | 0 | 10 | | Sell | 85 | 0 | 23 |
| | | Buy | Hold | Sell | | | Buy | Hold | Sell |
| | Predicted | | | | | Predicted | | | |

However, confusion matrix distribution for technical indicators show that model built with technical indicators achieved better individual accuracy scores for Buy and Sell decisions compared to OHLCV. For TI, Buy prediction accuracy score is 52.2% whereas it is 49.7% for OHLCV. Sell prediction accuracy score is 63.9% for TI, whereas it is 55.6% for OHLCV.

Using margin value as 0.007, confusion matrix shown in Table 8 is produced for OHLCV and Technical Indicators dataset for further analysis.

Table 8. Confusion Matrix of DNN Models for Margin 0.007

| OHLCV | | | | | Technical Indicators | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Actual | Buy | 57 | 19 | 0 | Actual | Buy | 33 | 14 | 29 |
| | Hold | 64 | 36 | 0 | | Hold | 24 | 36 | 40 |
| | Sell | 47 | 14 | 0 | | Sell | 18 | 7 | 36 |
| | | Buy | Hold | Sell | | | Buy | Hold | Sell |
| | Predicted | | | | | Predicted | | | |

Analysing the Table 8, it is seen that for Margin 0.007, likewise general accuracy performance, technical indicators produced better individual class

prediction results compared to OHLCV. Success rate of Buy, Hold, and Sell decisions for TIs are 44.0%, 63.2%, and 34.3% respectively. Success rates of the same decisions for OHLCV are 34.0%, 52.2% and 0%. Even for uniform distributed classes with margin 0.007 model could not learn the third class with OHLCV data.

## 8.2 Long Short-Term Memory (LSTM)

Experiments with LSTM cells showed that model performed efficiently with utilization of two hidden layers for every temporal length within the scope of this research. However, number of hidden nodes to obtain optimal results were different for different temporal length values. Number of epochs and number of hidden nodes to obtain best performing model are shown in Table 9.

Table 9. Hyperparameters of LSTM Models

| Temporal Length | Margin 0.001 | | | | Margin 0.007 | | | |
|---|---|---|---|---|---|---|---|---|
| | OHCLV | | TI | | OHLCV | | TI | |
| | # of Hidden Nodes | # of Epochs | # of Hidden Nodes | # of Epochs | # of Hidden Nodes | # of Epochs | # of Hidden Nodes | # of Epochs |
| 10 | 14 | 19 | 14 | 19 | 26 | 11 | 94 | 7 |
| 20 | 14 | 9 | 18 | 17 | 38 | 7 | 70 | 7 |
| 30 | 14 | 16 | 16 | 20 | 40 | 7 | 22 | 12 |
| 40 | 14 | 11 | 16 | 26 | 88 | 7 | 84 | 7 |
| 50 | 14 | 10 | 16 | 22 | 72 | 7 | 10 | 14 |
| 60 | 14 | 13 | 14 | 23 | 46 | 7 | 52 | 12 |
| 70 | 14 | 11 | 14 | 21 | 18 | 7 | 52 | 8 |
| 80 | 14 | 13 | 18 | 19 | 50 | 7 | 70 | 7 |
| 90 | 14 | 10 | 18 | 22 | 42 | 7 | 70 | 11 |
| 100 | 14 | 12 | 40 | 18 | 46 | 7 | 52 | 21 |

Using the hyperparameters shown in Table 9, the obtained accuracy and F1 performance scores are shown in Table 10. It can be seen from Table 10 that,

utilizing technical indicators produced better accuracy and F1 scores for all of the temporal lengths for both margin values.

Table 10. Performance Results of LSTM Models

| | Margin 0.001 | | | | Margin 0.007 | | | |
|---|---|---|---|---|---|---|---|---|
| | OHCLV | | TI | | OHLCV | | TI | |
| Temporal Length | Acc. | F1 Weighted | Acc. | F1 Weighted | Acc. | F1 Weighted | Acc. | F1 Weighted |
| 10 | 0.4912 | 0.3236 | 0.4956 | 0.4969 | 0.3377 | 0.2278 | 0.4824 | 0.4129 |
| 20 | 0.4862 | 0.3182 | 0.5183 | 0.5023 | 0.3532 | 0.3344 | 0.4908 | 0.4364 |
| 30 | 0.4856 | 0.3174 | 0.5000 | 0.4812 | 0.3510 | 0.3149 | 0.4663 | 0.3927 |
| 40 | 0.4899 | 0.3221 | 0.5202 | 0.5000 | 0.3485 | 0.3126 | 0.4848 | 0.4001 |
| 50 | 0.4893 | 0.3216 | 0.5212 | 0.5001 | 0.3404 | 0.3058 | 0. 5106 | 0. 4636 |
| 60 | 0.4888 | 0.3209 | 0.5449 | 0.5272 | 0.3258 | 0.2819 | 0.5056 | 0.4435 |
| 70 | 0.5059 | 0.3400 | 0.5416 | 0.5270 | 0.3274 | 0.2598 | 0.4762 | 0.3753 |
| 80 | 0.5127 | 0.3474 | 0.5633 | 0.5509 | 0.3417 | 0.2750 | 0.4620 | 0.4014 |
| 90 | 0.5202 | 0.3560 | 0.5540 | 0.5416 | 0.3581 | 0.2977 | 0.4864 | 0.4430 |
| 100 | 0.5362 | 0.3743 | 0.5434 | 0.5316 | 0.3696 | 0.3283 | 0.4783 | 0.3855 |

Analysing the results for margin 0.007 from Table 10, it is seen that using technical indicators as feature set produced significantly better accuracy and F1 weighted results for every temporal length from 10 to 100. For margin 0.007, the best performing temporal length for LSTM algorithm is analysed to be 50 days. Considering that the output space is evenly distributed, obtaining the accuracy result 51.06% with technical indicators is significantly better compared to OHLCV data with a positive difference of around 14.1%.

