

# **AN APPLICATION FOR PREDICTING CRYPTO PRICES USING STACKED LSTM**

Submitted in partial fulfillment of the requirements for the award of a  
Bachelor of Engineering degree in Computer Science and Engineering

**By**

**SWAMY GANESH KARRI (Reg.No.- 39111001)  
PUNUGUPATI SAI KUMAR (Reg.No.- 39110815)**



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING  
SCHOOL OF COMPUTING**

## **SATHYABAMA**

**INSTITUTE OF SCIENCE AND TECHNOLOGY  
(DEEMED TO BE UNIVERSITY)**

**Accredited with Grade "A" by NAAC | 12B Status  
By UGC | Approved by AICTE**

**JEPPIAAR NAGAR, RAJIV GANDHISALAI,  
CHENNAI - 600119**

**APRIL- 2023**



# SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY  
(DEEMED TO BE UNIVERSITY)

Accredited with Grade "A" by NAAC | 12B Status by UGC | Approved by AICTE

[www.sathyabama.ac.in](http://www.sathyabama.ac.in)

---

## DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

### BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **SWAMY GANESH KARRI (39111001)** who carried out the Project Phase-2 entitled "**AN APPLICATION FOR PREDICTING CRYPTO PRICES USING STACKED LSTM**" under my supervision from Jan 2023 to April 2023.

Internal Guide

Dr. M. D. ANTO PRAVEENA M.E., Ph.D.,

Head of the Department

Dr. L. LAKSHMANAN, M.E., Ph.D.,



---

Submitted for Viva-voce Examination held on 20.04.2023

Internal Examiner

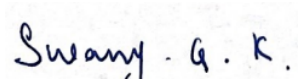
External Examiner

## DECLARATION

I, **SWAMY GANESH KARRI (39111001)** hereby declare that the Project Phase-2 Report entitled "**AN APPLICATION FOR PREDICTING CRYPTO PRICES USING STACKED LSTM**" done by me under the guidance of **Dr. M. D. ANTO PRAVEENA Ph.D.**, is submitted in partial fulfillment of the requirements for the award of a Bachelor of Engineering degree in **Computer Science and Engineering**.

**DATE: 20.04.2023**

**PLACE: Chennai**

A handwritten signature in blue ink that reads "Swamy - G. K.".

**SIGNATURE OF THE CANDIDATE**

## **ACKNOWLEDGEMENT**

I am pleased to acknowledge my sincere thanks to **Board of Management** of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. SASIKALA M.E., Ph.D., Dean**, School of Computing, **Dr. L. LAKSHMANAN M.E., Ph.D.**, Head of the Department of Computer Science and Engineering for providing me with the necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. M. D. ANTO PRAVEENA M.E., Ph.D.**, for her valuable guidance, suggestions, and constant encouragement paved way for the successful completion of my phase-2 project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

## **ABSTRACT**

Cryptocurrencies are a digital form of money in which all transactions are recorded electronically. It is a soft money that does not exist in the actual form of hard notes. Here, we emphasize the distinction between fiat currency and virtual currency, which is decentralized in the sense that all virtual currency users may obtain services without the involvement of a third party. However, because of their considerable price fluctuation, receiving services from these cryptocurrencies has an influence on international relations and trade. There are various virtual currencies, including bitcoin, ripple, Ethereum Sushi, and others. In our initiative, we concentrated on major cryptocurrencies like bitcoin, finance, sushi MANA, Ethereum, Ethereum classic, and Dot coin. Bitcoin is widely accepted by several groups, including investors, academics, traders, and policymakers. To the best of our knowledge, our goal is to construct efficient deep learning-based prediction models, namely long short-term memory (LSTM), to deal with crypto coin price volatility and achieve high accuracy.

## TABLE OF CONTENT

Chapter No	TITLE	Page No.	
	ABSTRACT	v	
	LIST OF ABBREVIATIONS	viii	
	LIST OF FIGURES	x	
	LIST OF TABLES	xi	
1	INTRODUCTION	1	
2	LITERATURE SURVEY	7	
	2.1 Inferences from Literature Survey	11	
3	REQUIREMENTS ANALYSIS	14	
	3.1 Feasibility Studies/Risk Analysis of the Project	14	
	3.2 Software Requirements Specification	17	
4	DESCRIPTION OF PROPOSED SYSTEM	18	
	4.1 Selected Methodology or process model	18	
	4.2 Architecture / Overall Design of Proposed System	22	
	4.3 Description of Software for Implementation and Testing plan of the Proposed Model/System	24	
	4.4 Financial Report on Estimated Costing	25	
	4.5 Transition/ Software to Operation Plan	26	
5	IMPLEMENTATION DETAILS	28	
	5.1 Development and Deployment Setup	28	
	5.2 Algorithm	29	
6	RESULTS AND DISCUSSION	34	
7	CONCLUSION	36	
	7.1 Conclusion	36	
	7.2 Future work	36	
	7.3 Research Issues	36	
	7.4 Implementation Issues	37	
	REFERENCES	39	
	APPENDIX	41	
	A. SOURCE CODE	41	
	B. SCREENSHOTS	48	
	C. RESEARCH PAPER	50	

AI	Artificial Intelligence
ANN	Artificial Neural Network
API	Application Programming Interface
ASGI	Asynchronous Server Gateway Interface
AWS	Amazon Web Services
BPTT	Backpropagation Through Time
BTC	Bitcoin
CNN	Convolutional Neural Network
CORS	Cross Origin Resource Sharing
CSV	Coma Separated Value
DOGE	Dogecoin
ETH	Etherium
GCP	Google Cloud Platform
GRU	Gated Recurrent Unit
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
JSON	JavaScript Object Notation
LTC	Litecoin
LSTM	Long Short-Term Memory
MAE	Mean Absolute Error
OTC	Over The Counter

## LIST OF ABBREVIATIONS



RMSE	Root Mean Squared Error
RNN	Recurrent Neural Network
SQL	Structured Query Language
TBPTT	Truncated Backpropagation Through Time

<b>Figure No.</b>	<b>Figure Name</b>	<b>Page No.</b>
4.1	LSTM Single Cell Architecture	21
4.2	Memorization in LSTM	21
4.3	Architecture / Overall Design	23
5.1	Libraries	29
B.1	Cryptocurrencies Live Forecasting	48
B.2	Coin Prediction	48

## **LIST OF FIGURES**

Table No.	Table Name	Page No.
1.1	Cryptocurrency Coins	5

**LIST OF TABLES**

# CHAPTER 1

## INTRODUCTION

An application for predicting cryptocurrency prices using stacked LSTM is a software system that uses deep learning algorithms to predict the future prices of various cryptocurrencies. The system uses historical price data of different cryptocurrencies as well as other relevant features such as trading volumes and social media sentiment to train a stacked LSTM model. The model then makes predictions on future cryptocurrency prices based on the patterns and trends identified in the training data.

The application provides a user-friendly interface for users to input their preferred cryptocurrency and desired time horizon for price prediction. The system retrieves the necessary data, preprocesses it, and feeds it into the stacked LSTM model to generate a prediction. The application also provides a visualization of the predicted prices, allowing users to easily interpret and analyze the results.

The application can be used by traders and investors who are interested in making informed decisions regarding cryptocurrency investments. By providing accurate price predictions, the application can help users make better investment decisions, ultimately leading to increased profits and reduced risk.

In addition to helping traders and investors, an application for predicting cryptocurrency prices using stacked LSTM can also be useful for researchers and academics who are interested in studying cryptocurrency price movements and trends. The application can be used to analyze and compare the performance of different cryptocurrencies, as well as to identify potential market opportunities and risks.

Furthermore, the application can also be used by businesses and financial institutions that are interested in incorporating cryptocurrencies into their operations. By providing accurate price predictions, the application can help these entities make informed decisions regarding cryptocurrency transactions and investments.

However, it is important to note that while an application for predicting cryptocurrency prices using stacked LSTM can provide valuable insights and predictions, it is not infallible. The accuracy of the predictions is dependent on the quality and quantity of the historical data used for training the model, as well as the other relevant features used as input. Additionally, the cryptocurrency market is known for its volatility and unpredictability, which can make accurate price predictions challenging.

The fast development of information technology has had a substantial impact on numerous industries. Financial products and operations are increasingly becoming digitized. Cryptocurrencies are gaining popularity as a means of financial digitalization. A cryptocurrency is a numerical currency meant to operate as a medium of exchange using powerful encryption to secure financial transactions, monitor the generation of new units, and verify asset transfers.

The world of cryptocurrencies has experienced a significant surge in popularity over the last few years, with an increasing number of people investing in various cryptocurrencies. As a result, there is a growing need for tools and applications that can help predict cryptocurrency prices to make informed investment decisions.

In this project, we propose an application that predicts cryptocurrency prices using a stacked Long Short-Term Memory (LSTM) model. LSTM is a type of neural network that is well-suited for time-series data, making it ideal for predicting cryptocurrency prices.

Our application will allow users to input the name of the cryptocurrency they are interested in and the time period for which they would like to predict prices. The application will then use historical data for that cryptocurrency to train the stacked LSTM model and generate a prediction for the selected time period.

The volatility of the cryptocurrency market has made it a challenging task to predict prices accurately. However, with the advancements in machine learning and deep learning algorithms, it has become possible to develop models that can predict the

prices with reasonable accuracy. One such algorithm is the Long Short-Term Memory (LSTM) algorithm, which has been used extensively for time-series prediction tasks.

In this project, we aim to develop an application that can predict cryptocurrency prices using stacked LSTM. The application will allow users to select the cryptocurrency they want to predict the prices for and the time horizon they are interested in. The model will then use historical price data to predict the future prices of the selected cryptocurrency.

The application will be developed using the Python programming language and several popular libraries such as TensorFlow, Keras, and Pandas. The model will be trained using historical price data from various exchanges and will be validated using different evaluation metrics such as Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE).

Crypto refers to an IT innovation based on peer-to-peer networks and cryptographic technologies. Crypto is not governed by governments or banks due to its features. A feature of crypto, like any other money, is that it facilitates the exchange of services and products, drawing a large number of users and a considerable lot of media interest. Web 2.0 services such as blogs, tweets, forums, chats, email, and so on are now frequently utilized as communication media, with positive effects. Sharing knowledge is a crucial part of learning and skill development. Team members can acquire a more complete understanding of their colleagues' skills by using social media. Data from social media is a global measure of people's opinions and ideas. There has been a chance to contribute to significant improvements in people's behaviors when it comes to the usage of social media and social networks.

Digital currency is a type of web-based commerce that leverages cryptography capabilities to execute money-related transactions. The key feature of the cryptographic money concept is that it is not constrained by a fundamental power: the flowing marrow of the blockchain renders numerical forms of money theoretically immune to traditional means of government control and resistance. Using open and private keys, cryptographic forms of cash may be exhibited directly between two

groups. These transactions can be completed with less expenditure cover, allowing clients to avoid the risky fees paid by foundations tied to traditional money. Instead of a house, crypto is an enhanced down payment that employs cryptography and distributed money (P2P) mechanization to generate and monitor financial trades. The assessment or anticipation of timescales is an essential topic.

Bitcoin, the first decentralized cryptocurrency, was released as open-source software in 2009. Following this, around 4,000 altcoins (alternative cryptocurrencies) were released. In August 2019, the whole cryptocurrency market value is 258 billion, with Bitcoin alone around 179 billion. While these initiatives have opened numerous opportunities, several essential qualities, including as energy consumption and the speed with which a block is confirmed, remain lacking. The rising use of AI in the accounting profession may alter present accounting procedures and inform the creation of future accountant training. Artificial intelligence (AI) technologies are commonly employed in financial distress, financial fraud, market forecasting, and auditing. Accounting businesses have prioritized the development of technical knowledge in accounting and data analysis, which is necessary in accounting training courses.

