NETAJI SUBHAS INSTITUTE OF TECHNOLOGY, BIHTA, PATNA

INTERNSHIP REPORT

ON

P.I. DAPP : BLOCKCHAIN BASED DECENTRALISED APPLICATION FOR CROP DISEASE PREDICTION AND INSURANCE

SUBMITTED IN THE PARTIAL FULFILLMENT OF THE REQUIREMENT

OF

BACHELOR OF TECHNOLOGY (COMPUTER SCIENCE AND ENGINEERING)

BY

ARYABHATTA KNOWLEDGE UNIVERSITY, PATNA, BIHAR

SUBMITTED BY:

Swadha Kumari (181029)

Karan Kumar (181040)

Priti Kumari (181038)

Vth sem

SESSION 2018-22

An ISO 9001:2008 Certified Institution

Approved by AICTE, New Delhi

Affiliated to Aryabhatta Knowledge University, Patna, Bihar

NETAJI SUBHAS INSTITUTE OF TECHNOLOGY, BIHTA, PATNA

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SESSION- 2018-22

ACKNOWLEDGEMENT

The internship opportunity, we had with IIT Patna, was a great chance for learning and professional development. Therefore, we consider ourselves lucky and would like to express our deepest gratitude to Dr. Raju Halder, Assistant Professor, IIT Patna, who inspite of having a busy schedule, heartily guided and encouraged us throughout the internship.

We would also like to convey our deepest gratitude to Mr. Gopal Krishna and Mr. Subhash Chandra Pandit, Assistant Professor, NSIT Patna, for taking part in useful decisions and giving necessary advice and guidance and also for arranging all facilities to make the training experience easier.

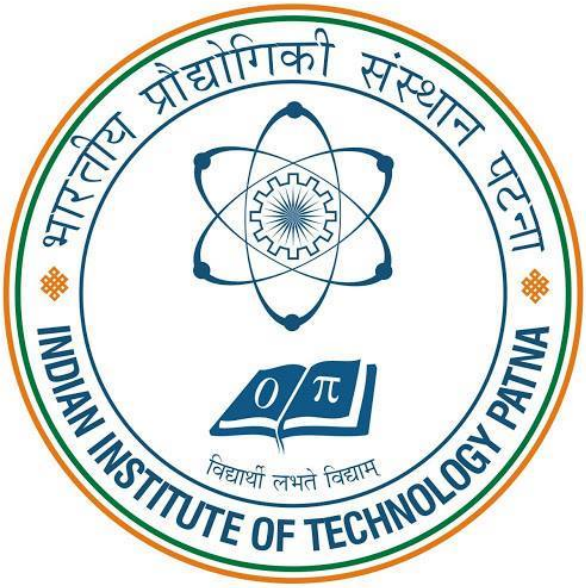
Last but not the least, we would like to thank our family and friends, for all that they meant to us during the crucial times of the completion of our internship.

Swadha Kumari (181029)

Karan Kumar (181040)

Priti Kumari (181038)

ABOUT THE ORGANISATION



Indian Institute of Technology Patna (abbreviated IIT Patna or IITP) is an autonomous institute of education and research in science, engineering and technology located in Patna, India. It is recognized as an Institute of National Importance by the Government of India. It is one of the new IITs established by an Act of the Indian Parliament on August 6, 2008.

The permanent campus of IIT Patna is located at Bihta which is approximately 30 km from Patna and has been fully operational since 2015.

CAMPUS AND LOCATION

IIT Patna's campus is located at Amhara, Bihta, 35 km from Patna at a 501 acres site. The foundation stone of the IIT Patna, Bihta Campus was laid by Kapil Sibal in 2011. IIT Patna started its new session (from July 2015) in its permanent campus located at Bihta. The campus at Bihta was inaugurated by Prime Minister Narendra Modi on 25 July 2015. Earlier, the institute was operating from a temporary 10 acres complex in Pataliputra colony, Patna, in buildings which have been renovated and were previously used by the Naveen Government Polytechnic.

The main building of the campus can be viewed directly from the campus main gate. It houses all the administrative offices, the Central Library, the Directorate, offices of all the Deans and other officials. The Department of Computer Science is evolving since the inception of IIT Patna in the year 2008 and owns some of the prestigious researchers of the country like Prof. Pushpak Bhattacharyya, Dr. Jimson Mathew, Dr. Arijit Mondal, and others. The major objective of the Department is to impart high quality education and research in India. Other Departments like Electrical Engineering and Mechanical Engineering also reside within the aegis of the department. The major research areas of the department include Communications, Signal Processing, VLSI, Electric Drives, Power Systems and Power Electronics. The Department has online access to IEL, Science Direct, Springer and other online journals. Software like MATLAB and

GAMS are available with the Department in order to accelerate the research. Instructional laboratories for Basic Electronics, Analog Electronics Digital Electronics, Digital Signal Processing, Embedded Systems, VLSI, Electrical Machines, Power Electronics and Power Systems are fully operational.All the academic lectures take place in the Tutorial Block. The five-storey building also temporarily houses the laboratories in Computer Science, Physics and Chemistry for the undergraduate students. It also has the Computer Centre of the institute with total 18 GBPS internet connections and adequate server infrastructure. Laboratories in Basic Electronics, Analog Electronics, Digital Electronics, VLSI, Control, Instrumentation and Communication are also located in this building. The Department of Chemical and Biochemical Engineering is a newly established department, that was instituted in 2015 with the efforts of Dr. Sujoy Kumar Samanta, faculty of IIT Patna. It is located in the Block-6 of the campus. The department accepted the first batch of undergraduate students in 2016 and presently has 9 faculty members. The Block-6 houses state-of-the-art research facilities for its young and dynamic faculty members in various research fields ranging from Clean fuel production, CO2 capture and storage, Pinch analysis, Microwave-assisted heating, Process modelling and simulation to Bio-resource engineering and Wastewater treatment. The department has also set up various laboratories for its undergraduate students including Heat transfer laboratory, Mechanical Operations laboratory, Chemical Reaction Engineering laboratory, Process simulation laboratory among many others. The science block has classrooms, faculty rooms and laboratories for Physics, Chemistry and Mathematics. These include the material science research and optics research facilities of the Physics Department, the chemistry instruments and chemistry research facilities of the Chemistry Department and the computational research facility of the Mathematics Department. The Department of Humanities and Social Sciences is located in Block 6. The department has also set up a Centre for Endangered Language Studies to work for the minor/ tribal/endangered languages of the state of Bihar and adjoining areas. In addition, the centre aims to collaborate with the institutes and universities of the neighbouring states to work for minor/ tribal/endangered languages along the state borders. The Mechanical Engineering Workshop building houses all the Mechanical Engineering Laboratories, viz. Basic and Conventional Manufacturing Lab, Advanced Manufacturing Lab, CAD/CAM Lab, Dynamics Lab, Fluid Mechanics Lab, Heat and Mass Transfer Lab,Instruments and Control Lab, IC Engines Lab, Material Testing Lab, Metrology and Metallographic Lab, Robotics Lab and Polymer Engineering Lab.

ACADEMICS

UNDERGRADUATE PROGRAMS

IIT Patna awards Bachelor of Technology degrees in the following disciplines:

- Bachelor of Technology in Computer Science and Engineering
- Bachelor of Technology in Electrical Engineering
- Bachelor of Technology in Mechanical Engineering
- Bachelor of Technology in Civil and Environmental Engineering
- Bachelor of Technology in Chemical Engineering
- Bachelor of Technology in Metallurgical and Materials Engineering

Admission to these programs for 225 seats is through the Joint Entrance Examination (JEE)-Advanced, taken by students seeking admission into the IITs after completing 10+2 years of schooling. The Curriculum consists of eight semesters spread over four academic years. A student takes five to six theory courses in addition to laboratory courses in each semester. A student's performance is evaluated based on a credit system. Credits are allotted to each course depending on the number of lecture/tutorial/laboratory hours per week.

POSTGRADUATE PROGRAMS

IIT Patna postgraduate programs include:

- Master of Technology
- Ph.D.

The M.tech program, started in 2012, awards degrees in following disciplines:

- Master of Technology in Mathematics and Computing
- Master of Technology in Nanotechnology
- Master of Technology in Mechatronics
- Master of Technology in Computer Science and Engineering
- Master of technology in Communication Systems Engineering
- Master of technology in Metallurgical and Materials Engineering
- Master of Technology in Civil and Infrastructure Engineering
- Master of Technology in VLSI and Embedded Systems
- Master of Technology in Mechanical engineering

IIT Patna is the first IIT in the whole IIT system to start a M.tech program in Nanotechnology. Admission to M.tech is through GATE after which an interview is held to screen the shortlisted candidates. Sponsored Candidates are not required to appear in GATE and are directly called for interview on applying for admission to M.Tech Program (if selected).

Started in 2009, Phd degrees are awarded by all the departments. Requirements for admission into the Ph.D. programs include a master's degree and prior academic achievement. Students undergo an interview before gaining admission.

RANKINGS

Internationally, IIT Patna was ranked 141 among BRICS nations by the QS World University Rankings of 2019.

IIT Patna was ranked 22 among Engineering Colleges in India by the National Institutional Ranking Framework (NIRF) in 2019 and 58 overall.

Contents

Abstract

The concept of Smart Agriculture is the key to keep moving the world's agriculture to a more productive and sustainable path. Recent advancements in Information and Communication Technology (ICT) have the potential to address some of the environmental, economic, and technical challenges as well as opportunities in this sector. This report is based on the application of machine learning fused with Blockchain technology to form a decentralized application. In this model, two streams are adopted: (i) Predicting the crop disease with deep convolutional neural network (ii) Providing a decentralized crop insurance system. The main objective of the smart agricultural system is to improve the protection of crops. The above goal is achieved by systematically monitoring the field by uploading the images of crop fields. The monitoring process involves collecting information and using the model's predict and insurance features.The use of smart contracts is made to keep the machine learning model and establish conditions for insurance, which in turn automate insurance payment.

1. INTRODUCTION

Plant pathogens can be fungal, bacterial, viral or nematodes and can damage plant parts above or below the ground. Identifying symptoms and knowing when and how to effectively control diseases is an ongoing challenge for growers of cereals , pulses crops. Bacteria are single celled prokaryotic (no membrane around nucleus) microorganisms that are either free-living in soil or water or diseases of plants. There are many types of viruses, viroid, prions and syndromes that have the potential to affect plant health. Viruses pose a serious risk for primary producers, as they can impact on market access and agricultural production. some of the world's major agricultural and livestock diseases.[2]

Good biosecurity measures on the property are vital for preventing the spread of plant diseases. Viruses can be spread by insect vectors. There are no pesticides that can be used to kill viruses, however they can be reduced and controlled by controlling these insect vectors with pesticides. Farmers know they lose crops to pests and plant diseases, but scientists have found that on a global scale, pathogens and pests are reducing crop yields for five major food crops by 10 percent to 40 percent, according to a report by a UC Agriculture and Natural Resources scientist and further members of the International Society for Plant Pathology. Wheat, rice, maize, soybean, and potato yields are reduced by pathogens and animal pests, including insects, scientists found in a global survey of crop health experts. Therefore, in the absence of assured and controlled water supply, the agricultural productivity in India is bound to be low. [2]

Observing all these problems and challenges in agriculture, scientists from different parts of the world have volunteered and contributed for the recent advancements that the agriculture sector has observed over time.

Field-ready serological tests such as lateral flow devices (LFDs) are commonly used as diagnostic tools to aid disease management decision making, to back up diagnoses based on symptoms, and as a triage tool to prescreen plants for specified target diseases. In addition to digital systems widely available for distance diagnosis, several free forms of software can be used to map and share the presence of diseases, these include apps like RustMapper, University of Florida's Digital Diagnostic and Identification System (DDIS) and Farmer's Friend aims to provide advice on treating common pests and diseases. The use of drones and remote sensing techniques coupled with spectroscopy-based methods - these techniques may be very useful as a rapid preliminary identification of primary infection. Biosensors based on phage display and biophotonics can also instantaneously detect infections.

These advance expert systems use concepts of building a greater intelligence quotient for the machines which develop decision making and predictive capabilities in the machines. Still, there are challenges that the agricultural sector is subdued to. The farmers are subjected under the heavy burdens of loans once their crop gets wasted. The insurance system for the crops need to be upgraded to work efficiently with the real time data of the fields.

Now, this is where. a collaboration of machine learning and blockchain could be put to use. The machine learning algorithm would detect and predict the diseases paying attention to all

the minute and essential details that a naked eye could miss easily. Blockchain, here, would be responsible for an insurance smart contract, being highly secure and transparent simultaneously.

The techniques like blockchain and machine learning complement each other really well. Machine learning gives the computer the ability to learn over time without being repeatedly programmed and without any human intervention. It primarily makes decisions without human help, thus making machines more autonomous. On the other hand, blockchain's primary function is making secured transactions between participants. It helps in eradicating middlemen who try making money off every transaction made.

The report primarily focuses on a decentralised application which provides a single window for all the services as per the requirements of a consumer. It provides an interface for the users to apply for an insurance and they could find the details of the diseases, that their crops are subjected to just by uploading their images, simultaneously.

The backend of the proposed model is highly efficient in segmenting those images and predicting the diseases due to its higher precision and decisive intelligence quotients. Moreover, the collaboration of Blockchain with the model contributes in making this application decentralised, thereby innovating the insurance processes to be hassle-free. It builds the trust among the users and maintains the integrity as well.

2. RELATED WORK

Agriculture, farming or husbandry is a vital occupation since the history of mankind is maintained. The name agriculture represents all entities that come under the linear sequence of links of food chain for human beings. As humans are the smartest living species on this planet, so their smartness always provokes them to change and to innovate. This provoking has led to invention of wheel, advancements in living standards and styles, languages, life spending methodologies and countless more achievements. The advancements in agriculture are necessary to balance the demand and supply as population is increasing day by day. As compared to the last fifty years and earlier, the demand of food has accelerated. To overcome the requirements, the deployment of modern technology over this vital source for humans is intolerable. With the use of modern and advanced technologies, efficiency of the agricultural industry can be improved, where it not only includes better productivity, but also lessens intra-field and inter-field losses present in conventional methods. From the beginning, agriculture is crucial part of human society due to the reality that man and agriculture are directly related to each other. This fact leads towards the advancement and enhancement of the typical, inappropriate and time consuming methodologies, used for agriculture. The fast moving world, new trends and technological advancement has changed the life style of people. Emerging new technologies are becoming an important part of routine. Smart homes and grid, smart cities smart campus, and smart farming are some of the whole advanced and upgraded, information and communication technologies that are helping humans to save time and get faster and accurate outcomes. [9]

2.1 TRENDS AND CONVENTIONS

Trends and research conventions are mainly focused on precise agriculture, database integration system and network information, virtual agriculture, expert systems, the connotation and extension, development stage and the impact of agricultural modernization for economic growth and improved life style of rural areas.Due to the direct impact on human life, agriculture is stepping towards modernization steadily. New trends are being introduced often, in order to meet the technological advancements. Comparing the past and continuing technological implementations it can be understood easily how most of the communication and information technologies are playing an effective role in modern agriculture. An increase in the demand for agriculture as a consequence of an abruptly growing population will enhance the need of efficient and actuate infrastructure support, in-order to fulfill the agricultural requirements of the modern society, without any interruption in its production.

Hybrid architecture for localized agricultural information dissemination is the client server architecture, in conjunction with the mobile applications on smart-phones, which can be used to deliver the precise agricultural information to the farmers. Geographical data of mobile phones can further localize the required information needs of the farmers. The cattle and farm manage-

ment, by using RFID tags and the recognition of cattle with the help of image processing, can lead to a decrease in the probability of viral spread.Considering these facts, the green houses are increasing in their popularity, every day.

According to recent trends and technological development in Wireless Sensor Networks, it has been made possible to use WSN in the monitoring and control of the greenhouse parameter, in precision agriculture. Actuated sensor networks are being deployed for the management of green houses. Using wireless sensor networks will reduce the chances of human errors that can occur while investigating the facts, about the ideal method of irrigation suitable for all weather conditions, types of soils and different crop cultures. The usage of advanced technologies and automated machines, which is making the world soar to greater heights, experiences a lag when it comes to the farming either due to the lack of awareness or because of the unavailability of advanced facilities in the market, leading towards poverty in farming.

In order to make the market more accessible to the farmers, the concept of e-farming is introduced. E-farming is the web application that will help the farmers to perform the agro-marketing leading to achieve success and increase in their standard of living. Smart agriculture is composed of many different technological implementations. These applications are replacing the tough, unreliable and time consuming traditional farming techniques with efficient, reliable and sustainable smart agriculture. Water irrigation context aware farming, pesticide control, remote monitoring, security control, environmental monitoring, precious agriculture, machine and process control, vehicle guidance, animal feeding facilities, trace-ability system, food packaging and inspection etc are a few examples.[9]

2.2 MAKING AGRICULTURE SMART

Generally speaking, if a machine/artifact or any system does something that we think an intelligent person can do, we consider the machine to be smart. Any system, process and domain is said to be smart if follows different levels of intelligence:

• Adapting: The term adapting refers to the change to meet any particular requirements in terms of smart agriculture the changes would be referred to as environmental

• Sensing: The ability to sense the changes in surrounding or to observe any change.

• Inferring:It basically refers to conclusion which is based on results and observations.

• Learning:After getting conclusions and observed results the learning can be used to improve the methodologies used previously. It involves different type of information.

• Anticipating:It relates to thinking of something new and innovative which going to be happened or we can say it as the next level of anything.

The processes referred to any intelligent system which has ability to sense and monitor and then change its parameters according to the need.Jotting all the initials, smart agriculture can be composed of these main paradigms;

- Smart Consumer
- Smart Farmer
- Smart farms

Smart consumer tends to the online access, for any end user to get information related to the productivity, particularly the consumer electronics in which one can be able to buy and sell the productions directly from the farm. This involves internet applications, web application, data base and online stores etc. The other end of smart consumer is the smart farmer side. It is the main node from where the farmer can directly interact with open market, without any extra expenses and involvement of third party. Any farm management system can be used to manage these outside activities. This node is then connected with the smart farm, which implies sensor nodes for humidity, moisture, weather, irrigation system, ware house management, cattle, pesticide detection and monitoring

In the present era, there are various and constantly evolving technologies available. Many of them are suitable, in various locations and scenarios. These technologies are fairly equal to use and fine in their output regardless of their deployment in either rural or urban areas. While we are discussing agriculture and relevant technologies, there are two major groups to discuss; the sensors available and the communication platforms.

The Sensors available for remote deployments are not for a single measurement or coordinate collections. They are especially designed for gathering a bunch of information from concerned entity. The main composite sensors are available for climate, soil and plants.[9]

According to the International Assessment of Agricultural Science and Technology for Development (IAASTD).The widespread realization is that despite significant scientific and technological achievements in our ability to increase agricultural productivity, we have been less attentive to some of the unintended social and environmental consequences of our achievements. We are now in a good position to reflect on these consequences and to outline various policy options to meet the challenges ahead, perhaps best characterized as the need for food and livelihood security under increasingly constrained environmental conditions from within and outside the realm of agriculture and globalized economic systems.The synthesis proposed some of the factors like equitable environment and sustainability which can be used to improve the rural livelihoods and can be useful in reducing hunger and poverty. In compliance with these factors, biological diversity and services for ecosystem rapid changes in climate and availability of water are some of the major concerns which are addressed. To fulfill the diverse needs of human life, there is a need for sustainability which requires the concern of the international collaboration.Apart from the technological advancement making its way towards smart agriculture, in order to benefit this industry which the humans directly associated with, along with its long lasting benefits also has some issues related to the agricultural advancement.

The main barrier in the development of agriculture is the human behavior towards adapting new technology. It has always been hard for a common man to adopt something different from the traditional method. Most people from the ruler areas are associated with the farming and are not much educated and independent in technological advancement. This factor has widened the gap between the modern and ruler areas. It is evident that the electronic media can be a great means of reducing the hesitation of adapting new technologies by commercials and on-air campaigns.

There are many issues directly associated with agriculture like grid, crop and soil monitoring, irrigation, pesticide and fertilization applications, and cattle farming. Information and communication technology can be a practical tool for overcoming these technical issues. Climate Changes The climatic change is biggest issue in agricultural paradigm, which directly affects each and every factor associated with farming. This natural conflict directly influences productivity and quality, leaving lasting and long-term impacts on food security. Quick solutions are needed for

this issue. Pre-weather detection, temperature monitoring, climate changes, moisture levels, air flow and pressure, rain and extreme weather prediction are few of the many solutions. [9]

3. PRELIMINARIES

3.1 BLOCKCHAIN

A blockchain is a constantly growing ledger which keeps a permanent record of all the transactions that have taken place in a secure, chronological, and immutable way. A blockchain is a chain of blocks which contain information. Each block records all of the recent transactions, and once completed goes into the blockchain as a permanent database. Each time a block gets completed, a new block is generated. Blockchain technology can be integrated into multiple areas. The primary use of blockchains is as a distributed ledger for cryptocurrencies. It shows great promise across a wide range of business applications like Banking, Finance, Government, Healthcare, Insurance, Media.[8]

It is a system of recording information in a way that makes it difficult or impossible to change, hack, or cheat the system. It is a system of recording information in a way that makes it difficult or impossible to change, hack, or cheat the system. Bitcoin was designed to solve this problem by using a specific type of database called a blockchain. Most normal databases, such as an SQL database, have someone in charge who can change the entries (e.g. giving themselves a million X dollars). Blockchain is different because nobody is in charge; it's run by the people who use it. What's more, bitcoins can't be faked, hacked or double spent – so people that own this money can trust that it has some value.

