ISSN: 2229-6956 (ONLINE) ICTACT JOURNAL ON SOFT COMPUTING, APRIL 2023, VOLUME: 13, ISSUE: 03
DOI: 10.21917/ijsc.2023.0411
SPECIAL ISSUE ON INNOVATIONS AND INTELLIGENCE FOR COMPUTING AND TECHNOLOGIES (IICT-2023)

BLOCKCHAIN ENABLED, COLLABORATIVE PLATFORM FOR AI AS A SERVICE

Venkata Raghava Kurada and Pallav Kumar Baruah
Department of Mathematics and Computer Science, Sri Sathya Sai Institute of Higher Learning, India

Abstract

With the advent of technology, modern human activities produce a huge amount of data. This vast amount of data facilitates a better model training, thus creating accurate predictions. But most of the business entities lack the facilities and resources to develop an AI system. There is a need for a platform, to which the business can outsource the process of data collection, model development, and its deployment. These models should be tailored for each use case. The work presented here attempts to address these issues using blockchain and incremental learning. The transactions and user identification in the platform are implemented using blockchain, thus maintaining the ownership of the model and dataset in a transparent, immutable and decentralized manner. Incremental learning algorithms are employed to facilitate the real-time updation of the model. All the models and the datasets collected in the platform are considered resources. The platform opens up an avenue for developing a marketplace for data and trained models.

Keywords:

Blockchain, Incremental Learning, AI as a Service, Data Market Place

1. INTRODUCTION

Recent years have witnessed the explosion of Machine Learning algorithms; the models are meant to accomplish complex tasks without compromising the performance. These machine learning models are increasingly used in the decision automation process. The development of a machine learning model generally depends on two factors, availability of the data and model training. The process of developing and fine-tuning these models is highly centralized in nature and beyond the reach of the common user. There is also the problem of collecting large data sets, where often times these datasets are proprietary and the process of data collection is often plagued with data brokers. This led to the development of the new field called AI as a Service (AIaaS) [1]. Typically, AIaaS refers to outsourcing the process of model development and maintenance to an external entity. One of the key objectives of AIaaS is to make AI accessible to business entities, irrespective of size of the organization or technical skills required. To foster AI diffusion, many cloud service providers such as Amazon [2], Google [3], and Azure [4] have started to offer AIaaS, but these are highly centralized in nature and there is the issue of identification and authentication of users, with respect to the cloud service providers, for example, IAM [32] by Amazon.

There is a need for the cloud-based platform, where a user is empowered to create, train and deploy a model with a click of a button. This platform should also enable the community of users to share data, receive incentives, use the platform to build various data repositories from multiple sources in a highly secure manner while maintaining the integrity of the data. Such a platform should also enable the user to obtain and use a purpose specific AI model which trained on the data available in the platform and target to achieve a specific task for the user. The service provided by the platform is to enable users to trade the resources like datasets and trained models that are generated by the platform. The platform also needs to update the model and datasets on the fly with the trending streaming data. The key challenges to realize such a platform are the establishing of trust in its user base and maintain the ownership of the authentic resources transparently. As the cloud is a centralized entity, the platform should guarantee that the user privacy is preserved.

Incremental learning [5] is a subset of Machine Learning where the model trained is continuously on the input data. The main objective of incremental learning is to improve the current model knowledge extensively and adapt to the data that is used, without forgetting its existing knowledge. Incremental learning is also known as Online Learning or Continuous Learning.

Blockchain [6] is a distributed ledger that is decentralized in nature. The transactions that take place in the ledger are immutable in nature, which offers accountability. Blockchain can be used as a trust-establishing platform between the trust less parties [17]. Each and every user has a set of keys namely private and public, with the help of cryptography these keys can be used to provide authentication and identity management. Because of the traceability, the blockchain can be used for record-keeping where the ownership of the datasets or trained models is maintained.

This work is of significance because it addresses the need for a platform that leverages on a decentralized ledger to facilitate collaborative data collection and transparent data marketplace while protecting user privacy. The use of the blockchain in the platform ensures a greater degree of trust. With its property of immutability and decentralized nature, the ownership of the resources can be maintained transparently by using Blockchain. However, there is a computational challenge to create a composite application using blockchain [35]. In order to circumvent this challenge, one needs to perform most of the computation off chain, thereby avoiding the actual performance bottleneck of computation. In our work, we achieve this by offloading the computation to the resources on the cloud. This enables the platform to create, train a wide array of models efficiently without any performance bottleneck.

The main contributions of the work are:

- The design and implementation of a platform that facilitates development and creation of data repositories from multiple sources.
- This implementation of the platform also demonstrates creating and training of models using the data repositories available in the platform.
- Design and implementation of decentralized data marketplace.
- Retraining and updating the models in real time with streaming data.
- The issues of trust, authentication, ownership is taken care of by using blockchain.