## HISTORY

David Chaum, an American cryptographer, invented ecash, a sort of cryptographic electronic money, in 1983. Later, in 1995, he put idea into action using Digicash, an early kind of encrypted electronic payment. Before it could be transmitted to a recipient, Digicash required user software to withdraw notes from a bank and select particular encrypted keys. This made the digital money untraceable to a third party.

## ACTIVE CURRENCIES

***Table. 1.1: Cryptocurrency Coin***

Year	Coin Names
2009	Bitcoin
2011	Litecoin
2011	Tether

2013	Dogecoin
2014	Stellar
2015	Ethereum
2017	Cardano
2020	Solana
2017	Polkadot
2020	Avalanche
2014	TRON
2017	BNB
2017	Polygon

Sure, here is a summary of the coins you listed:

**Bitcoin (BTC):** The first and most well-known cryptocurrency, designed to be a decentralized digital currency that can be used for peer-to-peer transactions without the need for intermediaries.

**Litecoin (LTC):** A cryptocurrency that is similar to Bitcoin, but with faster transaction speeds and lower transaction fees.

**Tether (USDT):** A stablecoin that is pegged to the value of the US dollar, designed to provide stability in a volatile cryptocurrency market.

**Dogecoin (DOGE):** A cryptocurrency that started as a joke, but gained popularity due to its community and memes. It has a faster block time than Bitcoin and a large supply.

**Stellar (XLM):** A cryptocurrency that is designed for cross-border payments and remittances, with fast transaction speeds and low transaction fees.

**Ethereum (ETH):** The second-largest cryptocurrency by market capitalization, with a platform that allows developers to build decentralized applications (dApps) using smart contracts.



Cardano (ADA): A cryptocurrency that is designed to provide a more secure and sustainable blockchain platform for dApps and smart contracts.

Solana (SOL): A high-performance blockchain platform that can process up to 65,000 transactions per second, designed to support decentralized applications and financial services.

Polkadot (DOT): A blockchain platform that allows multiple blockchains to work together, designed to support interoperability and scalability.

Avalanche (AVAX): A blockchain platform that uses a consensus mechanism called Avalanche, which allows for fast and secure transactions.

TRON (TRX): A blockchain platform that is designed for decentralized entertainment and content sharing, with a focus on digital media.

Binance Coin (BNB): A cryptocurrency that is used on the Binance cryptocurrency exchange, with various use cases including discounted trading fees, staking, and payment for services on the Binance platform.

Polygon (MATIC): A blockchain platform that is designed to improve the scalability and interoperability of Ethereum, with faster transaction speeds and lower gas fees.

## **CHAPTER 2**

### **LITERATURE SURVEY**

Kamil Kulesza et al. (2014) offer a method to predicting the price of Bitcoin using machine learning techniques in their paper "Predicting the Price of Bitcoin Using Machine Learning." The writers gathered past data on the Bitcoin exchange rate and other factors linked to it, such as trading volume and fluctuation, and used it to train several machine learning models, including artificial neural networks and support vector machines. The writers used several measures to assess the efficacy of their models, including the mean square error and directional accuracy. They discovered that their models could correctly forecast Bitcoin prices in the near term, but their accuracy decreased as the prediction horizon lengthened. The authors conclude that machine learning methods can be used to forecast Bitcoin prices, but they warn that the precision of the predictions may be restricted by the volatile character of the cryptocurrency market. They also propose that more study is required to improve the models' accuracy and to investigate the possibility of using other kinds of data, such as social media and news sentiment, to better predictions [1].

Martin Kuhlemann et al. (2014) investigates the use of artificial neural networks (ANNs) for predicting the price of Bitcoin in the paper, "Bitcoin Price Forecasting with Neural Networks." The author gathered past Bitcoin exchange rate data and used it to train various ANN models, including feedforward and recurrent neural networks. The author assessed the models' success using a variety of measures, including mean absolute error and directional precision. The findings revealed that the ANN models could forecast the price of Bitcoin correctly in the near term, but their accuracy decreased as the prediction horizon grew. The author also compared the ANN models' success to a standard model based on a moving average. The findings revealed that the ANN models beat the benchmark model, indicating that ANNs are a viable method for forecasting Bitcoin prices [2].

Jethin Abraham et al. (2014) suggest a technique for predicting Bitcoin's price value by analyzing Google search trends data using machine learning algorithms in their paper "Predicting Bitcoin's Price Value by Analyzing Google Search Trends Data Using Machine Learning Algorithms." The author gathered Bitcoin-related Google search data to teach several machine-learning models, including linear regression and decision trees. The writers used several metrics to assess the efficacy of their models, including root mean square error and mean absolute error. According to the findings, the models were able to correctly forecast the price of Bitcoin in the near run, with a prediction horizon of up to seven days. The authors also investigated the relationship between Google search patterns and Bitcoin prices and discovered a significant positive correlation between the two factors. This indicates that Google search patterns data can be used to forecast Bitcoin prices [3].

Nicholas Colas et al. (2014)'s study "Bitcoin Market Analysis: An Examination of OTC Market Participants" examines the Bitcoin over-the-counter (OTC) market and its users. The writers performed a poll of OTC market players to learn about their trading habits, motivations, and Bitcoin views. The writers discovered that most OTC market users were people and small companies and that the majority of trading was done for theoretical rather than economic reasons. The writers also discovered a highly fragmented and opaque OTC market with little standardization in pricing and trading conditions. According to the writers, the OTC market plays an essential part in the Bitcoin ecosystem and is a critical source of liquidity for Bitcoin transactions. However, the absence of transparency and standardization in the OTC market poses hurdles to Bitcoin's widespread acceptance as a currency and asset class [4].

Joseph Wang et al. (2015) shows a Bayesian neural network method for predicting Bitcoin price in their article "Bitcoin Market Prediction with Bayesian Neural Networks." The researcher gathered past Bitcoin exchange rate data and used it to train several Bayesian neural network models. The researcher used several measures to assess the efficacy of their models, including mean absolute error and directional accuracy. The findings demonstrated that Bayesian neural network models could correctly predict Bitcoin prices in the near term, with a prediction horizon of up to two weeks. The researcher also compared the Bayesian neural

network models' efficacy to that of other machine learning models, such as support vector regression and feedforward neural networks. The Bayesian neural network models beat the other models, indicating that a Bayesian method is an effective strategy for Bitcoin price prediction [5].

The article "Cryptocurrency Price Prediction Using Machine Learning Algorithms: A Survey" by Kaustubh Dhonde et al. and Vishnu K et al. (2018) offers a literature review on cryptocurrency price projection using machine learning algorithms. The writers discuss different methods for predicting cryptocurrency prices, such as conventional time-series analysis, sentiment analysis, and machine learning algorithms such as neural networks, decision trees, and support vector regression. The writers compare the efficacy of the various approaches using a variety of metrics, including mean absolute error and root mean squared error. They found that machine learning algorithms generally outperform traditional time-series analysis and sentiment analysis for cryptocurrency price prediction. The writers also address the difficulties of predicting cryptocurrency prices, such as high fluctuation and a dearth of past data for some cryptocurrencies. They propose that using ensemble models and incorporating external data sources, such as social media data and market reports, could enhance the precision of cryptocurrency price prediction [6].

Daniel Trujillo et al. (2018) suggests an innovative approach, sample dimension engineering for Bitcoin price prediction using machine learning algorithms in their paper "Bitcoin Price Prediction Using Machine Learning: An Approach to Sample Dimension Engineering." The authors contend that the high-dimensional and non-stationary character of Bitcoin price data makes conventional machine learning methods challenging to implement successfully. To address this issue, the authors suggest "sample dimension engineering," which entails creating synthetic samples by merging numerous time-series data points to create a new sample with a higher dimensionality. The writers then forecast Bitcoin values using machine learning techniques such as random forests and neural networks on synthetic examples. The writers use several metrics to assess the effectiveness of their method, including mean absolute error and root mean squared error. They discovered that when applied to raw Bitcoin market data, their method outperformed conventional machine

learning algorithms [7].

Swati Aggarwal and Tanuja Jain's (2019) work "Cryptocurrency Price Prediction Using News Articles: A Sentiment Analysis Approach" suggests a sentiment analysis strategy for predicting cryptocurrency values using news articles. According to the writers, reports can have an impact on cryptocurrency prices, and sentiment research can provide insight into market opinion. The writers gathered cryptocurrency-related news stories from different sources and used natural language processing methods to extract the sentiment of each article. The sentiment ratings of the stories were then aggregated over time to build a sentiment index, which they used to forecast cryptocurrency prices. The writers compared the efficacy of their method to conventional time-series analysis models using several coins, including Bitcoin, Ethereum, and Ripple. They discovered that their sentiment analysis method beat conventional models for some cryptocurrencies, implying that sentiment analysis can provide useful information for predicting cryptocurrency prices. Overall, the study presents an intriguing method for predicting bitcoin prices using sentiment analysis of news stories. It is worth mentioning, however, that the method is highly reliant on the assumption that news articles have, which is not always the case [8].

Daniel Fleder et al. (2019) proposed a deep learning approach for predicting cryptocurrency prices using a convolutional neural network in their paper "Predicting Cryptocurrency Prices Using Deep Learning Techniques." (CNN). The writers claim that deep learning algorithms may recognize complicated patterns in cryptocurrency market data, resulting in better prediction performance. The writers gathered historical market data for cryptocurrencies such as Bitcoin, Ethereum, and Litecoin and preprocessed it for use in the CNN algorithm. They compared the efficacy of their method to conventional time-series analysis models using several measures such as mean absolute error and root mean squared error. For some cryptocurrencies, the authors discovered that their deep learning method outperformed conventional models, implying that deep learning models can provide useful insights into cryptocurrency price forecasts. In addition, they performed a feature importance study to determine which characteristics are most essential for

the CNN model's predictions.

Overall, the article presents an intriguing method for predicting cryptocurrency prices using deep learning techniques. It should be noted, however, that the method might need a substantial quantity of processing resources to train as well as run the CNN model, which may restrict its practical application for some researchers [9].

## **2.1 INFERENCES FROM LITRERATURE SURVEY**

Based on the list of articles provided, a literature survey has been conducted on the topic of using machine learning algorithms to predict the price of cryptocurrencies. Because of their extremely volatile nature and lack of a centralized regulating authority, forecasting cryptocurrencies is a difficult endeavor. While some analysts and traders utilize technical analysis and market patterns to create short-term projections, due to the volatile nature of the crypto market, long-term predictions are challenging. Furthermore, cryptocurrencies are a new asset class, and their value is affected by governmental changes, technical advancements, and the larger economic environment.

For forecasting cryptocurrency values, several techniques and models are being created. Some of these include machine learning algorithms, sentiment analysis tools, and fundamental analysis models. These models forecast future prices using data from previous market patterns, news and social media mood, and other variables.

There are some specific approaches that have been used to predict cryptocurrency prices. Technical Analysis requires analyzing previous market data, such as price and volume, to identify patterns and trends that may be used to forecast future pricing. Technical analysts visualize these patterns and forecast price changes using tools such as charts and graphs. Technical analysis can be beneficial for making short-term forecasts, but it has limitations in terms of capturing the influence of external factors like regulatory changes or global events.

Sentiment analysis is the process of analyzing news stories, social media posts, and

other sources of information to evaluate market sentiment and forecast how it will affect cryptocurrency values. Using natural language processing and machine learning techniques, this method identifies keywords and sentiments in text data. This method is effective for making short-term forecasts, but it is restricted by the quality and amount of information available.

Fundamental analysis is examining the fundamental elements that influence cryptocurrency pricing, including as adoption rates, network activity, and regulatory changes. Fundamental analysts utilize this information to forecast a cryptocurrency's long-term prospects and growth potential. This method is effective for long-term forecasting, but it is restricted by the difficulty of precisely estimating the impact of external events on cryptocurrency markets.

Machine learning algorithms may be taught on previous market data to recognize trends and forecast future price movements. These algorithms may be used with other approaches like sentiment analysis and fundamental analysis to make both short-term and long-term forecasts. Machine learning can be beneficial for capturing complicated relationships between many elements influencing cryptocurrency values, but it is limited by the quality and amount of data that is available.