In the financial industry, blockchain can allow the quicker settlement of trades. It does not take a lengthy process for verification, settlement, and clearance. It is because of a single version of agreed-upon data available between all stakeholders. blockchain register transactions in a chronological order which certifies the inalterability of all operations, means when a new block is added to the chain of ledgers, it cannot be removed or modified. Blockchain certifies and verifies the identities of each interested parties. This removes double records, reducing rates and accelerates transactions.[8]

Blockchain uses very advanced cryptography to make sure that the information is locked inside the blockchain. It uses Distributed Ledger Technology where each party holds a copy of the original chain, so the system remains operative, even the large number of other nodes fall. It allows each party to transact directly with each other without requiring a third-party intermediary.

It is decentralized because there is no central authority supervising anything. There are standards rules on how every node exchanges the blockchain information. This method ensures that all transactions are validated, and all valid transactions are added one by one. Blockchain Cryptocurrency is a type of digital asset which is used to exchange value between parties. It uses strong cryptography to secure financial transactions and control the creation of new units of that currency and verify the transfer of assets.

Cryptocurrency does not exist physically. We know that the government prints the government

currencies like fiat currency such as Dollar, Yen or Yuan itself. It means there is a centralized institution exists which can create thousands or millions or billions more of that currency. Unlike government currencies like bitcoin, these type of currencies is created by the same mathematical formulas that make the cryptocurrency work. Thus, cryptocurrencies use decentralized control, which works through distributed ledger technology that serves as a public financial transaction database.[5]

Bitcoin is generally known as the first decentralized cryptocurrencies. It is created by the network of thousands of specific nodes called miners. The miners can process the bitcoin transactions in the blockchain network. Since the release of bitcoin, more than 4000 alternative variants of bitcoin are available now. In the blockchain technology, bitcoin is the best-known implementation of the blockchain. There is a lot of development and the direction is based on the premise of what blockchain does to enable Bitcoin to happen. We can learn and expand how it can spread into so many different areas.[5]

A blockchain distributed ledger is a type of database that is consensually shared, replicated, and synchronized among the members of a decentralized network. All the information on this ledger is securely and accurately stored using cryptography. This information can be accessed by using keys and cryptographic signatures. The distributed ledger allows transactions to have public witnesses, which makes cyberattack more difficult. It records the transactions such as the exchange of assets or data, among the participants in the network.[1]

Blockchain technology has enormous potential in creating trustless, decentralized applications. But it is not perfect. There are certain barriers which make the blockchain technology not the right choice and unusable for mainstream application. We can see the limitations of blockchain technology in the following image. There is a lot of discussion about blockchain, but people do not know the true value of blockchain and how they could implement it in different situations.

Today, there are a lot of developers available who can do a lot of different things in every field. But in the blockchain technology, there are not so many developers available who have specialized expertise in blockchain technology. Hence, the lack of developers is a hindrance to developing anything on the blockchain. In immutable, we cannot make any modifications to any of the records. It is very helpful if you want to keep the integrity of a record and make sure that nobody ever tampers with it. But immutability also has a drawback. We can understand this, in the case, when you want to make any revisions, or want to go back and make any reversals. For example, you have processed payment and need to go back and make an amendment to change that payment. As we know, blockchain is built on cryptography, which implies that there are different keys, such as public keys and private keys. When you are dealing with a private key, then you are also running the risk that somebody may lose access to your private key. It happens a lot in the early days when bitcoin wasn't worth that much. People would just collect a lot of bitcoin, and then suddenly forgot what the key was, and those may be worth millions of dollars today.[1]

Blockchain like bitcoin has consensus mechanisms which require every participating node to verify the transaction. It limits the number of transactions a blockchain network can process. So bitcoin was not developed to do the large scale volumes of transactions that many of the other institutions are doing. Currently, bitcoin can process a maximum of seven transactions per second. In the blockchain, we know that a block can be created in every 10 minutes. It is because every transaction made must ensure that every block in the blockchain network must reach a common consensus. Depending on the network size and the number of blocks or nodes involved

in a blockchain, the back-and-forth communications involved to attain a consensus can consume a considerable amount of time and resources. Proof of Work(PoW) is the original consensus algorithm in a blockchain network. The algorithm is used to confirm the transaction and creates a new block to the chain. In this algorithm, minors (a group of people) compete against each other to complete the transaction on the network. The process of competing against each other is called mining. As soon as miners successfully created a valid block, he gets rewarded. The most famous application of Proof of Work(PoW) is Bitcoin.

Producing proof of work can be a random process with low probability. In this, a lot of trial and error is required before a valid proof of work is generated. The main working principle of proof of work is a mathematical puzzle which can easily prove the solution. Proof of work can be implemented in a blockchain by the Hashcash proof of work system.[1]

By design, a blockchain is resistant to modification of the data. It is "an open, distributed ledger that can record transactions between two parties efficiently and in a verifiable and permanent way". For use as a distributed ledger, a blockchain is typically managed by a peer-to-peer network collectively adhering to a protocol for inter-node communication and validating new blocks. Once recorded, the data in any given block cannot be altered retroactively without alteration of all subsequent blocks, which requires consensus of the network majority. Although blockchain records are not unalterable, blockchains may be considered secure by design and exemplify a distributed computing system with high Byzantine fault tolerance. Decentralized consensus has therefore been claimed with a blockchain.

Blockchain was invented by a person (or group of people) using the name Satoshi Nakamoto in 2008 to serve as the public transaction ledger of the cryptocurrency bitcoin. The identity of Satoshi Nakamoto remains unknown to date. The invention of the blockchain for bitcoin made it the first digital currency to solve the double-spending problem without the need of a trusted authority or central server. The bitcoin design has inspired other applications, and blockchains that are readable by the public are widely used by cryptocurrencies. Blockchain is considered a type of payment rail. Private blockchains have been proposed for business use. Computerworld called the marketing of such blockchains without a proper security model "snake oil".

3.1.1 HISTORY

Cryptographer David Chaum first proposed a blockchain-like protocol in his 1982 dissertation "Computer Systems Established, Maintained, and Trusted by Mutually Suspicious Groups." Further work on a cryptographically secured chain of blocks was described in 1991 by Stuart Haber and W. Scott Stornetta. They wanted to implement a system where document timestamps could not be tampered with. In 1992, Haber, Stornetta, and Dave Bayer incorporated Merkle trees to the design, which improved its efficiency by allowing several document certificates to be collected into one block.

The first blockchain was conceptualized by a person (or group of people) known as Satoshi Nakamoto in 2008. Nakamoto improved the design in an important way using a Hashcash-like method to timestamp blocks without requiring them to be signed by a trusted party and introducing a difficulty parameter to stabilize rate with which blocks are added to the chain. The design was implemented the following year by Nakamoto as a core component of the cryptocurrency bitcoin, where it serves as the public ledger for all transactions on the network.

In August 2014, the bitcoin blockchain file size, containing records of all transactions that

have occurred on the network, reached 20 GB (gigabytes). In January 2015, the size had grown to almost 30 GB, and from January 2016 to January 2017, the bitcoin blockchain grew from 50 GB to 100 GB in size. The ledger size had exceeded 200 GiB by early 2020.

The words block and chain were used separately in Satoshi Nakamoto's original paper, but were eventually popularized as a single word, blockchain, by 2016.

According to Accenture, an application of the diffusion of innovations theory suggests that blockchains attained a 13.5 percent adoption rate within financial services in 2016, therefore reaching the early adopters phase. Industry trade groups joined to create the Global Blockchain Forum in 2016, an initiative of the Chamber of Digital Commerce.

In May 2018, Gartner found that only 1 percent of CIOs indicated any kind of blockchain adoption within their organisations, and only 8 percent of CIOs were in the short-term "planning or [looking at] active experimentation with blockchain". [8]

3.1.2 STRUCTURE

A blockchain is a decentralized, distributed, and oftentimes public, digital ledger consisting of records called blocks that is used to record transactions across many computers so that any involved block cannot be altered retroactively, without the alteration of all subsequent blocks. This allows the participants to verify and audit transactions independently and relatively inexpensively. A blockchain database is managed autonomously using a peer-to-peer network and a distributed timestamping server. They are authenticated by mass collaboration powered by collective self-interests. Such a design facilitates robust workflow where participants' uncertainty regarding data security is marginal. The use of a blockchain removes the characteristic of infinite reproducibility from a digital asset. It confirms that each unit of value was transferred only once, solving the long-standing problem of double spending. A blockchain has been described as a value-exchange protocol.A blockchain can maintain title rights because, when properly set up to detail the exchange agreement, it provides a record that compels offer and acceptance.

3.1.2.1 Blocks

Blocks hold batches of valid transactions that are hashed and encoded into a Merkle tree. Each block includes the cryptographic hash of the prior block in the blockchain, linking the two. The linked blocks form a chain. This iterative process confirms the integrity of the previous block, all the way back to the original genesis block.

Sometimes separate blocks can be produced concurrently, creating a temporary fork. In addition to a secure hash-based history, any blockchain has a specified algorithm for scoring different versions of the history so that one with a higher score can be selected over others. Blocks not selected for inclusion in the chain are called orphan blocks. Peers supporting the database have different versions of the history from time to time. They keep only the highest-scoring version of the database known to them. Whenever a peer receives a higher-scoring version (usually the old version with a single new block added) they extend or overwrite their own database and retransmit the improvement to their peers. There is never an absolute guarantee that any particular entry will remain in the best version of the history forever. Blockchains are typically built to add the score of new blocks onto old blocks and are given incentives to extend with new blocks rather than overwrite old blocks. Therefore, the probability of an entry becoming superseded

decreases exponentially as more blocks are built on top of it, eventually becoming very low. For example, bitcoin uses a proof-of-work system, where the chain with the most cumulative proof-of-work is considered the valid one by the network. There are a number of methods that can be used to demonstrate a sufficient level of computation. Within a blockchain the computation is carried out redundantly rather than in the traditional segregated and parallel manner. Block time

The block time is the average time it takes for the network to generate one extra block in the blockchain. Some blockchains create a new block as frequently as every five seconds. By the time of block completion, the included data becomes verifiable. In cryptocurrency, this is practically when the transaction takes place, so a shorter block time means faster transactions. The block time for Ethereum is set to between 14 and 15 seconds, while for bitcoin it is on average 10 minutes. Hard forks This section is an excerpt from Fork (blockchain) § Hard fork[edit]

A hard fork is a rule change such that the software validating according to the old rules will see the blocks produced according to the new rules as invalid. In case of a hard fork, all nodes meant to work in accordance with the new rules need to upgrade their software. If one group of nodes continues to use the old software while the other nodes use the new software, a permanent split can occur. For example, Ethereum has hard-forked to "make whole" the investors in The DAO, which had been hacked by exploiting a vulnerability in its code. In this case, the fork resulted in a split creating Ethereum and Ethereum Classic chains. In 2014 the Nxt community was asked to consider a hard fork that would have led to a rollback of the blockchain records to mitigate the effects of a theft of 50 million NXT from a major cryptocurrency exchange. The hard fork proposal was rejected, and some of the funds were recovered after negotiations and ransom payment. Alternatively, to prevent a permanent split, a majority of nodes using the new software may return to the old rules, as was the case of bitcoin split on 12 March 2013. [8]

3.1.2.2 Decentralisation

By storing data across its peer-to-peer network, the blockchain eliminates a number of risks that come with data being held centrally. The decentralized blockchain may use ad hoc message passing and distributed networking.

Peer-to-peer blockchain networks lack centralized points of vulnerability that computer crackers can exploit; likewise, it has no central point of failure. Blockchain security methods include the use of public-key cryptography.A public key (a long, random-looking string of numbers) is an address on the blockchain. Value tokens sent across the network are recorded as belonging to that address. A private key is like a password that gives its owner access to their digital assets or the means to otherwise interact with the various capabilities that blockchains now support. Data stored on the blockchain is generally considered incorruptible.

Every node in a decentralized system has a copy of the blockchain. Data quality is maintained by massive database replication and computational trust. No centralized "official" copy exists and no user is "trusted" more than any other. Transactions are broadcast to the network using software. Messages are delivered on a best-effort basis. Mining nodes validate transactions, add them to the block they are building, and then broadcast the completed block to other nodes. Blockchains use various time-stamping schemes, such as proof-of-work, to serialize changes. Alternative consensus methods include proof-of-stake. Growth of a decentralized blockchain is accompanied by the risk of centralization because the computer resources required to process larger amounts of data become more expensive. [5]

3.1.2.3 Openness

Open blockchains are more user-friendly than some traditional ownership records, which, while open to the public, still require physical access to view. Because all early blockchains were permissionless, controversy has arisen over the blockchain definition. An issue in this ongoing debate is whether a private system with verifiers tasked and authorized (permissioned) by a central authority should be considered a blockchain. Proponents of permissioned or private chains argue that the term "blockchain" may be applied to any data structure that batches data into time-stamped blocks. These blockchains serve as a distributed version of multiversion concurrency control (MVCC) in databases.Just as MVCC prevents two transactions from concurrently modifying a single object in a database, blockchains prevent two transactions from spending the same single output in a blockchain. Opponents say that permissioned systems resemble traditional corporate databases, not supporting decentralized data verification, and that such systems are not hardened against operator tampering and revision.Nikolai Hampton of Computerworld said that "many in-house blockchain solutions will be nothing more than cumbersome databases," and "without a clear security model, proprietary blockchains should be eyed with suspicion." Permissionless

The great advantage to an open, permissionless, or public, blockchain network is that guarding against bad actors is not required and no access control is needed. This means that applications can be added to the network without the approval or trust of others, using the blockchain as a transport layer.

Bitcoin and other cryptocurrencies currently secure their blockchain by requiring new entries to include a proof of work. To prolong the blockchain, bitcoin uses Hashcash puzzles. While Hashcash was designed in 1997 by Adam Back, the original idea was first proposed by Cynthia Dwork and Moni Naor and Eli Ponyatovski in their 1992 paper "Pricing via Processing or Combatting Junk Mail".

In 2016, venture capital investment for blockchain-related projects was weakening in the USA but increasing in China. Bitcoin and many other cryptocurrencies use open (public) blockchains. As of April 2018, bitcoin has the highest market capitalization. Permissioned (private) blockchain

Permissioned blockchains use an access control layer to govern who has access to the network.[40] In contrast to public blockchain networks, validators on private blockchain networks are vetted by the network owner. They do not rely on anonymous nodes to validate transactions nor do they benefit from the network effect.[citation needed] Permissioned blockchains can also go by the name of 'consortium' blockchains.[citation needed] Disadvantages of private blockchain

Nikolai Hampton pointed out in Computerworld that "There is also no need for a '51 percent' attack on a private blockchain, as the private blockchain (most likely) already controls 100 percent of all block creation resources. If you could attack or damage the blockchain creation tools on a private corporate server, you could effectively control 100 percent of their network and alter transactions however you wished." This has a set of particularly profound adverse implications during a financial crisis or debt crisis like the financial crisis of 2007–08, where politically powerful actors may make decisions that favor some groups at the expense of others, and "the bitcoin blockchain is protected by the massive group mining effort. It's unlikely that any private blockchain will try to protect records using gigawatts of computing power — it's time consuming and expensive." He also said, "Within a private blockchain there is also no 'race'; there's no

incentive to use more power or discover blocks faster than competitors. This means that many in-house blockchain solutions will be nothing more than cumbersome databases." Blockchain analysis

The analysis of public blockchains has become increasingly important with the popularity of bitcoin, Ethereum, litecoin and other cryptocurrencies. A blockchain, if it is public, provides anyone who wants access to observe and analyse the chain data, given one has the know-how. The process of understanding and accessing the flow of crypto has been an issue for many cryptocurrencies, crypto-exchanges and banks.The reason for this is accusations of blockchain enabled cryptocurrencies enabling illicit dark market trade of drugs, weapons, money laundering etc. A common belief has been that cryptocurrency is private and untraceable, thus leading many actors to use it for illegal purposes. This is changing and now specialised tech-companies provide blockchain tracking services, making crypto exchanges, law-enforcement and banks more aware of what is happening with crypto funds and fiat crypto exchanges. The development, some argue, has led criminals to prioritise use of new cryptos such as Monero.The question is about public accessibility of blockchain data and the personal privacy of the very same data. It is a key debate in cryptocurrency and ultimately in blockchain. [5]

3.1.3 USES

Blockchain technology can be integrated into multiple areas. The primary use of blockchains today is as a distributed ledger for cryptocurrencies, most notably bitcoin. There are a few operational products maturing from proof of concept by late 2016. Businesses have been thus far reluctant to place blockchain at the core of the business structure.

3.1.3.1 Cryptocurrencies

Most cryptocurrencies use blockchain technology to record transactions. For example, the bitcoin network and Ethereum network are both based on blockchain. On 8 May 2018 Facebook confirmed that it would open a new blockchain group which would be headed by David Marcus, who previously was in charge of Messenger. Facebook's planned cryptocurrency platform, Libra, was formally announced on June 18, 2019. [5]

3.1.3.2 Smart contracts

Blockchain-based smart contracts are proposed contracts that can be partially or fully executed or enforced without human interaction. One of the main objectives of a smart contract is automated escrow. An IMF staff discussion reported that smart contracts based on blockchain technology might reduce moral hazards and optimize the use of contracts in general. But "no viable smart contract systems have yet emerged." Due to the lack of widespread use their legal status is unclear. [1]

3.1.3.3 Financial services

Major portions of the financial industry are implementing distributed ledgers for use in banking, and according to a September 2016 IBM study, this is occurring faster than expected.

Banks are interested in this technology because it has potential to speed up back office settlement systems.

Banks such as UBS are opening new research labs dedicated to blockchain technology in order to explore how blockchain can be used in financial services to increase efficiency and reduce costs.

Berenberg, a German bank, believes that blockchain is an "overhyped technology" that has had a large number of "proofs of concept", but still has major challenges, and very few success stories.

In December 2018, Bitwala launched Europe's first regulated blockchain banking solution that enables users to manage both their bitcoin and euro deposits in one place with the safety and convenience of a German bank account. The bank account is hosted by the Berlin-based solarisBank.

The blockchain has also given rise to Initial coin offerings (ICOs) as well as a new category of digital asset called Security Token Offerings (STOs), also sometimes referred to as Digital Security Offerings (DSOs). STO/DSOs may be conducted privately or on a public, regulated stock exchange and are used to tokenize traditional assets such as company shares as well as more innovative ones like intellectual property, real estate, art, or individual products. A number of companies are active in this space providing services for compliant tokenization, private STOs, and public STOs.

3.1.3.4 Video games

A blockchain game CryptoKitties, launched in November 2017. The game made headlines in December 2017 when a cryptokitty character - an in-game virtual pet - was sold for more than US100,000 dollars. CryptoKitties illustrated scalability problems for games on Ethereum when it created significant congestion on the Ethereum network with about 30 percent of all Ethereum transactions being for the game.

CryptoKitties also demonstrated how blockchains can be used to catalog game assets (digital assets).

3.1.3.5 Energy trading

Blockchain is also being used in peer-to-peer energy trading.

3.1.3.6 Supply chain

There are a number of efforts and industry organizations working to employ blockchains in supply chain management.

Mining — Blockchain technology allows wholesalers, retailers, and customers to track the origins of gems stones and other precious commodities. In 2016, The Wall Street Journal reported that the blockchain technology company, Everledger was partnering with IBM's blockchain-based tracking service to trace the origin of diamonds to insure that they were ethically mined.

Food supply — Blockchain technology is being used to allow consumers to track the provenance of beef and other food products from their origins to stores and restaurants. Walmart and

IBM are running a trial to use a blockchain-backed system for supply chain monitoring for lettuce and spinach — all nodes of the blockchain are administered by Walmart and are located on the IBM cloud. One cited benefit is that the system will enable rapid tracing of contaminated produce. Fogo de Chao announced a partnership with HerdX that will allow that company's blockchain-based technology that will enable suppliers, wholesalers, and diners to trace the origins of the beef served in their restaurants.

Blockchain software development — The Linux Foundation's blockchain initiative, Hyperledger Grid develops open components for blockchain supply chain solutions. The goal of the project, said the foundation, was to "accelerate the development of blockchain-based solutions to cross-industry supply chain problems."

3.1.3.7 Domain Names

Blockchain domain names are another use of blockchain on the rise. Unlike regular domain names, blockchain domain names are entirely an asset of the domain owner and can only be controlled by the owner through a private key. Blockchain domains, pave way to having sites that are more resistant to censorship and thus enabling freedom of speech as there are no authorities or individuals that can intervene on controlling a domain except the private key holder. Again, they are a better option to replace the traditional cryptocurrency wallet addresses as one can easily memorize the domain and use it for receiving payments.

Organizations providing blockchain domain name services include Unstoppable Domains, Namecoin and Ethereum Name Services.