- The computational hurdles due to use of blockchain is overcome by offloading the computation to the cloud.

1.1 LITERATURE REVIEW

In this section, we will mention some key work done in conjunction with Machine Learning and Blockchain. Justin D Harris [6] proposed an on-chain method to train various models like perceptron and Naive Bayes, the results were discussed in [7]. The model is generated using a smart contract, that is executed on the blockchain. Similar work is discussed by Chen, Xuhui et al. [8], in which authors used on chain learning. Petrović, Nenad [9] limited the role of the blockchain for billing the user based on the extent of resource usage.

Marcelletti, et al [10] discuss the advantages of encompassing a Blockchain in MLOps. The usage of blockchain in a specialized privacy preserving machine learning, termed federated learning, has been discussed in [11] [12] [13]. A new method of developing the AIaaS on the blockchain called Secured AIaaS was proposed by Nicholas Six et al in [30]. In this work, the blockchain is used as a marketplace, where different entities like client, algorithm provider, infrastructure provider and data provider come together to generate a model. A specific model is generated using a given data set for a given user using a particular compute resource as a result of execution of smart contract.

1.2 PRELIMINARIES

1.2.1 Blockchain:

The idea of the Blockchain was initially proposed by the two physicists Stuart Haber and W Scott Stornetta [14], intending to deploy a system that prevents manipulation of past data by the future generation and also to prevent network spam. The same idea is extended using certain cryptographic methods coupled with techniques from game theory for exchanging a digital currency in, what is known as Bitcoin [15].

Blockchain is a decentralized and distributed ledger that is maintained by all the nodes of the network. To maintain the homogeneity of the records across the different nodes, various Consensus algorithms [16] are used. Blockchain is the database that holds the records of transactions in the form of blocks. The immutability in the blockchain is achieved using chaining. Each hash of the current block is stored in the next block. If the malicious actor wants to modify the content of a transaction in a certain block, the change is reflected in the hash of all the succeeding blocks. This new hash does not match the hash that is stored in the successive block, which leads to detection of any malicious changes. Thus, the malicious actor should change the content of all the blocks until the last block, which is computationally infeasible. The critical attribute of the blockchain lies in its ability to establish trust between the trust less parties. Zavolokina, Liudmila et al. [17] has discussed how trust-supporting design elements may be implemented to foster an end user trust in a blockchain platform.

1.2.2 Ethereum:

Ethereum [18] is developed by Vitalik Buterin along with Gavin Wood. Initially, Ethereum employed Proof of Work to maintain the consensus between the nodes, however the network adopted Proof of Stake [16] for consensus mechanism, in 2022. It uses Ether [18] as the native cryptocurrency. The key factor which separates the Ethereum blockchain from the others is the introduction of the Ethereum Virtual Machine (EVM) [18]. An EVM is the one single entity that is maintained by almost all the nodes of the network. Another breakthrough is the introduction of the Smart Contract [18], [20]. The main idea is to deploy the protocol that should maintain and ensure the immutable, continuous operations of the Smart Contract. Gas is the fuel of the Ethereum, which is charged by the miners to make the transactions. It has the capability to process 7–9 transactions per second and is plagued by scalability [19]. However, after adapting to Proof of Stake for consensus the performance improved by 18.6% as reported by Kapengut et al. [33].

1.2.3 Smart Contract:

Smart Contract [20] is the concept that is introduced in Ethereum. A smart contract is a code that exists on the network and is executed on the Ethereum Virtual Machine (EVM). This pioneering work by V Buterin [18] on smart contract rendered the Ethereum blockchain into a Turing machine and converted it to a computing platform. Solidity is the language used to program a smart contract on Ethereum [21]. Solidity is a high-level language which is highly influenced by the languages like C++, Java and JavaScript. The introduction of the EVM offered the capacity of executing a piece of code that is written either in Solidity or Assembly [18]. The smart contract can perform the transactions, send and receive funds and validate the stakes. This concept offered the flexibility to the developer to add if-else statements, along with the ability to move data on the fly. These codes have the ability to perform and take part in the transactions. Since these types of code deal with a huge amount of money and high-value assets, a bug or a loophole in the code can cause severe damage in terms of the economy. In order to mitigate these bugs, open-source libraries like Oppenzeplin can be used [31]. The other strategy is to minimize the number of the smart contract, which is adopted in our work.

1.2.4 Decentralized Application:

DApps is the short form for Decentralized Applications [18], these are often referred to as Distributed Applications as well. These types of applications use the smart contract as the backend. The front end of the application is generally a web page or android application. Each user on the blockchain network has a public key (often referred to as address) which can be used to send funds to an account.