The best performing temporal length for OHLCV data for margin 0.007 in terms of accuracy performance is 100 days. Even for temporal length of 100 days, model using technical indicators achieved better performance and F1 Weighted scores compared to OHLCV data.

Again, in order to understand the model performance for margin 0.001 better, confusion matrixes need to be used to understand model's capability of predicting

different classes due to imbalanced class distribution for margin 0.001. The models which produce the best accuracy results for margin 0.001 is for temporal length 100 for OHLCV data and temporal length 80 for technical indicator data. The confusion matrixes produced for these models are shown in Table 11.

Table 11. Confusion Matrix of LSTM Models for Margin 0.001

| OHLCV Temporal Length 100 | | | | | Technical Indicators Temporal Length 80 | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Actual | Buy | 74 | 0 | 0 | Actual | Buy | 50 | 0 | 31 |
| | Hold | 6 | 0 | 0 | | Hold | 2 | 0 | 5 |
| | Sell | 58 | 0 | 0 | | Sell | 31 | 0 | 39 |
| | | Buy | Hold | Sell | | | Buy | Hold | Sell |
| | Predicted | | | | | Predicted | | | |

Results clearly show that the LSTM model for which OHLCV data is used could not learn the under-sampled classes efficiently and predicted the oversampled class “Buy” for all even for its best accuracy performance in test set. However, the LSTM model which is trained by technical indicator predicted “Buy” class with 60.2% and predicted “Sell” class with 52.0%. Although this model as well could not predict any output values for class “Hold”, the results it achieves for “Buy” and “Hold” decisions are significantly better compared to OHLCV data.

Using margin value as 0.007, confusion matrix shown in Table 12 is produced for OHLCV and Technical Indicators dataset for further analysis.

Table 12. Confusion Matrix of LSTM Models for Margin 0.007

| OHLCV Temporal Length 100 | | | | | Technical Indicators Temporal Length 80 | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Actual | Buy | 33 | 12 | 2 | Actual | Buy | 25 | 26 | 4 |
| | Hold | 36 | 15 | 4 | | Hold | 14 | 66 | 4 |
| | Sell | 28 | 5 | 3 | | Sell | 12 | 32 | 5 |
| | | Buy | Hold | Sell | | | Buy | Hold | Sell |
| | Predicted | | | | | Predicted | | | |

Analysing the Table 12, it is seen that for Margin 0.007, likewise general accuracy performance, technical indicators produced better individual class prediction results compared to OHLCV. Success rate of Buy, Hold, and Sell decisions for TIs are 49.0%, 53.2%, and 38.4% respectively. Success rates of the same decisions for OHLCV are 34.0%, 46.9% and 33.3%.

Additional metrics for best performing LSTM models in terms of accuracy with temporal length 50 for margin 0.007 for technical indicators, temporal length 100 for margin 0.007 for OHLCV data, temporal length 80 for margin 0.001 for technical indicators and temporal length 100 for margin 0.001 for OHLCV data are shown in Table 13.

Table 13. Additional Performance Metrics for LSTM Models with Best Accuracy

| Metrics | Margin 0.001 | | Margin 0.007 | |
|---|---|---|---|---|
| | OHLCV | Technical Indicators | OHLCV | Technical Indicators |
| | Temporal Length 100 | Temporal Length 80 | Temporal Length 100 | Temporal Length 40 |
| Test Accuracy | 0.5362 | 0.5633 | 0.3696 | 0.5106 |
| Precision Macro | 0.1787 | 0.3741 | 0.3808 | 0.4690 |
| Precision Weighted | 0.2875 | 0.5392 | 0.3896 | 0.4815 |
| Recall Macro | 0.3333 | 0.3915 | 0.3527 | 0.4474 |
| F1 Macro | 0.2327 | 0.3825 | 0.3121 | 0.4225 |
| F1 Weighted | 0.3744 | 0.5509 | 0.3283 | 0.4635 |

Analysing Table 13, from a general approach, it can be said that utilization of technical indicators for both margin values 0.001 and 0.007 produced better results in terms of monitored metrics.

## 8.3 Gated Recurrent Units (GRU)

Experiments with GRU cells showed that model required additional hidden layers to handle long term dependencies after certain levels of temporal length. Similar to LSTM algorithms, number of hidden layers to obtain optimal results were also different for different temporal length values.

Using the hyperparameters shown in Table 14, accuracy and F1 performance scores shown in Table 15. It can be seen from Table 15 that, likewise DNN and LSTM algorithms, utilizing technical indicators produced better accuracy and F1 scores for all of the temporal lengths for both margin values for GRU models.

Table 14. Hyperparameters of GRU Models

| Temporal Length | Margin 0.001 | | | | Margin 0.007 | | | |
|---|---|---|---|---|---|---|---|---|
| | OHCLV | | TI | | OHLCV | | TI | |
| | # of Hidden Nodes | # of Epochs | # of Hidden Nodes | # of Epochs | # of Hidden Nodes | # of Epochs | # of Hidden Nodes | # of Epochs |
| 10 | 16 | 19 | 24 | 17 | 12 | 10 | 50 | 11 |
| 20 | 16 | 13 | 28 | 25 | 18 | 10 | 50 | 11 |
| 30 | 16 | 15 | 10 | 46 | 14 | 10 | 50 | 7 |
| 40 | 16 | 27 | 24 | 26 | 20 | 8 | 50 | 13 |
| 50 | 16 | 14 | 26 | 19 | 38 | 7 | 40 | 8 |
| 60 | 16 | 16 | 24 | 16 | 18 | 9 | 80 | 7 |
| 70 | 16 | 14 | 24 | 18 | 42 | 7 | 46 | 12 |
| 80 | 16 | 16 | 14 | 14 | 18 | 7 | 50 | 9 |
| 90 | 16 | 8 | 28 | 40 | 18 | 8 | 14 | 12 |
| 100 | 16 | 9 | 14 | 26 | 86 | 9 | 14 | 10 |