3.1.3.8 Other uses

Blockchain technology can be used to create a permanent, public, transparent ledger system for compiling data on sales, tracking digital use and payments to content creators, such as wireless users or musicians. In 2017, IBM partnered with ASCAP and PRS for Music to adopt blockchain technology in music distribution. Imogen Heap's Mycelia service has also been proposed as blockchain-based alternative "that gives artists more control over how their songs and associated data circulate among fans and other musicians."

New distribution methods are available for the insurance industry such as peer-to-peer insurance, parametric insurance and microinsurance following the adoption of blockchain.[91][92] The sharing economy and IoT are also set to benefit from blockchains because they involve many collaborating peers. Online voting is another application of the blockchain. The use of blockchain in libraries is being studied with a grant from the U.S. Institute of Museum and Library Services.

Other designs include:

Hyperledger is a cross-industry collaborative effort from the Linux Foundation to support blockchain-based distributed ledgers, with projects under this initiative including Hyperledger Burrow (by Monax) and Hyperledger Fabric (spearheaded by IBM). Quorum – a permissionable private blockchain by JPMorgan Chase with private storage, used for contract applications. Tezos, decentralized voting.[37]:94 Digital Asset Modeling Language (DAML) by Digital Asset Holdings is a smart contract language based on Glasgow Haskell Compiler. Proof of Existence is an online service that verifies the existence of computer files as of a specific time.

3.1.4 TYPES

Currently, there are at least four types of blockchain networks — public blockchains, private blockchains, consortium blockchains and hybrid blockchains.

3.1.4.1 Public blockchains

A public blockchain has absolutely no access restrictions. Anyone with an Internet connection can send transactions to it as well as become a validator (i.e., participate in the execution of a consensus protocol).[self-published source?] Usually, such networks offer economic incentives for those who secure them and utilize some type of a Proof of Stake or Proof of Work algorithm.

Some of the largest, most known public blockchains are the bitcoin blockchain and the Ethereum blockchain.

3.1.4.2 Private blockchains

A private blockchain is permissioned. One cannot join it unless invited by the network administrators. Participant and validator access is restricted. To distinguish between open blockchains and other peer-to-peer decentralized database applications that are not open ad-hoc compute clusters, the terminology Distributed Ledger (DLT) is normally used for private blockchains.

3.1.4.3 Hybrid blockchains

A hybrid blockchain has a combination of centralized and decentralized features. The exact workings of the chain can vary based on which portions of centralization decentralization are used.

3.1.4.4 Sidechains

A sidechain is a designation for a blockchain ledger that runs in parallel to a primary blockchain. Entries from the primary blockchain (where said entries typically represent digital assets) can be linked to and from the sidechain; this allows the sidechain to otherwise operate independently of the primary blockchain (e.g., by using an alternate means of record keeping, alternate consensus algorithm, etc.). [1]

3.1.5 ACADEMY RESEARCH

Blockchain panel discussion at the first IEEE Computer Society TechIgnite conference

In October 2014, the MIT Bitcoin Club, with funding from MIT alumni, provided undergraduate students at the Massachusetts Institute of Technology access to 100 dollars of bitcoin. The adoption rates, as studied by Catalini and Tucker (2016), revealed that when people who typically adopt technologies early are given delayed access, they tend to reject the technology.

3.1.5.1 Adoption decision

Motivations for adopting blockchain technology have been investigated by researchers. Janssen et al. provided a framework for analysis.[106] Koens Poll pointed out that adoption could be heavily driven by non-technical factors. Based on behavioral models, Li discussed the differences between adoption at individual level and at organization level.

3.1.5.2 Collaboration

Scholars in business and management have started studying the role of blockchains to support collaboration. It has been argued that blockchains can foster both cooperation (i.e., prevention of opportunistic behavior) and coordination (i.e., communication and information sharing). Thanks to reliability, transparency, traceability of records, and information immutability, blockchains facilitate collaboration in a way that differs both from the traditional use of contracts and from relational norms. Contrary to contracts, blockchains do not directly rely on the legal system to enforce agreements. In addition, contrary to the use of relational norms, blockchains do not require trust or direct connections between collaborators.

3.1.5.3 Blockchain and internal audit

The need for internal audit to provide effective oversight of organizational efficiency will require a change in the way that information is accessed in new formats. Blockchain adoption requires a framework to identify the risk of exposure associated with transactions using blockchain. The Institute of Internal Auditors has identified the need for internal auditors to address this transformational technology. New methods are required to develop audit plans that identify threats and risks. The Internal Audit Foundation study, Blockchain and Internal Audit, assesses these factors. The American Institute of Certified Public Accountants has outlined new roles for auditors as a result of blockchain.

3.1.5.4 Energy use of proof-of-work blockchains

The Bank for International Settlements has criticized the public proof-of-work blockchains for high energy consumption.Nicholas Weaver, of the International Computer Science Institute at the University of California, Berkeley examines blockchain's online security, and the energy efficiency of proof-of-work public blockchains, and in both cases finds it grossly inadequate. The 31—45 TWh of electricity used for bitcoin in 2018 produced 17—22.9 MtCO2.

3.1.5.5 Journals

In September 2015, the first peer-reviewed academic journal dedicated to cryptocurrency and blockchain technology research, Ledger, was announced. The inaugural issue was published in December 2016.The journal covers aspects of mathematics, computer science, engineering, law, economics and philosophy that relate to cryptocurrencies such as bitcoin.

The journal encourages authors to digitally sign a file hash of submitted papers, which are then timestamped into the bitcoin blockchain. Authors are also asked to include a personal bitcoin address in the first page of their papers for non-repudiation purposes. [1]

3.2 ETHEREUM

Ethereum is the second-largest cryptocurrency platform by market capitalization, behind bitcoin. It is a decentralized open source blockchain featuring smart contract functionality. Ether is the cryptocurrency generated by Ethereum miners as a reward for computations performed to secure the blockchain. Ethereum serves as the platform for over 260,000 different cryptocurrencies, including 47 of the top 100 cryptocurrencies by market capitalization.

Ethereum provides a decentralized virtual machines, the Ethereum Virtual Machine (EVM), which can execute scripts using an international network of public nodes./The virtual machine's instruction set, in contrast to others like bitcoin script, is turing -complete. "Gas", an internal transaction pricing mechanism, is used to mitigate spam and allocate resources on the network. [11]

Ethereum is open access to digital money and data-friendly services for everyone – no matter your background or location. It's a community-built technology behind the cryptocurrency Ether (ETH) and thousands of applications you can use today. Ethereum is a technology that lets you send cryptocurrency to anyone for a small fee. It also powers applications that everyone can use and no one can take down.

Ethereum is currently developing and planning to implement a series of upgrades called Ethereum 2.0. Current specifications for Ethereum 2.0 include a transition the proof of stake and an increase in transaction throughput using technology. Ethereum was initially described in a white paper by a programmer and co-founder in late 2013 with a goal of building decentralized applications. Buterin had argued that Bitcoin needed a scripted language for application development. Failing to gain agreement, he proposed the development of a new platform with a more general scripting language.

Ethereum was announced at the North American Bitcoin Conference in Miami, in January 2014. Several codenamed prototypes of the Ethereum platform were developed by the Ethereum Foundation, as part of their Proof-of-Concept series, prior to the official launch of the Frontier network. "Olympic" was the last of these prototypes, and public beta pre-release. The Olympic network provided users with a bug bounty of 25,000 Ether for stress testing the limits of the Ethereum blockchain. As with other cryptocurrencies, the validity of each Ether is provided by a blockchain, which is a continuously growing list of records, called blocks, which are linked and secured using cryptocurrency.[11]

By design, the blockchain is inherently resistant to modification of the data. It is an open, distributed ledger that records transactions between two parties efficiently and in a verifiable and permanent way. Unlike Bitcoin, Ethereum operates using accounts and balances in a manner called state transitions. This does not rely upon unspent transaction output (UTXOs). The state denotes the current balances of all accounts and extra data. The state is not stored on the blockchain, it is stored in a separate merkle Patricia tree. [11]

A cryptocurrency wallet stores the public and private keys or "addresses" which can be used to receive or spend ether. These can be generated through BIP 39 style mnemonics for a BIP 32 "HD Wallet". In Ethereum, this is unnecessary as it does not operate in a UTXO scheme. With the private key, it is possible to write in the blockchain, effectively making an Ether transaction.To send the Ethereum value token Ether to an account, you need the hash of the public key of that account. Ethereum accounts are pseudonymous in that they are not linked to individual persons, but rather to one or more specific addresses.

Ether is a fundamental token for operation of Ethereum, which thereby provides a public distributed ledger for transactions. It is used to pay for gas, a unit of computation used in transactions and other state transitions. Mistakenly, this currency is also referred to as Ethereum.It is listed under the ticker symbol ETH and traded on cryptocurrency exchanges, and the Greek uppercase Xi character () is generally used for its currency symbol. It is also used to pay for transaction fees and computational services on the Ethereum network. [4]

Ethereum's smart contracts are based on different computer languages, which developers use to program their own functionalities. Smart contracts are high-level programming abstractions that are compiled down to EVM bytecode and deployed to the Ethereum blockchain for execution. The Ethereum Virtual Machine (EVM) is the runtime environment for smart contracts in Ethereum. It is a 256-bit register stack, designed to run the same code exactly as intended. It is the fundamental consensus mechanism for Ethereum. The formal definition of the EVM is specified in the Ethereum Yellow Paper. In Ethereum all smart contracts are stored publicly on every node of the blockchain, which has costs.[4]

Being a blockchain means it is secure by design and is an example of a distributed computing system with high Byzantine fault tolerance. The downside is that performance issues arise in that every node is calculating all the smart contracts in real time, resulting in lower speeds.] As of January 2016, the Ethereum protocol could process about 25 transactions per second. In comparison, the Visa payment platform processes 45,000 payments per second leading some to question the scalability of Ethereum. Ethereum-based customized software and networks, independent from the public Ethereum chain, are being tested by enterprise software companies. [11]

Ethereum is the second-largest cryptocurrency platform by market capitalization, behind Bitcoin. It is a decentralized open source blockchain featuring smart contract functionality. Ether is the cryptocurrency generated by Ethereum miners as a reward for computations performed to secure the blockchain. Ethereum serves as the platform for over 260,000 different cryptocurrencies, including 47 of the top 100 cryptocurrencies by market capitalization.

Ethereum provides a decentralized virtual machine, the Ethereum Virtual Machine (EVM), which can execute scripts using an international network of public nodes. The virtual machine's instruction set, in contrast to others like Bitcoin Script, is Turing-complete. "Gas", an internal transaction pricing mechanism, is used to mitigate spam and allocate resources on the network.

Ethereum was proposed in late 2013 by Vitalik Buterin, a cryptocurrency researcher and programmer. Development was funded by an online crowdsale that took place between July and August 2014. The system then went live on 30 July 2015, with 72 million coins minted. This accounts for about 65 percent of the total circulating supply in April 2020.[non-primary source needed]

In 2016, as a result of an exploitation of a flaw in The DAO project's smart contract software, and subsequent theft of Dollar 50 million worth of ether, Ethereum was split into two separate blockchains. The new separate version became Ethereum (ETH) with the theft reversed, and the original chain continued as Ethereum Classic (ETC).

Ethereum is currently developing and planning to implement a series of upgrades called Ethereum 2.0. Current specifications for Ethereum 2.0 include a transition to proof of stake and an increase in transaction throughput using sharding technology.

3.2.1 HISTORY OF ETHEREUM

Ethereum was initially described in a white paper by Vitalik Buterin, a programmer and co-founder of Bitcoin Magazine, in late 2013 with a goal of building decentralized applications. Buterin had argued that Bitcoin needed a scripting language for application development. Failing to gain agreement, he proposed the development of a new platform with a more general scripting language.

Ethereum was announced at the North American Bitcoin Conference in Miami, in January 2014. During the same time as the conference, a group of people rented a house in Miami: Gavin Wood, Charles Hoskinson, and Anthony Di Iorio, a Torontonian who financed the project. Di Iorio invited friend Joseph Lubin, who invited reporter Morgen Peck, to bear witness. Six months later the founders met again in a house in Zug, Switzerland, where Buterin told the founders that the project would proceed as a non-profit. Hoskinson left the project at that time.

Ethereum has an unusually long list of founders. Anthony Di Iorio wrote "Ethereum was founded by Vitalik Buterin, Myself, Charles Hoskinson, Mihai Alisie, Amir Chetrit (the initial 5) in December 2013. Joseph Lubin, Gavin Wood, Jeffrey Wilke were added in early 2014 as founders." Formal development of the Ethereum software project began in early 2014 through a Swiss company, Ethereum Switzerland GmbH (EthSuisse). The basic idea of putting executable smart contracts in the blockchain needed to be specified before the software could be implemented; this work was done by Gavin Wood, then chief technology officer, in the Ethereum Yellow Paper that specified the Ethereum Virtual Machine. Subsequently, a Swiss non-profit foundation, the Ethereum Foundation (Stiftung Ethereum), was created as well. Development was funded by an online public crowdsale during July–August 2014, with the participants buying the Ethereum value token (ether) with another digital currency, Bitcoin.

While there was early praise for the technical innovations of Ethereum, questions were also raised about its security and scalability.

In 2019, an Ethereum foundation employee named Virgil Griffith was arrested by the US government for presenting at a blockchain conference in North Korea. [11]

3.2.2 ETHEREUM 2.0

Development is currently underway[39] for a major upgrade to the Ethereum platform, known as Ethereum 2.0.

The Ethereum 2.0 upgrade (also known as Serenity) is designed to be launched in three phases:

- "Phase 0" will create the Beacon Chain, a proof-of-stake blockchain.
- "Phase 1" will create shard chains and connect them to the Beacon Chain.
- "Phase 2" will implement state execution in the shard chains.

The current Ethereum 1.0 chain is expected to become one of the shards of Ethereum 2.0. Ethereum 2.0 has five main design goals:

- Minimize complexity by simplifying the Ethereum blockchain, even at the cost of efficiency.

- Improve up-time and keep the Ethereum network live during major network splits.
- Ensure longevity by building Ethereum 2.0 with elements which are either quantum secure or can be easily swapped out for quantum secure replacements when available.
- Increase security by using design techniques which allow a large number of validators to secure the network by staking their ETH holdings.
- Reduce barriers to entry, making it possible for a typical laptop to process or validate shards.

3.2.3 CHARACTERISTICS

As with other cryptocurrencies, the validity of each Ether is provided by a blockchain, which is a continuously growing list of records, called blocks, which are linked and secured using cryptography. By design, the blockchain is inherently resistant to modification of the data. It is an open, distributed ledger that records transactions between two parties efficiently and in a verifiable and permanent way.

Unlike Bitcoin, Ethereum operates using accounts and balances in a manner called state transitions. This does not rely upon unspent transaction outputs (UTXOs). The state denotes the current balances of all accounts and extra data. The state is not stored on the blockchain, it is stored in a separate Merkle Patricia tree. A cryptocurrency wallet stores the public and private "keys" or "addresses" which can be used to receive or spend ether. These can be generated through BIP 39 style mnemonics for a BIP 32 "HD Wallet". In Ethereum, this is unnecessary as it does not operate in a UTXO scheme. With the private key, it is possible to write in the blockchain, effectively making an Ether transaction.

To send the Ethereum value token Ether to an account, you need the Keccak-256 hash of the public key of that account. Ethereum accounts are pseudonymous in that they are not linked to individual persons, but rather to one or more specific addresses. [11]

3.2.4 FEATURES OF ETHEREUM

- Ether
- Addresses
- Ethereum Virtual Machine (EVM)
- Decentralized applications (Dapps)
- Decentralized autonomous organizations (DAOs)
- Smart Contracts [11]

3.2.4.1 Ether

Ether is a fundamental token for operation of Ethereum, which thereby provides a public distributed ledger for transactions. It is used to pay for gas, a unit of computation used in transactions and other state transitions. Mistakenly, this currency is also referred to as Ethereum.

It is listed under the ticker symbol ETH and traded on cryptocurrency exchanges, and the Greek uppercase Xi character () is generally used for its currency symbol. It is also used to pay for transaction fees and computational services on the Ethereum network.

Ether (ETH) is Ethereum's cryptocurrency. It is the fuel that runs the network. It is used to pay for the computational resources and the transaction fees for any transaction executed on the Ethereum network. Like Bitcoins, ether is a peer-to-peer currency. Apart from being used to pay for transactions, ether is also used to buy gas, which is used to pay for the computation of any transaction made on the Ethereum network.

Also, if you want to deploy a contract on Ethereum, you will need gas, and you would have to pay for that gas in ether. So gas is the execution fee paid by a user for running a transaction in Ethereum. Ether can be utilized for building decentralized applications, building smart contracts, and making regular peer-to-peer payments.

3.2.4.2 Addresses

Ethereum addresses are composed of the prefix "0x", a common identifier for hexadecimal, concatenated with the rightmost 20 bytes of the Keccak-256 hash (big endian) of the ECDSA public key (the curve used is the so-called secp256k1, the same as Bitcoin). In hexadecimal, 2 digits represent a byte, meaning addresses contain 40 hexadecimal digits.

An example of an Ethereum address is 0xb794f5ea0ba39494ce839613fffba74279579268. Contract addresses are in the same format, however, they are determined by sender and creation transaction nonce. User accounts are indistinguishable from contract accounts given only an address for each and no blockchain data. Any valid Keccak-256 hash put into the described format is valid, even if it does not correspond to an account with a private key or a contract. This is unlike Bitcoin, which uses base58check to ensure that addresses are properly typed.

3.2.4.3 Ethereum Virtual Machine

The Ethereum Virtual Machine (EVM) is the runtime environment for smart contracts in Ethereum. It is a 256-bit register stack, designed to run the same code exactly as intended. It is the fundamental consensus mechanism for Ethereum. The formal definition of the EVM is specified in the Ethereum Yellow Paper. Ethereum Virtual Machines have been implemented in C++, C, Go, Haskell, Java, JavaScript, Python, Ruby, Rust, Elixir, Erlang, and soon, WebAssembly (currently under development).

EVM is designed to operate as a runtime environment for compiling and deploying Ethereum-based smart contracts. EVM is the engine that understands the language of smart contracts, which are written in the Solidity language for Ethereum. EVM is operated in a sandbox environment—basically, you can deploy your stand-alone environment, which can act as a testing and development environment, and you can test your smart contract (use it) "n" number of times, verify it, and then once you are satisfied with the performance and the functionality of the smart contract, you can deploy it on the Ethereum main network.

Any programming language in the smart contract is compiled into the bytecode, which the EVM understands. This bytecode can be read and executed using the EVM. One of the most popular languages for writing a smart contract in Solidity. Once you write your smart contract in Solidity, that contract gets converted into the bytecode and gets deployed on the EVM. And thereby EVM guarantees security from cyberattacks.

WORKING OF EVM

Suppose person A wants to pay person B 10 ethers. The transaction will be sent to the EVM using a smart contract for a fund transfer from A to B. To validate the transaction; the Ethereum network will perform the proof-of-work consensus algorithm.

The miner nodes on Ethereum will validate this transaction—whether the identity of A exists or not, and if A has the requested amount to transfer. Once the transaction is confirmed, the ether will be debited from A's wallet and will be credited to B's wallet, and during this process, the miners will charge a fee to validate this transaction and will earn a reward.

All the nodes on the Ethereum network execute smart contracts using their respective EVMs.

PROOF OF WORK

Every node in the Ethereum network has:

The entire history of all the transactions—the entire chain The history of the smart contract, which is the address at which the smart contract is deployed, along with the transactions associated with the smart contract The handle to the current state of the smart contract The goal of the miners on the Ethereum network is to validate the blocks. For each block of a transaction, miners use their computational power and resources to get the appropriate hash value by varying the nonce. The miners will vary the nonce and pass it through a hashing algorithm—in Ethereum, it is the Ethash algorithm.

This produces a hash value that should be less than the predefined target as per the proof-of-work consensus. If the hash value generated is less than the target value, then the block is considered to be verified, and the miner gets rewarded.

When the proof of work is solved, the result is broadcast and shared with all the other nodes to update their ledger. If other nodes accept the hashed block as valid, then the block gets added to the Ethereum main blockchain, and as a result, the miner receives a reward, which as of today stands at three ethers. Plus the miner gets the transaction fees that have been generated for verifying the block. All the transactions that are aggregated in the block—the cumulative transaction fees associated with all the transactions are also given as a reward to the miner.

PROOF OF STAKE

In Ethereum, a process called proof of stake is also under development. It is an alternative to proof of work and is meant to be a solution to minimize the use of expensive resources spent on mining using proof of work. In proof of stake, the miner—who is the validator—can validate the transactions based on the number of crypto coins he or she holds before actually starting the mining. So based on the accumulation of crypto coins the miner has beforehand, he or she has a higher probability of mining the block. However, proof of stake is not widely used as of now compared to proof of work.

GAS

Just like we need fuel to run a car, we need gas to run applications on the Ethereum network. To perform any transaction within the Ethereum network, a user has to make a payment—shell out ethers—to get a transaction done, and the intermediary monetary value is called gas. On the

Ethereum network, gas is a unit that measures the computational power required to run a smart contract or a transaction. So if you have to do a transaction that updates the blockchain, you would have to shell outgas, and that gas costs ethers.

In Ethereum, the transaction fees are calculated using a formula. For every transaction, there is gas and its correlated gas price. The amount of gas required to execute a transaction multiplied by the gas price equals the transaction fees. "Gas limit" refers to the amount of gas used for the computation and the amount of ether a user is required to pay for the gas.

