Blockchain as a Service: An Autonomous, Privacy Preserving, Decentralized Architecture for Deep Learning

Gihan J. Mendis, Moein Sabounchi, Jin Wei, *Member, IEEE* and Rigoberto Roche'

***Abstract*—Deep learning algorithms have recently gained attention due to their inherent capabilities and the application opportunities that they provide. Two of the main reasons for the success of deep learning methods are the availability of processing power and *big data*. Both of these two are expensive and rare commodities that present limitations to the usage and implementation of deep learning. Decentralization of the processing and data is one of the most prevalent solutions for these issues. This paper proposes a cooperative decentralized deep learning architecture. The contributors can train deep learning models with private data and share them to the cooperative data-driven applications initiated elsewhere. Shared models are fused together to obtain a better model. In this work, the contributors can both design their own models or train the models provided by the initiator. In order to utilize an efficient decentralized learning algorithm, blockchain technology is incorporated as a method of creating an incentive-compatible market. In the proposed method, Ethereum blockchain's scripting capabilities are employed to devise a decentralized deep learning mechanism, which provides much higher, collective processing power and grants access to large amounts of data, which would be otherwise inaccessible. The technical description of the mechanism is described and the simulation results are presented.**

***Index Terms*—Deep learning, Privacy preserving learning, Decentralize learning, Blockchain**

I. Introduction

Deep learning methods are a class of emerging Artificial Neural Network (ANN)-based machine learning techniques with deep and hierarchical structures, which are capable of extracting complex features from raw data and thus they are effective in completing nontrivial tasks [1]. Various deep learning techniques have been proposed and implemented such as deep belief networks (DBNs), deep convolutional neural networks (DCNNs), deep recurrent neural networks (DRNNs) [2]–[5]. Deep learning techniques have been widely used in various application areas such as: speech recognition, natural language processing (NLP), audio and music processing, image recognition, and machine vision [6]–[14]. Thanks to algorithmic breakthroughs, these techniques outperform other earlier machine learning techniques in many situations where large amounts of data are available [1], [15], [16].

Although the benefits of deep learning are undeniable, its requirement of having a collected big data data-set, introduces a risk in privacy and confidentiality. Large, web-based business organizations and end-user electronic/software production companies contribute to the research of deep learning and implementations of deep learning methods to improve their products and services. Since the success of deep learning largely depends on the availability of data, these organizations gather and preserve end-user data such as photos, speech, video, interaction and preference data in centralized collections for future use. End-users do not have control over how the collected data will be used and the organization or other third parties may be able to exploit privacy-sensitive information. As an example, the online movie rental service Netflix launched the competition "Netflix Prize" to find an algorithm to predict user rating on a movie. In doing so, the company released "Netflix Prize" data-set that contains anonymous movie ratings of nearly 500,000 subscribers of Netflix [17]. In [18], Narayanan *et al.* showed that it is possible to identify the subscribers' records in a data-set and uncover their apparent political preferences and other potentially sensitive information. With the rise of awareness for data privacy, end-users have become reluctant to share their data. Also, in another perspective, user data are becoming a valuable asset. Therefore, it is meaningful to develop a reliable infrastructure in which end-users or data creators are able to securely transfer data in a privacy preserving manner and get compensated for their contributions. Furthermore, it will be immensely beneficial for the progress of research and application development, if there are frameworks that multiple, distrustful parties can cooperate and contribute, with their private data and processing power while being able to preserve their privacy. Additionally, in some domains, such as the healthcare sector, federal and civil service offices, there is an abundance of valuable data which due to privacy laws and regulations, cannot be shared with third parties. Inspired by the demands on data acquisition and privacy preservation, in this paper we propose a blockchain-empowered, privacy preserving, decentralized deep learning infrastructure, in which the distrusted parties are able to share their domain-specific data and computing solutions.