Analysing the results for margin 0.007 from Table 15, likewise DNN and LSTM algorithms, it is seen that using technical indicators as feature set produced significantly better accuracy and F1 weighted results for every temporal length from 10 to 100. For margin 0.007, the best performing temporal length for GRU algorithm is analysed to be 10 days. Considering that the output space is almost evenly

distributed, obtaining the accuracy result 52.19% with technical indicators is significantly better compared to OHLCV data with a difference of around 17.1%.

Table 15. Performance Results of GRU Models

| | Margin 0.001 | | | | Margin 0.007 | | | |
|---|---|---|---|---|---|---|---|---|
| | OHCLV | | TI | | OHLCV | | TI | |
| Temporal Length | Acc. | F1 Weighted | Acc. | F1 Weighted | Acc. | F1 Weighted | Acc. | F1 Weighted |
| 10 | 0.4912 | 0.3236 | 0.4824 | 0.4847 | 0.3509 | 0.3083 | 0.5219 | 0.4613 |
| 20 | 0.4862 | 0.3181 | 0.5229 | 0.5062 | 0.3486 | 0.3287 | 0.5046 | 0.4371 |
| 30 | 0.4856 | 0.3174 | 0.5048 | 0.4910 | 0.3510 | 0.3013 | 0.4951 | 0.4573 |
| 40 | 0.4899 | 0.3221 | 0.5151 | 0.4919 | 0.3535 | 0.3212 | 0.5101 | 0.4495 |
| 50 | 0.4893 | 0.3215 | 0.5319 | 0.5107 | 0.3457 | 0.3026 | 0.5053 | 0.4842 |
| 60 | 0.4888 | 0.3209 | 0.5280 | 0.5093 | 0.3371 | 0.2852 | 0.5000 | 0.4030 |
| 70 | 0.5059 | 0.3399 | 0.5357 | 0.5228 | 0.3512 | 0.3029 | 0.4881 | 0.4628 |
| 80 | 0.5127 | 0.3474 | 0.5379 | 0.5073 | 0.3671 | 0.3121 | 0.5127 | 0.4906 |
| 90 | 0.5202 | 0.3560 | 0.5405 | 0.5294 | 0.3851 | 0.3298 | 0.4662 | 0.4436 |
| 100 | 0.5362 | 0.3743 | 0.5289 | 0.4996 | 0.3985 | 0.3535 | 0.4565 | 0.4266 |

The best performing temporal length for OHLCV data for margin 0.007 in terms of accuracy performance is 100 days. Even for temporal length of 100 days, GRU model using technical indicators achieved better performance and F1 Weighted scores compared to OHLCV data.

In order to analyse the model performance for margin 0.001 efficiently, due to imbalanced class distribution for margin 0.001, confusion matrixes need to be used to understand model's capability of predicting different classes. The GRU model which produces the best accuracy result for margin 0.001 are the model with temporal length 90 for technical indicators and the model with temporal length 100 for OHLCV data. The confusion matrixes produced for GRU model's temporal length 90 and 100 are shown in Table 16.

Table 16. Confusion Matrix of GRU Models for Margin 0.001

| OHLCV Temporal Length 100 | | | | | Technical Indicators Temporal Length 90 | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Actual | Buy | 74 | 0 | 0 | Actual | Buy | 46 | 0 | 31 |
| | Hold | 6 | 0 | 0 | | Hold | 3 | 0 | 3 |
| | Sell | 58 | 0 | 0 | | Sell | 31 | 0 | 34 |
| | | Buy | Hold | Sell | | | Buy | Hold | Sell |
| | Predicted | | | | | Predicted | | | |

Results in Table 16 clearly show that, similar to the LSTM model the GRU model for which OHLCV data is used could not learn the under-sampled classes efficiently and predicted the oversampled class “Buy” for all even for its best accuracy performance in test set. However, the GRU model which is trained by technical indicator predicted “Buy” class with 57.5% and predicted “Sell” class with 50.0%. Although this model as well could not predict any output values for class “Hold”, the results it achieves for “Buy” and “Hold” decisions are significantly better compared to OHLCV data.

Using margin value as 0.007, confusion matrix shown in Table 17 is produced for OHLCV and Technical Indicators dataset for further analysis.

Table 17. Confusion Matrix of GRU Models for Margin 0.007

| OHLCV | | | | | Technical Indicators | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Actual | Buy | 33 | 13 | 1 | Actual | Buy | 23 | 35 | 3 |
| | Hold | 35 | 20 | 0 | | Hold | 6 | 89 | 4 |
| | Sell | 28 | 6 | 2 | | Sell | 16 | 45 | 7 |
| | | Buy | Hold | Sell | | | Buy | Hold | Sell |
| | Predicted | | | | | Predicted | | | |

Analysing the Table 17, it is seen that for Margin 0.007, likewise general accuracy performance, technical indicators produced better individual class prediction results compared to OHLCV. Success rate of Buy, Hold, and Sell decisions for TIs are 51.1%, 52.6%, and 50.0% respectively. Success rates of the same decisions for OHLCV are 34.4%, 51.2% and 66.7%. For OHLCV data since

the model attempted to predict majority of its decisions as “Buy” for which model is not quite successful, the overall accuracy performance of OHLCV reduced significantly, whereas the balanced success of the model for TIs produced a better overall accuracy result.