ETHEREUM VIRTUAL MACHINE GAS

To understand the gas limit and the gas price, let's consider an example using a car. Suppose your vehicle has a mileage of 10 kilometers per liter and the amount of petrol is 1 dollar per liter. Then driving a car for 50 kilometers would cost you five liters of petrol, which is worth 5 dollars. Similarly, to perform an operation or to run code on Ethereum, you need to obtain a certain amount of gas, like petrol, and the gas has a per-unit price, called gas price.

If the user provides less than the amount of gas to run a particular operation, then the process will fail, and the user will be given the message "out of gas." And Gwei, as noted above, is the lowest denomination of ether used for measuring a unit of a gas price.

ETHEREUM MINING VS. BITCOIN MINING

The hashing algorithm is the primary difference between Ethereum mining and Bitcoin mining.

Bitcoin uses SHA-256, and Ethereum uses Ethash. The average time taken on Bitcoin for mining a block is 10 minutes, whereas on Ethereum it is 12 to 15 seconds. As of today, the mining reward for Bitcoin is 12.5 bitcoins; for Ethereum it's three ethers plus the transaction fee—the cumulative transaction fees of all the transactions of a block. As of April 10, 2019, the value of 1 bitcoin is 5249.03 dollars, whereas one ether is 180.89 dollars.

3.2.4.4 Decentralized Applications (Dapps)

Let's compare decentralized applications with traditional applications. When you log in to Twitter, for example, a web application gets displayed that is rendered using HTML. The page will call an API to access your data (your information), which is centrally hosted. It's a simple process: your front end executes the backend API, and the API goes and fetches your data from a centralized database.

DAPP

If we transform this application into a decentralized application when you log in, the same web application gets rendered, but it calls a smart contract-based API to fetch the information from the blockchain network. So the API is replaced by a smart contract interface, and the smart contract will bring the data from the blockchain network, which is its backend.

That blockchain network is not a centralized database; it's a decentralized network in which the participants of the network (the miners) validate (verify) all the transactions that are happening using the smart contract on the blockchain network. So any transaction or action happening on a Twitter-type application that has now been transformed will be a decentralized transaction.

A Dapp consists of a backing code that runs on a distributed peer-to-peer network. It is a software designed to work in the Ethereum network without being controlled by a centralized system, as mentioned, and that is the primary difference: it provides direct interaction between the end-users and the decentralized application providers.

An application qualifies as a Dapp when it is open-source (its code is on Github), and it uses a public blockchain-based token to run its applications. A token acts as fuel for the decentralized application to run. Dapp allows the backend code and data to be decentralized, and that is the primary architecture of any Dapp.

3.2.4.5 Decentralized Autonomous Organizations (DAOs)

A DAO is a digital organization that operates without hierarchical management; it works in a decentralized and democratic fashion. So basically a DAO is an organization in which the decision-making is not in the hands of a centralized authority but preferably in the hands of certain designated authorities or a group or designated people as a part of an authority. It exists on a blockchain network, where it is governed by the protocols embedded in a smart contract, and thereby, DAOs rely on smart contracts for decision-making—or, we can say, decentralized voting systems—within the organization. So before any organizational decision can be made, it has to go through the voting system, which runs on a decentralized application.

Here's how it works. People add funds through the DAO because the DAO requires funding in order to execute and make decisions. Based on that, each member is given a token that represents that person's percentage of shares in the DAO. Those tokens are used to vote in the DAO, and the proposal status is decided based on the maximum votes. Every decision within the organization has to go through this voting process.

3.2.4.6 Smart Contracts

Ethereum's smart contracts are based on different computer languages, which developers use to program their own functionalities. Smart contracts are high-level programming abstractions that are compiled down to EVM bytecode and deployed to the Ethereum blockchain for execution. They can be written in Solidity (a language library with similarities to C and JavaScript), Serpent (similar to Python, but deprecated), LLL (a low-level Lisp-like language), and Mutan (Go-based, but deprecated). There is also a research-oriented language under development called Vyper (a strongly-typed Python-derived decidable language).

Smart contracts can be public, which opens up the possibility to prove functionality, e.g. self-contained provably fair casinos.

One issue related to using smart contracts on a public blockchain is that bugs, including security holes, are visible to all but cannot be fixed quickly. One example of this is the 17 June 2016 attack on The DAO, which could not be quickly stopped or reversed.

There is ongoing research on how to use formal verification to express and prove non-trivial properties. A Microsoft Research report noted that writing solid smart contracts can be extremely difficult in practice, using The DAO hack to illustrate this problem. The report discussed tools that Microsoft had developed for verifying contracts, and noted that a large-scale analysis of published contracts is likely to uncover widespread vulnerabilities. The report also stated that it is possible to verify the equivalence of a Solidity program and the EVM code.

A smart contract is a simple computer program that facilitates the exchange of any valuable asset between two parties. It could be money, shares, property, or any other digital asset that you want to exchange. Anyone on the Ethereum network can create these contracts. The contract consists primarily of the terms and conditions mutually agreed on between the parties (peers).

The primary feature of a smart contract is that once it is executed, it cannot be altered, and any transaction done on top of a smart contract is registered permanently—it is immutable. So even if you modify the smart contract in the future, the transactions correlated with the original contract will not get altered; you cannot edit them.

The verification process for the smart contracts is carried out by anonymous parties of the network without the need for a centralized authority, and that's what makes any smart contract execution on Ethereum a decentralized execution.

The transfer of any asset or currency is done in a transparent and trustworthy manner, and the identities of the two entities are secure on the Ethereum network. Once the transaction is successfully done, the accounts of the sender and receiver are updated accordingly, and in this way, it generates trust between the parties.

SMART CONTRACTS VS. TRADITIONAL CONTRACT SYSTEMS

In conventional contract systems, you sign an agreement, then you trust and hire a third party for its execution. The problem is that in this type of process, data tampering is possible. With smart contracts, the agreement is coded in a program. A centralized authority does not verify the result; it is confirmed by the participants on the Ethereum blockchain-based network. Once a contract is executed, the transaction is registered and cannot be altered or tampered, so it removes the risk of any data manipulation or alteration.

Let's take an example in which someone named Zack has given a contract of 500 dollars to someone named Elsa for developing his company's website. The developers code the agreement of the smart contract using Ethereum's programming language. The smart contract has all the conditions (requirements) for building the website. Once the code is written, it is uploaded and deployed on the Ethereum Virtual Machine (EVM).

EVM is a runtime compiler to execute a smart contract. Once the code is deployed on the EVM, every participant on the network has a copy of the contract. When Elsa submits the work on Ethereum for evaluation, each node on the Ethereum network will evaluate and confirm whether the result given by Elsa has been done as per the coding requirements, and once the result is approved and verified, the contract worth 500 dollars will be self-executed, and the payment will be paid to Elsa in ether. Zack's account will be automatically debited, and Elsa will be credited with 500 dollars in ether.

3.2.5 COMPARISON TO BITCOIN

Ethereum is different from Bitcoin (the cryptocurrency with the largest market capitalization as of August 2020) in several aspects:

- Its block time is 13 seconds, compared with 10 minutes for bitcoin.
- Mining of Ether generates new coins at a usually consistent rate, occasionally changing during hard forks, while for bitcoin the rate halves every 4 years.
- For proof-of-work, it uses the Ethash algorithm which reduces the advantage of specialized ASICs in mining.
- Transaction fees differ by computational complexity, bandwidth use and storage needs (in a system known as gas), while bitcoin transactions compete by means of transaction size, in bytes.

- Ethereum uses an accounting system where values in Wei (the smallest denomination of 1 Ether, 1 ETH = 1018 Wei) are debited from accounts and credited to another, as opposed to Bitcoin's UTXO system, which is more analogous to spending cash and receiving change in return. [11]

3.2.6 APPLICATIONS

Ethereum apps are written in one of seven different Turing-complete languages. Developers use the language to create and publish applications which they know will run inside Ethereum. The stablecoins Tether and DAI , and the prediction market Augur are examples of applications that run on Ethereum.

Many uses have been proposed for the Ethereum platform, including ones that are impossible or unfeasible. Use case proposals have included finance, the internet-of-things, farm-to-table produce, electricity sourcing and pricing, and sports betting. Ethereum is (as of 2017) the leading blockchain platform for initial coin offering projects, with over 50 percent market share.

3.2.6.1 Enterprise Software

Ethereum-based customized software and networks, independent from the public Ethereum chain, are being tested by enterprise software companies. Interested parties include Microsoft, IBM, JPMorgan Chase, Deloitte, R3, Innovate UK (cross-border payments prototype). Barclays, UBS and Credit Suisse are experimenting with Ethereum.

3.2.6.2 Permissioned Ledgers

Ethereum-based permissioned blockchain variants are used and being investigated for various projects.

- J. P. Morgan Chase is developing JPM Coin on a permissioned-variant of Ethereum blockchain dubbed "Quorum". It's designed to tow the line between private and public in the realm of shuffling derivatives and payments. The idea is to satisfy regulators who need seamless access to financial goings-on, while protecting the privacy of parties that don't wish to reveal their identities nor the details of their transactions to the general public.

- Royal Bank of Scotland has announced that it has built a Clearing and Settlement Mechanism (CSM) based on the Ethereum distributed ledger and smart contract platform.

3.2.6.3 Voting Systems

As we've seen with DAO, voting systems are adopting Ethereum. The results of polls are publicly available, ensuring a transparent and fair democratic process by eliminating voting malpractices.

3.2.6.4 Banking Systems

Ethereum is getting adopted widely in banking systems because with Ethereum's decentralized system; it is challenging for hackers to gain unauthorized access. It also allows payments on an Ethereum-based network, so banks are also using Ethereum as a channel to make remittances and payments.

3.2.6.5 Shipping

Deploying Ethereum in shipping helps with the tracking of cargo and prevents goods from being misplaced or counterfeited. Ethereum provides the provenance and tracking framework for any asset required in a typical supply chain.

3.2.6.6 Agreements

With Ethereum smart contracts, agreements can be maintained and executed without any alteration. So in an industry that has fragmented participants, is subject to disputes, and requires digital contracts to be present, Ethereum can be used as a technology for developing smart contracts and for digitally recording the agreements and the transactions based on them.

3.3 INTERPLANETARY FILE SYSTEM (IPFS)

IPFS (Interplanetary File System) is a peer to peer, version controlled, content-addressed file system. It makes use of Computer Science concepts like Distributed Hash Table, BitSwap (Inspired by BitTorrent), MerkleDag (Inspired by The Git Protocol). There are multiple applications currently being built on top of IPFS. The current default way to exchange data across the Internet is HTTP, but it fails in some cases. Large files cannot be transferred using HTTP, data is not permanent on HTTP, HTTP mainly uses a Client-Server protocol which leads to low latency and makes it difficult to establish a peer to peer connection, also real-time media streaming is difficult on HTTP. All of these failures are overcome using IPFS.

Unlike HTTP which is IP addressed, an IPFS network is content addressed. Which means, when any data is uploaded on an IPFS network, it returns a Hash and the data is then requested using that hash. Anyone can provide storage on the IPFS network and everyone is incentivized with crypto tokens. Data is distributed and replicated throughout the network which leads to data permanence. While requesting data it searches for the nearest copy of that data which leads to high latency and overcomes any bottleneck points.

As the data is completely distributed, it has no scope for centralization of data. IPFS could be seen as a single BitTorrent swarm, exchanging objects within one Git repository. The current default way to exchange data across the Internet is HTTP, but it fails in some cases. Large files cannot be transferred using HTTP, data is not permanent on HTTP, HTTP mainly uses a Client-Server protocol which leads to low latency and makes it difficult to establish a peer to peer connection, also real-time media streaming is difficult on HTTP. All of these failures are overcome using IPFS.

IPFS could be seen as a single BitTorrent swarm, exchanging objects within one Git repository. Distributed hash table is used to store and retrieve data across nodes in the network. It is

a class which is similar to hash tables. Using a DHT, any node on the network can request the value corresponding to a hash key.

Blockchain exchange is used in the BitTorrent Protocol (also known as BitSwap) to exchange data between nodes. It is a peer to peer file sharing protocol which coordinates data exchange between untrusted swarms .It uses a tit for tat strategy which rewards nodes that contribute to each other and punishes nodes that only request resources. This helps an IPFS node in retrieving multiple parts of data parallelly.

It uses Merkle DAG similar to the one used in the Git Version Control system. It is used to track change to files on the network in a distributed friendly way. Data is content-addressed, by the cryptographic hash of the content. Every node on the network is identified using a NodeID which is nothing but the hash of its public key. Everyone on the network can store files on their local. storage and they are incentivized to do so. Each node maintains a DHT which is used to find out Ids of other peers on the network and what data those peers can serve.

Users in a local network can communicate with each other, even if the Wide Area network is blocked for some reason. Since no servers are required, creators can distribute their work without any cost. Data loads faster as it has higher bandwidth. IPFS installation has a lot of hasseles, it is not at all user friendly. IPFS consumes lot of bandwidth which is not appreciated by metered internet users. IPFS currently is used by tech enthusiast and normal people don't tend to set up their own node, which leads to shortage of nodes on the network. IPFS synthesizes various best systems and protocols to date. IPFS is an ambitious vision of a new decentralized Internet infrastructure, upon which many different kinds of applications can be built in the future.

The list of problems go on and it is no surprise that a technology more than 20 years old is becoming more noticeably outdated in an age of technological innovation. IPFS provides the distributed storage and file system that the Internet needs to achieve its true potential. Instead of downloading files from single servers, in IPFS, you ask peers in the network to give you a path to a file rather than it coming from a central server. This enables high volume data distribution with high efficiency, historic versioning, resilient networks, and persistent availability of content secured and verified through cryptographic hashing and distributed across a network of peers. The design of the protocol provides historic versioning of the Internet like with Git. Each file and all blocks within it are given a unique identifier, which is a cryptographic hash. Duplicates are removed across the network and version history is tracked for every file.

IPFS links file structures to each other using Merkle links and every file can be found by human-readable names using a decentralized naming system called IPNS. Content has a unique identifier that is the cryptographic hash of the file. Content has a unique identifier that is the cryptographic hash of the file. Data is verified with its checksum, so if the hash changes, then IPFS will know the data is tampered with. Integration of IPFS with blockchain technology seems to be a perfect fit. [7]

IPFS allows users to not only receive but host content, in a similar manner to BitTorrent. As opposed to a centrally located server, IPFS is built around a decentralized system of user-operators who hold a portion of the overall data, creating a resilient system of file storage and sharing. Any user in the network can serve a file by its content address, and other peers in the network can find and request that content from any node who has it using a distributed hash table (DHT).

IPFS was launched in an alpha version in February 2015, and by October of the same year was described by TechCrunch as "quickly spreading by word of mouth." The logo of the online en-

cyclopedia Wikipedia has an IPFS hash with the following code: QmRW3V9znzFW9M5FYbit SEvd5dQrPWGvPvgQD6LM22Tv8D. It can be accessed with that hash over HTTP by a public gateway or a local IPFS instance.

3.3.1 DESIGN

IPFS allows users to not only receive but host content, in a similar manner to BitTorrent. As opposed to a centrally located server, IPFS is built around a decentralized system of user-operators who hold a portion of the overall data, creating a resilient system of file storage and sharing. Any user in the network can serve a file by its content address, and other peers in the network can find and request that content from any node who has it using a distributed hash table (DHT).

3.3.2 HISTORY

IPFS was launched in an alpha version in February 2015, and by October of the same year was described by TechCrunch as "quickly spreading by word of mouth."

The Catalan independence referendum, taking place in September–October 2017, was deemed illegal by the Constitutional Court of Spain and many related websites were blocked. Subsequently, the Catalan Pirate Party mirrored the website on IPFS to bypass the High Court of Justice of Catalonia order of blocking.

Phishing attacks have also been distributed through Cloudflare's IPFS gateway since July 2018. The phishing scam HTML is stored on IPFS, and displayed via Cloudflare's gateway. The connection shows as secure via a Cloudflare SSL certificate.

The IPStorm botnet, first detected in June 2019, uses IPFS, so it can hide its command-and-control amongst the flow of legitimate data on the IPFS network.Security researchers had worked out previously the theoretical possibility of using IPFS as a botnet command-and-control system. [7]

3.3.3 PROBLEMS WITH HTTP

Let's say you are sitting in a lecture hall, and the professor asks you to go to a specific website. Every student in the lecture makes a request to that website and are given a response. This means that the same exact data was sent individually to each student in the room. If there are 100 students, then that's 100 requests and 100 responses. This is obviously not the most efficient way to do things. Ideally, the students will be able to leverage their physical proximity to more efficiently retrieve the information they need.

HTTP also presents a big problem if there is some problem in the networks line of communication and the client is unable to connect with the server. This can happen if an ISP has an outage, a country is blocking some content, or if the content was simply deleted or moved. These types of broken links exist everywhere on the HTTP web.

The location-based addressing model of HTTP encourages centralization. It's convenient to trust a handful of applications with all our data but because of this much of the data on the web becomes siloed. This leaves those providers with enormous responsibility and power over our information.

3.3.4 WORKING OF IPFS

IPFS seeks to create a permanent and distributed web. It does this by using a content-addressed system instead of HTTP's location-based system.

An HTTP request would look like http://10.20.30.40/folder/file.txt

An IPFS request would look like /ipfs/QmT5NvUtoM5n/folder/file.txt

Instead of using an location address, IPFS uses a representation of the content itself to address the content. This is done using a cryptographic hash on a file and that is used as the address. The hash represents a root object and other objects can be found in its path. Instead of talking to a server, you gain access to this "starting point" of data. This way the system leverages physical proximity. If someone very close to me has what I want, I'll get it directly from them instead of connecting to a central server. In the lecture example from earlier, the students in the classroom can pull the data from each other without all having to establish their own communication with the a server. With HTTP you are asking what is at a certain location whereas with IPFS you are asking where a certain file is. In order to accomplish this, IPFS synthesizes a few successful ideas from other peer-to-peer systems.

To store data, IPFS uses a Distributed Hash Table, or DHT. Once we have a hash, we ask the peer network who has the content located at that hash and we download the content directly from the node that has the data I want. Data is transferred between the nodes in the network using mechanisms similar to BitTorrent.

A user looking for some content on the IPFS web finds neighbors who have access to that content. They then download small bits of the content from those neighbors. On top of the DHT and the BitTorrent protocols, IPFS uses a Merkle Tree. This is a data structure similar to the one Git uses as a version control system and the protocol used in the bitcoin blockchain. In Git, its used to track versions of source code, whereas in IPFS it's used to track content across the entire web. [7]

3.3.5 IPFS AND BLOCKCHAIN

Because of the similarity in their structure, IPFS and blockchains can work well together. In fact, Juan Benet, the inventor of IPFS calls this a "great marriage." IPFS is one of a few projects that are part of a group called Protocol Labs, which was also founded by Benet. Some projects from Protocol Labs closely related to IPFS are IPLD (Inter-Planetary Linked Data) and Filecoin. IPLD is a data model for distributed data structures like blockchains. This model allows for easy storage and access of blockchain data through IPFS. Users willing to store IPFS data will be rewarded with Filecoin. IPLD allows users to seamlessly interact with multiple blockchains and has been integrated with Ethereum and Bitcoin.

IPFS and other projects from Protocol Labs are ambitious by nature. The idea of a permanent web that is resilient and efficient were no doubt also the goals of the original inventors of our internet protocols. However, over time as our usage of the web changed, weaknesses in these protocols became evident. Although it is in its early stages, IPFS shows promise in being a crucial piece of a new decentralized technology stack.

3.3.6 OTHER NOTABLE USES

- IPFS was used to create a mirror of Wikipedia, which allows people living in jurisdictions where Wikipedia is blocked to access the content of Wikipedia. That archived version of Wikipedia is a limited immutable copy that cannot be updated.
- Filecoin, also inter-related to IPFS and developed by Juan Benet and Protocol Labs, is an IPFS-based cooperative storage cloud.
- Cloudflare runs a distributed web gateway to simplify, speed up, and secure access to IPFS without needing a local node.
- Microsoft's self-sovereign identity system, Microsoft ION, builds on the Bitcoin blockchain and IPFS through a Sidetree-based DID network.
- Brave uses Origin Protocol and IPFS to host its decentralized merchandise store.
- Opera for Android has default support for IPFS, allowing mobile users to browse IPFS:// links to access data on the IPFS network.

3.4 MACHINE LEARNING

Machine learning (ML) is the study of computer algorithms that improve automatically through experience.It is seen as a subset of artificial intelligence. Machine learning algorithms build a mathematical model based on sample data, known as "training data", in order to make predictions or decisions without being explicitly programmed to do so. Machine learning algorithms are used in a wide variety of applications, such as email filtering and computer vision, where it is difficult or infeasible to develop conventional algorithms to perform the needed tasks.