Recently, blockchain technology has paved its way into different fields, many outside the realm of finance. The authors of [19] have investigated, the possibility of using a blockchain as a replacement for traditional software connectors and shared data storage systems. Software structure is generally comprised of software blocks that are connected together using software connectors (middleware). The blockchain incorporation can guaranty the security and optimize the transaction rate of data transfer and thus make the interactions faster and more secure. Dennis and Owen in [20] have developed a reputation generation system that can be used in the blockchain-enabled systems. The have solved the issue of personal bias from

reputation evaluation by including hashing functions in the rating processes. Watanabe *et al.* in [21] have proposed a new consensus protocol for the verification of transactions in the blockchain. They have proposed to record a trail of the consensus into a transaction as the proof of a party's consent. In their model, the recipient of each transaction, generates a whole new transaction proving his consent. Then a chain of transactions is formed between contractors, where the last contractor will be linked to the first contractor, to form a closed loop. To secure this process, a key-pair is assigned to the contract data and each e-signature between two contractors.

In [22], Shokri *et al.* proposed a privacy preserving deep learning mechanism with secure multi-party computations to update the single initial deep learning model. Federated learning is one machine learning algoirthm where the goal is to train a high-quality centralized model while having a distributed training data-set [23]–[25]. In these methods, the deep learning model is centralized while multiple parties contribute to the model training in a manner that guaranties the privacy-preservation. In [26], Kokkinos *et al.* discussed distributed privacy preserving data mining that can build several neural networks while a confidence ratio-based method is used for selecting the best model.

These established methods are effective in accomplishing privacy preserving data processing. Practically, the success of decentralized privacy preserving data processing, requires voluntary cooperation of multiple distrustful but rational parties and financial compensation can be an effective incentive for the cooperation [27], [28]. However, there is no effective solution on how to qualitatively evaluate and financially incentivize the contributions of the individuals; local computing units and data assets. To address this challenge, we exploit the Ethereum blockchain and decentralized deep learning techniques to develop an autonomous privacy preserving cooperative deep learning platform [29]–[31]. Our platform mainly consists of application initiators, computing contributors, and verification contributors. The application initiators announce their data-driven applications and start the associated cooperative deep learning. Multiple distrustful computing parties participate in a cooperation, using their local data assets to develop and announce the appropriate computing models, for the given data-driven applications. The announced models are evaluated by decentralized verification contributors and are rewarded financially according to the evaluations provided the verification contributors. By doing so, various deep learning models developed by multiple distrustful computing contributors are traded and fused via secure and peer-to-peer transactions. Additionally, the privacy preservation of the local data asset is also achieved.

The proposed method requires an effective fusion strategy of trained deep learning models. Fusion of multiple learning models is an active area of research and fusion strategies in literature can be divided into two main categories: (1) Late fusion, which consists of predicting the labels based on the labels given by each learned model, and (2) Early fusion, which consists of taking the features' vectors given by each of the learned models as the inputs and learning a classifier on top of them. For most practical applications, late fusion has proven to be a cheap and effective solution [32]–[34]. However, in [35], Neverova *et al.*indicates that early fusion may be a more optimal way to combine learned models. In this work, we consider the early fusion approach. However, the feature vectors we consider for fusion are features with the highest level of abstraction. We consider two strategies for model fusion, one is to use a fully connected structure with a single hidden layer to map concatenated features to labels, and the other is to implement gradual fusion using the ModDrop algorithm introduced by Neverova *et al.*, in [35]. The authors would like to claim that the technologies presented in this paper has been include in a provisional patent [36].

The next section describes the problem setting for privacy preserving decentralized deep learning. Section III details our proposed blockchain-empowered cooperative deep learning mechanism. Simulation and results are shown in Section IV followed by final remarks and the conclusion in Section V.