Additional metrics for best performing GRU models in terms of accuracy with temporal length 10 for margin 0.007 for technical indicators, temporal length 100 for margin 0.007 for OHLCV data, temporal length 90 for margin 0.001 for technical indicators and temporal length 100 for margin 0.001 for OHLCV data are shown in Table 18.

Table 18. Additional Performance Metrics for GRU Models with Best Accuracy

| Metrics | Margin 0.001 | | Margin 0.007 | |
|---|---|---|---|---|
| | OHLCV | Technical Indicators | OHLCV | Technical Indicators |
| | Temporal Length 100 | Temporal Length 90 | Temporal Length 100 | Temporal Length 10 |
| Test Accuracy | 0.5362 | 0.5405 | 0.3986 | 0.5219 |
| Precision Macro | 0.1787 | 0.3583 | 0.5077 | 0.5126 |
| Precision Weighted | 0.2875 | 0.5188 | 0.4954 | 0.5150 |
| Recall Macro | 0.3333 | 0.3735 | 0.3738 | 0.4479 |
| F1 Macro | 0.2327 | 0.3658 | 0.3299 | 0.4184 |
| F1 Weighted | 0.3744 | 0.5294 | 0.3535 | 0.4613 |

Analysing Table 18, from a general perspective, it can be said that utilization of technical indicators for both margin values 0.001 and 0.007 produced better results in terms of monitored metrics.

## 8.4 Results

Results for three different deep learning models show that using technical indicators in feature set produced better test accuracy results compared to using raw open, high, close, low and volume data.

For the margin value 0.007, the output space is evenly distributed. Comparing the accuracy results for the evenly distributed output set, GRU achieved best performance in terms of accuracy as shown in Table 19.

Table 19. Accuracy Performance Comparison of Models for Margin 0.007

| Algorithm | Accuracy Score with Technical Indicators | Accuracy Score with OHLCV | Individual Class Prediction Score with Technical Indicators | Individual Class Prediction Score with OHLCV |
|---|---|---|---|---|
| Deep Neural Network (DNN) | 44.30% | 39.24% | Buy 44.0% | Buy 34.0% |
| | | | Hold 63.2% | Hold 52.2% |
| | | | Sell 34.3% | Sell 0% |
| Long Short-Term Memory (LSTM) | 51.06% | 36.96% | Buy 49.0% | Buy 34.0% |
| | | | Hold 53.2% | Hold 46.9% |
| | | | Sell 38.4% | Sell 33.3% |
| Gated Recurrent Unit (GRU) | 52.19% | 39.85% | Buy 51.1% | Buy 38.4% |
| | | | Hold 52.6% | Hold 51.2% |
| | | | Sell 50.0% | Sell 66.6% |

Analysing the table, it can be said that with using technical indicators, all three models achieved overall accuracy scores close to 50% for an evenly distributed output space in terms of class samples. It is shown that utilization of technical indicators significantly increased the performance of the models to establish better prediction values in a multiclass, complex, non-linearity time series prediction problem.

For margin value 0.001, it is analysed that all three models for which OHLCV data is used were inefficient in terms of learning under-sampled classes. It is seen from confusion matrixes shown in Table 7, Table 11 and Table 16, that the models predicted with oversampled class for majority of their predictions. However, utilizing technical indicators for the same margin produced significant improvement for learnings of the model which can be seen from confusion matrixes shown in Table 7, Table 11 and Table 16.

Verifying that the models actually learn with utilization of technical indicators and can predict different classes of the output space, accuracy score comparison for the three models are shown in Table 20. For margin 0.001, LSTM achieved best accuracy score with reasonable individual class prediction scores.

Table 20. Accuracy Performance Comparison of Models for Margin 0.001

| Algorithm | Accuracy Score with Technical Indicators | Accuracy Score with OHLCV | Individual Class Prediction Score with Technical Indicators | Individual Class Prediction Score with OHLCV |
|---|---|---|---|---|
| Deep Neural Network (DNN) | 54.01% | 50.21% | Buy 52.2% | Buy 49.8% |
| | | | Hold 0% | Hold 0% |
| | | | Sell 63.8% | Sell 55.6% |
| Long Short-Term Memory (LSTM) | 56.33% | 53.62% | Buy 60.2% | Buy 53.6% |
| | | | Hold 0% | Hold 0% |
| | | | Sell 52.0% | Sell 0% |
| Gated Recurrent Unit (GRU) | 54.05% | 53.62% | Buy 57.5% | Buy 53.6% |
| | | | Hold 0% | Hold 0% |
| | | | Sell 50.0% | Sell 0% |

As seen in Table 20, with the imbalanced output space in terms of class distribution, none of the models could learn a pattern to recognize the class which

has the least number of samples “Hold”. However, when trained with technical indicators, all three models produced reasonable and acceptable values for predicting Buy and Sell decisions for margin 0.001.

Considering that this research takes the one-way transaction cost into account while predicting, the results also show parallelism with the Stylised Fact number four which is argued by Menkhoff and Taylor (2006). The fact number four argues that the profitability through technical analysis on currencies can still be achieved in spite of the existence of transaction costs and interest rate costs.

# CHAPTER 9

# CONCLUSIONS AND FUTURE WORK

McNally et al. (2016) state that utilizing technical indicators or feature engineering should not make significant differences in terms of prediction performance as deep learning algorithms are expected to be capable of picking meanings from data through further training executions.

However, this research showed that utilization of technical indicators in feature set produces better prediction accuracy performances compared to using solely raw open, high, close, low and volume data for feed forward DNNs, LSTM and GRU models with ten different temporal lengths.