Crypto Wallet is the specialized software that performs tasks like key management and connecting the user to the specific blockchain. These act as middleware between the web application and the backend. It performs tasks such as invoking a smart contract and conducting transactions on behalf of the user. The smart contracts deployed on a blockchain along with the crypto wallets are used in the development of the DApps. DApps have the front end implemented in the various JavaScript frameworks like Vue, React, and Svelte. Svelte is used in our work because of its efficient performance.

1.2.5 Incremental Learning:

Incremental learning [22] is a subset of Machine Learning, where the model is trained on a continuous input data stream. The main objective of incremental learning is to improve a model knowledge extensively and adapt to the data that it is trained on, without forgetting its existing knowledge. Unlike the traditional

Machine Learning algorithms, Incremental Learning in its purest essence encompasses a model which learns with one sample at a time. This set of algorithms excels in situations where the rate of data availability varies with time. Such systems are suitable for applications such as IoT, Physical Cyber Systems where the stream processing of the data plays a crucial role as proposed in the papers [23], [24]. The python module, River [25] is used to implement the Incremental Learning algorithms. River module can be considered as a wrap-up for both creame and multi-flow by sklearn.

Incremental learning can be classified into two types:

- *Reactive Learning:* The situations in which one does not have control over the data arrival. For example, the number of requests per minute on a server.
- *Proactive Learning:* The situations in which one has control over the data arrival. For example, data received in the IoT network.

1.2.6 AI as a Service:

AI as a Service [1] can be defined as AIaaS as cloud-based systems providing on-demand services to organizations and individuals to deploy, develop, train, and manage AI models. Reflecting on this broad definition reveals that AIaaS not only relates to AI software and applications available on-demand, such as chatbots using natural language processing, but also covers tools and resources needed to develop, operate and maintain AI models.

AIaaS [1] aims to make AI accessible and inexpensive to all organizations, regardless of size, technological sophistication, or AI funding. Without the need to master difficult algorithms or technologies, AIaaS leads users through the process of designing, implementing, or using data analytics models. Users can therefore concentrate on their core competencies, such as training and configuring their AI models, rather than worrying about installation, maintenance, and related administrative issues.

2. SYSTEM DESIGN AND ARCHITECTURE

2.1 SYSTEM DESIGN

System design constitutes the skeleton of a framework and aids the development of the system from its theory. As described earlier, the purpose of this proposed framework is to create a transparent, privacy centric, blockchain enabled platform that serves as a Data marketplace.

As visible in Fig.1, the proposed framework or the system has three layers. These layers are explained further below. The users of the proposed framework consist of data collectors, data contributors, and resource users.

2.1.1 Front End Layer:

The front end is the layer where the various users interact with the system to perform tasks like creating a data schema, contributing to the data schema, trading the resources and training a model. The user interacts with the system and its functionalities through a browser, which is referred to as the DApp browser in this context. The DApp browser contains functions that facilitate interactions with both the blockchain layer and the computation layer. For the computation layer, the front-end module acts as a gateway for authentication.

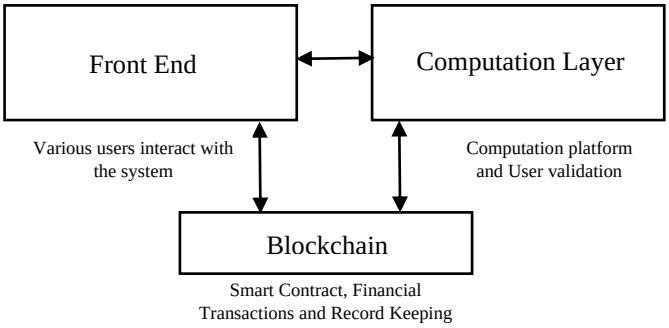

Fig.1. The Building Blocks of the Platform

2.1.2 Blockchain Layer:

Smart contract mainly constitutes this layer. A smart contract that is deployed on the blockchain network is used to maintain the record of ownership of a resource. The functions in smart contract are invoked by both the computation layer and front-end layer. The front end uses the data from the blockchain to validate the owner of the resources while trading a particular resource. Whereas, the computation layer is used to add, update these records. In order to reduce the gas, the proposed platform uses only one smart contract. The implementation of the smart contract is explained in the following sections.

2.1.3 Computation Layer:

A database and server constitute this layer. This layer resides off-chain, in order to leverage on the greater computation power. The database is used to store the user details like the signature, which is used for authentication purposes. In order to preserve the user privacy; it is designed based on zero-knowledge proof [33]. On the other hand, the server is where all the computation tasks like training the model, updating the models and basic application related jobs takes place. This layer uses the blockchain layer as a verification tool to authenticate and identify the user.

As we observe that the three layers are independent by design, due to this feature the hardware can be scaled up and down.