II. Problem Setting

Two of the main factors for the thriving of deep learning are: (1) The availability of sizable data-sets with generalized distributions, commonly known as big data, and (2) The availability of the computational power to process this big data, mainly in the form of large-scale GPU clusters. Because of this, most profitable parties in the field of deep learning are large organizations, which hold both valuable big data and sufficient computational power to process it. As illustrated in Fig. 1(a), these large organizations collect data from data contributors to advance the capabilities of their deep learning techniques. One of the biggest challenges for creating large-scale data-sets via data acquisition, from multiple parties, is the issue of privacy and the related concerns. Potential data providers may not get motivated to share the data because of the high potential for data privacy violations. Additionally, collecting tremendous raw data from multiple parties results in a huge demand on communication bandwidth and a dramatically increased attack surface. Furthermore, a large amount of computational power is required by the central server to process the collected big data.

One solution is the implementation of distributed learning architectures. As shown in Fig. 1(b), in distributed learning, rather than collecting and processing data in a single central server, data processing is distributed partially to the individual data providers. By doing so, the distributed learning is implemented in such a way, that the computing contributors process their local data by training their own deep learning model and then share the trained model with a central controlling agent. Since the data are not shared, we can say that the data privacy is preserved in this architecture. Also, since only the parameter set of the deep learning model is shared, the demand for bandwidth is greatly reduced. Furthermore, the deep learning models are trained in distributed locations with smaller sets of data, and thus the computational power required by the individual computing contributors is much lower, compared with that of a central server. However, in this solution, the deep learning architecture is fully controlled by a centralized agent. Therefore, the deep learning architecture in

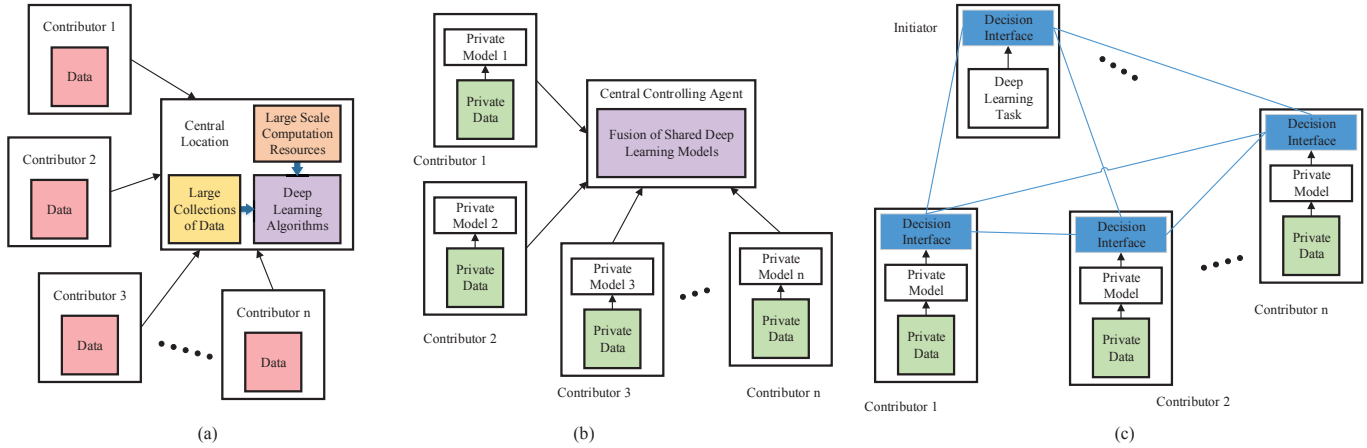

Fig. 1. (a) Centralized deep learning architecture where data are collected to centralized server with high processing and storage capability; (b) Distributed deep learning architecture where partial of training is distributed to the data contributors and the training process is fully controlled by a central controlling agent; (c) Cooperative and decentralized deep learning architecture with no central agents facilitated by blockchain service infrastructure.

Fig. 1(b) is susceptible to a single point of failure. To address this issue, the individual computing contributors should have control on their computation models. To mitigate this, we improve the distributed deep learning architecture presented in Fig. 1(b) and achieve the cooperative and decentralized deep learning architecture, which is the architecture that will be proposed in this paper, as shown in Fig. 1(c). In this architecture, the untrustful, computing contributors have full control on their own deep learning models. Additionally, the individual contributors are able to participate or leave the computing architecture, without disturbing the functionality and efficiency of the overall learning process. Furthermore, the participation of the computing contributors is motivated by financial rewards that they shall recieve according to the value of their contribution. To achieve these objectives, we exploit the Ethereum blockchain and design the smart contract to secure the peer-to-peer transactions between the multiple distrustable parties to enable the autonomous cooperative deep learning [31].