For all the monitored metrics such as Accuracy, Precision, Recall and F1, the models trained with technical indicators achieved significantly better performance results for both imbalanced and balanced output spaces in terms of class distribution. It can be derived from the results that since the amount of data for daily technical analysis will be limited compared to minute-based data or high frequency trading (HFT) applications, utilization of technical indicators should bring significant advantages to traders.

Considering that the number of rows in our dataset is limited for deep learning applications, as a future work, minute-based data can be collected and utilized to investigate whether deep learning models can learn non-linear relationships attained by technical indicators employed in this research. Because with utilization of more data, the deep learning models might be able to construct non-linear relationships directly from raw data more effectively. Expanding the dataset could also produce reasonable individual class prediction scores for OHLCV data.

Additionally, for this research technical indicator feature set is directly fed into the model without applying any feature selection methodologies and it is observed that deep learning methodologies are capable of handling a wide feature set even with a limited amount of data. However, application of feature selection methodologies might have positive effects on the accuracy prediction performance with eliminating irrelevant features from the input set beforehand.

Moreover, in order to understand the best contributing set of technical indicators to prediction performance, technical indicators can be eliminated one by one or by groups and resulting features that are left can be used to generate the models and test their prediction performances. While eliminating the technical indicators to form technical indicator groups, a strategy which is oriented around eliminating highly correlated technical indicators can be followed.

As an additional feature selection strategy, technical indicators can be grouped based on their categories and a category-based analysis can be applied to analyse the prediction power of different technical indicator categories. Furthermore, feature set can be extended with the technical indicators which are not available in TALib python library. Expanding the feature set with new technical indicators could embed more meaning to the input set and can eventually help models to produce better results.

In addition to the technical indicators, in order to improve the prediction performance, models can be fed with time-series data of various financial assets such as stocks, options, indices or data which belongs to different currency parities such as USD/EURO and TL/USD.

As another future research topic, learning models can be combined with the components which are provided by fundamental analysis methodologies such as

macroeconomic factors, industrial trends and regulations to be able to analyse the relations of Bitcoin movements with global fundamental factors.

This research showed that increasing the ratio of number of samples to establish uniformity helps model to recognize all the classes in the dataset. Another research topic can be tuning the margin value to detect the certain points for which model can recognize all the classes in the network and attempt making predictions for all of the classes.

As another possible future research, the same prediction architecture and approach can be used to predict two, three days or one week of output periods instead of a single day to have a projection on a future period and establish a clearer perspective towards the upcoming movements in the market.

Furthermore, the data fed to the model can be grouped under subsets arbitrarily based on the observed structural breaks in the historical time-series of Bitcoin (Lento, 2008). Since in between the structural breaks, the market behaviour is expected to be more stable, the models can achieve better prediction scores in between two structural breaks.

Additionally, models can be tested with different subsets of data which shares the same characteristics in terms of price distribution between different structural breaks. With leveraging the similar patterns between structural breaks, which are isolated arbitrarily from the charts or through statistical analysis, models can be tested against different test data to measure and validate the prediction performance with different data groups. Moreover, the technical indicator input set used for this research are constructed with the default period parameters. Optimizing these period parameters of technical indicators for Bitcoin domain can also have positive effects on the prediction performance.

# REFERENCES


Alessandretti, L., ElBahrawy, A., Aiello, L. M., & Baronchelli, A. (2018) Anticipating cryptocurrency prices using machine learning. *Complexity, 2018,* 1-16. https://doi.org/10.1155/2018/8983590

Alonso, M.N., Estrada, G.B., & Moulin, A. (2019). Deep learning in finance: prediction of stock returns with Long-Short Term Memory Networks. In Guida, T. (Ed.), *Big data and machine learning in quantitative investment*. Wiley, Cornwall, UK. doi: https://doi.org/10.1002/9781119522225.ch13

Appel, G. (1979). *The Moving Average Convergence-Divergence method.* Great Neck, NY: Signalert.

Arora, N., & Parmiala, M. (2019). *Financial analysis: Stock market prediction using deep learning algorithms.* Paper presented at International Conference on Sustainable Computing in Science, Technology & Management (SUSCOM-2019). doi: https://dx.doi.org/10.2139/ssrn.3358252

Ata, R. (2015). Artificial neural networks applications in wind energy systems: A review. *Renewable and Sustainable Energy Reviews, 49*, 534-562. doi: https://doi.org/10.1016/j.rser.2015.04.166

Ayodele, O. T. (2010). Types of machine learning algorithms. In Y. Zagang (Eds.) *New advances in machine learning.* Portsmouth, U.K: Intech. Retrieved from http://cdn.intechweb.org/pdfs/10694.pdf

Barboza, F., Kimura, H., & Altman, E. (2017). Machine learning models and bankruptcy prediction. *Expert Systems with Applications, 83,* 405–417. doi: https://doi.org/10.1016/j.eswa.2017.04.006

Bollinger, J. (2002). *Bollinger on Bollinger Bands.* McGraw-Hill, New York.

Böhme, R., Christin, N., Edelman, B, & Moore, T. (2015). Bitcoin: Economics, technology, and governance. *Journal of Economic Perspectives, 29*(2): 213-238. doi: 10.1257/jep.29.2.213

Chande, T. S., & Kroll, S. (1994). *The new technical trader: Boost your profit by plugging into the latest indicators.* Wiley Finance Editions, New York, USA. Retrieved from http://dl.fxf1.com/files/books/english/Chande%20Kroll%20-%20The%20New%20Technical%20Trader.pdf

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*. Paper presented at 2016 7th International Conference on Cloud Computing and Big Data (CCBD). doi: 10.1109/CCBD.2016.027

Chen, T., & Guestrin, C. (2016). *XGBoost: A scalable tree boosting system*. Paper presented at Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM. doi: 10.1145/2939672.2939785