Machine learning is closely related to computational statistics, which focuses on making predictions using computers. The study of mathematical optimization delivers methods, theory and application domains to the field of machine learning. Data mining is a related field of study, focusing on exploratory data analysis through unsupervised learning.In its application across business problems, machine learning is also referred to as predictive analytics. [6]

3.4.1 OVERVIEW

Machine learning involves computers discovering how they can perform tasks without being explicitly programmed to do so. It involves computers learning from data provided so that they carry out certain tasks. For simple tasks assigned to computers, it is possible to program algorithms telling the machine how to execute all steps required to solve the problem at hand; on the computer's part, no learning is needed. For more advanced tasks, it can be challenging for a human to manually create the needed algorithms. In practice, it can turn out to be more effective to help the machine develop its own algorithm, rather than having human programmers specify every needed step.

The discipline of machine learning employs various approaches to teach computers to accomplish tasks where no fully satisfactory algorithm is available. In cases where vast numbers of

potential answers exist, one approach is to label some of the correct answers as valid. This can then be used as training data for the computer to improve the algorithm(s) it uses to determine correct answers. For example, to train a system for the task of digital character recognition, the MNIST dataset of handwritten digits has often been used.

3.4.1.1 Machine learning approaches

Machine learning approaches are traditionally divided into three broad categories, depending on the nature of the "signal" or "feedback" available to the learning system:

- Supervised learning: The computer is presented with example inputs and their desired outputs, given by a "teacher", and the goal is to learn a general rule that maps inputs to outputs.
- Unsupervised learning: No labels are given to the learning algorithm, leaving it on its own to find structure in its input. Unsupervised learning can be a goal in itself (discovering hidden patterns in data) or a means towards an end (feature learning).
- Reinforcement learning: A computer program interacts with a dynamic environment in which it must perform a certain goal (such as driving a vehicle or playing a game against an opponent). As it navigates its problem space, the program is provided feedback that's analogous to rewards, which it tries to maximize.

Other approaches have been developed which don't fit neatly into this three-fold categorisation, and sometimes more than one is used by the same machine learning system. For example topic modeling, dimensionality reduction or meta learning.

3.4.2 HISTORY AND RELATIONSHIP TO OTHER FIELDS

The term machine learning was coined in 1959 by Arthur Samuel, an American IBMer and pioneer in the field of computer gaming and artificial intelligence. A representative book of the machine learning research during the 1960s was the Nilsson's book on Learning Machines, dealing mostly with machine learning for pattern classification. Interest related to pattern recognition continued into the 1970s, as described by Duda and Hart in 1973. In 1981 a report was given on using teaching strategies so that a neural network learns to recognize 40 characters (26 letters, 10 digits, and 4 special symbols) from a computer terminal.

Tom M. Mitchell provided a widely quoted, more formal definition of the algorithms studied in the machine learning field: "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E. This definition of the tasks in which machine learning is concerned offers a fundamentally operational definition rather than defining the field in cognitive terms. This follows Alan Turing's proposal in his paper "Computing Machinery and Intelligence", in which the question "Can machines think?" is replaced with the question "Can machines do what we (as thinking entities) can do?". [6]

3.4.2.1 Artificial intelligence

As a scientific endeavor, machine learning grew out of the quest for artificial intelligence. In the early days of AI as an academic discipline, some researchers were interested in having machines learn from data. They attempted to approach the problem with various symbolic methods, as well as what were then termed "neural networks"; these were mostly perceptions and other models that were later found to be reinventions of the generalized linear models of statistics. Probabilistic reasoning was also employed, especially in automated medical diagnosis.

However, an increasing emphasis on the logical, knowledge-based approach caused a rift between AI and machine learning. Probabilistic systems were plagued by theoretical and practical problems of data acquisition and representation.By 1980, expert systems had come to dominate AI, and statistics was out of favor.Work on symbolic/knowledge-based learning did continue within AI, leading to inductive logic programming, but the more statistical line of research was now outside the field of AI proper, in pattern recognition and information retrieval. 755 Neural networks research had been abandoned by AI and computer science around the same time. This line, too, was continued outside the AI/CS field, as "connectionism", by researchers from other disciplines including Hopfield, Rumelhart and Hinton. Their main success came in the mid-1980s with the reinvention of backpropagation.

Machine learning, reorganized as a separate field, started to flourish in the 1990s. The field changed its goal from achieving artificial intelligence to tackling solvable problems of a practical nature. It shifted focus away from the symbolic approaches it had inherited from AI, and toward methods and models borrowed from statistics and probability theory. As of 2019, many sources continue to assert that machine learning remains a subfield of AI. Yet some practitioners, for example Dr Daniel Hulme, who teaches AI and runs a company operating in the field, argues that machine learning and AI are separate.

3.4.2.2 Data mining

Machine learning and data mining often employ the same methods and overlap significantly, but while machine learning focuses on prediction, based on known properties learned from the training data, data mining focuses on the discovery of (previously) unknown properties in the data (this is the analysis step of knowledge discovery in databases). Data mining uses many machine learning methods, but with different goals; on the other hand, machine learning also employs data mining methods as "unsupervised learning" or as a preprocessing step to improve learner accuracy. Much of the confusion between these two research communities (which do often have separate conferences and separate journals, ECML PKDD being a major exception) comes from the basic assumptions they work with: in machine learning, performance is usually evaluated with respect to the ability to reproduce known knowledge, while in knowledge discovery and data mining (KDD) the key task is the discovery of previously unknown knowledge. Evaluated with respect to known knowledge, an uninformed (unsupervised) method will easily be outperformed by other supervised methods, while in a typical KDD task, supervised methods cannot be used due to the unavailability of training data.

3.4.2.3 Optimization

Machine learning also has intimate ties to optimization: many learning problems are formulated as minimization of some loss function on a training set of examples. Loss functions express the discrepancy between the predictions of the model being trained and the actual problem instances (for example, in classification, one wants to assign a label to instances, and models are trained to correctly predict the pre-assigned labels of a set of examples). The difference between the two fields arises from the goal of generalization: while optimization algorithms can minimize the loss on a training set, machine learning is concerned with minimizing the loss on unseen samples.

3.4.2.4 Statistics

Machine learning and statistics are closely related fields in terms of methods, but distinct in their principal goal: statistics draws population inferences from a sample, while machine learning finds generalizable predictive patterns. According to Michael I. Jordan, the ideas of machine learning, from methodological principles to theoretical tools, have had a long pre-history in statistics. He also suggested the term data science as a placeholder to call the overall field.

Some statisticians have adopted methods from machine learning, leading to a combined field that they call statistical learning.

3.4.3 THEORY

Main articles: Computational learning theory and Statistical learning theory

A core objective of a learner is to generalize from its experience. Generalization in this context is the ability of a learning machine to perform accurately on new, unseen examples/tasks after having experienced a learning data set. The training examples come from some generally unknown probability distribution (considered representative of the space of occurrences) and the learner has to build a general model about this space that enables it to produce sufficiently accurate predictions in new cases.

The computational analysis of machine learning algorithms and their performance is a branch of theoretical computer science known as computational learning theory. Because training sets are finite and the future is uncertain, learning theory usually does not yield guarantees of the performance of algorithms. Instead, probabilistic bounds on the performance are quite common. The bias–variance decomposition is one way to quantify generalization error.

For the best performance in the context of generalization, the complexity of the hypothesis should match the complexity of the function underlying the data. If the hypothesis is less complex than the function, then the model has under fitted the data. If the complexity of the model is increased in response, then the training error decreases. But if the hypothesis is too complex, then the model is subject to overfitting and generalization will be poorer.

In addition to performance bounds, learning theorists study the time complexity and feasibility of learning. In computational learning theory, a computation is considered feasible if it can be done in polynomial time. There are two kinds of time complexity results. Positive results show that a certain class of functions can be learned in polynomial time. Negative results show that certain classes cannot be learned in polynomial time. [6]

3.4.4 APPROACHES

3.4.4.1 Types of learning algorithms

The types of machine learning algorithms differ in their approach, the type of data they input and output, and the type of task or problem that they are intended to solve.

SUPERVISED LEARNING: A support vector machine is a supervised learning model that divides the data into regions separated by a linear boundary. Here, the linear boundary divides the black circles from the white.

Supervised learning algorithms build a mathematical model of a set of data that contains both the inputs and the desired outputs. The data is known as training data, and consists of a set of training examples. Each training example has one or more inputs and the desired output, also known as a supervisory signal. In the mathematical model, each training example is represented by an array or vector, sometimes called a feature vector, and the training data is represented by a matrix. Through iterative optimization of an objective function, supervised learning algorithms learn a function that can be used to predict the output associated with new inputs. An optimal function will allow the algorithm to correctly determine the output for inputs that were not a part of the training data. An algorithm that improves the accuracy of its outputs or predictions over time is said to have learned to perform that task.

Types of supervised learning algorithms include Active learning, classification and regression. Classification algorithms are used when the outputs are restricted to a limited set of values, and regression algorithms are used when the outputs may have any numerical value within a range. As an example, for a classification algorithm that filters emails, the input would be an incoming email, and the output would be the name of the folder in which to file the email.

Similarity learning is an area of supervised machine learning closely related to regression and classification, but the goal is to learn from examples using a similarity function that measures how similar or related two objects are. It has applications in ranking, recommendation systems, visual identity tracking, face verification, and speaker verification.

UNSUPERVISED LEARNING:

Unsupervised learning algorithms take a set of data that contains only inputs, and find structure in the data, like grouping or clustering of data points. The algorithms, therefore, learn from test data that has not been labeled, classified or categorized. Instead of responding to feedback, unsupervised learning algorithms identify commonalities in the data and react based on the presence or absence of such commonalities in each new piece of data. A central application of unsupervised learning is in the field of density estimation in statistics, such as finding the probability density function. Though unsupervised learning encompasses other domains involving summarizing and explaining data features.

Cluster analysis is the assignment of a set of observations into subsets (called clusters) so that observations within the same cluster are similar according to one or more predesignated criteria, while observations drawn from different clusters are dissimilar. Different clustering techniques make different assumptions on the structure of the data, often defined by some similarity metric and evaluated, for example, by internal compactness, or the similarity between members of the same cluster, and separation, the difference between clusters. Other methods are based on estimated density and graph connectivity.

SEMI-SUPERVISED LEARNING:

Semi-supervised learning falls between unsupervised learning (without any labeled training data) and supervised learning (with completely labeled training data). Some of the training examples are missing training labels, yet many machine-learning researchers have found that unlabeled data, when used in conjunction with a small amount of labeled data, can produce a considerable improvement in learning accuracy.

In weakly supervised learning, the training labels are noisy, limited, or imprecise; however, these labels are often cheaper to obtain, resulting in larger effective training sets.

REINFORCEMENT LEARNING:

Reinforcement learning is an area of machine learning concerned with how software agents ought to take actions in an environment so as to maximize some notion of cumulative reward. Due to its generality, the field is studied in many other disciplines, such as game theory, control theory, operations research, information theory, simulation-based optimization, multi-agent systems, swarm intelligence, statistics and genetic algorithms. In machine learning, the environment is typically represented as a Markov Decision Process (MDP). Many reinforcement learning algorithms use dynamic programming techniques. Reinforcement learning algorithms do not assume knowledge of an exact mathematical model of the MDP, and are used when exact models are infeasible. Reinforcement learning algorithms are used in autonomous vehicles or in learning to play a game against a human opponent.

SELF LEARNING:

Self-learning as machine learning paradigm was introduced in 1982 along with a neural network capable of self-learning named Crossbar Adaptive Array (CAA).[34] It is a learning with no external rewards and no external teacher advices. The CAA self-learning algorithm computes, in a crossbar fashion, both decisions about actions and emotions (feelings) about consequence situations. The system is driven by the interaction between cognition and emotion.[35] The self-learning algorithm updates a memory matrix W =||w(a,s)|| such that in each iteration executes the following machine learning routine:

In situation s perform action a; Receive consequence situation s'; Compute emotion of being in consequence situation v(s'); Update crossbar memory w'(a,s) = w(a,s) + v(s').

It is a system with only one input, situation s, and only one output, action (or behavior) a. There is neither a separate reinforcement input nor an advice input from the environment. The backpropagated value (secondary reinforcement) is the emotion toward the consequence situation. The CAA exists in two environments, one is behavioral environment where it behaves, and the other is genetic environment, wherefrom it initially and only once receives initial emotions about situations to be encountered in the behavioral environment. After receiving the genome (species) vector from the genetic environment, the CAA learns a goal seeking behavior, in an environment that contains both desirable and undesirable situations.[36]

FEATURE LEARNING:

Several learning algorithms aim at discovering better representations of the inputs provided during training. Classic examples include principal components analysis and cluster analysis. Feature learning algorithms, also called representation learning algorithms, often attempt to preserve the information in their input but also transform it in a way that makes it useful, often as a pre-processing step before performing classification or predictions. This technique allows reconstruction of the inputs coming from the unknown data-generating distribution, while not being necessarily faithful to configurations that are implausible under that distribution. This replaces manual feature engineering, and allows a machine to both learn the features and use them

to perform a specific task.

Feature learning can be either supervised or unsupervised. In supervised feature learning, features are learned using labeled input data. Examples include artificial neural networks, multilayer perceptrons, and supervised dictionary learning. In unsupervised feature learning, features are learned with unlabeled input data. Examples include dictionary learning, independent component analysis, autoencoders, matrix factorization and various forms of clustering.

Manifold learning algorithms attempt to do so under the constraint that the learned representation is low-dimensional. Sparse coding algorithms attempt to do so under the constraint that the learned representation is sparse, meaning that the mathematical model has many zeros. Multilinear subspace learning algorithms aim to learn low-dimensional representations directly from tensor representations for multidimensional data, without reshaping them into higher-dimensional vectors. Deep learning algorithms discover multiple levels of representation, or a hierarchy of features, with higher-level, more abstract features defined in terms of (or generating) lower-level features. It has been argued that an intelligent machine is one that learns a representation that disentangles the underlying factors of variation that explain the observed data.

Feature learning is motivated by the fact that machine learning tasks such as classification often require input that is mathematically and computationally convenient to process. However, real-world data such as images, video, and sensory data has not yielded to attempts to algorithmically define specific features. An alternative is to discover such features or representations through examination, without relying on explicit algorithms. Sparse dictionary learning Main article: Sparse dictionary learning

Sparse dictionary learning is a feature learning method where a training example is represented as a linear combination of basis functions, and is assumed to be a sparse matrix. The method is strongly NP-hard and difficult to solve approximately. A popular heuristic method for sparse dictionary learning is the K-SVD algorithm. Sparse dictionary learning has been applied in several contexts. In classification, the problem is to determine the class to which a previously unseen training example belongs. For a dictionary where each class has already been built, a new training example is associated with the class that is best sparsely represented by the corresponding dictionary. Sparse dictionary learning has also been applied in image de-noising. The key idea is that a clean image patch can be sparsely represented by an image dictionary, but the noise cannot. Anomaly detection Main article: Anomaly detection

In data mining, anomaly detection, also known as outlier detection, is the identification of rare items, events or observations which raise suspicions by differing significantly from the majority of the data.[46] Typically, the anomalous items represent an issue such as bank fraud, a structural defect, medical problems or errors in a text. Anomalies are referred to as outliers, novelties, noise, deviations and exceptions.

In particular, in the context of abuse and network intrusion detection, the interesting objects are often not rare objects, but unexpected bursts in activity. This pattern does not adhere to the common statistical definition of an outlier as a rare object, and many outlier detection methods (in particular, unsupervised algorithms) will fail on such data, unless it has been aggregated appropriately. Instead, a cluster analysis algorithm may be able to detect the micro-clusters formed by these patterns.

Three broad categories of anomaly detection techniques exist.Unsupervised anomaly detection techniques detect anomalies in an unlabeled test data set under the assumption that the majority of the instances in the data set are normal, by looking for instances that seem to fit

least to the remainder of the data set. Supervised anomaly detection techniques require a data set that has been labeled as "normal" and "abnormal" and involves training a classifier (the key difference to many other statistical classification problems is the inherently unbalanced nature of outlier detection). Semi-supervised anomaly detection techniques construct a model representing normal behavior from a given normal training data set and then test the likelihood of a test instance to be generated by the model. Robot learning

In developmental robotics, robot learning algorithms generate their own sequences of learning experiences, also known as a curriculum, to cumulatively acquire new skills through self-guided exploration and social interaction with humans. These robots use guidance mechanisms such as active learning, maturation, motor synergies and imitation. Association rules Main article: Association rule learning See also: Inductive logic programming

Association rule learning is a rule-based machine learning method for discovering relationships between variables in large databases. It is intended to identify strong rules discovered in databases using some measure of "interestingness".

Rule-based machine learning is a general term for any machine learning method that identifies, learns, or evolves "rules" to store, manipulate or apply knowledge. The defining characteristic of a rule-based machine learning algorithm is the identification and utilization of a set of relational rules that collectively represent the knowledge captured by the system. This is in contrast to other machine learning algorithms that commonly identify a singular model that can be universally applied to any instance in order to make a prediction. Rule-based machine learning approaches include learning classifier systems, association rule learning, and artificial immune systems.

Based on the concept of strong rules, Rakesh Agrawal, Tomasz Imieliński and Arun Swami introduced association rules for discovering regularities between products in large-scale transaction data recorded by point-of-sale (POS) systems in supermarkets. Information can be used as the basis for decisions about marketing activities such as promotional pricing or product placements. In addition to market basket analysis, association rules are employed today in application areas including Web usage mining, intrusion detection, continuous production, and bioinformatics. In contrast with sequence mining, association rule learning typically does not consider the order of items either within a transaction or across transactions.

Learning classifier systems (LCS) are a family of rule-based machine learning algorithms that combine a discovery component, typically a genetic algorithm, with a learning component, performing either supervised learning, reinforcement learning, or unsupervised learning. They seek to identify a set of context-dependent rules that collectively store and apply knowledge in a piecewise manner in order to make predictions.

Inductive logic programming (ILP) is an approach to rule-learning using logic programming as a uniform representation for input examples, background knowledge, and hypotheses. Given an encoding of the known background knowledge and a set of examples represented as a logical database of facts, an ILP system will derive a hypothesized logic program that entails all positive and no negative examples. Inductive programming is a related field that considers any kind of programming languages for representing hypotheses (and not only logic programming), such as functional programs.

Inductive logic programming is particularly useful in bioinformatics and natural language processing. Gordon Plotkin and Ehud Shapiro laid the initial theoretical foundation for inductive machine learning in a logical setting. Shapiro built their first implementation (Model Inference

System) in 1981: a Prolog program that inductively inferred logic programs from positive and negative examples. The term inductive here refers to philosophical induction, suggesting a theory to explain observed facts, rather than mathematical induction, proving a property for all members of a well-ordered set.

3.4.4.2 Models

Performing machine learning involves creating a model, which is trained on some training data and then can process additional data to make predictions. Various types of models have been used and researched for machine learning systems. Artificial neural networks An artificial neural network is an interconnected group of nodes, akin to the vast network of neurons in a brain. Here, each circular node represents an artificial neuron and an arrow represents a connection from the output of one artificial neuron to the input of another.

Artificial neural networks (ANNs), or connectionist systems, are computing systems vaguely inspired by the biological neural networks that constitute animal brains. Such systems "learn" to perform tasks by considering examples, generally without being programmed with any task-specific rules.

An ANN is a model based on a collection of connected units or nodes called "artificial neurons", which loosely model the neurons in a biological brain. Each connection, like the synapses in a biological brain, can transmit information, a "signal", from one artificial neuron to another. An artificial neuron that receives a signal can process it and then signal additional artificial neurons connected to it. In common ANN implementations, the signal at a connection between artificial neurons is a real number, and the output of each artificial neuron is computed by some non-linear function of the sum of its inputs. The connections between artificial neurons are called "edges". Artificial neurons and edges typically have a weight that adjusts as learning proceeds. The weight increases or decreases the strength of the signal at a connection. Artificial neurons may have a threshold such that the signal is only sent if the aggregate signal crosses that threshold. Typically, artificial neurons are aggregated into layers. Different layers may perform different kinds of transformations on their inputs. Signals travel from the first layer (the input layer) to the last layer (the output layer), possibly after traversing the layers multiple times.

The original goal of the ANN approach was to solve problems in the same way that a human brain would. However, over time, attention moved to performing specific tasks, leading to deviations from biology. Artificial neural networks have been used on a variety of tasks, including computer vision, speech recognition, machine translation, social network filtering, playing board and video games and medical diagnosis.

Deep learning consists of multiple hidden layers in an artificial neural network. This approach tries to model the way the human brain processes light and sound into vision and hearing. Some successful applications of deep learning are computer vision and speech recognition.