III. Blockchain-Empowered Cooperative Deep Learning Mechanism

The overview of our proposed blockchain-empowered, cooperative and decentralized deep learning mechanism is illustrated in Fig. 2. As shown in Fig. 2, in our proposed mechanism, the individual participants can act as application initiators, computing contributors, or verification contributors. Application initiators that announce the data-driven applications are responsible for defining the computing tasks, such as the properties of input data and the expected output of the tasks. They provide a sample set of data for verification, defining expected accuracies, and stating the financial compensation commitments. Optionally, the initiators can recommend the structure of the computing model. Computing contributors that own a certain local data asset will design and train an appropriate machine learning model for the given task and publish it to the verification contributors after the model is trained sufficiently. If the initiators recommend

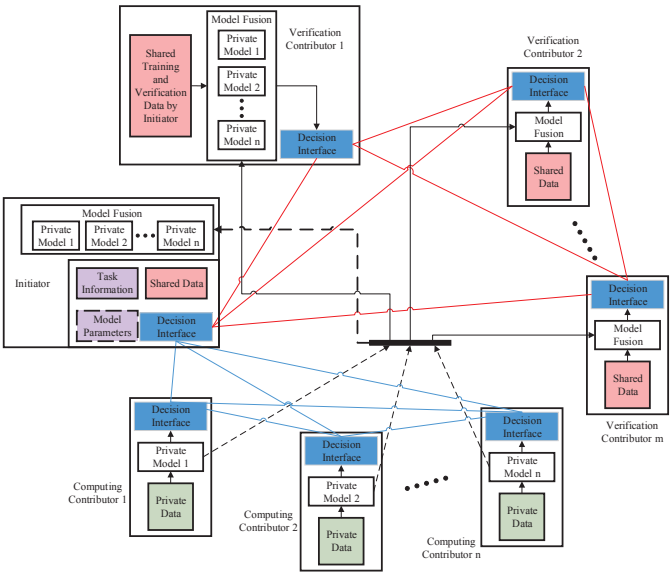

Fig. 2. Overview of a cooperative deep learning task on our proposed blockchain-empowered cooperative deep learning mechanism. Where each type of line indicates functionality as follows: solid black arrow: transfer of deep learning model, dashed black arrow: conditional transfer of deep learning model, blue lines: decentralized deep learning architecture facilitated by blockchain service infrastructure, and red lines: decentralized verification architecture facilitated by blockchain service infrastructure.

certain computing model structures, computing contributors optimize the recommended structure instead of training a new model. After receiving the computing models announced by the computing contributors, the verification contributors are in charge of evaluating the contribution of the computing contributors. They justify whether the overall performance of the data-driven applications, announced by the associated initiator, is improved by integrating their models. In the proposed architecture, the verification contributors involved in the process in a passive way, by providing the hardware resources to perform verification tasks in a random and decentralized

manner. Majority voting amongst the verification contributors is used to determine the contribution of the corresponding computing contributors and get compensated from initiators. Additionally, the verification contributors also get compensated from initiators for their effort. According to the smart contract in Ethereum blockchain, after paying the financial compensation to the computing and verification contributors, the application initiators gain the access to the rewarded computing models. Then they can use the same strategy used by the verification contributors to fuse the individual computing model and achieve an integrated deep learning model. The architecture is considered asynchronous and adaptive since the computing and verification contributors can leave or join the task at their convenience.