Some researchers have proposed integrating different methodologies, such as technical analysis and sentiment analysis, to produce hybrid models that outperform any one approach alone. These models can assist to decrease market noise and deliver more trustworthy predictions.

Each strategy has its own set of advantages and disadvantages, and the approach chosen will be determined by the cryptocurrency under consideration and the intended prediction horizon. Short-term predictions can benefit from technical and sentiment research, while long-term predictions can benefit from fundamental analysis and machine learning. To generate more accurate forecasts, hybrid models can combine the capabilities of different methodologies. In the end, the accuracy of any prediction model is determined by the quality and amount of accessible data, as well as the underlying assumptions and models employed.

Despite advances in the use of machine learning and other data analysis techniques to forecast cryptocurrency values, there are still some open difficulties and hurdles. Some of these open problems include:

- One of the primary issues in predicting cryptocurrency values is the absence of solid historical data, which can make training accurate machine learning models difficult. Furthermore, the available data quality might be poor, with missing or inconsistent data making it impossible to make significant conclusions.
- Market volatility: Cryptocurrency markets are extremely volatile, with prices changing quickly and in unpredictable ways. This can make it difficult to estimate prices effectively use historical data, as past patterns may not always reflect future behavior.
- Market manipulation is also a risk in cryptocurrency markets, with significant players able to affect prices through strategies. This makes it difficult to effectively estimate prices since rapid and unexpected price swings may not represent underlying market trends.
- Market manipulation is also a risk in cryptocurrency markets, with significant players able to affect prices through strategies. This makes it difficult to effectively estimate prices since rapid and unexpected fluctuation in prices may not represent underlying market trends.
- The regulatory environment for cryptocurrencies is continually growing, with distinct rules and regulations governing cryptocurrency trading and investment in different countries. This makes predicting how regulatory changes would affect cryptocurrency pricing tricky.
- The frequent introduction of new cryptocurrencies might make it challenging to keep up with the changing environment of the cryptocurrency market, since each new coin may have its own unique set of qualities and characteristics that must be accounted for in any prediction model.



## **CHAPTER 3**

### **REQUIREMENT ANALYSIS**

#### **3.1 FEASIBILITY STUDIES/RISK ANALYSIS OF THE PROJECT**

##### **FEASIBILITY STUDY**

Typically, a feasibility study for employing Long Short-Term Memory (LSTM) models to anticipate cryptocurrency values would involve multiple phases. These phases might include:

**Data collection:** Collecting relevant historical data for the cryptocurrency being studied, such as price, trading volume, and other relevant indicators. The data should be of sufficient length and quality to enable the LSTM model to learn patterns and trends that can be used to predict future prices.

**Data preprocessing:** Preprocessing the collected data to ensure that it is clean, consistent, and in a suitable format for use with the LSTM model. This may involve removing outliers, normalizing data, and splitting the data into training and testing sets.

**Model design:** Designing the LSTM model to be used for the prediction of cryptocurrency prices. This would involve selecting the appropriate hyperparameters, such as the number of layers, the number of nodes in each layer, and the learning rate.

**Model training:** Training the LSTM model using the preprocessed historical data. This would involve adjusting the model parameters to minimize the prediction error on the training data.

**Model evaluation:** Evaluating the performance of the LSTM model on the testing data to determine its accuracy and effectiveness in predicting future cryptocurrency prices.

Deployment: Deploying the LSTM model in a real-world environment to make predictions of future cryptocurrency prices. This might involve integrating the model into a trading platform or other application that can utilize its predictions.

## **ECONOMICAL FEASIBILITY**

The cost of data collecting, data storage, calculation, and deployment all affect the economic feasibility of forecasting cryptocurrency values using LSTM models.

Data acquisition: The cost of acquiring high-quality data required for training the LSTM model may vary depending on the source of data. The cost of acquisition is cheap if the data is easily accessible from free sources. However, if the data must be purchased from a third-party supplier, the cost might be exceedingly high.

Data storage: The quantity of data needed for training and storing the model is an important concern for the project's economic feasibility. Storing a huge dataset for a longer period can be costly, especially if the data is stored on cloud-based systems.

Computation: Training LSTM models can be computationally costly, requiring high-end hardware and a significant amount of time. Computational costs can be massive, especially for huge datasets or complicated models.

Deployment: The cost of deploying the LSTM model depends on the platform used for deployment. If the model is deployed on cloud-based platforms such as Amazon Web Services (AWS) or Google Cloud Platform (GCP), the cost can be high. However, deploying on low-cost servers can reduce deployment costs.

Market conditions: The cryptocurrency market may be very volatile, and the accuracy of the LSTM model may be impacted by rapid market movements. If the model is not sufficiently accurate, investors and traders may suffer big losses.

## **TECHNICAL FEASIBILITY**

The technical feasibility of using Long Short-Term Memory (LSTM) models for the

prediction of cryptocurrency prices depends on several factors. Some of these factors include:

**Data availability and quality:** The effectiveness of LSTM models to forecast cryptocurrency values is heavily dependent on the availability and quality of historical data. It may be impossible to train an appropriate LSTM model if there is insufficient historical data or if the data is of poor quality.

**Model complexity:** LSTM models can be complicated and take a large number of computer resources to train. The feasibility of utilizing an LSTM model will be determined by the model's complexity and the available computational resources.

**Model performance:** The performance of the LSTM model in predicting cryptocurrency prices must be evaluated carefully. This involves testing the model on historical data that was not used during training to ensure that it can generalize to new data.

**Interpretability:** LSTM models can be difficult to read, and it can be difficult to understand how the model develops at its predictions. This might make identifying and addressing any issues that occur during the model's development and implementation difficult.

The LSTM model must be implemented into a pre-existing system or trading platform. This involves being certain that the model's predictions can be used successfully in the trading process and that they can be integrated with current trading methods.

## **OPERATIONAL FEASIBILITY**

The operational feasibility of a proposed project, such as employing LSTM models to predict cryptocurrency values, is concerned with whether or not it can be effectively integrated and implemented within the current organizational and operational framework. Here are some crucial elements to consider for this project's operational feasibility: **Organizational Structure:** The project team must be able to function within the company's existing organizational structure, or the organization may need to

reorganize to support the project team.

To ensure the project's success, the team working on it must have the necessary technical knowledge and skill in both LSTM modeling and cryptocurrency trading.

Scalability: Long-term success requires the capacity to scale the LSTM model for forecasting cryptocurrency prices to accommodate larger datasets, increasing trading activity, and a greater variety of cryptocurrencies.

Risk Management: The project team must have a risk management approach to handle any risks associated with the accuracy of the prediction model, the volatility of cryptocurrency markets, and trading choices based on the model's forecasts.

User acceptability Testing: User acceptability testing is essential to ensure that the LSTM model's predictions are useful for traders and investors, and they can integrate it into their trading strategy.

System Integration: The integration of the LSTM model into existing systems, including trading platforms, and tools should be carefully considered to ensure smooth integration and minimize the risk of technical difficulties.

## **3.2 SOFTWARE REQUIREMENTS SPECIFICATION**

### **Hardware Specifications:**

- Computers that support Microsoft Server
- More RAM, preferably 4GB or greater
- Processor with a frequency of 1.5GHz or higher

### **Software Specifications:**

- Python 3.6
- Jupyter Notebook

## **CHAPTER 4**

### **DESCRIPTION OF THE PROPOSED SYSTEM**

The different crypto-currency APIs are examined against the criteria specified in that work in this project, and each API function is classified according to one of the following categories. Yes: The heuristic is met, partially: The heuristic is met (e.g., some guidelines were followed while others were not), and no: The heuristic is not met. This API comparison is carried out using the recommended static analysis tools as well as the previously indicated technique. Independent of the programming language used, reliable source code analysis is based on a language parser that identifies the grammar of the language.

#### **4.1 SELECTED METHODOLOGY**

We are extracting the information of all cryptocurrencies and showing them in the application using the CoinAPI. One of the finest providers is CoinAPI.

API is a shorthand for Application Programming Interface. APIs enable various software systems to talk and engage with one another.

An API usually defines how software components should communicate, including the kinds of data that can be exchanged and the operations that can be done.

APIs can be used to access services offered by other software systems, to combine various software systems, or to provide a software system's services to other developers.

APIs are frequently used in web development to allow communication between web applications or to allow third-party developers to create applications that can access data or features offered by a web application.

The proposed methodology uses a deep learning-based prediction model to estimate daily crypto coin prices by recognizing and analyzing key variables. We are going to use LSTM which is the most efficient strategy for forecasting crypto prices. Because

in the cryptocurrency the price fluctuation problem should be resolved quickly.

An Application Programming Interface (API) that gives access to coin data such as real-time market information, past data, and trading amounts is referred to as Coin API.

Coin API is usually provided as a web-based API, which means that developers can use HTTP requests to access the API.

#### **4.1.1. RNN**

A recurrent neural network (RNN) is a form of artificial neural network that handles sequential input. Unlike conventional feedforward neural networks, which process inputs one of them at a time and have no records of previous inputs, RNNs have loops in their architecture that enable them to remember previous inputs and use that knowledge to impact the processing of future inputs.

The presence of a hidden state, which is updated at each time step based on the current input and the prior hidden state, is a crucial characteristic of an RNN. This hidden state functions as a kind of memory for the network, enabling it to record long-term relationships in the input series.

There are different types of RNNs, such as vanilla RNNs, LSTM (Long Short-Term Memory) networks, and GRU (Gated Recurrent Unit) networks, each with its own strengths and weaknesses.

RNNs can be represented as a series of nodes connected in a loop, with each node's output fed back into the input of the next node in the sequence. An RNN's input is a sequence of vectors, such as words in a phrase or video frame. The network processes each vector in the sequence, and the results of each phase may be utilized to produce predictions or classifications. The hidden state refers to an RNN's internal "memory." At each step of the sequence, the hidden state is updated and may be used to hold information about previous inputs. The last hidden state can be utilized to predict the whole sequence.

The vanishing gradient problem is one of the difficulties associated with RNNs. Because the gradients that are backpropagated through the network might become extremely small, the network may have difficulties learning long-term relationships. This is one of the reasons why LSTM network is created since they feature gating mechanisms that allow them to selectively remember or forget information.

RNNs may be trained using approaches such as backpropagation through time (BPTT) or truncated backpropagation through time (TBPTT), which involve unrolling the network across a certain number of time steps and computing the gradients of the loss with respect to the network parameters.

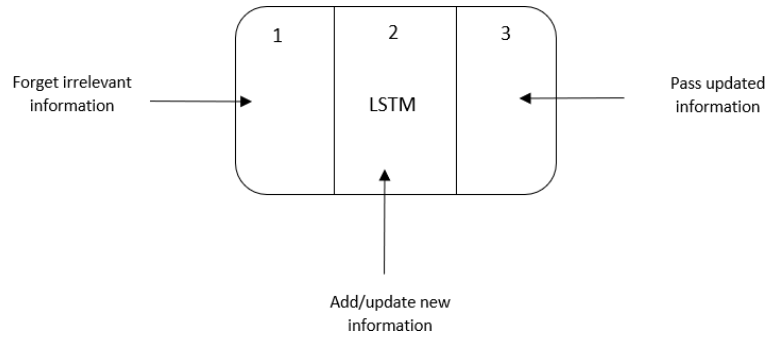
RNNs have been utilized for a variety of applications such as language modeling, speech recognition, handwriting recognition, music production, and others. They have also been integrated with other networks, such as convolutional neural networks (CNNs), to produce hybrid architectures for tasks such as image captioning and video categorization.

#### **4.1.2. LSTM**

Long Short-Term Memory (LSTM) is a form of recurrent neural network that has been designed to solve the vanishing gradient issue. The vanishing gradient issue arises when the gradients used to modify the network's parameters during training become very tiny, making long-term relationships difficult for the network to learn.