2.2 TASKS

2.2.1 Creating a Data Schema:

Data Schema is an abstract concept concerning the proposed framework, it is the collection of the features that a data contributors desired, along with features data type. Data schema offers flexibility and customizability to the data collector. In the proposed framework the data type is restricted to 4 types, integer, boolean, float and string. The data collector is expected to provide at least 15 data records that satisfy the schema, these are used to train a filter model that is used in the data cleaning process. The data contributor is also expected to provide the target, the number of data records he intends to collect via the platform and bounty, and the incentives that were offered to every contributor for each data record. After the successful creation of the data schema in the computation layer, the ID of the model along with the address of the owner is entered into the blockchain through the blockchain layer.

2.2.2 Contributing to Data Schema:

After the creation of the data schema, all the data contributors can view it from the front-end module of the proposed platform along with other details like incentives and the feedback for the model. The proposed platform implements the idea proposed in [6] to filter the data records. In work by Justin et al. [6] proposes to use a model that is trained to accuracy <70% as a filter to determine the authenticity of the particular record. When a data contributor provides a data record with the target variable, compared to the prediction of the filter model, if the error is less than a threshold, then the particular record is deemed as a good data record and gets accepted. On acceptance, the data contributor will receive the incentive. As incremental learning is used, which facilitates updating the filter model frequently, thus leading to effective data filtering.

2.2.3 Training a Model:

Training the model is fully supported by the computation layer. The resource user selects the type of algorithm that suits his requirements. The proposed platform provides a set of choices based on the user responses, the system suggests the best-suited learning type, i.e. Incremental, Conventional ML. The key difference between these two approaches is the update frequency of the filter. The incremental learning-based filter is updated for every data record collected, whereas the conventional machine learning-based models are updated until a threshold number of records are collected. The same is the case with the actual predicting models as well.

2.2.4 Trading the Resources:

Resources in this context refer to either fully trained models or collected datasets. As mentioned earlier, the computation layer keeps a record of the ownership of a particular resource in the blockchain. Through the frontend layer of the system, resources users can view all the available resources. The process starts with the seller quoting a price to the resource, to which the buyer agrees. The computation layer will validate the ownership of the buyer, by comparing the resource id and the buyer address from the blockchain layer. After successfully validating the ownership, the buyer wires the funds to the seller via blockchain. The record of the ownership is updated after the fund transfer.

2.3 SMART CONTRACT

Smart Contract is used for maintaining the record of the data collectors and their resources. The proposed platform uses a contract named Details Holder. This contract is used to maintain the record of ownership of a resource. The contract contains the getter and setter methods for user details and the resource id. It makes use of the events to interact with the front-end layer and computation layer. Computation layer validates the ownership of the resources.

Algorithm1 Smart Contract of DetailsHolder

New Resource Creation:

function newUser(id,
resourceid,userid,sign,proof)
  if (proof==sign) then
  add the details of user and resource
    return (success)
  else
    return (failure)
  end if
end function

Retrieve Data:

function getData(id, index)
  retrieve data of the specified user
  return (user record)
  return (failure)
end function

Update the New User:

function updateUser(id,id1,index)
  if (msg.sender==computationlayer)
    Delete the resource id from the old user map list
    Add the resource id to the new user map list
    Return (success)
  else
    Return (failure)
  end if
end function

The Algorithm1 explains the functioning of the smart contract for DetailsHolder. This algorithm has five functions that are used for new resource creation, retrieving the user details and updating the owner of the resource. The first function of Algorithm 1 *newuser*, is to be performed by the user, and it includes five variables, id, resourceid, userid, sign and proof; these are used for adding the new user and the resource details in a mapping which the other function to access the user details. This map contains an id and an array of name resources which contains all the resource id that is owned or created by the user. The second function is getData and it is invoked by the server in the computation layer, it takes two parameters id and index. Id variable is used as the index for the mapping, whereas the index is used for retrieving the particular resource. In order to verify the server is invoking this function, msg.sender is used and if the value of it is the same as the server from the computation layer, the details of the user are retrieved and returned else the failure message is emitted as an event. The 'msg.sender' is the term used by Solidity which contains the address of the caller. The last function named updateUser takes the id, id1 and index as arguments. The arguments id and id1 correspond to the index of the map list of the new and old owners respectively. The index is used to delete the respective resource id from the user structure.

2.4 TECHNOLOGIES USED

2.4.1 Metamask:

Metamask is one of the leading crypto wallets that is offered by Consensys [26]. It is the gateway to Web3 which lets the user buy, store and send tokens globally. It is open-source software that is maintained by 8–10 developers and has wide community support. Metamask lets users manage their private keys and various local client wallets. It even provides a secure interface to review and take part in the transaction that is initialized by the DApp. It is available as both a Browser extension and an Android

Application as well. The latest version of the Metamask was developed to be more robust by removing the web3 dependencies.