DECISION TREES:

Decision tree learning uses a decision tree as a predictive model to go from observations about an item (represented in the branches) to conclusions about the item's target value (represented in the leaves). It is one of the predictive modeling approaches used in statistics, data mining and machine learning. Tree models where the target variable can take a discrete set of values are called classification trees; in these tree structures, leaves represent class labels and branches represent conjunctions of features that lead to those class labels. Decision trees where the tar-

get variable can take continuous values (typically real numbers) are called regression trees. In decision analysis, a decision tree can be used to visually and explicitly represent decisions and decision making. In data mining, a decision tree describes data, but the resulting classification tree can be an input for decision making. Support vector machines Main article: Support vector machines

Support vector machines (SVMs), also known as support vector networks, are a set of related supervised learning methods used for classification and regression. Given a set of training examples, each marked as belonging to one of two categories, an SVM training algorithm builds a model that predicts whether a new example falls into one category or the other. An SVM training algorithm is a non-probabilistic, binary, linear classifier, although methods such as Platt scaling exist to use SVM in a probabilistic classification setting. In addition to performing linear classification, SVMs can efficiently perform a non-linear classification using what is called the kernel trick, implicitly mapping their inputs into high-dimensional feature spaces. Illustration of linear regression on a data set. Regression analysis Main article: Regression analysis

Regression analysis encompasses a large variety of statistical methods to estimate the relationship between input variables and their associated features. Its most common form is linear regression, where a single line is drawn to best fit the given data according to a mathematical criterion such as ordinary least squares. The latter is often extended by regularization (mathematics) methods to mitigate overfitting and bias, as in ridge regression. When dealing with non-linear problems, go-to models include polynomial regression (for example, used for trendline fitting in Microsoft Excel), Logistic regression (often used in statistical classification) or even kernel regression, which introduces non-linearity by taking advantage of the kernel trick to implicitly map input variables to higher dimensional space. Bayesian networks Main article: Bayesian network A simple Bayesian network. Rain influences whether the sprinkler is activated, and both rain and the sprinkler influence whether the grass is wet.

A Bayesian network, belief network or directed acyclic graphical model is a probabilistic graphical model that represents a set of random variables and their conditional independence with a directed acyclic graph (DAG). For example, a Bayesian network could represent the probabilistic relationships between diseases and symptoms. Given symptoms, the network can be used to compute the probabilities of the presence of various diseases. Efficient algorithms exist that perform inference and learning. Bayesian networks that model sequences of variables, like speech signals or protein sequences, are called dynamic Bayesian networks. Generalizations of Bayesian networks that can represent and solve decision problems under uncertainty are called influence diagrams. Genetic algorithms Main article: Genetic algorithm

A genetic algorithm (GA) is a search algorithm and heuristic technique that mimics the process of natural selection, using methods such as mutation and crossover to generate new genotypes in the hope of finding good solutions to a given problem. In machine learning, genetic algorithms were used in the 1980s and 1990s. Conversely, machine learning techniques have been used to improve the performance of genetic and evolutionary algorithms.

3.4.4.3 Training models

Usually, machine learning models require a lot of data in order for them to perform well. Usually, when training a machine learning model, one needs to collect a large, representative sample of data from a training set. Data from the training set can be as varied as a corpus of text,

a collection of images, and data collected from individual users of a service. Overfitting is something to watch out for when training a machine learning model. Federated learning

Federated learning is an adapted form of Distributed Artificial Intelligence to training machine learning models that decentralizes the training process, allowing for users' privacy to be maintained by not needing to send their data to a centralized server. This also increases efficiency by decentralizing the training process to many devices. For example, Gboard uses federated machine learning to train search query prediction models on users' mobile phones without having to send individual searches back to Google. [6]

4. P.I. Dapp : OUR PROPOSED APPROACH

The system herein proposes a decentralised application via which a farmer could easily apply for an insurance of his crops, based on the crop type, namely, the rabi and kharif crops, the area of farm and the location of the place. Here, the farmer will have to submit some of the images of his crops. A machine learning model at the backend would go through these images and if the crop are not spoilt from before, the insurance would be approved and the farmer would be allotted a unique policy identification number for that specific crop, and he would have to pay a certain premium amount, which specifically depends on the crop.

Now, at a later stage, if the farmer finds that his crops have got affected with a certain disease and he does not know the severity of the disease and up to what extent this disease would cost him, he could simply approach this application. Here, he will just have to upload the images of the affected portions and within some seconds, the application would tell him about the disease that the crop is subjected to. Apart from this, the farmer would be offered a coverage amount based on the severity of the disease, like if the crops could no longer be cured, then the full coverage amount would be offered, however, if they could be cured, then a certain percentage of the coverage amount would be given to the farmers.

Moreover, if there are farmers who have not taken any kind of insurance for their crops, and their crops are on the verge of getting wasted, they could also approach this application to get a detailed information of the disease, as this system is trained with the machine learning models which have a high accuracy and are highly precisive. This is highly beneficial in situations when the farmers could not study their crops with their naked eye observation and miss out the small signs, however, this system could study out the smallest details on the leaves and would detect even the diseases that are at a premature stage.

The system herein is designed to be a decentralised application, as here there is no intervention of third-party mediators but the information of the policy and the crops of the farmers remains on the blockchain ledger which could not be tampered with. This hereby builds the trust among the farmers. The backend code here runs on the decentralised peer-to-peer networks, as opposed to the typical applications where the backend code is running on centralised servers. This proves to be hassle free in comparison to the traditional methods and the age-old conventions of getting an insurance policy and getting them approved in cases of wastage. This has incorporated a single window system for the farmers, by providing all the services via a single application just through some clicks.

4.1 SYSTEM DESIGN

4.1.1 MODEL FLOW DIAGRAM

The basic flow of our decentralized model is in a way were we have two smart contracts, first one is "mlmodel.sol" and the second one is "insurance.sol". The "mlmodel.sol" interact with our machine learning model and every image we pass that can predict the disease in the leaf and the information generated will be stored in a variable named predicted and "insurance.sol" deals with all the forms of insurance and through this smart contract we take new insurance for any crop and we can also claim insurance of any existing crop. We have also an "app.js" and this is JavaScript file and its deals with IPFS because for this we need an image and storage capacity of blockchain is not sufficient for store an image and it is also very costly or expensive so we have used one of the alternatives is IPFS and in IPFS when we upload an image to IPFS it automatically gets stored in IPFS server and generated hash and the generated hash can easily be passed to machine learning to predict the disease.

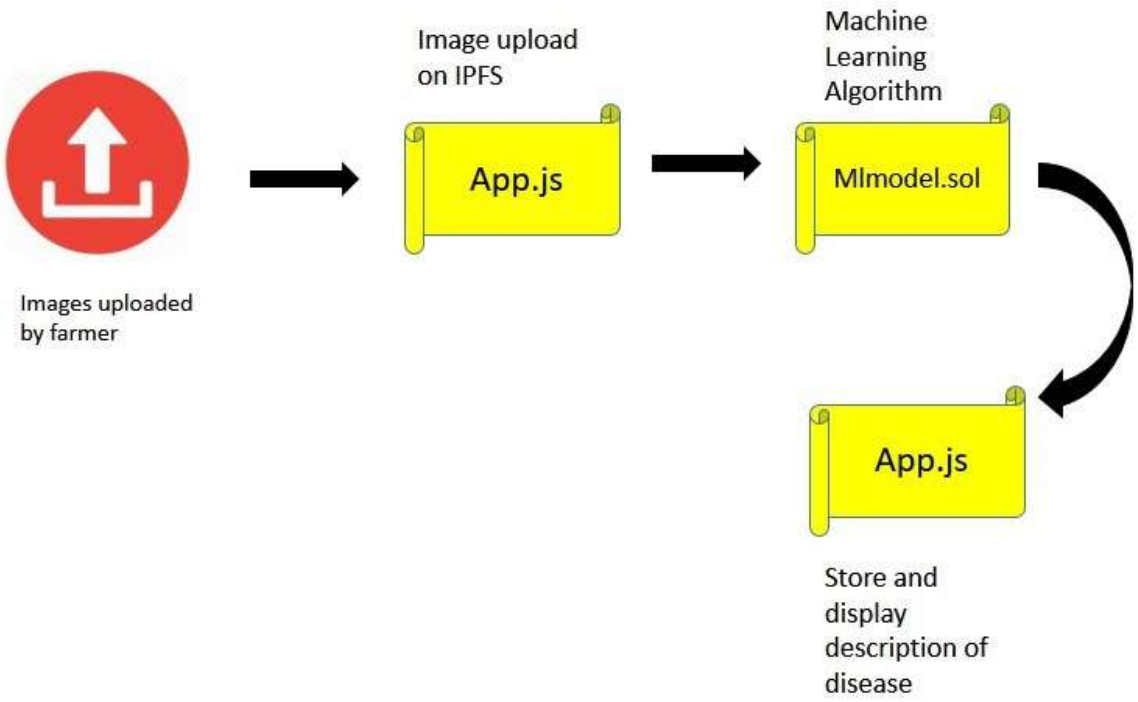


Figure 4.1: Flow to check disease

In this flow diagram firstly we have to upload the images. The uploaded images will further go to app.js which is JavaScript file and after that, it will get uploaded to IPFS and hash will be generated. The generated hash will go to the mlmodel.sol and the generated information will go back to the JavaScript file and then disease information will be display.

In this flow diagram, we have two options first one is to buy the insurance, and the second one

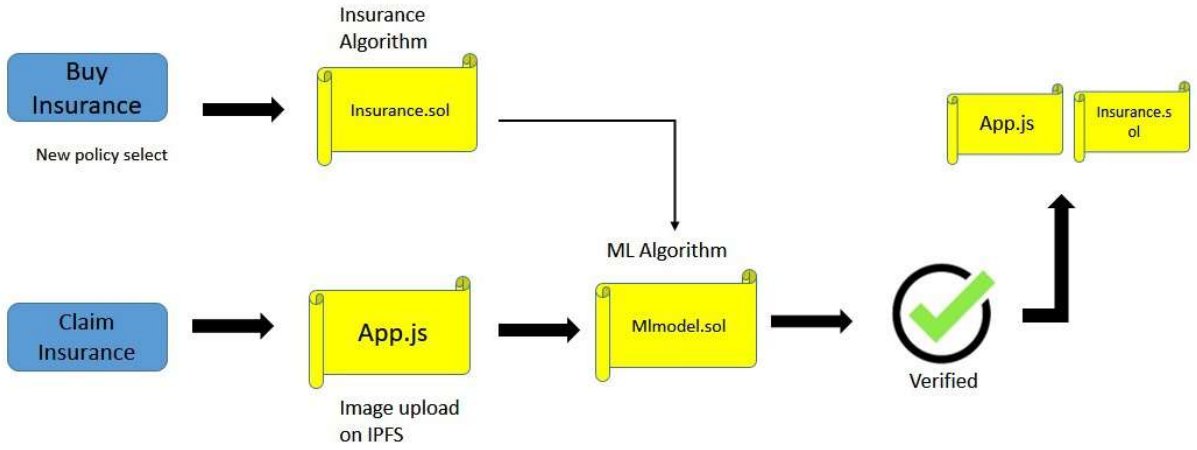


Figure 4.2: Flow to claim Insurance

is claim insurance. In buy insurance we select a new policy first through insurance algorithm we purchase new insurance and after that, we will go-to machine learning algorithm to see whether the disease is being predicted or not and after it we will also see whether there is already a disease or not and if there is no disease then we will not give insurance. In claim insurance, under this, we upload images through IPFS to app.js after this machine learning algorithm will verify whether there is a disease present or not. After verification app.js will display information regarding verification and insurance.sol will display information regarding weather insurance is claimed or not.

4.1.2 MODEL STATE DIAGRAM

The state flow diagram of the model has three primary files that influence the workflow of the model. It includes two smart contracts, namely "mlmodel.sol" and "insurance.sol" and a javascript file saved as "app.js". When someone uploads the images, through the "app.js" file it'll be stored on the IPFS and a cryptographic hash is generated.

The hash is then passed to the "mlmodel.sol" file, there the hash will be used as a parameter for the predict function and then it'll return a result with the callback function. Then the result will proceed to the claim function inside "insurance.sol" file. When the condition of the claim function is fulfilled it'll call the makerequest function, which will determine the claim amount.

Inside the "insurance.sol" file there's a newPolicy function that'll ask the user's details stored

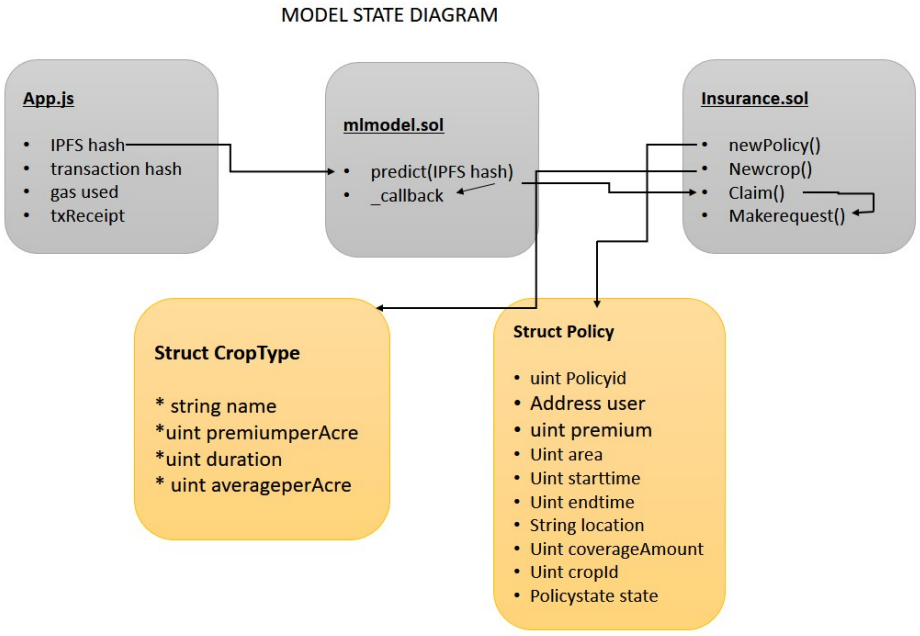


Figure 4.3: Model State Diagram

in the Policy structure. The details include Policy id, Adress, Premium Amount, Coverage Area, Start and End time, Crop id, Policy state, etc. The end time is calculated based on the crop type (Rabi and Kharif). Moreover, the user can add a new crop with the newcrop function built in the "insurance.sol" It has structure named as cropType which will store the Crop name, Premium per acre, Duration, etc.

4.2 DECENTRALISED CROP INSURANCE

In the modern age, the Blockchain revolution has popularised the decentralization of every application. Farmers frequently face the problems of crop damage because of several plant diseases. Many insurance companies are covers these factors, but the farmers that are insured face challenges to get their coverage outlay from the insurance companies.

Furthermore, the insurance corporations always act as a middleman in all this process. To remove the middleman and immediately settle insurance claims by the user, this decentralized insurance system is made. The DAPP has various options to buy the insurance, Users can opt for their policy by specifying the type of crop, start time, area and all other necessary details.

The end time is automatically calculated depending on the type of crop (i.e. Rabi or Kharif). The users pay the required premium fee to the smart contract using the p2p network system. After buying, the users will get a unique Policy ID which needs to be preserved. In case of crop damage, the users can use the claim option and enter their Policy ID. In order to calculate the

coverage amount, some parameters have been defined in the contract depending on the type of disease.

4.3 MACHINE LEARNING MODEL

In order to get the prediction of the crop diseases from their images, there was a need to have a machine learning model at the backend. Now, this model was built up in python language with the help of a dataset obtained from the kaggle website.

The objectives of the dataset were to train a model using images of training dataset to 1) Accurately classify a given image from testing dataset into different diseased category or a healthy leaf; 2) Accurately distinguish between many diseases, sometimes more than one on a single leaf; 3) Deal with rare classes and novel symptoms; 4) Address depth perception—angle, light, shade, physiological age of the leaf; and 5) Incorporate expert knowledge in identification, annotation, quantification, and guiding computer vision to search for relevant features during learning.

This dataset contained approximately 3642 leaf images and their labels were under four classes that is leaves that are healthy, leaves with multiple diseases, leaves with rust and leaves with scabs.

After the loading of this dataset, the images had to be preprocessed to obtain all the important features of the leaves. Scaling, normalisation of pixels, edge detection and splitting and merging of respective colour channels was done. After which, the processed images over here were appended into a different array.

Now this array was splitted into training, validation and testing datasets. Thereafter, a sequential model, consisting of densenet121, global average pooling2d and a dense layer of 4 neurons, was defined. Here, in the densenet121 layer approximately 29089792 pre trained imagenet weights were used in order to get a high accuracy.

The model summary was as follows:

LAYER (TYPE)	OUTPUT SHAPE	PARAM
densenet121 (Functional)	None, 6, 6, 1024	7037504
global average pooling2d	None, 1024	0
dense (Dense)	None, 4	4100
Total params: 7,041,604		
Trainable params: 6,957,956		
Non-trainable params: 83,648		

The model obtained was compiled using rmsprop optimizer and categorical crossentropy loss. Now, the training dataset was then made to pass via the sequential model for about 200 epochs, obtaining a training categorical accuracy of 1 at the end. Then, the prediction was done and after that a classification report was obtained, wherein, the model gave the f1-score, by the weighted average of all the four classes, of 0.98. Thereby, proving to give an accuracy of about 98 percent in the predictions of the foliar disease.

At later stages, this model was turned into an API and then oraclized into the smart contract to connect it further to the proposed decentralised application.

4.4 TURNING MACHINE LEARNING MODEL TO AN API

Web APIs have made it easy for cross-language applications to work well. If a frontend developer needs to use your ML Model to create an ML-powered web application, they would just need to get the URL Endpoint from where the API is being served. In simple words, an API is a (hypothetical) contract between 2 software saying if the user software provides input in a pre-defined format, the later with extending its functionality and provide the outcome to the user software.

Essentially, APIs are very much like web applications, but instead of giving the styled HTML page, APIs tend to return data in a standard data-exchange format such as JSON, XML, etc.

Once the developer has the desired output they can style it whatever the way they want. There are many popular ML APIs available out there as well. IBM Watson's ML API which is capable of the following:

• Machine Translation - Helps translate text in different language pairs.

• Message Resonance – To find out the popularity of a phrase or word with a predetermined audience.

• Question and Answers - This service provides direct answers to the queries that are triggered by primary document sources.

• User Modelling – To make predictions about the social characteristics of someone from a given text.

4.4.1 USING FLASK

Web service is a form of API only that assumes that an API is hosted over a server and can be consumed. Flask is a web service development framework in Python. It is not the only one in Python, there a couple of others as well such as Django, Falcon, Hug, etc. Flask is very minimal.

Flask framework comes with an inbuilt light-weighted web server that needs minimal configuration, and it can be controlled from the Python code. Flask will load the already persisted model into memory when the application starts. Moreover, it'll create an API endpoint that takes input variables, transform them into the appropriate format, and returns predictions.

Flask is used with our Densenet machine learning model to create an API for further use in smart contracts. It'll be used as a core state to utilize our model.

4.5 ORACLIZING THE API

The smart contracts built are like a closed box, the code inside it cannot communicate with the external world on their own. However, it is really important that the smart contracts communicate with the machine learning API, in order to predict the diseases via the user interface for the images uploaded by the farmers. One option for this communication is by using services such as

Oraclize. It acts as a data carrier and a reliable connection between APIs and the decentralised application.

To use Oraclize on testrpc and truffle, ethereum-bridge needs to be installed and this is a very essential requirement for the service. Thereafter, the oracliseAPI is imported from the github link in the smart contract wherein further, a contract is initialized and an event is generated with a description as parameter.

Then a function is declared for the prediction which is of payable type and here the access specifier is defined public. Now here, the generated event and a predefined function oraclize query is called with a parameter "URL" as the interaction made here is by the URL of the ML model API. The URL of the API is specified as the second parameter of the function under the json command, owing to the fact that the API is written in the json format at the backend mapped with the keys and values.

When the function predict communicates with the API, it gets a unique queryId, the result of the prediction and a proof of the authentication. After this, a callback function is called by itself, which contains the parameters like queryId, prediction result and the proof. This function also checks whether the call back address is similar to that of the message sender or not, once it gets authenticated, the predicted result is sent to the insurance smart contract.

The insurance smart contract further checks the different classes of the predicted result and accordingly decides the coverage amount that the farmer would get. Simultaneously, the information of the predicted disease also gets displayed on the user interface to let them know about the disease that their crop is subjected to or may suffer with.

4.6 IPFS INTERACTION WITH SMART CONTRACT

Now, as the user uploads an image on the decentralised application, the image needs to get stored at some place within the application, and as this is a decentralised application there is no server for storage and the data gets stored on the blockchain ledger. However, this is a time consuming task, therefore, there was a need for the interplanetary file system(IPFS) where the data is distributed over the peer-to-peer network and does not take much time for uploading or retrieval of data.

When dealing with IPFS storage, one can directly install IPFS on their systems and thereafter can initialise the daemon and could use the website to upload their file and from there only, it could be accessed as well.

However, to make the application a single place for all the services, it needs to be integrated with the user interface and for this one needs to unbox react firstly at the directory of workspace and thereafter the IPFS-api needs to be installed and after that the IPFS-infura.io is introduced as the host for the data uploading purposes. Now, another javascript file is created to import the web3 and to fetch the abi of the solidity files to interact with the front end.

Hence, whenever the user uploads an image on the application, a capture file event is generated and here the image file is read as the array buffer. The console of the user interface obtains the web3 account and metamask transaction account information. The file is then uploaded on the IPFS, which further generates a unique hash value, namely the IPFS hash, for the image file. This hash is sent to the user interface which gets displayed on the screen for the user's convenience to further check on the IPFS website itself for their image. In addition to the IPFS hash, the

ethereum contract address and the transaction hash also get displayed for future correspondence and reference.