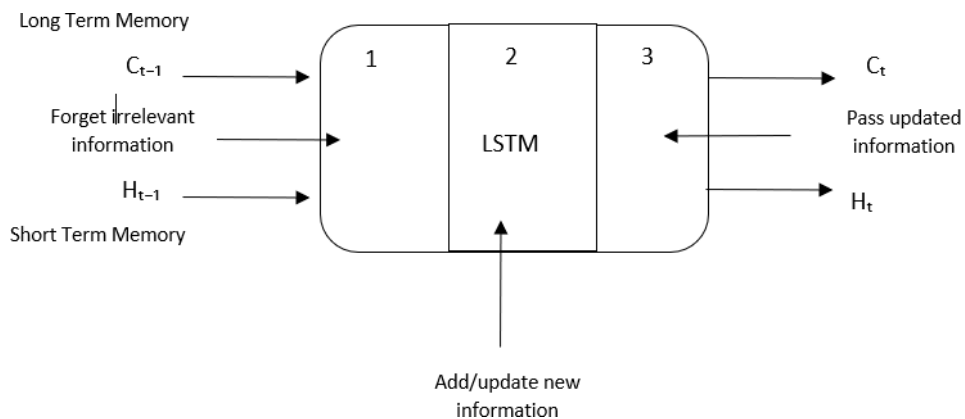
LSTMs resolve this issue by adding a new type of memory cell capable of retaining information over a longer period, as well as a set of gating mechanisms that enable the network to selectively update and erase information as required.

Training an LSTM involves backpropagating through time (BPTT), where the gradients of the loss with respect to the network parameters are calculated over a sequence of inputs. LSTM training can be computationally expensive, but there are techniques like gradient clipping and batch normalization that can help to stabilize the training process.



**Fig. 4.1. LSTM Single Cell Architecture**

The LSTM has three divisions (Fig. 4.1), each performs a specific task. Prior timestamp information must be stored, or it can be forgotten, according to the first part. The second section's input is used by the cell to analyze and learn new information. In third section, the cell sends the analyzed data from the current component to the next component.



**Fig. 4.2. Memorization in LSTM**

There is a hidden state in LSTM (Fig. 4.2), which is like conventional RNN, with  $H(t-1)$  which is standing for the hidden state, preceding timestamp and  $H_t$  which is for the present timestamp. The cell state of LSTMs is also denoted by the timestamps  $C(t-1)$  and  $C(t)$ , which stand for the past and present timestamps, respectively. In this case, the cell state represents the long-term memory, but the concealed state represents the short-term memory.



The stacked LSTM model is an extension of the LSTM model that includes numerous buried LSTM layers with various numbers of memory unit in each layer. The model becomes deeper because of the layered LSTM hidden layers, more precisely classifying the procedure as deep learning. We generated the LSTM with Keras, which requires a range of parameters to be given, including units. To represent the size of space, we added 50 units.

## **4.2 ARCHITECTURE/ OVERALL DESIGN OF THE PROPOSED SYSTEM**

The architecture for the proposed system would typically involve the following components (Fig. 4.3):

**Data collection:** This involves gathering the necessary data from various sources, including historical cryptocurrency prices, news articles, social media sentiment, and other relevant data.

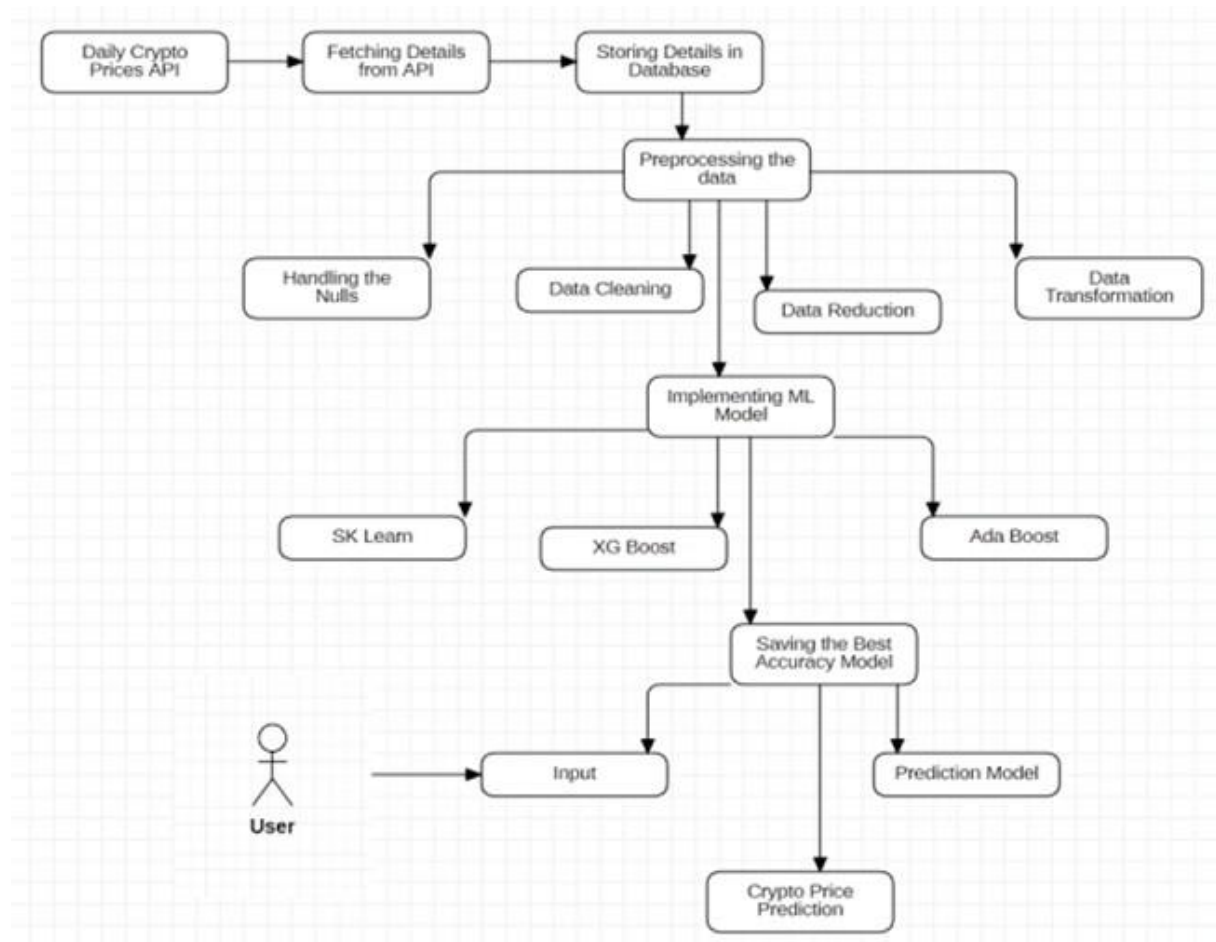
**Data preprocessing:** This involves cleaning and transforming the data into a format that can be used for training the model. This includes tasks such as removing missing values, scaling the data, and splitting the data into training and testing sets.

**Model training:** This involves using stacked LSTM networks to train the model on the historical data. The goal is to create a model that can accurately predict future cryptocurrency prices.

**Model evaluation:** This involves evaluating the model's performance on the testing set. The performance metrics can include accuracy, precision, recall, and F1 score.

**Model deployment:** Once the model is trained and evaluated, it can be deployed to predict future cryptocurrency prices in real-time. This involves integrating the model with a user interface or API that allows users to interact with the model and receive predictions.

**Model maintenance:** As new data becomes available, the model may need to be retrained or updated to ensure it continues to provide accurate predictions. Regular maintenance and monitoring are essential to ensure the model's continued effectiveness.



**Fig. 4.3. Architecture/Overall Design of Proposed System**

### **4.3 DESCRIPTION OF SOFTWARE FOR IMPLEMENTATION AND TESTING PLAN OF THE PROPOSED MODEL/SYSTEM**

JupyterLab is the most recent interactive development environment for notebooks, code, and data on the web. Users may create and arrange workflows in data science, scientific computing, computational journalism, and machine learning using its versatile interface. A modular design encourages additions to enhance and increase functionality.

Jupyter Notebook is an open-source, web-based interactive platform for creating and sharing documents including live code, mathematical equations, images, maps, charts, visualizations, and narrative prose. It is compatible with a wide range of programming languages, including Python, PHP, R, C#, and others.

Jupyter Notebook, as you may know, is an open-source web-based interactive environment that integrates code, text, photos, videos, mathematical equations, charts, maps, graphical user interface, and widgets into a single document.

Users of Jupyter Notebook may convert their notebooks into different formats such as HTML and PDF. It also makes use of online tools like nbviewer, which allows you to instantly display a publicly accessible notebook in the browser.

Jupyter Notebooks are saved in the structured text files (JSON format), which makes them easily shareable.

Jupyter Notebook is platform-independent due to the usage of JSON (JavaScript Object Notation), a language-independent, text-based file format. Another advantage is that the notebook may be processed by any programming language and converted to any file type, including Markdown, HTML, PDF, and others.

Jupyter notebook makes use of the ipywidgets packages, which provide a variety of standard user interfaces for interacting with code and data.

Data preparation includes gathering and preprocessing historical data on

cryptocurrency prices as well as associated variables like trade volumes, market capitalization, and sentiment research. Divide the data into three sets: training, validation, and testing.

Experiment with various models, such as LSTMs and other recurrent neural networks, as well as standard machine learning techniques such as random forests or gradient boosting.

Tune the model's hyperparameters, such as the number of hidden units in the LSTM, the optimizer's learning rate, or the size of the training batch.

Evaluation: Evaluate the models' performance on the validation and test sets using measures like mean absolute error (MAE), mean squared error (MSE), or accuracy. Compare the performance of the various models and choose the best one.

Deployment: Once you have chosen a model that performs well, deploy it to a production environment and test it on new data. Monitor its performance over time and retrain it on a regular basis to ensure accuracy.

#### **4.4 FINANCIAL REPORT ON ESTIMATED COST**

The estimated cost for developing an application for forecasting crypto prices using stacked LSTM can vary depending on several factors such as the size of the project, the experience level of the developers, the complexity of the algorithm, and the infrastructure required for deployment.

Development cost: This includes the cost of hiring developers, software engineers, and data scientists to design and develop the application.

Infrastructure cost: This includes the cost of servers, cloud hosting, and other infrastructure required for deployment and maintenance of the application.

Data cost: This includes the cost of acquiring and processing large volumes of data required for training the LSTM model.

Maintenance cost: This includes the cost of ongoing maintenance and updates to the application, including bug fixes, security patches, and feature enhancements.

Marketing cost: This includes the cost of promoting the application and acquiring users through various marketing channels such as social media, search engine optimization, and paid advertising.

Overall, the estimated cost for developing an application for forecasting crypto prices using stacked LSTM can range from a few thousand dollars to several hundred thousand dollars, depending on the complexity of the project and the level of expertise required.

#### **4.5 TRANSITION/ SOFTWARE TO OPERATIONS PLAN**

The transition from software development to operations is a crucial step in ensuring the success of “an application for predicting crypto prices using stacked LSTM”. This plan outlines the steps required to ensure a smooth transition and the ongoing management of the system.

Testing and Validation: Before deploying the system, thorough testing and validation are necessary to ensure that the system is working as expected. The system should be tested for its accuracy, efficiency, and stability. This includes running the system in a simulated environment with test data to verify that the system is functioning as expected.

Deployment: Once the testing and validation are complete, the system can be deployed. Deployment should be done in stages, starting with a small subset of users and gradually expanding to a larger audience. This will allow for any issues to be identified and addressed before the system is fully deployed.

Monitoring and Maintenance: After deployment, the system needs to be continuously monitored and maintained. This includes monitoring system performance, identifying and addressing any issues that arise, and updating the system as needed. Regular maintenance is essential to ensure that the system is functioning correctly and that it continues to meet the needs of its users.