2.4.2 Svelte JS:

Svelte JS [27] is an open-source JavaScript front end compiler that was developed by Rich Harris and is currently maintained by the Svelte Core Team members [27]. It is a JavaScript framework that compiles the HTML template and manipulates the DOM directly without Virtual DOM, unlike Vue.js or React JS [28]. It is a component-based framework, which uses HTML, CSS and JavaScript with no extra syntax, unlike ReactJS and Vue.js. During the build and deployment phase, the Svelte Framework compiles the written code into standalone JavaScript modules. As the components are statically mounted, thus decreasing the workload on the browser.

2.4.3 Flask:

Flask is an open-source micro web Framework that was written by Armin Ronacher of Pocono [29]. It was developed by a group of enthusiasts of the Python Group in 2004 [29]. It is referred to as a micro framework because it does not necessitate the use of any specific tools or libraries. Furthermore, it doesn't have a database abstraction layer, form validation, or any other components that rely on third-party libraries to perform common tasks. Extensions, on the other hand, can be used to add application features as if they were built into Flask itself. Object-relational mappers, form validation, upload handling, various open authentication technologies, and several framework-related tools all have extensions.

2.4.4 MySQL:

MySQL is an open-source, community-driven database management system. It is an implementation of the Relational Database Management System. In RDBMS, the table data structure is used to store and retrieve the data. The source code of MySQL is primarily written in C and Prostar C. It was developed by Michael Widenius. The MySQL stack work with the operating system to create, retrieve and manipulate the tables that contain the data. It also enforces the ACID properties, which are maintained along with Identity and Access Control and backup creation. Furthermore, it uses Structured Query Language to create and extract data from the tables. It was currently owned by the Oracle Corporation and Nominated as the DB of the Year, 2019. MySQL is often used as LAMP stack, which stands for Linux, Apache, MySQL, Perl/PHP/Python

2.5 INTERACTIONS

The computation layer and the front-end layer are connected by the Fetch API. Fetch API is the modern alternative to the XMLHTTPRequest. The data is transferred in JSON format between the front end and the computation layers. Users can interact with the blockchain directly from the front end via the Metamask browser extension. An API named windows.ethereum exposed by Metamask is used for generating signatures and verifying them.

The smart contract used in the proposed framework is invoked by the address that is generated while deploying the smart contract. The events that are emitted by the smart contracts are listened to in the front-end layer, thus facilitating the interaction between the Front End and the Blockchain Layer. All the verification is offloaded to the DApp Browser, thus reducing the computation in the blockchain layer.

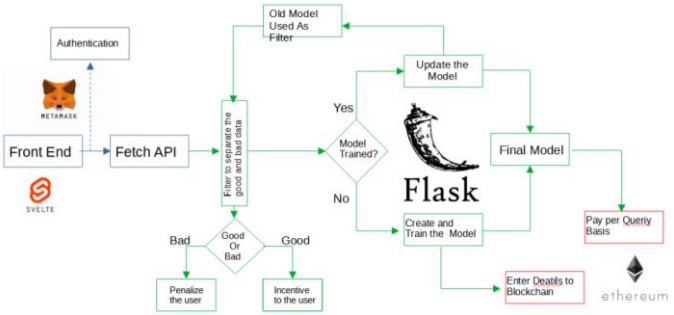

Fig.2. Depicting the Interaction between Different Technologies

2.6 DISTINGUISHING FEATURES OF PLATFORM

2.6.1 Cryptography Based Authentication:

Unlike the conventional web applications where the credentials are stored in the database and verified against each other while the user is logged in, the platform uses a cryptographic approach. The private key of the user is used to encrypt the user credentials to generate a signature. This signature is stored in the backend to authenticate the user for future logins.

2.6.2 High Availability of the Model:

With the usage of incremental learning, the model can be updated on a real-time basis. The computational cost associated with the operations like updating, predicting and learning is less than conventional machine learning algorithms. Thus, reducing the carbon footprint of the models.

2.6.3 Restricted Use of Blockchain:

As mentioned in previous sections. The blockchain is known for its immutability and distributed nature. But it also suffers from scalability issues. The proposed platform uses the blockchain as a record-keeping platform, which is used to authenticate the user and their ownership of the particular resource on the platform. Because of the traceability aspect of the blockchain, all the cryptographic transactions are conducted on the blockchain.

2.6.4 Filter Based Data Collection:

As proposed in [6], the proposed platform uses a semi-trained model to differentiate between the good data contributor from all data contributors. Since the filter can be updated in real-time, the robustness of the data collection process increases with time, i.e. more records are collected, and the more the filter model has trained the higher accuracy.