Chen, Y., & Hao, Y. (2017). A feature weighted support vector machine and K-Nearest Neighbor algorithm for stock market indices prediction. *Expert Systems with Applications, 80,* 340-355. doi: https://doi.org/10.1016/j.eswa.2017.02.044

Cheng, J., & Greiner, R. (2001). Learning Bayesian belief network classifiers: Algorithms and system. In E. Stroulia & S. Matwin (Eds.), *AI*. LNAI 2056. doi: https://doi.org/10.1007/3-540-45153-6_14

Cho, K., van Merrienboer, B., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). *Learning phrase representations using RNN encoder-decoder for statistical machine translation.* Retrieved from https://arxiv.org/pdf/1406.1078v3.pdf

Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2014*). Empirical evaluation of Gated Recurrent Neural Networks on sequence modelling*. Retrieved from https://arxiv.org/pdf/1412.3555.pdf

Donalek, C. (2011, April). *Supervised and unsupervised learning*. Retrieved from http://www.astro.caltech.edu/~donalek/Teaching.html

Dozat, T. (2016). *Incorporating Nesterov momentum into Adam.* Paper presented at Workshop Track ICLR 2016. Retrieved from https://openreview.net/pdf?id=OM0jvwB8jIp57ZJjtNEZ

Droke, C., (2001). *Moving averages simplified.* Marketplace Books, Columbia, MD. Retrieved from https://www.traderslibrary.com/pdf/pdf_9818935_y85j.pdf

Du, Y., Wang, W., & Wang, L. (2015). *Hierarchical recurrent neural network for skeleton based action recognition*. Paper presented at IEEE Conference on Computer Vision and Pattern Recognition. Retrieved from https://www.cv-foundation.org/openaccess/content_cvpr_2015/papers/Du_Hierarchical_Recurrent_Neural_2015_CVPR_paper.pdf

Fischer, T., & Krauss, C. (2018). Deep learning with Long Short-Term Memory Networks for financial market predictions. *European Journal of Operational Research, 270*(2), 654-669. doi: https://doi.org/10.1016/j.ejor.2017.11.054

Foruzan, E., Scott, S. D., & Lin J., (2015). *A comparative study of different machine learning methods for electricity prices forecasting of an electricity market.* Paper presented at 2015 North American Power Symposium (NAPS). doi: 10.1109/NAPS.2015.7335095

Gers, F.A., Schmidhuber, J., & Cummins, F. (2000). Learning to forget: Continual prediction with LSTM. *Neural Computation, 12*(10), 2451-2471. doi: 10.1049/cp:19991218

Ghiassi, M. & Saidane, H. (2005). A dynamic artificial neural network model for forecasting time series events. *International Journal of Forecasting, 21*(2), 341-362. doi: https://doi.org/10.1016/j.ijforecast.2004.10.008

Granville, J.E. (1963). *Granville's new key to stock market profits*. Englewood Cliffs, N. J., Prentice Hall

Guresen, E., Kayakutlu, G., & Daim, T. U. (2011). Using artificial neural network models in stock market index prediction. *Expert Systems with Applications, 38(8),* 10389-10397. doi: https://doi.org/10.1016/j.eswa.2011.02.068

Hackeling, G. (2014). *Mastering machine learning with Scikit-Learn*. Packt Publishing. Retrieved from http://www.smallake.kr/wp-content/uploads/2017/03/Mastering-Machine-Learning-with-scikit-learn.pdf

Heaton, J. (2005). *Introduction to neural network for Java.* Heaton Research. Inc. St. Louis. Retrieved from https://books.google.com.tr/books?hl=tr&lr=&id=Swlcw7M4uD8C&oi=fnd&pg=PR35&dq=).+Introduction+to+Neural+Network+for+Java,+Heaton+Research&ots=TKz6Mb9-v4&sig=A50fWEBGYhYjmpsBnF7W4KPMG40&redir_esc=y#v=onepage&q=).%20Introduction%20to%20Neural%20Network%20for%20Java%2C%20Heaton%20Research&f=false

Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T. N., & Kingsbury B. (2012). Deep neural networks for acoustic modelling in speech recognition: The shared views of four research groups. *Signal Processing Magazine, IEEE, 29*(6), 82–97. doi: 0.1109/MSP.2012.2205597

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

Hsu, P.H., Taylor, M.P., Wang, Z. (2016). Technical trading: Is it still beating the foreign exchange market. *Journal of International Economics, 106,* 188-208. doi: https://doi.org/10.1016/j.jinteco.2016.03.012

Jiang, Z., & Liang, J. (2017). *Cryptocurrency portfolio management with deep reinforcement learning*. Paper presented at 2017 Intelligent Systems Conference (IntelliSys). doi: 10.1109/IntelliSys.2017.8324237

Jordan, M. I., & Mitchell, T. M., (2015). Machine learning: Trends, perspective, and prospects. *Science 349*(6245), 255-260. doi: 10.1126/science.aaa8415

Karmiani, D., Kazi, R., Nambisan, A., Shah, A., & Kable, V. (2019). *Comparison of predictive algorithms: Backpropagation, SVM, LSTM and Kalman Filter for stock market*. Paper presented at Dubai, United Arab Emirates, United Arab Emirates, 2019 Amity International Conference on Artificial Intelligence (AICAI). doi: 10.1109/AICAI.2019.8701258

Khaidem, L., Saha, S., & Dey, S. R. (2016). *Predicting the direction of stock market prices Using random forest*. Retrieved from https://arxiv.org/pdf/1605.00003.pdf

Khare, K., Darekar, O., Gupta, P., & Attar, V. Z. (2017). *Short term stock price prediction using deep learning.* Paper presented at 2017 2nd IEEE International Conference RTEICT. doi: 10.1109/RTEICT.2017.8256643