5. PROOF OF CONCEPT

5.1 DEPENDENCIES

5.1.1 TRUFFLE

Truffle is a framework for building, testing, and deploying applications on the Ethereum network that was founded by Tim Coulter. The Truffle Framework consists of three primary development frameworks for Ethereum smart contract and decentralized application (dApp) development called Truffle, Ganache, and Drizzle.

npm install -g truffle

5.1.1.1 Requirements

- NodeJS v8.9.4 or later
- Windows, Linux or Mac OS X

Truffle also requires that one have a running Ethereum client which supports the standard JSON RPC API (which is nearly all of them). There are many to choose from, and some better than others for development. We'll discuss them in detail in the Choosing an Ethereum client section

To use most Truffle commands, one need to run them against an existing Truffle project. So the first step is to create a Truffle project.

One can use the 'truffle unbox ' command to download any of the Truffle Boxes. For example: truffle unbox metacoin

Once this operation is completed, one will have a project structure with the following items:

- contracts/: Directory for Solidity contracts
- migrations/: Directory for scriptable deployment files
- test/: Directory for test files for testing your application and contracts
- truffle-config.js: Truffle configuration file

All of the contracts are located in your project's contracts/ directory. As contracts are written in Solidity, all files containing contracts will have a file extension of .sol. Associated Solidity libraries will also have a .sol extension. With a bare Truffle project (created through truffle init), you're given a single Migrations.sol file that helps in the deployment process. If you're using a

Truffle Box, you will have multiple files here. To compile a Truffle project, change to the root of the directory where the project is located and Upon first run, all contracts will be compiled. Upon subsequent runs, Truffle will compile only the contracts that have been changed since the last compile. If one like to override this behaviour, run the above command with the –all option. Artifacts of your compilation will be placed in the build/contracts/ directory, relative to your project root. (This directory will be created if it does not exist.) These artifacts are integral to the inner workings of Truffle, and they play an important part in the successful deployment of your application. You should not edit these files as they'll be overwritten by contract compilation and deployment.Truffle supports dependencies installed via both EthPM and NPM. To import contracts from a dependency, use the following syntaximport "somepackage/SomeContract.sol";

5.1.2 GANACHE

Ganache is a personal blockchain for rapid Ethereum and Corda distributed application development. You can use Ganache across the entire development cycle; enabling you to develop, deploy, and test your dApps in a safe and deterministic environment.

Ganache UI is desktop application supporting both Ethereum and Corda technology. In addition, an Ethereum version of ganache is available as a command-line tool: ganache-cli (formerly known as the TestRPC).

1. Install Ganache

Next, double-click on the downloaded file, follow the prompts, and you're up and running.

2. Create a Workspace

When you open Ganache for the first time, you'll see the home screen. On this screen you're prompted to load an existing workspace (if any exist), create a new custom workspace, or quickstart a one-click blockchain with default options. For now, let's go with a quickstart workspace. Select the desired blockchain from the QUICKSTART drop down; you can choose to start an Ethereum node or Corda network, then click the QUICKSTART button.

5.1.2.1 Linking a truffle project

To link a project, enter the settings by clicking the gear icon in the upper right.

One should be seeing the WORKSPACE settings pane; if not, you can get there by clicking the WORKSPACE tab in the top left.

From here, there is a section labelled TRUFFLE PROJECTS. Beneath this box, click the button ADD PROJECT. A file selection popup will appear. Navigate to the folder of your Truffle project, and select the truffle-config.js or truffle.js configuration file. The file you pick must be either named truffle-config.js or truffle.js for Ganache to correctly load it. After selecting the file, one will see it listed in the TRUFFLE PROJECTS section.

You can add multiple projects to a workspace. After you're finished adding projects you can click the SAVE AND RESTART (SAVE WORKSPACE if this is a new workspace) button in the top right. After at least one workspace has been created, the home screen will now have a list of workspaces for you to choose from. You can scroll through the list to find the desired workspace, and then load the workspace by clicking its name.

5.1.3 REACT

React (also known as React.js or ReactJS) is an open-source JavaScript library for building user interfaces or UI components. It is maintained by Facebook and a community of individual developers and companies. React can be used as a base in the development of single-page or mobile applications. However, React is only concerned with rendering data to the DOM, and so creating React applications usually requires the use of additional libraries for state management and routing. Redux and React Router are respective examples of such libraries.

5.1.3.1 React Truffle Box

This box comes with everything you need to start using smart contracts from a react app. This is as barebones as it gets, so nothing stands in your way.

React has been designed from the start for gradual adoption, and you can use as little or as much React as you need. Whether you want to get a taste of React, add some interactivity to a simple HTML page, or start a complex React-powered app, the links in this section will help you get started.

5.1.3.2 Online Playgrounds

If you're interested in playing around with React, you can use an online code playground. Try a Hello World template on CodePen, CodeSandbox, Glitch, or Stackblitz.

If you prefer to use your own text editor, you can also download this HTML file, edit it, and open it from the local filesystem in your browser. It does a slow runtime code transformation, so we'd only recommend using this for simple demos.

5.1.3.3 Add React To A Website

You can add React to an HTML page in one minute. You can then either gradually expand its presence, or keep it contained to a few dynamic widgets.

5.1.3.4 Create a new React App

When starting a React project, a simple HTML page with script tags might still be the best option. It only takes a minute to set up!

As your application grows, you might want to consider a more integrated setup. There are several JavaScript toolchains we recommend for larger applications. Each of them can work with little to no configuration and lets you take full advantage of the rich React ecosystem. Learn how.

An element describes what you want to see on the screen:

const element = <h1>Hello, world</h1>;

Unlike browser DOM elements, React elements are plain objects, and are cheap to create. React DOM takes care of updating the DOM to match the React elements.

5.1.3.5 Components

React code is made of entities called components. Components can be rendered to a particular element in the DOM using the React DOM library. When rendering a component, one can pass in values that are known as "props".

ReactDOM.render(<Greeter greeting="Hello World!" />,

document.getElementById('myReactApp'));

The two primary ways of declaring components in React is via functional components and class-based components.

Parallel native technology for creating reusable building blocks of the web — Web Components. Advantage over React components — ability to create components not only for React but also for Angular, other libraries/frameworks, and for projects without any external dependency.

5.1.3.6 Functional Components

Functional components are declared with a function that then returns some JSX.

const Greeting = (props) => Hello, props.name! ;

5.1.3.7 Class Based Components

Class-based components are declared using ES6 classes.

class ParentComponent extends React.Component

state = color: 'green' ;

render()

return (

<ChildComponent color=this.state.color />

);

5.1.3.8 Virtual DOM

Another notable feature is the use of a virtual Document Object Model, or virtual DOM. React creates an in-memory data-structure cache, computes the resulting differences, and then updates the browser's displayed DOM efficiently. This process is called reconciliation. This allows the programmer to write code as if the entire page is rendered on each change, while the React libraries only render subcomponents that actually change. This selective rendering provides a major performance boost. It saves the effort of recalculating the CSS style, layout for the page and rendering for the entire page.

5.1.3.9 Lifecycle methods

Lifecycle methods use a form of hooking that allows execution of code at set points during a component's lifetime.

- shouldComponentUpdate allows the developer to prevent unnecessary re-rendering of a component by returning false if a render is not required.

- componentDidMount is called once the component has "mounted" (the component has been created in the user interface, often by associating it with a DOM node). This is commonly used to trigger data loading from a remote source via an API.
- componentWillUnmount is called immediately before the component is torn down or "unmounted". This is commonly used to clear resource demanding dependencies to the component that will not simply be removed with the unmounting of the component (e.g., removing any setInterval() instances that are related to the component, or an "eventListener" set on the "document" because of the presence of the component)
- render is the most important lifecycle method and the only required one in any component. It is usually called every time the component's state is updated, which should be reflected in the user interface.

5.1.3.10 Javascript XML

JSX, or JavaScript XML , is an extension to the JavaScript language syntax. Similar in appearance to HTML, JSX provides a way to structure component rendering using syntax familiar to many developers. React components are typically written using JSX, although they do not have to be (components may also be written in pure JavaScript). JSX is similar to another extension syntax created by Facebook for PHP called XHP.

React does not attempt to provide a complete "application library". It is designed specifically for building user interfaces and therefore does not include many of the tools some developers might consider necessary to build an application. This allows the choice of whichever libraries the developer prefers to accomplish tasks such as performing network access or local data storage. Common patterns of usage have emerged as the library matures.

5.1.3.11 Use of the flux architecture

To support React's concept of unidirectional data flow (which might be contrasted with AngularJS's bidirectional flow), the Flux architecture represents an alternative to the popular model-view-controller architecture. Flux features actions which are sent through a central dispatcher to a store, and changes to the store are propagated back to the view. When used with React, this propagation is accomplished through component properties.

Flux can be considered a variant of the observer pattern. A React component under the Flux architecture should not directly modify any props passed to it, but should be passed callback functions that create actions which are sent by the dispatcher to modify the store. The action is an object whose responsibility is to describe what has taken place: for example, an action describing one user "following" another might contain a user id, a target user id, and the type USERFOLLOWEDANOTHERUSER.

The stores, which can be thought of as models, can alter themselves in response to actions received from the dispatcher.

This pattern is sometimes expressed as "properties flow down, actions flow up". Many implementations of Flux have been created since its inception, perhaps the most well-known being Redux, which features a single store, often called a single source of truth. [3]

5.1.4 WEB3

Web 3 refers to an Internet that is made possible by decentralized networks, such as Bitcoin and Ethereum. The key innovation of these networks is the creation of platforms that no single entity controls, yet everyone can still trust. That's because every user and operator of these networks must follow the same set of hard-coded rules, known as consensus protocols.

The secondary innovation is that these networks allow value or money to be transferred between accounts. These two things—decentralization and Internet money—are the keys to understanding Web 3.

With Web 3, the network is decentralized, so no one entity controls it, and the decentralized applications (dapps) that are built on top of the network are open. The openness of the decentralized web means that no single party can control data or limit access. Anyone is able to build and connect with different dapps without permission from a central company.

On Web 3, money is native. Instead of having to rely on the traditional financial networks that are tied to governments and restricted by borders, money on Web 3 is instant, global, and permissionless. It also means tokens and cryptocurrencies can be used to design completely new business models and economies, a field increasingly becoming known as tokenomics.

For example, advertising on the decentralized web would not need to rely on selling users' data to advertisers, but could instead reward users with a token for viewing ads.

5.1.4.1 Getting started

The web3.js library is a collection of modules that contain functionality for the ethereum ecosystem.

web3-eth is for the ethereum blockchain and smart contracts.

web3-shh is for the whisper protocol, to communicate p2p and broadcast.

web3-bzz is for the swarm protocol, the decentralized file storage.

web3-utils contains useful helper functions for Dapp developers.

5.1.4.2 Adding Web3.JS

First you need to get web3.js into your project. This can be done using the following methods:

npm: npm install web3

yarn: yarn add web3

pure js: link the dist/web3.min.js

After that you need to create a web3 instance and set a provider.

Most Ethereum-supported browsers like MetaMask have an EIP-1193 compliant provider available at window.ethereum.

For web3.js, check Web3.givenProvider.

If this property is null you should connect to a remote/local node.

// In Node.js use: const Web3 = require('web3');

let web3 = new Web3(Web3.givenProvider || "ws://localhost:8545"); That's it! now you can use the web3 object.

5.1.4.3 Callbacks Promises Events

To help web3 integrate into all kinds of projects with different standards we provide multiple ways to act on asynchronous functions.

Most web3.js objects allow a callback as the last parameter, as well as returning promises to chain functions.

Ethereum as a blockchain has different levels of finality and therefore needs to return multiple "stages" of an action. To cope with requirement we return a "promiEvent" for functions like web3.eth.sendTransaction or contract methods. This "promiEvent" is a promise combined with an event emitter to allow acting on different stages of action on the blockchain, like a transaction.

PromiEvents work like a normal promises with added on, once and off functions. This way developers can watch for additional events like on "receipt" or "transactionHash".

web3.eth.sendTransaction(from: '0x123...', data: '0x432...')

.once('sending', function(payload) ...)

.once('sent', function(payload) ...)

.once('transactionHash', function(hash) ...)

.once('receipt', function(receipt) ...)

.on('confirmation', function(confNumber, receipt, latestBlockHash) ...)

.on('error', function(error) ...)

.then(function(receipt) // will be fired once the receipt is mined);

5.1.4.4 JSON interface

The json interface is a json object describing the Application Binary Interface (ABI) for an Ethereum smart contract.

Using this json interface web3.js is able to create JavaScript object representing the smart contract and its methods and events using the web3.eth.Contract object.

5.1.4.5 Specification

Functions

- type: "function", "constructor" (can be omitted, defaulting to "function"; "fallback" also possible but not relevant in web3.js);
- name: the name of the function (only present for function types);
- constant: true if function is specified to not modify the blockchain state;
- payable: true if function accepts ether, defaults to false;
- stateMutability: a string with one of the following values:
  - pure (specified to not read blockchain state)
  - view (same as constant above)

  - nonpayable and payable (same as payable above)

- inputs: an array of objects, each of which contains:
  - name: the name of the parameter
  - type: the canonical type of the parameter
- outputs: an array of objects same as inputs, can be omitted if no outputs exist.

Events

- type: always "event"
- name: the name of the event;
- inputs: an array of objects, each of which contains:
  - name: the name of the parameter;
  - type: the canonical type of the parameter.
  - indexed: true if the field is part of the log's topics, false if it one of the log's data segment.

5.1.5 METAMASK PLUG-IN

MetaMask is a very useful tool that plays a pivotal role in allowing you to make your foray into the world of blockchain. First and foremost, MetaMask is an Ethereum wallet. It allows one to

• Create accounts for use in the various Ethereum networks

• It maintains the private keys for accounts so that one can export them or import new accounts.

• Switch between the various Ethereum networks, so that accounts can reflect the correct balance for each network

• Perform transactions between accounts

• One can transfer Ethers from one account to another. One can also hold tokens in your MetaMask accounts.

• One can also view your detail transactions on Etherscan, a blockchain explorer.

MetaMask is an extension for accessing Ethereum enabled distributed applications, or "Dapps" in your browser!

The extension injects the Ethereum web3 API into every website's javascript context, so that dapps can read from the blockchain.

MetaMask also lets the user create and manage their own identities (via private keys, local client wallet and hardware wallets like Trezor^{TM}), so when a Dapp wants to perform a transaction and write to the blockchain, the user gets a secure interface to review the transaction, before approving or rejecting it.

Because it adds functionality to the normal browser context, MetaMask requires the permission to read and write to any webpage. You can always "view the source" of MetaMask the way you do any Chrome extension.

Enables access to:

- Web 3.0
- Dapps
- NFTs
- erc20
- tokens
- ICOs
- erc271

5.1.5.1 Installing Metamask

The easiest way to install MetaMask is to use the Chrome browser and install MetaMask as a Chrome extension.

To install the MetaMask extension:

• Launch Chrome and navigate to the Chrome Web Store.

• Search for MetaMask.

• You should now be able to see the MetaMask extension in the search result. Click Add to Chrome (see Figure 5-2). You will be prompted to add MetaMask to Chrome. Click Add extension

Once MetaMask is added to Chrome, you will be able to see its icon appear on the top right corner of the browser

5.1.6 NODE.JS

Node.js is an open-source, cross-platform, JavaScript runtime environment (Framework) that executes JavaScript code outside a web browser. Node.js lets developers use JavaScript to write command line tools and for server-side scripting—running scripts server-side to produce dynamic web page content before the page is sent to the user's web browser. Consequently, Node.js represents a "JavaScript everywhere" paradigm, unifying web-application development around a single programming language, rather than different languages for server- and client-side scripts.

Though .js is the standard filename extension for JavaScript code, the name "Node.js" doesn't refer to a particular file in this context and is merely the name of the product. Node.js has an event-driven architecture capable of asynchronous I/O. These design choices aim to optimize throughput and scalability in web applications with many input/output operations, as well as for real-time Web applications (e.g., real-time communication programs and browser games).

The Node.js distributed development project was previously governed by the Node.js Foundation, and has now merged with the JS Foundation to form the OpenJS Foundation, which is facilitated by the Linux Foundation's Collaborative Projects program.

Corporate users of Node.js software include GoDaddy, Groupon, IBM, LinkedIn, Microsoft, Netflix, PayPal, Rakuten, SAP, Voxer, Walmart, and Yahoo!.

Node.js allows the creation of Web servers and networking tools using JavaScript and a collection of "modules" that handle various core functionalities. Modules are provided for file

system I/O, networking (DNS, HTTP, TCP, TLS/SSL, or UDP), binary data (buffers), cryptography functions, data streams, and other core functions. Node.js's modules use an API designed to reduce the complexity of writing server applications.

JavaScript is the only language that Node.js supports natively, but many compile-to-JS languages are available. As a result, Node.js applications can be written in CoffeeScript, Dart, TypeScript, ClojureScript and others.

Node.js is primarily used to build network programs such as Web servers. The most significant difference between Node.js and PHP is that most functions in PHP block until completion (commands only execute after previous commands finish), while Node.js functions are non-blocking (commands execute concurrently or even in parallel, and use callbacks to signal completion or failure).

Node.js is officially supported on Linux, macOS and Microsoft Windows 8.1 and Server 2012 (and later), with tier 2 support for SmartOS and IBM AIX and experimental support for FreeBSD. OpenBSD also works, and LTS versions available for IBM i (AS/400). The provided source code may also be built on similar operating systems to those officially supported or be modified by third parties to support others such as NonStop OS and Unix servers.

5.1.6.1 Platform Architecture

Node.js brings event-driven programming to web servers, enabling development of fast web servers in JavaScript. Developers can create scalable servers without using threading, by using a simplified model of event-driven programming that uses callbacks to signal the completion of a task. Node.js connects the ease of a scripting language (JavaScript) with the power of Unix network programming.

Node.js was built on the Google V8 JavaScript engine since it was open-sourced under the BSD license. It is proficient with internet fundamentals such as HTTP, DNS, TCP. JavaScript was also a well-known language, making Node.js accessible to the web development community.

5.1.6.2 Technical Details

Node.js is a JavaScript runtime environment that processes incoming requests in a loop, called the event loop.

Threading

Node.js operates on a single-thread event loop, using non-blocking I/O calls, allowing it to support tens of thousands of concurrent connections without incurring the cost of thread context switching. The design of sharing a single thread among all the requests that use the observer pattern is intended for building highly concurrent applications, where any function performing I/O must use a callback. To accommodate the single-threaded event loop, Node.js uses the libuv library—which, in turn, uses a fixed-sized thread pool that handles some of the non-blocking asynchronous I/O operations.

A thread pool handles the execution of parallel tasks in Node.js. The main thread function call posts tasks to the shared task queue, which threads in the thread pool pull and execute. Inherently non-blocking system functions such as networking translate to kernel-side non-blocking sockets, while inherently blocking system functions such as file I/O run in a blocking way on their own

threads. When a thread in the thread pool completes a task, it informs the main thread of this, which in turn, wakes up and executes the registered callback.

A downside of this single-threaded approach is that Node.js doesn't allow vertical scaling by increasing the number of CPU cores of the machine it is running on without using an additional module, such as cluster, StrongLoop Process Manager, or pm2. However, developers can increase the default number of threads in the libuv thread pool. The server operating system (OS) is likely to distribute these threads across multiple cores. Another problem is that long-lasting computations and other CPU-bound tasks freeze the entire event-loop until completion.[citation needed]

Node.js uses libuv to handle asynchronous events. Libuv is an abstraction layer for network and file system functionality on both Windows and POSIX-based systems such as Linux, macOS, OSS on NonStop, and Unix.

The core functionality of Node.js resides in a JavaScript library. The Node.js bindings, written in C++, connect these technologies to each other and to the operating system.

V8

V8 is the JavaScript execution engine which was initially built for Google Chrome. It was then open-sourced by Google in 2008. Written in C++, V8 compiles JavaScript source code to native machine code at runtime. As of 2016, it also includes Ignition, a bytecode interpreter.

Package Management

npm is the pre-installed package manager for the Node.js server platform. It installs Node.js programs from the npm registry, organizing the installation and management of third-party Node.js programs. Packages in the npm registry can range from simple helper libraries such as Lodash to task runners such as Grunt.

Unified API

Node.js can be combined with a browser, a database that supports JSON data (such as Postgres, MongoDB, or CouchDB) and JSON for a unified JavaScript development stack. With the adaptation of what were essentially server-side development patterns such as MVC, MVP, MVVM, etc., Node.js allows the reuse of the same model and service interface between client side and server side.

Event Loop

Node.js registers with the operating system so the OS notifies it of connections and issues a callback. Within the Node.js runtime, each connection is a small heap allocation. Traditionally, relatively heavyweight OS processes or threads handled each connection. Node.js uses an event loop for scalability, instead of processes or threads. In contrast to other event-driven servers, Node.js's event loop does not need to be called explicitly. Instead, callbacks are defined, and the server automatically enters the event loop at the end of the callback definition. Node.js exits the event loop when there are no further callbacks to be performed.

Web Assembly

Node.js supports WebAssembly and as of version 14.x has an experimental support of WASI, the WebAssembly System Interface. [10]

5.1.7 SOLIDITY COMPILER

5.1.7.1 Version

Solidity versions follow semantic versioning and in addition to releases, nightly development builds are also made available. The nightly builds are not guaranteed to be working and despite best efforts they might contain undocumented and/or broken changes. We recommend using the latest release. Package installers below will use the latest release.