2.6.5 Off-Loaded Computation:

As mentioned in previous sections, the proposed platform uses the Computation Layer, which is independent of the blockchain layer. This facilitates the dynamic provision of the resource. It should also be taken into consideration that there are considerable technical restrictions in the Solidity [21], for instance, lack of support for the floating-point operation, which is crucial for the machine learning models. The higher computation leads to higher gas prices. These problems can be mitigated by off chaining these operations to the computation layer.

2.6.6 Hot Swamp of the Model:

In order to ensure the availability of the latest model that is trained on more data and limit the downtime of the production model, the proposed platform uses the model real-time model updates. All the models are stored in the form of pickle format. These pickle files are loaded into the server, after getting trained on the latest data, the newly trained model is used as the production model.

2.7 USAGE SCENARIO FOR PROPOSED PLATFORM

The Fig.3 depicts the basic usage scenario of the proposed framework. The system mainly has two entities, i.e., User and the Computation Layer. Users can be further classified as Data Contributors and Data Collectors. When the data collector intends to collect data, he creates a data schema. The computation layer verifies the authenticity of the data collector and creates a request for the data collection in the data marketplace.

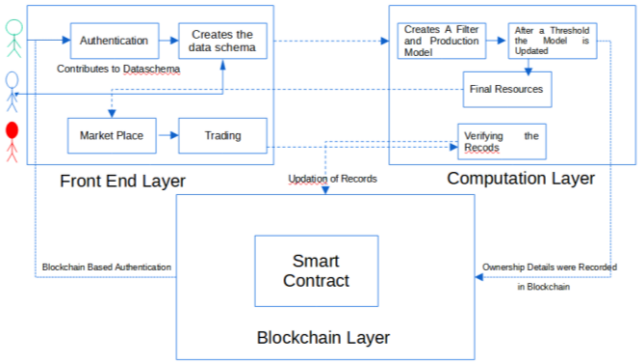

Fig.3. Flow of the Proposed Platform

This request is then reflected on the front end, where all the data contributors can view it and make contributions. The contributor data points are fed into a filter model, which makes an initial prediction. If the error between the initial prediction and the contributor target variable value falls under a threshold, the particular data point is accepted, and the contributor will receive the incentives in the form of cryptocurrency. If the error is greater than the threshold, then the particular data record is rejected. Once the production model is robust enough to make accurate predictions, the data collector can be made available to other users on the platform, either in the form of pay per query basis or by selling it. Thus, creating a self-sustained economy.

3. RESULT

3.1 EXPERIMENTAL SETUP

For testing the performance of the proposed framework, we have conducted experiments on a system with following configurations:

- AMD Ryzen 5 4600H with Radeon Graphics @ 3.0GHz
- 16 GB of memory with Pop! OS 22.04 LTS

The model implemented in this experiment is Linear Regression on mtcars [35]. The smart contract is developed and tested in Solidity using the Truffle Suite. The default iterations are limited to 5. The python code was timed with the help of jupyter magic tools - timeit. In both cases, the mean of all the 50 runs was taken. The results are presented in Fig.4.

3.2 EVALUATION METRICS

In section 1.1, we discuss two different approaches of model generation using a blockchain in [7],[30]. For Secured AIaaS [30], the key performance metric is the amount of gas consumed. It is known the value of gas depends on various factors like the economic value of the crypto token and the type of network that is used. In a paper titled, Analysis of Models for Decentralized and Collaborative AI on Blockchain [7] the authors use the gas price and Accuracy as the evaluation metrics.

In both the above cases [7], [30], model is generated on chain which leads to higher gas consumption; therefore it is necessary to keep gas consumption as one of the key performance metric. In [7] the work is about training a model on blockchain; therefore accuracy is a performance metric along with gas consumption.

In contrast to that, our work is to provide a platform for AI as a Service. By design the computation is off chained, in order to avoid huge gas consumption for training of the model. In our design the training of the model occurs in the Computation Layer which is deployed as a different service. The effective parameters for these types of services are execution time and train data size.

Moreover, the platform is designed to be deployed as a cloud service. Therefore, parameters like availability and execution time for a service can be considered as a useful indicator of performance. A service in this case is training of a model on a given size of dataset. Therefore, execution time for a given size of data set seem to be the best way to evaluate the performance of the platform. Therefore, we considered relationship between execution time and train data size to be the key performance indicator for the proposed platform. We explain the two indicators as follows:

- *Execution Time:* With respect to python, it can be defined as the time duration to the execution of a code snippet. Whereas in case of smart contract, it can be defined as the duration between the transaction conformation and its execution in the blockchain network.
- *Train Data Size:* Refers to the number of data records present in the training data. To maintain the uniformity between all the test cases, the test train classifier from sklearn is used.

3.3 DISCUSSION

The x-axis of the graph represents the size of the test dataset; y-axis is the execution time in milliseconds. The greater the execution time slower the model. Therefore, it is desirable to have minimum execution time for better efficiency of the model.