Kingma, D.P., & Ba, J.L. (2014). *Adam: A method for stochastic optimization.* Paper presented at ICLR 2015. Retrieved from https://arxiv.org/pdf/1412.6980.pdf%20%22%20entire%20document

Kotsiantis, S. (2007). Supervised machine learning: A review of classification techniques. *Informatica, 31*(3), 249-264. Retrieved from http://www.informatica.si/index.php/informatica/article/viewFile/148/140

Kotsiantis, S. (2007). Supervised machine learning: A review of classification techniques. *Informatica, 31*(3), 251. Retrieved from http://www.informatica.si/index.php/informatica/article/viewFile/148/140

Lambert, D. R. (1983). Commodity Channel Index: Tool for trading cyclic trends. *Technical Analysis of Stocks and Commodities, 1*(5), 120-122.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature, 521,* 436-444. doi:10.1038/nature14539

Lee, J.W. (2001). *Stock price prediction using reinforcement learning.* Paper presented at IEEE International Joint Conference on Neural Networks. doi: 10.1109/ISIE.2001.931880

Lento, C. (2008). A combined signal approach to technical analysis on the S&P 500. *Journal of Business & Economics Research, 6*(8), 41-52.

Li, X., Xie, H., Wang, R., Cai, Y., Cao, J., Wang, F., Min, H., & Deng, X. (2016). Empirical analysis: Stock market prediction via Extreme Learning Machine. *Neural Computing and Applications, 27*(1), 67–78. doi: https://doi.org/10.1007/s00521-014-1550-z

Li, Z., & Tam, V. (2017). *A comparative study of a recurrent neural network and support vector machine for predicting price movements of stocks of different volatilities.* Paper presented at 2017 IEEE Symposium Series on Computational Intelligence (SSCI). doi: 10.1109/SSCI.2017.8285319

Malkiel, B. G. (2003). The efficient market hypothesis and its critics. *Journal of Economic Perspectives, 17*(1), 59-82. doi: 10.1257/089533003321164958

Marsland, S. (2015). Machine learning: An algorithmic perspective. In R. Herbrich & T. Graepel (Eds.), *Machine Learning & Pattern Recognition Series.* CRC Press Taylor & Francis Group. Retrieved from http://dai.fmph.uniba.sk/courses/ICI/References/marsland.machine-learning.2ed.2015.pdf

Masters, D., & Luschi, C. (2018). *Revisiting small batch training for deep neural networks*. Retrieved from https://arxiv.org/pdf/1804.07612.pdf

McNally S., Roche J., & Caton S. (2016). *Predicting the price of Bitcoin using machine learning*. Paper presented at 2018 26th Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP). doi: 10.1109/PDP2018.2018.00060

Menkhoff, L. (2010). The use of technical analysis by fund managers: International evidence. *Journal of Banking and Finance 34*(11), 2573-2586.

Menkhoff, L., & Taylor, M. P. (2006). The obstinate passion of foreign exchange professionals: Technical analysis. *Journal of Economic Literature, 45*(4), 936-972. doi: 10.1257/jel.45.4.936

Mittal, A., & Goel, A. (2013). *Stock prediction using Twitter sentiment analysis.* Retrieved from Stanford University http://cs229.stanford.edu/proj2011/GoelMittal-StockMarketPredictionUsingTwitterSentimentAnalysis.pdf

Mousavi, S. S., Schukat, M., & Howley, E., (2017). Traffic light control using deep policy-gradient and value-function-based reinforcement learning. *IET Intelligent Transport Systems, 11*(7), 417-423. doi: 10.1049/iet-its.2017.0153

Muchiri, P., Pintelon, L., Gelders, L., & Martin, H. (2011). Development of maintenance function performance measurement framework and indicators. *International Journal of Production Economics 131*(1), 295-302. doi: 10.1016/j.ijpe.2010.04.039

Ni, H., & Yin, H. (2009). Exchange rate prediction using hybrid neural networks and trading indicators. *Neurocomputing, 72*(13-15), 2815-2823. doi: https://doi.org/10.1016/j.neucom.2008.09.023

Pan, H. (2003). *A joint review of technical and quantitative analysis of the financial markets towards a unified science of intelligent finance.* Paper presented at Proceedings of 2003 Hawaii International Conference on Statistics and Related Fields, Hawaii, USA.

Patel, J., Shah, S., Thakkar, P., & Kotecha, K. (2014). Predicting stock and stock price index movement using trend deterministic data preparation and machine learning techniques. *Expert Systems with Applications, 42*(1), 259-268. doi: https://doi.org/10.1016/j.eswa.2014.07.040

Patel, J., Shah, S., Thakkar, P., & Kotecha, K. (2015). Predicting stock market index using fusion of machine learning techniques. *Expert Systems with Applications, 42*(4), 2162-2172. doi: https://doi.org/10.1016/j.eswa.2014.10.031

Persio, L. D., & Honchar, O. (2016). Artificial neural networks architectures for stock price prediction: Comparisons and applications. *International Journal of Circuits, Systems and Signal Processing, 10*, 403-413. Retrieved from https://iris.univr.it/retrieve/handle/11562/955101/60620/Artificial%20Neural%20Networks%20architectures%20for%20stock%20price%20prediction%20comparisons%20and%20applications.pdf

Raschka, S. (2015). *Python machine learning.* Packt Publishing Ltd, Birmingham. Retrieved from https://www.javiercancela.com/pymle-equations.pdf

Ratto, A. P., Merello, S., Oneto, L., Ma, Y., Malandri, L., & Cambria E. (2018). *Ensemble of technical analysis and machine learning for market trend prediction.* Paper presented at 2018 IEEE Symposium Series on Computational Intelligence (SSCI). doi: 10.1109/SSCI.2018.8628795