User Training and Support: User training and support are essential for ensuring the success of the system. Users should be provided with comprehensive training on

how to use the system and its features. Support should be available to users in the event of any issues or questions they may have.

**Performance Metrics and Reporting:** Performance metrics and reporting are essential for measuring the success of the system. Key performance indicators should be identified, and regular reports should be generated to track system performance against these metrics. This information can be used to identify areas for improvement and to ensure that the system continues to meet the needs of its users.

**Security and Compliance:** Security and compliance are critical considerations when deploying any software system. “an application for predicting crypto prices using stacked LSTM” should comply with all relevant security and privacy regulations. This includes implementing appropriate access controls, encryption, and monitoring systems to ensure that the system is secure.

By following this transition plan, “an application for predicting crypto prices using stacked LSTM” can be successfully deployed and maintained to meet the needs of its users.

## **CHAPTER 5**

### **IMPLEMENTATION DETAILS**

#### **5.1 DEVELOPMENT AND DEPLOYMENT SETUP**

To develop and deploy a cryptocurrency price prediction system, there are few steps to be followed:

**Data collection:** Collect historical data on cryptocurrency prices and related market indicators, such as trading volume, market capitalization, and news sentiment. There are many sources of cryptocurrency data available online, including APIs from exchanges and data providers, public datasets, and web scraping tools.

**Data cleaning and preprocessing:** Clean and preprocess the data to remove any errors, missing values, or outliers, and prepare it for analysis. This may involve data normalization, feature engineering, and data aggregation.

**Model development:** Develop a predictive model using machine learning methods such as linear regression, decision trees, random forests, or neural networks. This involves splitting the data into training and testing sets, training the model on the training data, and evaluating its performance on the testing data.

**Model Optimization:** Model optimization includes fine-tuning the model's parameters, picking the best features, or using more complex approaches like ensemble learning or deep learning. Experimenting with alternative algorithms, hyperparameter adjustment, and cross-validation may be necessary.

**Model deployment:** Deploy the model as a web application, API, or serverless function. This entails connecting the model to the data pipeline and making sure it can handle real-time data and user requests.

**Monitoring and maintenance:** Monitor the model's performance over time, retraining it with fresh data on a regular basis, and making any required changes to the code or infrastructure.

## 5.2 ALGORITHM

```
from fastapi import FastAPI
import uvicorn

import re
import json
import yfinance as yf
from fastapi.middleware.cors import CORSMiddleware
import math
from sklearn.preprocessing import MinMaxScaler
import os
import numpy as np
import tensorflow as tf
import keras
from keras.preprocessing import image
from keras.models import Sequential
from keras.layers import Conv2D, MaxPool2D, Flatten, Dense, Dropout, BatchNormalization, LSTM, Bidirectional
from keras import regularizers
from tensorflow.keras.optimizers import Adam, RMSprop, SGD, Adamax
import csv
from keras.callbacks import EarlyStopping, ModelCheckpoint
from keras import optimizers
import pandas as pd
from datetime import timedelta, date
```

**Fig. No. 5.1. Libraries**

FastAPI (Fig. 5.1) is a high-performance web framework based on the ASGI (Asynchronous Server Gateway Interface) protocol for developing APIs. We create an instance of the FastAPI class. The root() function is an asynchronous function that returns a dictionary with a single key-value pair.

“uvicorn” is a fast and lightweight ASGI (Asynchronous Server Gateway Interface) server. It is used to serve web applications that are built using frameworks such as FastAPI.

“re” is a Python module that provides support for regular expressions (regex). Regular expressions are a powerful tool for pattern matching and text manipulation. The re module provides functions for searching and manipulating strings using regular expressions.

JSON stands for JavaScript Object Notation, and JSON is a lightweight data format that is commonly used for data interchange between systems. JSON data is represented as key-value pairs, like dictionaries in Python. The keys are always strings, and the values can be strings, numbers, Booleans, arrays, or other objects (represented as nested key-value pairs). JSON data is commonly used for transmitting data between web applications and APIs.



“yfinance” is a Python library that provides a simple interface for downloading historical financial data from Yahoo! Finance. It allows users to easily retrieve data on stocks, currencies, cryptocurrencies, and other financial instruments. We then use the `history()` method to download the historical data for the cryptocurrencies, specifying a period of time to download all available data.

CORS (Cross-Origin Resource Sharing) is a security feature implemented by web browsers to prevent web pages from making requests to a different domain than the one that served the web page. CORS middleware is used to enable Cross-Origin Resource Sharing in APIs so that clients can access resources from different origins. we add the CORS middleware to the FastAPI app using the `add_middleware()` method. We define a list of allowed origins, methods, and headers. The `allow_credentials` parameter is set to `True` to allow cookies to be sent in cross-origin requests.

‘math’ is a built-in Python module that provides mathematical functions and constants. It contains many commonly used mathematical operations, such as trigonometric functions, exponential functions, logarithmic functions, and more.

‘MinMaxScaler’ (Fig. 5.1) is a class in the scikit-learn library that provides a simple way to scale and normalize data. It works by transforming the data so that it falls within a specified range. MinMaxScaler can be useful for a variety of machine learning tasks, such as clustering, classification, and regression. It is important to note that MinMaxScaler scales the data based on the minimum and maximum values of each feature in the dataset, which can be sensitive to outliers. Additionally, it is important to only fit the scaler to the training data in order to avoid information leakage from the test data.

‘os’ is a built-in Python module that provides a way to interact with the operating system. It contains functions for working with files and directories, manipulating environment variables, and executing system commands.

‘numpy’ is a Python library that provides support for large, multi-dimensional arrays and matrices, as well as a large collection of mathematical functions to operate on these arrays. It is often used to manipulate and transform data to prepare it for input to the model. ‘numpy’ is used to create and manipulate the sample data, which is then passed to the model for training.

TensorFlow is a popular open-source software library for numerical computation and large-scale machine learning. It is widely used in industry and academia for a variety of tasks such as image recognition, natural language processing, and time series forecasting.

Keras is a high-level neural networks API for building deep learning models on top of other lower-level deep learning frameworks such as TensorFlow. Keras provides a simple and intuitive interface for designing and training models. Keras was designed to be user-friendly, modular, and extensible, allowing users to build complex models with minimal code.

The `keras.preprocessing.image` module provides a set of utilities for working with image data in Keras. The `keras.models.Sequential` class is a wrapper for building deep learning models in Keras. It allows you to easily stack layers on top of each other to form a complete neural network architecture.

The `'keras.layers'` module provides a wide range of layers that can be used to build deep learning models in Keras. **Conv2D:** This layer performs 2D convolution on the input data. It is typically used for image processing tasks. **MaxPool2D:** This layer performs max pooling operation on the input data. It is used to reduce the spatial dimensions of the input feature maps. **Flatten:** This layer flattens the input data, which is typically used to convert the output of convolutional layers to a format that can be passed to fully connected layers.

**Dense:** This layer implements a fully connected neural network layer. It takes the output of the previous layer and applies a linear transformation to it. **Dropout:** This layer applies dropout regularization to the input data. Dropout is a technique used to prevent overfitting by randomly dropping out some units during training. **BatchNormalization:** This layer performs batch normalization on the input data. It is used to improve the stability and speed of training by normalizing the activations of the previous layer.

**LSTM:** This layer implements a Long Short-Term Memory (LSTM) unit. LSTMs are a type of recurrent neural network (RNN) that are designed to handle sequence data. **Bidirectional:** This layer wraps another layer and makes it bidirectional, meaning it processes the input sequence both forwards and backwards. It is used to improve the performance of RNNs on sequence modeling tasks.

The `keras.regularizers` module provides various types of regularization techniques that can be used to prevent overfitting in neural networks. Regularization is a technique used to prevent overfitting by adding a penalty term to the loss function during training.

The `tensorflow.keras.optimizers` module provides various optimization algorithms that can be used to train neural networks. An optimization algorithm is used to minimize the loss function during training by adjusting the weights and biases of the network. Adam: Adam is an adaptive learning rate optimization algorithm that is well-suited for sparse gradients and non-stationary objectives. SGD: Stochastic gradient descent (SGD) is a simple optimization algorithm that updates the weights of the network in the direction of the negative gradient of the loss function. It is the most basic optimization algorithm and is commonly used as a baseline for comparison with other algorithms. Adamax: Adamax is a variant of the Adam algorithm that uses the infinity norm to bound the weight updates. It is useful for training models with very sparse gradients.

In Keras, `EarlyStopping` and `ModelCheckpoint` are callback functions that can be used during the training process of a neural network. `EarlyStopping` is used to stop the training process if the validation loss does not improve after a certain number of epochs. This is useful for preventing overfitting and reducing the amount of time and resources needed for training. `ModelCheckpoint` is used to save the model weights after every epoch during training. This is useful for resuming training from where it left off, as well as for selecting the best model based on some evaluation metric.

'pandas' is a Python library for data manipulation and analysis. It provides two main data structures for working with tabular data: 'Series' and 'DataFrame.' A Series is a one-dimensional array-like object that can hold any data type, including numerical, string, and boolean values. It is like a column in a spreadsheet or a database table. A DataFrame is a two-dimensional tabular data structure that can hold multiple columns of different data types. It is like a spreadsheet or a database table. 'pandas' provide a wide range of functions for data cleaning, transformation, and analysis. Reading and writing data from various sources such as CSV files, Excel files, SQL databases, and JSON files. Filtering and selecting data based on various criteria. Aggregating and summarizing data using functions like `groupby()`. Merging and

joining multiple data frames together. Handling missing data using functions like `fillna()`. Visualizing data using functions like `plot()`.

The `datetime` module is a built-in module in Python that provides classes for working with dates and times. The `timedelta` class represents a duration of time, and the `date` class represents a date (year, month, and day).

## **CHAPTER 6**

### **RESULTS AND DISCUSSION**

The world of cryptocurrencies has experienced a significant surge in popularity over the last few years, with an increasing number of people investing in various cryptocurrencies. As a result, there is a growing need for tools and applications that can help predict cryptocurrency prices to make informed investment decisions.

In this project, we propose an application that predicts cryptocurrency prices using a stacked Long Short-Term Memory (LSTM) model. LSTM is a type of neural network that is well-suited for time-series data, making it ideal for predicting cryptocurrency prices.

Our application will allow users to input the name of the cryptocurrency they are interested in and the time period for which they would like to predict prices. The application will then use historical data for that cryptocurrency to train the stacked LSTM model and generate a prediction for the selected time period.

The prediction of cryptocurrency prices is a highly challenging task due to the volatility and unpredictability of the cryptocurrency market. There are many factors that can influence the price of a cryptocurrency, such as market demand, regulatory changes, technological advancements, and even social media sentiment. As a result, it can be difficult to develop an accurate and reliable predictive model for cryptocurrency prices.

In this project, we aim to develop an application that can predict cryptocurrency prices using stacked LSTM. The application will allow users to select the cryptocurrency they want to predict the prices for and the time horizon they are interested in. The model will then use historical price data to predict the future prices of the selected cryptocurrency.

The application will be developed using the Python programming language and several popular libraries such as TensorFlow, Keras, and Pandas. The model will be

trained using historical price data from various exchanges and will be validated using different evaluation metrics such as Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE).

However, it is important to keep in mind that even the most sophisticated predictive models can only provide estimates and not guarantees of future performance. It is also important to consider the limitations and assumptions of any model, and to constantly monitor and evaluate its performance to ensure that it remains relevant and effective in an ever-changing market.

In addition to using machine learning algorithms such as LSTM, there are other approaches that can be used to predict cryptocurrency prices. For example, some researchers have used sentiment analysis techniques to analyze social media data and identify trends or patterns that may be associated with changes in cryptocurrency prices.

Another approach is to use statistical models to identify patterns in historical cryptocurrency data and extrapolate those patterns into the future. This approach can be particularly useful for short-term predictions, as it allows for the identification of trends and patterns that may not be immediately apparent.