From the Fig.4, it can be said that as the test data size increases, the execution times increases linearly. The average execution time of the River implementation is 2.50 milliseconds (ms) with standard deviation of ±0.8ms, whereas the execution time for Solidity it is 28,738.7ms with standard deviation of ±10,467ms. When compared to the Solidity implementation, the River implementation is faster with magnitude of more than 10000x. From the above-mentioned factors, it can be said that the River (off chained) implementation is better than Solidity (on chain) for implementing the models on the proposed platform.

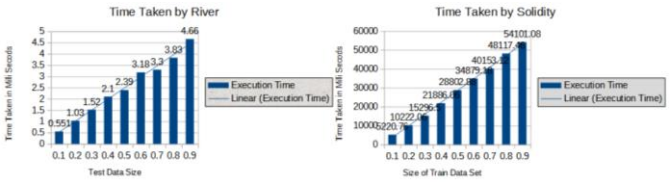

Fig.4. Comparison of Execution Time Between River and Solidity

4. CONCLUSION

In this paper, we discussed how blockchain technology in combination with the compute platform can be useful for collaborative data collection platforms and AI as a service Blockchain is used as an identity management and financial transaction platform. Also, the framework proposes certain measures to ensure the problem of the data storage as it is off chained to the cloud. In order to offer an efficient platform, various measures like the hot swap of the model, and cryptographic authentication were suggested.

Future work can be done in the following lines, the reduction of centralization by using decentralized storage solutions like Orbit DB or IPFS instead of MySQL. In order to establish the resilience of the design complex machine learning algorithms like Neural Networks, CNNs and LSTM can be implemented. In order to enhance the efficiency of the blockchain layer various different networks like Avalanche, Polkadot, Polygon can be used with the help of different suites like Truffle, Foundry, Hard Hat.

REFERENCES

[1] Parsaeefard, Saeedeh and Tabrizian, Iman and Leon-Garcia, "*Artificial Intelligence as a Services (AI-aaS) on Software-Defined Infrastructure*", Cotton Ginners' Handbook, 2019.

[2] Overview of Amazon Web Services, Available at https://docs.aws.amazon.com/whitepapers/latest/aws-overview/introduction.html, Accessed in 2022.

[3] Practitioners Guide to MLOps: A Framework for Continuous Delivery and Automation of Machine Learning, Available at https://services.google.com/fh/files/misc/practitioners_guide_to_mlops_whitepaper.pdf, Accessed in 2021.

[4] Azure Arc-enabled Machine Learning, Available at https://azure.microsoft.com/en-gb/resources/azure-arc-enabled-machine-learning-white-paper/, Accessed in 2022.

[5] A. Gepperth and Barbara Hammer, "Incremental Learning Algorithms and Applications", *Proceedings of European Symposium on Artificial Neural Networks*, pp. 1-13, 2016.

[6] D. Justin and B.W. Harris, "Decentralized and Collaborative Ai on Blockchain", *Proceedings of IEEE International Conference on Blockchain*, pp. 1-7, 2019.

[7] J.D. Harris, "Analysis of Models for Decentralized and Collaborative AI on Blockchain", *Proceedings of IEEE International Conference on Blockchain*, pp. 1-7, 2020.

[8] Xuhui Chen, "When Machine Learning Meets Blockchain: A Decentralized, Privacy-Preserving and Secure Design", *Proceedings of IEEE International Conference on Big Data*, pp. 1-13, 2018.

[9] Nenad Petrovic, "Model-Driven Approach to Blockchain-Enabled MLOps", *Proceedings of 9th IEEE International Conference on IcETRAN*, pp. 1-6, 2022.

[10] A. Marcelletti and Andrea Morichetta, "Exploring the Benefits of Blockchain Technology for MLOps Pipeline", *Proceedings of IEEE International Conference on Foundations of Consensus and Distributed Ledgers*, pp. 13-17, 2022.

[11] D.C. Nguyen, "Federated Learning meets Blockchain in Edge Computing: Opportunities and Challenges", *IEEE Internet of Things Journal*, Vol. 8, No. 16, pp. 12806-12825, 2021.

[12] Shiva Raj and Jinho Choi, "Federated Learning with Blockchain for Autonomous Vehicles: Analysis and Design Challenges", *IEEE Transactions on Communications*, Vol. 68, No. 8, pp. 4734-4746, 2020.

[13] H. Kim, "Blockchained on-Device Federated Learning", *IEEE Communications Letters*, Vol. 24, No. 6, pp. 1279-1283, 2019.

[14] System Haber and W. Scott Stornetta, "How to Time Stamp A Digital Document", *Proceedings of International Conference on the Theory and Application of Cryptography*, pp. 1-6, 1990.