Sak H., Senior, A., & Beufays, F. (2014). *Long Short-Term Memory recurrent neural network architectures for large scale acoustic modelling.* Paper presented at Proceedings of the Annual Conference of International Speech Communication Association (INTERSPEECH). Retrieved from https://arxiv.org/pdf/1511.07528.pdf

Schmidhuber, J. (2015), Deep learning in neural networks: An overview. *Neural Networks, 61,* 4-5. doi: https://doi.org/10.1016/j.neunet.2014.09.003

Selvin, S., Vinayakumar, R., Gopalakrishnan, E. A., Menon, V. K., & Soman, K. P. (2017). *Stock price prediction using LSTM, RNN and CNN-Sliding Window model*. Paper presented at 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI). doi: 10.1109/ICACCI.2017.8126078

Selvin, S., Vinayakumar, R., Gopalakrishnan, E. A., Menon, V. K., & Soman, K. P. (2017). *Stock price prediction using LSTM, RNN and CNN-Sliding Window model*. Paper presented at 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI), p. 1645. doi: 10.1109/ICACCI.2017.8126078

Shalev-Shwartz, S., & Ben-David, S. (2014). *Understanding machine learning: From theory to algorithms*. Cambridge: Cambridge University Press. doi:10.1017/CBO9781107298019.002

Shon, T., & Moon, J. (2007). A hybrid machine learning approach to network anomaly detection. *Information Sciences, 177*(18), 3799-3821. doi: https://doi.org/10.1016/j.ins.2007.03.025

Siegel, J. G., Shim, J.K., Qureshi, A.A., & Brauchler, J. (2014). *International encyclopaedia of technical analysis*, Routledge, New York. doi: https://doi.org/10.4324/9781315062105

Singh, A., Thakur, N., & Sharma A. (2016). *A review of supervised machine learning algorithms.* Paper presented at International Conference of Computing for Sustainable Global Development(INDIACom). Retrieved from https://ieeexplore.ieee.org/abstract/document/7724478/

Singh, R., & Srivastava, S. (2017). Stock prediction using deep learning. *Multimedia Tools Appl., 76*(18), 18569-18584. doi:10.1007/s11042-016-4159-7

Smola, A. J. (2008). *An introduction to machine learning*. Retrieved from http://www.hutter1.net/mlss08/files/smola.pdf

Smola, A. J. & Vishwanathan, S. V. N. (2008). *Introduction to machine learning*. Cambridge: Cambridge University Press. Retrieved from http://alex.smola.org/drafts/thebook.pdf

Song, Y. (2018). *Stock trend prediction: Based on machine learning methods.* Retrieved from https://cloudfront.escholarship.org/dist/prd/content/qt0cp1x8th/qt0cp1x8th.pdf

Song, Y., & Lu, Y. (2015). Decision tree methods: Applications for classification and prediction. *Shanghai Arch Psychiatry*, *27*(2), 130-135. doi: 10.11919/j.issn.1002-0829.215044

Tam, P. H., & Cuong, N. T., (2018). Effectiveness of investment strategies based on technical indicators: Evidence from Vietnamese stock markets. *Journal of Insurance and Financial Management, 3*(5), 55-68. Retrieved from https://www.researchgate.net/profile/Nguyen_Thanh_Cuong2/publication/324790132_Effectiveness_of_Investment_Strategies_Based_on_Technical_Indicators_Evidence_from_Vietnamese_Stock_Markets_ARTICLE_INFO_JEL_Classification_Keywords/links/5ae2727d0f7e9b28594a2805/Effectiveness-of-Investment-Strategies-Based-on-Technical-Indicators-Evidence-from-Vietnamese-Stock-Markets-ARTICLE-INFO-JEL-Classification-Keywords.pdf

Teixeira, L. A., & Oliveira, A. L. I. (2010). A method for automatic stock trading combining technical analysis and nearest neighbour classification. *Expert Systems with Applications 37*(10), 6885-6890. doi: https://doi.org/10.1016/j.eswa.2010.03.033

Utthammajai K., & Leesutthipornchai, P. (2015). Association mining on stock index indicators. *International Journal of Computer and Communication Engineering, 4*(1), 46-49.

Varon, J., & Soroka, A. (2016). *Stock trading with reinforcement learning*. Retrieved from https://static1.squarespace.com/static/563f45cfe4b0bff8503c3b20/t/58750e10e3df286f96adea20/1484066322582/Varon_Soroka_FinalProject.pdf

Wafi, A.S., Hassan, H., & Mabrouk, A. (2015). Fundamental analysis vs technical analysis in the Egyptian stock exchange – empirical study. *International Journal of Business and Management Study – IJBMS 2*(2), 212-218.

Wang, F., Zhang, Y., Rao, Q., Li, K., & Zhang, H. (2017). Exploring mutual information-based sentimental analysis with Kernel-Based Extreme Learning Machine for stock prediction. *Soft Computing, 21*(12), 3193–3205. doi: https://doi.org/10.1007/s00500-015-2003-z

Wilder, J. W. (1978). *New concepts in technical trading systems.* Trend Research, Hunter, North Carolina.

Yazdi, S. H. M., & Lashkari, Z. H. (2012). *Technical analysis of forex by Parabolic SAR indicator.* Paper presented at International Islamic Accounting and Finance Conference 2012.

Zarrabi, N., Snaith, S., & Coakley, J. (2017). FX technical trading rules can be profitable sometimes!. *International Review of Financial Analysis, 49,* 113-127. doi: 10.1016/j.irfa.2016.12.010

Zhou, Z., Li, X, & Zare, R. N. (2017). Optimizing chemical reactions with deep reinforcement learning. *ACS Cent. Sci.*, *3*(23), 1337-1344. doi: https://doi.org/10.1021/acscentsci.7b00492