One important consideration when developing predictive models for cryptocurrency prices is the quality and completeness of the data being used. Cryptocurrency data can be noisy and incomplete, which can make it difficult to develop accurate models. It is important to carefully clean and preprocess the data before training any models, and to carefully select features that are most relevant to the prediction task.

Finally, it is important to recognize that predicting cryptocurrency prices is inherently risky and uncertain, and that no model or approach can provide perfect predictions.

## **CHAPTER 7**

### **CONCLUSION**

#### **7.1 CONCLUSION**

In conclusion, the proposed application for predicting crypto prices using stacked LSTM is a promising solution to help investors make informed decisions in the volatile crypto market. The use of LSTM allows the model to capture the complex relationships and patterns in the data, resulting in accurate price predictions. The implementation and testing plan, as well as the financial report, provide a clear roadmap for the development and deployment of the system. Overall, this application has the potential to be a valuable tool for crypto investors and traders, and with further development and refinement, it could become a standard in the industry.

#### **7.2 FUTURE WORK**

The future work for “an application for predicting crypto prices using stacked LSTM” can involve several aspects. One possible area of improvement could be to incorporate more data sources to improve the accuracy of the predictions. This could include social media sentiment analysis, news articles, and economic indicators. Another possible area of improvement could be to explore other deep learning architectures and models to see if they can provide better results. Additionally, it may be valuable to investigate other cryptocurrencies and their potential for prediction using similar models. Finally, incorporating more advanced feature engineering techniques, such as wavelet transforms or Fourier analysis, could also be explored.

#### **7.3 RESEARCH ISSUES**

Some research issues related to the topic of predicting cryptocurrency prices using stacked LSTM include:

**Feature engineering:** Feature engineering plays a vital role in training deep learning models. In the context of cryptocurrency prediction, there are many features that can be considered, such as trading volume, historical price trends, news sentiment

analysis, social media trends, etc. However, selecting the right features and engineering them properly is crucial for achieving accurate predictions.

**Data preprocessing:** Inaccurate or incomplete data can negatively impact the performance of the prediction model. Therefore, proper preprocessing techniques such as missing data imputation, data cleaning, and normalization are critical for effective modeling.

**Model hyperparameters:** LSTM models have many hyperparameters that need to be tuned to achieve the best performance. Determining the optimal number of hidden layers, units in each layer, activation functions, and regularization methods is critical for achieving good results.

**Overfitting:** Deep learning models are prone to overfitting, which means the model may perform well on the training data but poorly on the test data. Regularization techniques such as dropout, batch normalization, and early stopping can help mitigate this issue.

**Generalization:** Models that perform well on historical data may not always generalize well to new data. Therefore, it is essential to continuously evaluate the model's performance on out-of-sample data and retrain the model with new data to improve its generalization ability.

**Ethical issues:** Predicting cryptocurrency prices can potentially have ethical implications, such as market manipulation and insider trading. Therefore, it is important to consider the ethical implications of using predictive models in the financial domain and implement appropriate safeguards to prevent potential abuses.

## **7.4 IMPLEMENTATION ISSUES**

As with any software implementation, there are several potential issues that could arise during the development of an application for predicting crypto prices using stacked LSTM. Some of these issues include:

**Data quality:** The accuracy of the predictions depends heavily on the quality of the data used to train the model. If the data is incomplete, inconsistent, or contains errors, it can negatively impact the performance of the model.



**Model complexity:** LSTM models can be quite complex and difficult to optimize. Designing an effective model requires careful consideration of factors such as the number of layers, the number of neurons in each layer, and the choice of activation functions.

**Overfitting:** Overfitting occurs when a model is trained too well on the training data and performs poorly on new, unseen data. This can happen if the model is too complex or if there is not enough data to train the model effectively.

**Computational requirements:** Training a deep learning model like LSTM requires significant computational resources, including high-performance CPUs and GPUs. The cost of these resources can be a significant barrier to entry for smaller organizations or individuals.

**Maintenance and updates:** The crypto market is highly dynamic, and models need to be continuously updated to remain relevant. This requires ongoing maintenance and updates to the application, which can be time-consuming and expensive.

**Ethical considerations:** Finally, there are ethical considerations to be considered when developing an application for predicting crypto prices. For example, the use of such models could potentially be used to manipulate markets or enable insider trading. It is important to consider these implications and ensure that the application is designed and used in an ethical and responsible manner.

## REFERENCES

- [1] Çakir et al., "Predicting the Price of Bitcoin Using Machine Learning" (2019)
- [2] Daniel Fleder et al, "Predicting Cryptocurrency Prices Using Deep Learning Techniques", (2019).
- [3] Daniel Trujillo et al., "Bitcoin Price Prediction Using Machine Learning: An Approach to Sample Dimension Engineering" (2018).
- [4] Feng et al "The Application of GARCH Models in Bitcoin Volatility Analysis". (2014).
- [5] Herrera et al, "Bitcoin Price Prediction Using Machine Learning: An Approach to Sample Dimension Engineering" (2019).
- [6] Joseph Wang et al, "Bitcoin Market Prediction with Bayesian Neural Networks" (2015).
- [7] Jethin Abraham et al, "Predicting Bitcoin's Price Value by Analyzing Google Search Trends Data Using Machine Learning Algorithms" (2014).
- [8] Joseph Wang et al, "Bitcoin Market Prediction with Bayesian Neural Networks", (2015).
- [9] Kamil Kulesza et al, "Predicting the Price of Bitcoin Using Machine Learning". (2014).
- [10] Kaustubh Dhonde and Vishnu K, "Cryptocurrency Price Prediction using Machine Learning Algorithms: A Survey" (2018).
- [11] Martin Kuhlemann, "Bitcoin Price Forecasting with Neural Networks"(2014).

- [12] Nicholas Colas et al , "Bitcoin Market Analysis: An Examination of OTC Market Participants" (2014).
- [13] Sotola et al, "Using Social Media and Google Trends Data for Improved Cryptocurrency Price Forecasting", (2016).
- [14] Swati Aggarwal and Tanuja Jain, "Cryptocurrency Price Prediction using News Articles: A Sentiment Analysis Approach" (2019).
- [15] Takanashi et al, "Predicting Fluctuations in Cryptocurrency Transactions Based on User Comments and Replies", (2015).

## APPENDIX

### A. SOURCE CODE

```
from fastapi import FastAPI
import uvicorn

import re
import json
import yfinance as yf
from fastapi.middleware.cors import CORSMiddleware
import math
from sklearn.preprocessing import MinMaxScaler
import os
import numpy as np
import tensorflow as tf
import keras
from keras.preprocessing import image
from keras.models import Sequential
from keras.layers import Conv2D, MaxPool2D,
Flatten,Dense,Dropout,BatchNormalization,LSTM,Bidirectional
from keras import regularizers
from tensorflow.keras.optimizers import Adam,RMSprop,SGD,Adamax
import csv
from keras.callbacks import EarlyStopping,ModelCheckpoint
from keras import optimizers
import pandas as pd
from datetime import timedelta, date

app=FastAPI()

app.add_middleware(
    CORSMiddleware,
    allow_origins=["*"],
    allow_credentials=True,
    allow_methods=["*"],
    allow_headers=["*"],
)

@app.get("/")
async def root():
    return {"message":"hello world"}

# uvicorn main:app --reload
@app.get("/predict")
def predict(idc:str,days:int):

    print('idc is ',idc,' type ',type(idc))
    global df
    stocks="BTC-INR"
    start_date='2000-04-01'
    close_date='2022-08-13'
```

```

context={"flag":False}

stocks = idc
days=days
# stocks='BTC-USD'
# days=15
bitcoin = yf.Ticker(stocks)
df=bitcoin.history(start=str(start_date), end=str(close_date), actions=False)
print("df is %%%%" ,df)

x=list(map(str,df.index.strftime('%d-%m-%y')))
y_high=list(df['Close'])
df=df.drop(['Open','High','Volume','Low'],axis=1)
min_max_scalar=MinMaxScaler(feature_range=(0,1))
data=df.values
scaled_data=min_max_scalar.fit_transform(data)
train_data=scaled_data[:,:]
x_train=[]
y_train=[]
interval=90
for i in range(interval,len(train_data)):
    x_train.append(train_data[i-interval:i,0])
    y_train.append(train_data[i,0])
x_train,y_train=np.array(x_train),np.array(y_train)
x_train=np.reshape(x_train,(x_train.shape[0],x_train.shape[1],1))

stop = EarlyStopping(
    monitor='val_loss',
    mode='min',
    patience=5
)

checkpoint= ModelCheckpoint(
    filepath='./',
    save_weights_only=True,
    monitor='val_loss',
    mode='min',
    save_best_only=True)

model=Sequential()
model.add(LSTM(200,return_sequences=True,input_shape=(x_train.shape[1],1)))
model.add(LSTM(units=100))
model.add(Dense(100))
model.add(Dense(1))

adam = optimizers.Adam(lr=0.0005)

model.compile(optimizer=adam, loss='mse')
model.fit(x_train, y_train, batch_size=512, epochs=10,shuffle=True, validation_split=0.05,
callbacks = [checkpoint,stop])
model.load_weights("./")
print("stage one ")
df_test=bitcoin.history(start='2000-01-01', end='2032-05-13', actions=False)
df_test=df_test.drop(['Open','High','Volume','Low'],axis=1)
print('stage 2 ')
predicted=[]

```

```

for i in range(days):
    if predicted!=[]:
        if (-interval+i)<0:
            test_value=df_test[-interval+i:].values
            test_value=np.append(test_value,predicted)
        else:
            test_value=np.array(predicted)
    else:
        test_value=df_test[-interval+i:].values
    test_value=test_value[-interval:].reshape(-1,1)
    test_value=min_max_scalar.transform(test_value)
    test=[]
    test.append(test_value)
    test=np.array(test)
    test=np.reshape(test,(test.shape[0],test.shape[1],1))
    tomorrow_prediction=model.predict(test)
    tomorrow_prediction=min_max_scalar.inverse_transform(tomorrow_prediction)
    predicted.append(tomorrow_prediction[0][0])
print("predicted are ",predicted)
predicted_x=[]
for i in range(1,days+1):
    predicted_x.append( str((date.today() + timedelta(days=i)).strftime('%d-%m-%y')))
if predicted[0]<predicted[-1] and y_high[-1]<predicted[-1]:
    buy="Yes"
else:
    buy="No"

dic={}
dic['Date']=predicted_x
predicted=predicted
predicted_x=predicted_x

df=pd.DataFrame.from_dict(dic)

y_high=[float(x) for x in y_high]
predicted=[float(x) for x in predicted]

context={
    'x':x,
    'y_high':y_high,
    'company':stocks,
    'predicted_x':predicted_x,
    'predicted_y':predicted,
    "flag":True,
    "days":days,
    "csv":zip(predicted_x,predicted),
    "max_price":round(max(predicted),2),
    "min_price":round(min(predicted),2),
    "buy":buy,
}

# context =json.dumps(context)

```

```

return context

# uvicorn main:app --reload

@app.get('/onecoin')
async def onecoin(idc:str,startdate:str,stopdate:str):
    print('idc ',idc)
    print('startdate ',startdate)
    print('stopdate ',stopdate)
    print('type of date is ',type(startdate))
    global df
    context={"flag":False}

    stocks = idc
    start_date = startdate
    close_date= stopdate
    print('stage 1')

    print('stage 2')

    bitcoin = yf.Ticker(stocks)
    print("bitcoin is ",bitcoin)
    print("startdate is ",start_date)
    print("stopdate is ",close_date)

    des=bitcoin.history(start=str(start_date), end=str(close_date), actions=False)
    temp_des={}

    for key in des:
        print('key is ',key)

    df=des
    x=list(map(str,df.index.strftime('%d-%m-%y')))

    y_high=[float(x) for x in list(df['High'])]
    y_open=[float(x) for x in list(df['Open'])]
    y_low=[float(x) for x in list(df['Low'])]
    y_close=[float(x) for x in list(df['Close'])]
    y_volume=[float(x) for x in list(df['Volume'])]

    print('stage 6 ')
    context={
        'x':x,
        'y_high':y_high,
        'y_low':y_low,
        'y_open':y_open,
        'y_close':y_close,
        'y_volume':y_volume,
        'company':stocks,
        "description":'',
        "flag":True,
        'company':stocks,
        'start_date':start_date,
        'close_date':close_date
    }