5.1.7.2 Remix

Access Remix online, you don't need to install anything. If you want to use it without connection to the Internet, go to https://github.com/ethereum/remix-live/tree/gh-pages and download the .zip file as explained on that page.

5.1.7.3 NPM

Use npm for a convenient and portable way to install solcjs, a Solidity compiler. The solcjs program has fewer features than the ways to access the compiler described further down this page. The Using the Commandline Compiler documentation assumes you are using the full-featured compiler, solc. The usage of solcjs is documented inside its own repository.

Note: The solc-js project is derived from the C++ solc by using Emscripten which means that both use the same compiler source code. solc-js can be used in JavaScript projects directly (such as Remix). Please refer to the solc-js repository for instructions.

npm install -g solc

5.1.7.4 Docker

We provide up to date docker builds for the compiler. The stable repository contains released versions while the nightly repository contains potentially unstable changes in the develop branch.

docker run ethereum/solc:stable –version

Currently, the docker image only contains the compiler executable, so you have to do some additional work to link in the source and output directories.

5.1.7.5 Binary Packages

We also have PPAs for Ubuntu, you can get the latest stable version using the following commands:

sudo add-apt-repository ppa:ethereum/ethereum

sudo apt-get update

sudo apt-get install solc

The nightly version can be installed using these commands:

sudo add-apt-repository ppa:ethereum/ethereum

sudo add-apt-repository ppa:ethereum/ethereum-dev

sudo apt-get update

sudo apt-get install solc

We are also releasing a snap package, which is installable in all the supported Linux distros. To install the latest stable version of solc:

sudo snap install solc

If you want to help testing the latest development version of Solidity with the most recent changes, please use the following:

sudo snap install solc –edge

Arch Linux also has packages, albeit limited to the latest development version:

pacman -S solidity

We distribute the Solidity compiler through Homebrew as a build-from-source version. Pre-built bottles are currently not supported.

brew update

brew upgrade

brew tap ethereum/ethereum

brew install solidity

If you need a specific version of Solidity you can install a Homebrew formula directly from Github.

View solidity.rb commits on Github.

Follow the history links until you have a raw file link of a specific commit of solidity.rb.

Install it using brew:

brew unlink solidity

Install 0.4.8

brew install

Gentoo Linux also provides a solidity package that can be installed using emerge:

emerge dev-lang/solidity

5.2 FRONT-END

The frontend of the proposed decentralised application provides various functionalities to the farmers. One could insure his crop through the application for a certain time period and if within that time period his crop gets harmed, he could easily upload the images of the affected leaves and if the machine learning model finds a damage to the crops, his damage would be covered by giving a specific coverage amount based on the disease caused.

The frontend here is developed using react scripts, which had to be installed on the system via the command- npm install react-scripts. The versions used were as follows: React: 16.11.0 React-dom: 16.11.0 React-scripts: 3.2.0

The frontend was linked with the IPFS-api in order to directly upload the images via IPFS and obtain the IPFS hash, ethereum contract address and the transaction hash and then the prediction is made whose result gets displayed on the screen.

The frontend also interacts with the insurance smart contract written in solidity to ensure the insurance functionalities effectively.

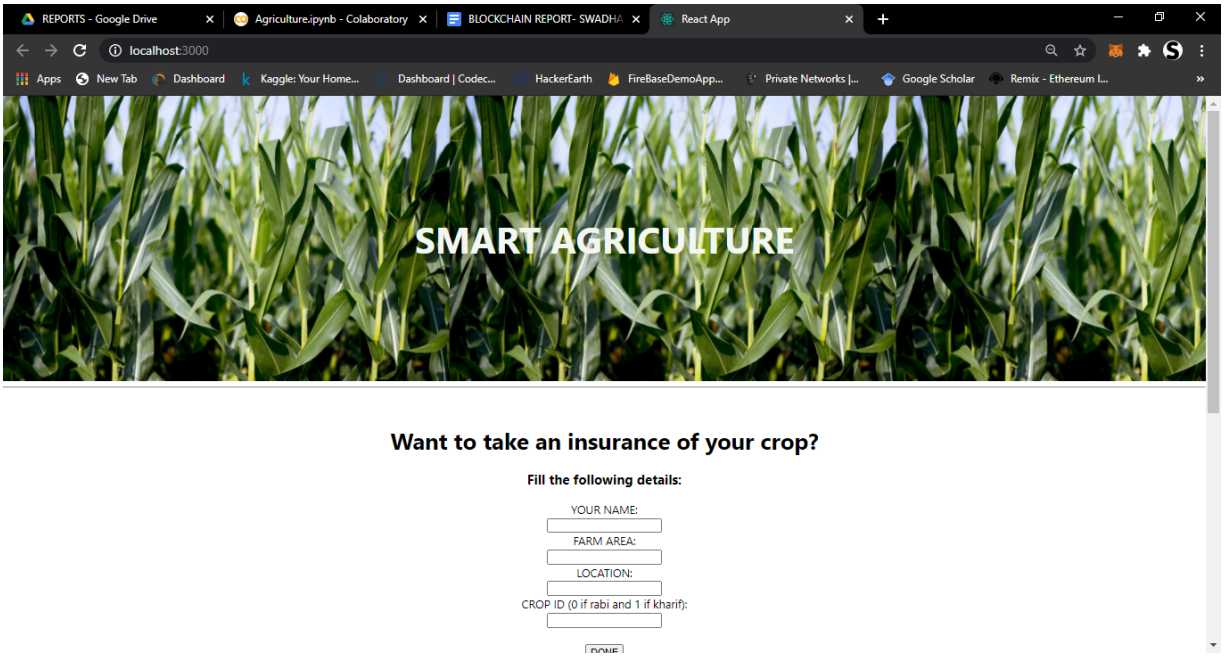


Figure 5.1: Part 1 of User Interface

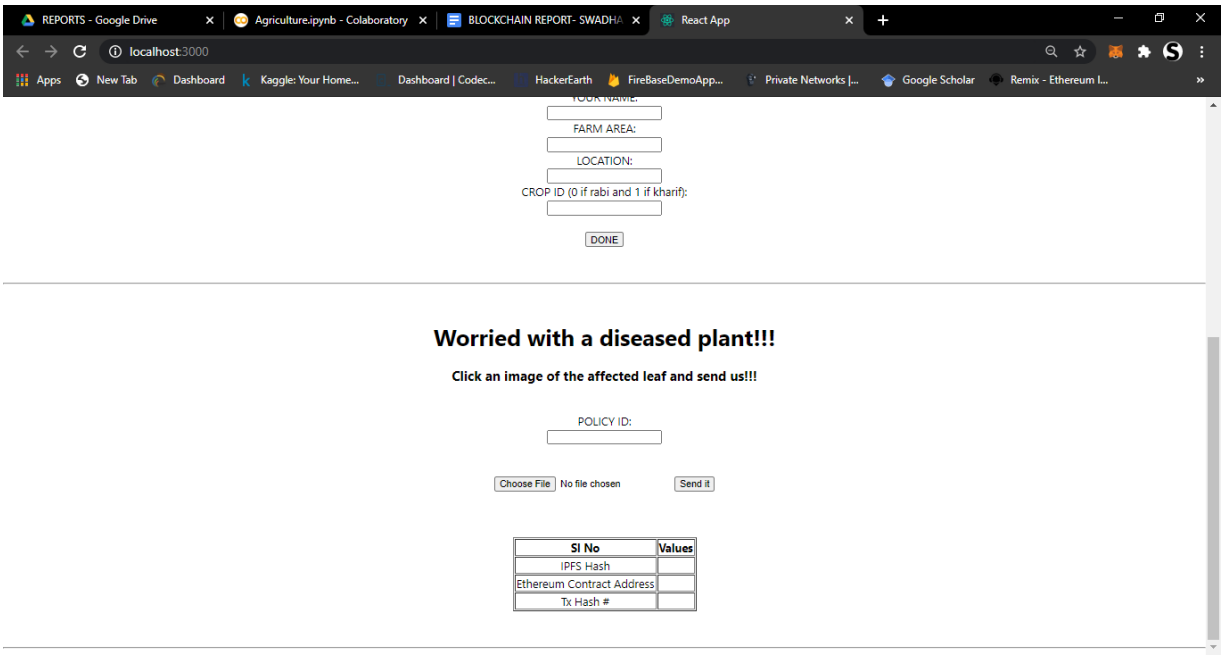


Figure 5.2: Part 2 of User Interface

6. ALGORITHMS

6.1 MACHINE LEARNING MODEL

6.1.1 Importing required libraries

Here, we needed to import certain libraries in python.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import os
import cv2
from keras.models import Sequential
from keras.applications import DenseNet121
from keras.layers import Dense, Dropout, Activation, Flatten, Embedding
from keras.layers import Conv1D, GlobalMaxPooling1D, MaxPooling1D, Conv2D,
MaxPooling2D, GlobalAveragePooling2D
from keras.optimizers import Adam
```

6.1.2 Preprocessing and Model building

Preprocessing of the data is highly important as it makes the model highly precise in grasping the features of the images.

```
for image in imagesort:
print(image)
image1 = cv2.imread(image)
image2 = cv2.resize(image1, (200, 200))
image3 = image2.astype('float32')
image4 = image3/255
imageread.append(image4)
imagearray = np.asarray(imageread)
```

Here, the resizing, scaling, normalisation and the edge detection was done for the image dataset.

Now, moving further, the model was built using densenet121, pooling, flattening and dense layers using their .

```
densemodel = Sequential()
densemodel.add(DenseNet121(includetop=False, weights='imagenet', inputshape=(200, 200, 3)))
densemodel.add(GlobalAveragePooling2D())
densemodel.add(Dense(4, activation='sigmoid'))
```

Later, the model was converted into an API for oraclizing it into the smart contracts(mlmodel.sol)

6.2 ORACLIZING ML MODEL

Now, here, a smart contract was made in which the oraclize or provable things functions were imported for communicating with the API of machine learning model built earlier.

After this, query was generated for prediction and callback function got reverted with the information of predicted class.

```
contract mlmodel is usingOraclize

uint public predictedclass;

event newOraclizeQuery(string description);

function callback(bytes32 queryId, uint result, bytes proof)
if(msg.sender != oraclizecbAddress()) revert();
predictedclass = result;

function predict(string memory hash) payable public
newOraclizeQuery("Oraclize query was sent, standing by for the answer...");
oraclizequery("URL", "json()", hash);
```

newOraclizeQuery was the event generated for the description. Moreover, json() command was used as the API beholds the values in a json file.

6.3 INSURANCE

Two structures CropType and Policy were defined at first for the crop types registered and for the cases where the consumer applies for a new policy.

Structure CropType:

string name;
uint premiumperAcre;
uint duration;
uint coverageperAcre;

Structure Policy:

string name;
uint area;
string location;
uint policyId;
address user;
uint premium;
uint startTime;
uint endTime;
uint coverageAmount;
uint claimAmount;
uint cropId;
policyState state;

Moving on, a function for new policy was defined which gets invoked whenever the user applies for a new policy.

function newPolicy (string memory name, uint area, string memory location, uint cropID)
public payable
uint pId = Policies.length+1;
userPolicies[msg.sender].push(pId);
Policy memory p = Policies[pId];
p.policyId = pId;
p.name = name;
p.area = area;
p.location = location;
p.cropId = cropID;
p.user = msg.sender;
p.premium = CropTypes[p.cropId].premiumperAcre * area;
p.startTime = now;
p.endTime = now + CropTypes[p.cropId].duration * 30*24*60*60;
p.coverageAmount = CropTypes[p.cropId].coverageperAcre * p.area;

```
p.state = policyState.Active;

require(msg.value == (p.premium),"INCORRECT PREMIUM AMOUNT");
```

Now, the claim function was defined which gets called when the ml model functions on an image and it finds certain diseases in the crops and user has insured the plant earlier.

```
function claim( uint predictedclass, uint Id) public

require(msg.sender == Policies[Id].user, "User Not Authorized");
require(Policies[Id].state == policyState.Active, "Policy Not Active");

if(now > Policies[Id].endTime)
Policies[Id].state = policyState.TimedOut;
revert("Policy's period has Ended.");

uint cl;

if(predictedclass == 0)
cl = 0;

else if(predictedclass == 1)
cl = 100;

else if(predictedclass == 2)
cl = 60;

else if(predictedclass == 3)
cl = 50;

makerequest(cl, Id);
```

After the successful execution of this function, the amount gets transferred to the account of the user if all the terms and conditions get fulfilled.

7. EXPERIMENTAL ANALYSIS

	PRECISION	RECALL	F1 SCORE	SUPPORT
0	0.93	0.93	0.95	280
1	0.94	0.96	0.93	234
2	0.95	0.97	0.99	318
3	0.92	0.92	0.98	260
micro avg	0.97	0.96	0.96	1092
macro avg	0.92	0.93	0.94	1092
weighted avg	0.97	0.99	0.98	1092
samples avg	0.99	0.96	0.95	1092

The table above depicts the result obtained for the machine learning model for the prediction of diseases under four classes: leaves that are healthy, leaves with multiple diseases, rust and scab.

According to the table, the f1-score obtained for the four classes 0,1,2 and 3 are 0.95, 0.93, 0.99, 0.98 respectively. However, in the cases of multi-class classification, the overall accuracy depends upon the f1-score of the weighted average which in our case is 0.98. Thereby, the model claims to give an accuracy of about 98 percent.

The model here, was built using densenet121 function from keras module because of its ability to concat the output of the previous layer with the future layer. Moreover, it uses pre-trained weights i.e., imagenet, which are highly accurate in grasping the vital features from the images. The DenseNet network is divided into Dense Blocks where the number of filters are different, but dimensions within the block are same. It is one of the latest neural networks for visual object recognition. It's quite similar to ResNet but has some fundamental differences.

A convolutional neural network is a deep learning algorithm which can take in an input image, assign importance to various objects in the image and be able to differentiate one from other. It is one of the main categories to do images recognition, images classifications. CNN image classifications takes an input image, process it and classify it under certain categories. In this model firstly convolutional neural network is used but this convolutional neural network doesn't provide good accuracy apart from this in place of convolutional network DenseNet network is used and this DenseNet network provide very good accuracy.

transaction hash	0xa56a5c15d106676609dae0055c1706a3bcbeb0eb47d8709a55fe37b9e6e45124
contract address	0x1b2ccbc328d17d7b7295f53d1ebbdd17e5f29e30
from	0x7e951cdca27d69b1d7a311bcc0cc56eb2805af86
to	Insurance.(constructor)
gas	3000000 gas
transaction cost	1486865 gas
execution cost	1104141 gas
hash	0xa56a5c15d106676609dae0055c1706a3bcbeb0eb47d8709a55fe37b9e6e45124
input	0x608...60033

transaction hash	0x85d507c1e1fd1aa62f76f1ab652c2b3b864b5fa93145901335f76ef576203a73
contract address	0x345f9019d0cb27a361e74aa0d8d5f36e65c56a89
from	0x7e951cdca27d69b1d7a311bcc0cc56eb2805af86
to	mlmodel.(constructor)
gas	3000000 gas
transaction cost	1395119 gas
execution cost	1026067 gas
hash	0x85d507c1e1fd1aa62f76f1ab652c2b3b864b5fa93145901335f76ef576203a73
input	0x608...20029

The table above depicts the result screen obtained for the solidity code of the "insurance.sol" file and the second table is for "mlmodel.sol". They were executed on the Remix IDE. The remix is a powerful, open-source tool that helps you write Solidity contracts straight from the browser. Written in JavaScript, Remix supports both usage in the browser and locally. The first row has a Transaction hash, it is an identifier used to uniquely identify a particular transaction. All on-chain transactions (the transactions from or to external addresses) have a unique id that can be seen in transaction details. The second row has an Ethereum contract address, it is the address of the place where the ETH is preserved. Then it displays the hash of the sender stored in the row named 'from' and sends it to the specified smart contract condition. The next row has gas, transaction cost, and execution cost. Gas is a unit that measures the amount of computational effort that it will take to execute certain operations. Every line of code in Solidity requires a certain amount of gas to be executed. Then the hash is the same transaction hash from above, the last row named 'input' is a unique hash formed while providing input.

8. DISCUSSION

Currently the dataset used is very limited in dynamics. There are numerous of crops and plants of dataset. Future plans is include adding specimens of leaf's of various plants and crops. Future dataset be cereals, pulses, rice and various fruits. These specimens will form future dataset and after that dataset will be trained in our model.

Data set includes various types of leaves. Some leaves show healthy characteristics and some shows some kind of disease. With help of model leaves can be examined, weather they are healthy or not. And if there is some kind of disease present it can also predict whether it is fungal or parasitic. For making model more accurate even sub-categories specimens will be added to dataset, like there are various varities of rices and various varities of oranges plant. Addition of various sub-category leaves in a particular crop/plant. Thus, more samples will lead to more accurate model in future.

Currently the examination of diseases is going basically through machine learning model. Image of apple leaves are examined by model for predicting diseases. This is current scenario of examination. Future plans includes hiring a specialist who will be a researcher for tackling the more complex unpredictable diseases. Model has a limitation because it can only work on basis of dataset.

If a rare disease in encountered who's sample is not yet available , model will fail there. Here, a specialist will examine sample physically . If it is a new disease then it will be his/her task to determine its dimension and type of diseases and specialist will also charge the fee. Through this we will come to know about the disease. The specialist will be paid for his work.

Currently insurance facility is available only for the apple plant in case of disease present. But future plans will broaden insurance dimensions. Like in the future there will be a insurance scheme for the stored foods too. If stored food gets damaged in period of storage, in that case crop damages due to natural disasters like flood and droughts. Broad insurance policy in future will greatly help farmers in future.

The proposed model enables the decentralised application to store the data on peer-to-peer networks implementing interplanetary file systems and a hypermedia protocol to make the web faster, safer and much more transparent. It abolishes the need of third party intermediates and a central server.

The traditional conventions have always laid focus on central authority to have the supreme decisive powers. There are cases where such authorities get biased and the user is denied his rights. Moreover, these insurance processes become so hassle that most of the farmers do not feel free to get insured due to lack of services.

However, the proposed model would provide a user friendly service which would help the farmers to take or claim an insurance just through some clicks. It would enforce transparency throughout the processes and there will be no cases of biasing as the ML Model will itself act as the proof of claiming insurance. The ML model working here is highly accurate and precise

and could easily identify even the diseases that could not be visualised by a naked eye.

9. CONCLUSION AND FUTURE SCOPE

In this model, a decentralized assisted plant disease prediction is proposed using deep convolutional neural networks. It works with a blending of machine learning and Blockchain technology. Apparently, the model has been able to predict 4 major diseases in apple plants with good accuracy, as well as gives the prompt to opt for decentralized insurance. The model enables any individual to help protect their crops. It has a simple web user interface, so It can be used by farmers all around the earth. In the future, the developed system will be linked to the National CropPest Management System to provide information on infection risks such as risk-based on the crop, mycelial growth rate, disease development speed, germination rate, and disease outbreak quantity.

The model intends to spread awareness among the masses by lessening the burden on the shoulders of the farmers and making the crop insured of losses. This would help the farmers as a boon in bad times.

The model is highly precise in detecting the diseases and this would help in reducing the wastage of crops by pests, pathogens and diseases. At the same time, being distributed and decentralised, it also maintains the security and integrity of data. Moreover, the model would also provide hassle free solutions to the farmers as compared to the traditional insurance methods. AI must be leveraged to increase the automation of tasks in agriculture and improve the yield while optimizing the use of natural resources. The implementation of smartphone android applications will be done in future.

References

[1] Imran Bashir. *Mastering Blockchain: Distributed ledger technology, decentralization, and smart contracts explained*. Packt Publishing Ltd, 2018.

[2] Gerald A Carlson. A decision theoretic approach to crop disease prediction and control. *American Journal of Agricultural Economics*, 52(2):216–223, 1970.

[3] Cory Gackenheimer. *Introduction to React*. Apress, 2015.

[4] Adem Efe Gencer, Soumya Basu, Ittay Eyal, Robbert Van Renesse, and Emin Gün Sirer. Decentralization in bitcoin and ethereum networks. In *International Conference on Financial Cryptography and Data Security*, pages 439–457. Springer, 2018.

[5] Garrick Hileman and Michel Rauchs. Global cryptocurrency benchmarking study. *Cambridge Centre for Alternative Finance*, 33:33–113, 2017.

[6] Donald Michie, David J Spiegelhalter, CC Taylor, et al. Machine learning. *Neural and Statistical Classification*, 13(1994):1–298, 1994.

[7] Muqaddas Naz, Fahad A Al-zahrani, Rabiya Khalid, Nadeem Javaid, Ali Mustafa Qamar, Muhammad Khalil Afzal, and Muhammad Shafiq. A secure data sharing platform using blockchain and interplanetary file system. *Sustainability*, 11(24):7054, 2019.

[8] Michael Nofer, Peter Gomber, Oliver Hinz, and Dirk Schiereck. Blockchain. *Business & Information Systems Engineering*, 59(3):183–187, 2017.

[9] Aqeel-ur Rehman. *Smart Agriculture: An Approach towards Better Agriculture Management*. 02 2015.

[10] Mithun Satheesh, Bruno Joseph D'mello, and Jason Krol. *Web development with MongoDB and NodeJs*. Packt Publishing Ltd, 2015.

[11] Gavin Wood et al. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum project yellow paper*, 151(2014):1–32, 2014.