[15] Satoshi Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash", Available at https://www.ussc.gov/sites/default/files/pdf/training/annual-national-training-seminar/2018/Emerging_Tech_Bitcoin_Crypto.pdf, Accessed in 2009.

[16] C. Zhang, Cangshuai Wu and Xinyi Wang, "Overview of Blockchain Consensus Mechanism", *Proceedings of International Conference on Big Data Engineering*, pp. 1-13, 2020.

[17] Liudmila Zavolokina, Noah Zani and Gerhard Schwabe, "Why should I Trust a Blockchain Platform? Designing for Trust in the Digital Car Dossier", *Proceedings of International Conference on Design Science Research in Information Systems and Technology*, pp. 1-13, 2019.

[18] Vitalik Buterin, "A Next-Generation Smart Contract and Decentralized Application Platform", Available at https://blockchainlab.com/pdf/Ethereum_white_paper-a_next_generation_smart_contract_and_decentralized_application_platform-vitalik-buterin.pdf, Accessed in 2014.

[19] Markus Schaffer and Gernot Salzer. "Performance and Scalability of Private Ethereum Blockchains", *Proceedings of International Conference on Business Process Management*, pp. 1-7, 2019.

[20] Jasvant Mandloi and Pratosh Bansal, "An Empirical Review on Blockchain Smart Contracts: Application and Challenges in Implementation", *International Journal of Computer Networks and Applications*, Vol. 7, No. 2, pp. 43-61, 2020.

[21] M. Wohrer and Uwe Zdun, "Smart Contracts: Security Patterns in the Ethereum Ecosystem and Solidity", *Proceedings of International Workshop on Blockchain Oriented Software Engineering*, pp. 43-56, 2018.

[22] Alexander Gepperth and Barbara Hammer, "Incremental Learning Algorithms and Applications", *Proceedings of European Symposium on Artificial Neural Networks*, pp. 1-5, 2016.

[23] Y. Liu and H. Song, "Class-Incremental Learning for Wireless Device Identification in IoT", *IEEE Internet of Things*, Vol. 8, No, 23, pp. 17227-17235, 2021.
[24] T. Li, S. Fong, R.C. Millham, J. Fiaidhi and S. Mohammed, "Fast Incremental Learning with Swarm Decision Table and Stochastic Feature Selection in an IoT Extreme Automation Environment", *IT Professional*, Vol. 21, No. 2, pp. 14-26, 2019.
[25] Jacob Montiel, Talel Abdessalem and Albert Bifet, "River: Machine Learning for Streaming Data in Python", *Proceedings of International Conference on Machine Learning*, pp.1-8, 2020.
[26] Nakhoon Choi and Heeyoul Kim, "A Blockchain-Based User Authentication Model using MetaMask", *Journal of Internet Computing and Services*, Vol. 20, No. 6, pp. 119-127, 2019.
[27] Svelte-Cybernetically Enhanced Web Apps, Available at https://svelte.dev/, Accessed in 2022.
[28] Mattias Levlin, "DOM Benchmark Comparison of the Front-End JavaScript Frameworks React", Angular, Vue, and Svelte", Available at https://www.doria.fi/handle/10024/177433, Accessed in 2020.
[29] Mufid Mohammad Robihul, "Design an MVC Model using Python for Flask Framework Development", *Proceedings of IEEE International Symposium on Electronics*, pp. 1-5, 2019.
[30] Andrea Perrichon-Chretien and Nicolas Herbaut. "Saiaas: A Blockchain-based Solution for Secure Artificial Intelligence as-a-Service", *Proceedings of International Conference on Deep Learning, Big Data and Blockchain*, pp. 1-7, 2022.
[31] Open Zepplin, Available at https://github.com/OpenZeppelin, Accessed in 2022.
[32] S. Moreschini, "MLOps for Evolvable AI Intensive Software Systems", Proceedings IEEE International Conference on Software Analysis, Evolution and Reengineering, pp. 1-13, 2022.
[33] AWS IAM UserGuide, Available at https://docs.aws.amazon.com/IAM/latest/UserGuide/iam-ug.pdf, Accessed in 2021.
[34] Elie Kapengut and Bruce Mizrach, "An Event Study of the Ethereum Transition to Proof-of-Stake", *Proceedings of European Symposium on Artificial Neural Networks*, pp. 1-8, 2022.
[35] M. Harikrishnan and K.V. Lakshmy, "Secure Digital Service Payments using Zero Knowledge Proof in Distributed Network", *Proceedings of International Conference on Advanced Computing and Communication Systems*, pp. 1-8, 2019.
[36] Salem Alqahtani and Murat Demirbas, "Bottlenecks in Blockchain Consensus Protocols", *Proceedings of IEEE International Conference on Omni-Layer Intelligent Systems*, pp. 1-8, 2021.