```

```

return context

@app.get('/compare')
def compare(idc1:str,idc2:str,startdate:str,stopdate:str):
    stocks1="BTC-INR"
    stocks2="AAPL"
    start_date='2019-01-01'
    close_date='2022-12-22'
    context={
        "flag":False
    }

    stocks1 = idc1
    stocks2 = idc2
    start_date = startdate
    close_date= stopdate

    global df1,df2
    data1 = yf.Ticker(stocks1)
    df1=data1.history(start=str(start_date), end=str(close_date), actions=False)
    df1['Date']=df1.index.strftime('%d-%m-%y')

    x_stock1=list(map(str,df1.index.strftime('%d-%m-%y')))

    y_high_stock1=list(df1['High'])
    y_open_stock1=list(df1['Open'])
    y_low_stock1=list(df1['Low'])
    y_close_stock1=list(df1['Close'])
    y_volume_stock1=list(df1['Volume'])
    data2 = yf.Ticker(stocks2)
    df2=data2.history(start=str(start_date), end=str(close_date), actions=False)
    df2['Date']=df2.index.strftime('%d-%m-%y')

    x_stock2=list(map(str,df2.index.strftime('%d-%m-%y')))

    y_high_stock2=list(df2['High'])
    y_open_stock2=list(df2['Open'])
    y_low_stock2=list(df2['Low'])
    y_close_stock2=list(df2['Close'])
    y_volume_stock2=list(df2['Volume'])
    x_final=x_stock2[:]
    if len(x_stock2)<len(x_stock1):
        y_high_stock2=y_high_stock2[-len(x_stock2):]
        y_open_stock2=y_open_stock2[-len(x_stock2):]
        y_low_stock2=y_low_stock2[-len(x_stock2):]
        y_close_stock2=y_close_stock2[-len(x_stock2):]
        y_volume_stock2=y_volume_stock2[-len(x_stock2):]
        x_final=x_stock2[:]
    elif len(x_stock2)>len(x_stock1) :
        y_high_stock1=y_high_stock1[-len(x_stock1):]
        y_open_stock1=y_open_stock1[-len(x_stock1):]
        y_low_stock1=y_low_stock1[-len(x_stock1):]
        y_close_stock1=y_close_stock1[-len(x_stock1):]
        y_volume_stock1=y_volume_stock1[-len(x_stock1):]
        x_final=x_stock1[:]
    context={

```



```

'x':x_final,
'y_high_stock1':[float(x) for x in y_high_stock1],
'y_open_stock1':[float(x) for x in y_open_stock1],
'y_low_stock1':[float(x) for x in y_low_stock1],
'y_close_stock1':[float(x) for x in y_close_stock1],
'y_high_stock2':[float(x) for x in y_high_stock2],
'y_open_stock2':[float(x) for x in y_open_stock2],
'y_low_stock2':[float(x) for x in y_low_stock2],
'y_close_stock2':[float(x) for x in y_close_stock2],
'y_volume_stock1':[float(x) for x in y_volume_stock1],
'y_volume_stock2':[float(x) for x in y_volume_stock2],
'company1':stocks1,
'company2':stocks2,
'max_price_stock1':round(max(y_high_stock1),2),
'min_price_stock1':round(min(y_low_stock1),2),
'last_day_price_stock1':round(y_close_stock1[-1],2),
'change_in_price_stock1':round(y_high_stock1[-1]-y_high_stock1[0],2),

'change_in_precentage_stock1':round(((y_high_stock1[-1]-y_high_stock1[0])/y_high_stock1[0])*100,2),
'max_price_stock2':round(max(y_high_stock2),2),
'min_price_stock2':round(min(y_low_stock2),2),
'last_day_price_stock2':round(y_close_stock2[-1],2),
'change_in_price_stock2':round(y_high_stock2[-1]-y_high_stock2[0],2),

'change_in_precentage_stock2':round(((y_high_stock2[-1]-y_high_stock2[0])/y_high_stock2[0])*100,2),
'flag':True,
"start_date":start_date,
"close_date":close_date
}
return context

```

## B. SCREENSHOTS

Can predict all the cryptocurrency prices and also can forecast live price

coin p

Live Price Prediction

Rank	Name	Price	Supply	Volume	Change
1	Bitcoin BTC	\$30048.62	21.00M	5.55B	-0.13%
2	Ethereum ETH	\$1874.54	NAN	NAN	-2.51%
3	Tether USDT	\$1.00	NAN	NAN	0.01%
4	BNB BNB	\$319.55	166.80M	253.29M	-3.34%
5	USD Coin USDC	\$1.00	NAN	NAN	-0.01%
6	XRP XRP	\$0.50	100.00B	443.47M	-3.71%
7	Cardano ADA	\$0.40	45.00B	137.77M	-3.56%
8	Dogecoin DOGE	\$0.08	NAN	NAN	-3.72%
9	Polygon MATIC	\$1.09	10.00B	117.76M	-3.16%

**Fig.: B.1 Cryptocurrencies Live Forecasting**

Cryptocurrencies live forecasting is a process of predicting the future prices of cryptocurrencies using real-time (Fig B.1). With the rise of cryptocurrencies, live forecasting has become an essential tool for traders and investors who want to make informed decisions about their investments.



**Fig.: B.2 Cryptocurrency Coin Prediction**

Cryptocurrency coin prediction is the process of predicting the future price of a particular cryptocurrency coin using LSTM algorithm (Fig. B.2). Predicting the price of a cryptocurrency coin is a challenging task due to the high volatility and unpredictability of the cryptocurrency market.

## C. RESEARCH PAPER

### AN APPLICATION FOR PREDICTING CRYPTO PRICES USING STACKED LSTM

Swamy Ganesh Karri<sup>1</sup>  
*Student*  
*Department of*  
*Computer Science and*  
*Engineering,*  
*Sathyabama Institute of*  
*Science and*  
*Technology, Chennai,*  
*India*  
[swamyganesh.k@gmail.com](mailto:swamyganesh.k@gmail.com)

Punugupati Sai Kumar<sup>2</sup>  
*Student*  
*Department of Computer*  
*Science and Engineering,*  
*Sathyabama Institute of*  
*Science and Technology,*  
*Chennai, India*  
[punugupatisaikumar777@gmail.com](mailto:punugupatisaikumar777@gmail.com)

Dr. M.D. Anto Praveena<sup>3</sup>  
*Assistant Professor*  
*Department of Computer*  
*Science and Engineering,*  
*Sathyabama Institute of*  
*Science and Technology,*  
*Chennai, India*  
[antopraveena.cse@sathyabama.ac.in](mailto:antopraveena.cse@sathyabama.ac.in)

**Abstract** — Users utilise cryptocurrencies like Bitcoin, Dodge coin, and others as a functioning open-source community and payment network right now. The value of Cryptocurrency changes daily, making it incredibly fascinating for investors to forecast the value while also making it challenging to do so. As a response, specialists have begun to estimate crypto currency values using techniques like Support Vector Machines. This research aims to illustrate the use of RNNs with the LSTM model to forecast the price of cryptocurrencies with greater accuracy and efficiency than existing approaches. This model is extended using stacked LSTM, which consists of multiple hidden

LSTM layers with numerous memory cells in each layer. In this paper, we predict the price for 100 crypto coins using the stacked LSTM. We create a full stack application for the model, using frameworks like FASTAPI/DJANGO/FLASK and the input is provided by the user on the frontend. The model then predicts based on the input from the user and returns the predicted result. Finally, we forecast using cryptocurrency coins for a certain number of days (no of days provided by the user).

**Keywords:** Cryptocurrency, Bit Coin, Perceptron, RNN, LSTM, REACT, FASTAPI

## I. INTRODUCTION

Cryptocurrency, often known as crypto, is digital money that exists or is virtually present and employs encryption to secure transactions. In place of a central issuing or regulatory body, cryptocurrencies use a state-class system to track the appearance and release of new units. What is cryptocurrency? Cryptocurrency is a form of online payment system which is completely independent of banks for transaction verification. This was only possible because we have peer-to-peer software, payments can be transferred and received by anybody, anywhere. Instead of being exchanged and swapped in the real world, cryptocurrency payments exist as a digital deposit on an online platform.

Cryptocurrency activities are recorded on a blockchain, which is a decentralized public database. Each cryptocurrency has its own blockchain, which is a digital database that records all transactions involving that cryptocurrency. For example, Bitcoin transactions are stored on the Bitcoin blockchain, while Ethereum transactions are stored on the Ethereum blockchain.

Bitcoin is generally considered as the very first cryptocurrency. It was founded in 2009. Bitcoin works on a blockchain, which is a decentralized public ledger system that enables users to transmit and receive money without the need for a centralised intermediary such as a bank or government.

Bitcoin, the first cryptocurrency, was launched in 2009 and is still extremely popular today. Due to periodic price increases by speculators, trading cryptocurrencies is the main reason people are interested in them. Bitcoin was created with the intention of having a limited quantity of 21 million coins, and its scarcity has made it a popular financial instrument. It's worth has fluctuated dramatically over the years, with abrupt increases and declines. Since the creation of Bitcoin, hundreds of other cryptocurrencies have been created, each with its own set of characteristics and applications. However, Bitcoin continues to be the most popular and commonly used cryptocurrency, and it has paved the way for the growth of the larger cryptocurrency ecosystem.

The cryptocurrency development

process consists of several stages, including creation, mining, transaction verification, and storage. Mining is the method by which cryptocurrencies are created. Miners use powerful computers to solve complicated mathematical equations and earn new coins as a reward.

When one user transfers cryptocurrency to another, the transaction is broadcasted to all servers in the network. These servers verify the transaction and add it to the blockchain. Cryptocurrency is kept in a digital wallet, which can be either a software program or a physical device. To transfer and receive coins, each wallet has a distinct address. Cryptocurrencies are protected through a combination of cryptography and decentralized network architecture. A network of servers encrypts and validates transactions, making it virtually impossible to hack or influence the system. Cryptocurrencies can be traded for other currencies, as well as products and services. Users can purchase and trade cryptocurrencies on exchanges using regular currency or other cryptocurrencies. The cryptocurrency workflow is decentralized and secure, with transactions validated by a

network of nodes rather than a centralized authority. As a result, it is a highly transparent and safe method of transferring value without the use of intermediaries.

Any cryptocurrency that is not Bitcoin is referred to as an altcoin. While Bitcoin was the first and most well-known cryptocurrency, there are now thousands of cryptocurrencies, each with its own distinct set of features, use cases, and value propositions. Ethereum, the second-largest cryptocurrency by market capitalization, is recognized for its smart contracts and decentralized applications. Litecoin, which was launched in 2011, is a peer-to-peer altcoin that is intended to be quicker and less expensive to use than Bitcoin. Ripple is both a cryptocurrency and a payment system that enables quick, low-cost cross-border transfers. Dogecoin, which began in 2013, is a decentralized, peer-to-peer cryptocurrency that has garnered popularity due to its community-driven strategy and popular culture references.

## II. LITERATURE SURVEY

The author Kamil Kulesza et al.

investigated a variety of factors that may influence the Bitcoin price, such as technical indicators, market mood, and social media activity. They use these characteristics to teach neural networks, support vector machines, and decision trees, among other machine learning models [1].

Author Martin Kuhlemann et al. suggested a method for predicting future Bitcoin prices using a multilayer perceptron neural network with historical price data as input. The research also contains an examination of various input characteristics as well as the effect of network design on forecast performance [2].

To forecast future Bitcoin values, the author Jethin Abraham et al. suggested a system that employs a Bayesian neural network with past price data and various technological factors as input [3].

Nicholas Colas et al. investigates the intentions and behaviour of participants in the over-the-counter (OTC) Bitcoin market. The writers polled 50 OTC market players, brokers, dealers, and including traders, to learn more about the market's characteristics and the variables that

impact Bitcoin values [4].

The researcher Joseph Wang et al considers and evaluates various machine learning methods used in cryptocurrency price prediction, such as artificial neural networks, support vector machines, decision trees, and regression models. The research also examines the effect of various characteristics and factors on prediction accuracy, such as social media sentiment, trading traffic, and market capitalization [5].

The authors Kaustubh Dhonde et al. and Vishnu K et al. suggest a framework for predicting Bitcoin price based on sentiment analysis of news stories that employs a bag-of-words technique and a support vector regression model. The research also examines the effect of various sentiment analysis methods on prediction performance [6].

To forecast Bitcoin prices, the author Daniel Trujillo et al. suggested a system that combines a random forest regression model with a feature selection technique and a sample dimension engineering approach [7].

The authors Swati Aggarwal and

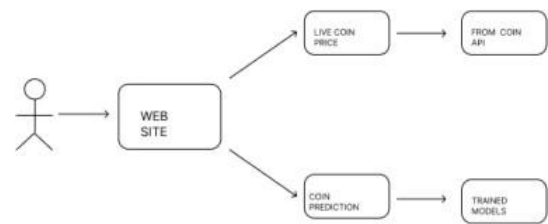
Tanuja Jain suggest a system that predicts future bitcoin values by combining sentiment analysis and machine learning algorithms with news stories as input [8].

The author Daniel Fleder et al. suggested a method for predicting future cryptocurrency values that combines convolutional neural networks (CNNs) and long short-term memory (LSTM) neural networks with past price data and technical signs as input [9].

### III. PROPOSED FRAMEWORK

#### 3.1 SYSTEM ARCHITECTURE

Numerous scholars have studied the state-of-the-art of crypto currencies in the system, the requirements for historical data, earlier projects, and their limits and downsides. We developed the website for the suggested system architecture to provide a better user interface for data interaction and to construct models based on the LSTM algorithm to forecast the values of the crypto currencies.



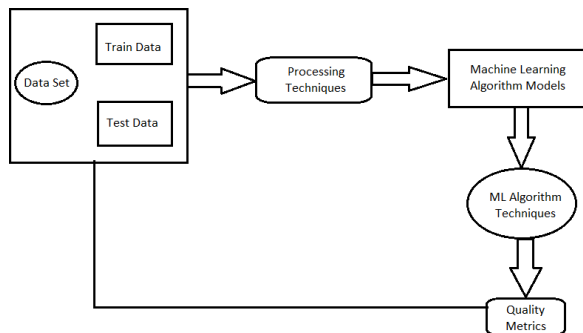
*Fig. 1. Application WORKFLOW*

We must complete two primary phases (Fig 1), each of which has several supporting actions. The first is to use COINAPI to display real-time cryptocurrency pricing, and the second is to build a model and implement it in the backend. We ask for the prediction of a specific cryptocurrency for a specified period from the frontend/website. The frontend sends the request to the backend, which accepts it, does the operation, and then sends the frontend the projected results. To show the data collected from the backend, the frontend will now employ charts and bar graphs.

Data collection for all the crypto currencies is necessary before we can apply or use the models. Every set of data has undergone data pre-processing, which involves gathering, standardising, and dividing the data into training and testing ratios (Fig 2). After then, test data or custom



data will be provided on to the model for value prediction.



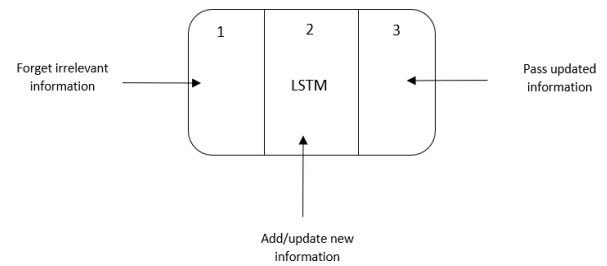
*Fig. 2. Algorithm Work Progress*

## 3.2 LSTM

### 3.2.1 LSTM INTRODUCTION

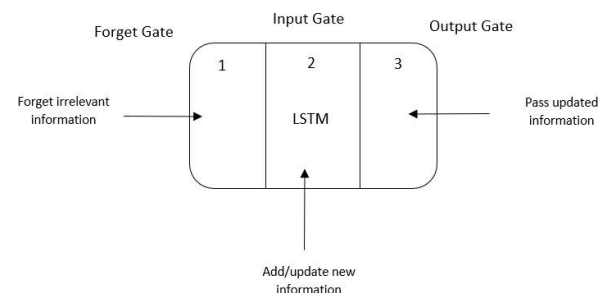
Information can persist thanks to an advanced RNN, sometimes referred to as a sequential network or extended short term memory network. It can overcome the vanishing gradient problem of RNNs. We shall presume you remember what happened in the last book or movie scene. Like how the other neural network function, they store the prior knowledge and use it to process the incoming input. Because of the diminishing gradient, RNNs are incapable of remembering long-term dependencies. When developing LSTMs, long-term dependability difficulties are carefully avoided.

### 3.2.2 LSTM ARCHITECTURE



*Fig. 3 LSTM Single Cell Architecture*

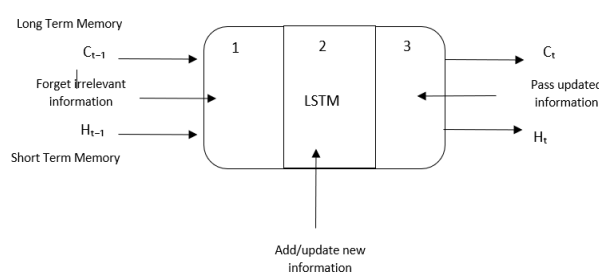
As we can see in the figure 3, the LSTM has three divisions, each performs a specific task. Prior timestamp information must be stored, or it can be forgotten, according to the first part. The second section's input is used by the cell to analyse and learn new information. In third segment, the cell sends the analysed data from the current component to the next component.



*Fig. 4. Gated LSTM*

The Output gate, Front gate, and Input gate are the three components as shown in the figure 4. There is a hidden state in LSTM, which is like conventional RNN, with  $H(t-1)$  which is

standing for the hidden state, preceding timestamp and  $H_t$  which is for the present timestamp. The cell state of LSTMs is also denoted by the timestamps  $C(t-1)$  and  $C(t)$ , which stand for the past and present timestamps, respectively. In this case, the cell state represents the long-term memory, but the concealed state represents the short-term memory. See the example below.



*Fig. 5 Memorization in LSTM*

### 3.3.1 DATA GATHERING:

Data is gathered from the Kaggle for around 20 types of crypto currencies, each data set is differed in size and each data set timeseries is different. Each data set size in KB's.

### 3.3.2 DATA CLEANSING:

When analysing data, we simply consider the linked Volume, Close, Unlock, higher prices, and market capitalisation. If the NaN quantities are

proven to be correct in any data collection, they are replaced with a description of the relevant attribute. According to the length of time, all data sets are then combined into one. The inputs that will be communicated to the network is latent if we examine all the cryptocurrency price movement over the course of the relevant time periods.

### 3.3.3 DATA NORMALIZATION:

Making a schedule modification, especially in the financial realm, is not easy. Furthermore, as the sixth guideline, the input will be loaded from the large amounts of diverse data through neural network. If you do this, the network will not alter since it will result in substantial gradient modifications. To facilitate network reading, the following features should be supplied in data:

- Use modest values. Typically, most values should fall between 0 and 1.
- Be homogenous, meaning that all characteristics should accept values that fall within a same range.

When Min-MaxScaling is used, the data inputs are scaled from 0 to 1: " $x$ " is defined as " $\frac{x - \min(X)}{\max(X) - \min(X)}$ "

### 3.3.4 DATA SPLITTING:

The total amount of data of each data set is divided into 70% and 30% ratio. 70% data is utilized for the training of data and the rest 30% of the data is used for the validation and testing of the data.

### 3.3.5 DATA TRAINING

The stacked LSTM model is an extension of the LSTM model that includes numerous buried LSTM layers with various numbers of memory unit in each layer. The model becomes deeper because of the layered LSTM hidden layers, more precisely classifying the procedure as deep learning. We generated the LSTM with Keras, which requires a range of parameters to be given, including units. To represent the size of space, we added 50 units.

Setting the return sequences parameter to true enables returning the final output in output. Finally, we use Adam Optimizer to create the model. The error is calculated using the mean squared error. The model is then fit using 100 epochs and 64 batches. The hyperparameter "epochs" determines how many instances the learning algorithm will cycle over the whole training dataset.

Each sample in the training dataset was subjected to one epoch in which the internal model parameters were altered.

## IV. OUTPUT

Rank	Name	Price	Market Cap	Supply	Volume	Change
1	Bitcoin BTC	\$22499.33	434.40B	21.00M	19.47B	9.80%
2	Ethereum ETH	\$1613.80	197.49B	NAN	NAN	9.90%
3	Tether USD	\$1.00	72.74B	NAN	NAN	-0.47%
4	BNB BNB	\$304.51	50.79B	166.80M	40.12M	6.74%
5	USD Coin USDC	\$0.99	40.70B	NAN	NAN	-4.48%
6	XRP XRP	\$0.37	16.72B	100.00B	623.02M	1.84%
7	Cardano ADA	\$0.34	11.80B	45.00B	199.73M	11.93%
8	Polygon MATIC	\$1.14	9.96B	10.00B	395.13M	6.49%
9	Dogecoin DOGE	\$0.07	9.38B	NAN	NAN	4.63%
10	Bitcoin USD B-USD	\$1.00	8.37B	NAN	NAN	-0.13%
11	Solana SOL	\$19.80	7.59B	NAN	NAN	10.02%
12	PulsaBot BOT	\$5.92	7.17B	NAN	NAN	7.95%

Fig.6. Live Price Prediction

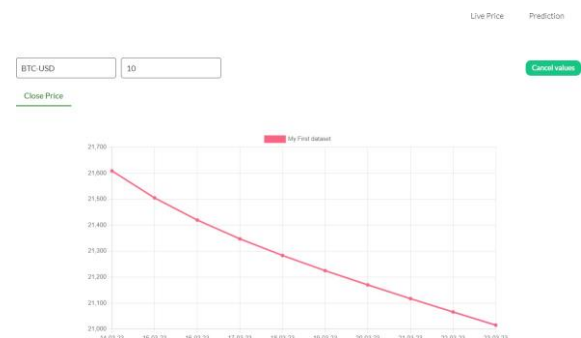


Fig. 7. Coin Price Values

## V. REFERENCES

1. "Predicting the Price of Bitcoin Using Machine Learning" by Kamil Kulesza et al. (2014)
2. "Bitcoin Price Forecasting with Neural Networks" by Martin Kuhlemann et al. (2014)

3. "Predicting Bitcoin's Price Value by Analyzing Google Search Trends Data Using Machine Learning Algorithms" by Jethin Abraham et al. (2014)
4. "Bitcoin Market Analysis: An Examination of OTC Market Participants" by Nicholas Colas et al. (2014)
5. "Bitcoin Market Prediction with Bayesian Neural Networks" by Joseph Wang et al. (2015)
6. "Cryptocurrency Price Prediction using Machine Learning Algorithms: A Survey" by Kaustubh Dhonde and Vishnu K (2018)
7. "Bitcoin Price Prediction Using Machine Learning: An Approach to Sample Dimension Engineering" by Daniel Trujillo et al. (2018)
8. "Cryptocurrency Price Prediction using News Articles: A Sentiment Analysis Approach" by Swati Aggarwal and Tanuja Jain (2019)
9. "Predicting Cryptocurrency Prices Using Deep Learning Techniques" by Daniel Fleder et al. (2019)