A. Blockchain as a Service

In our mechanism, the interactions between the participants, including application initiators, computing contributors, and verification contributors, for the cooperative and decentralized deep learning, are realized via the transactions deployed on the Ethereum Blockchain-based platform. The Truffle development suite, Oraclize API query services, and the Web3.js API [37]–[39], are the essential development tools deployed to build our platform. Additionally, the transactions between the participants of the cooperative deep learning are regulated by smart contracts written in solidity language, as shown in Fig. 3. The smart contracts mainly consist of two events. In the first event, the application initiator starts a data-driven task by specifying the details of the task, such as the properties of the input data, the expected outputs of the tasks, and financial compensations. The computing contributors achieve the appropriate computing models via off-chain training for the given tasks. They store the parameters of the models using InterPlanetary File System (IPFS) decentralized storage [40]. The achieved model is anounced by importing the returned hash values to the Decentralized Application (DApp), where the IPFS hash values are shown in the web application that uses smart contracts as a back-end [41]. After receiving the computing models, the verification contributors begin the off-chain evaluation and report their evaluation results via the web application. In the second event of the smart contract, the individual evaluations achieved by the verification contributors are fetched by the smart contract via the Web3.js API and then checked against a threshold, called minimum acceptable fitness rate (MAFR), specified by the initiator in the smart contract. If the majority of the evaluations exceeds the MAFR, the payable function monogamously transmits the Ethers to the accounts of the associated computing contributors via the Ethereum blockchain. Additionally, all the verification contributors obtain the financial compensations for their effort.

B. Cooperative and Decentralized Deep Learning

As illustrated in Fig. 2, the success of cooperative and decentralized deep learning requires an effective mechanism to integrate the various justified computing models, provided by the individual computing contributors to achieve a high performing functional computing model for the targeted task.

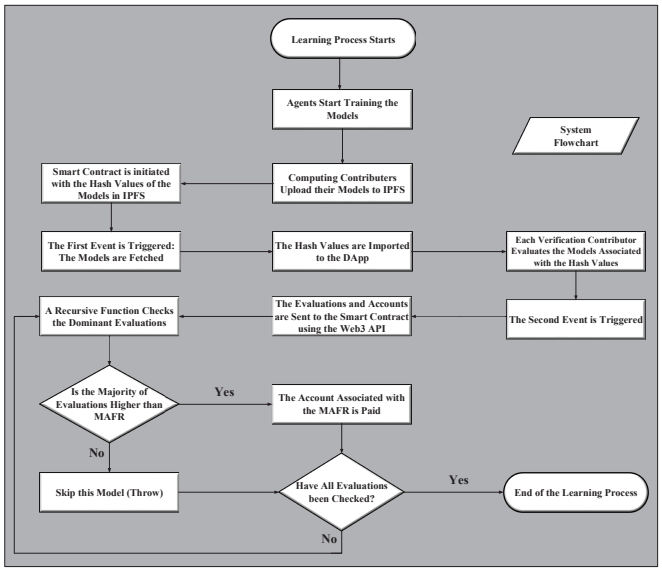

Fig. 3. The flowchart of the learning process.

Furthermore, in the strategy for fusing the individual models, it is necessary to treat the computing models, provided by the computing contributors, as separate entities, to ensure the fused structure is dynamic.

In this paper, a cooperative and decentralized deep learning mechanism is developed, under the following three assumptions. (1) The data-driven task considered is a classification task. (2) The computing contributors are rational and only use data that are consistent with the properties specified by the initiators in the smart contract. (3) All the computing contributors develop their computing models according to the given tasks. Figure 4 shows the structure of our strategy to integrate the computing models. The model-fusion strategy consists of two layers, each of which is achieved by calculating a weighted sum of the corresponding values of the previous layer. As shown in Fig. 4, the upper layer feature vectors from n shared models are concatenated to form a concatenated feature $\boldsymbol{fc}$, calculated based on $\mathbf{f_i}$, that is the upper layer feature vector of the ith model. A hidden layer $\boldsymbol{h}$ with a size of $\sum_{i=1}^{n} \mid l_i \mid$, where $\mid l_i \mid$ is the number of labeled classes in the ith model, is calculated as the weighted sum of the concatenated feature vector. We consider two strategies for learning the optimum values for weight matrices $\mathbf{A}$ and $\mathbf{B}$ in Fig. 4 as follows:

1) Strategy I: The hidden layer $\mathbf{h}$ and output layer $\mathbf{y}$ are calculated based on the fully connected weights $\mathbf{A}$ and $\mathbf{B}$ with no constraints, as shown in Eqs. (1) and (2), respectively. Weight matrices $\mathbf{A}$ and $\mathbf{B}$ are initiated randomly. The back-propagation algorithm is used to calculate the optimum values for the weight matrices $\mathbf{A}$ and $\mathbf{B}$ [42].

$$h_j = \sum_{i=1}^{|\mathbf{fc}|} A_{ij}^T \cdot fc_i \tag{1}$$

where $\mathbf{A}$ is a weight matrix and $\mathbf{fc}$ is the concatenated feature vector. The output of the final layer, $\mathbf{y}$, consists of probabilities

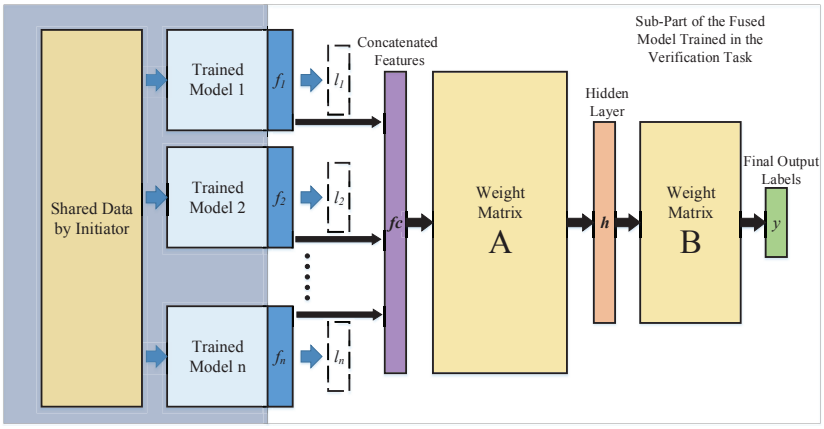

Fig. 4. The graph structure of the model fusion method.

calculated via a softmax function, which is shown as follows:

$$y_j = \frac{exp(\sum_{i=1}^{|\mathbf{h}|} B_{ij}^T \cdot h_i)}{\sum_{k=1}^{d} exp(\sum_{i=1}^{|\mathbf{h}|} B_{ik}^T \cdot h_i)} \tag{2}$$

where $\mathbf{B}$ is a weight matrix.

2) Strategy II: In this strategy, a gradual model-fusion is proposed, in which the correlation between the different computing models are learned, while the uniqueness of each computing model is preserved. The weight matrix $\mathbf{A}$ is initialized as a concatenated matrix formulated in Eq. (3), where a diagonal weight matrix $\mathbf{W_i}$ has dimensions $(\mid \mathbf{f}_i \mid, \mid \mathbf{l}_i \mid)$ and is initialized randomly. The weight matrix $\mathbf{B}$ is initialized with the concatenation of the identity matrices, as shown in Eq. (4), where n is the number of deep learning models fused and d denotes the number of class labels.

$$\mathbf{A}_{init} = \begin{bmatrix} \mathbf{W_1} & \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{W_2} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{W_3} & \dots & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & \mathbf{W_n} \end{bmatrix} \tag{3}$$

$$\mathbf{B}_{init} = \begin{bmatrix} w_{111}=1 & 0 & 0 & \dots & 0 \\ 0 & w_{122}=1 & 0 & \dots & 0 \\ 0 & 0 & w_{133}=1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & w_{1dd}=1 \\ w_{211}=1 & 0 & 0 & \dots & 0 \\ 0 & w_{222}=1 & 0 & \dots & 0 \\ 0 & 0 & w_{233}=1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & w_{2dd}=1 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ w_{n11}=1 & 0 & 0 & \dots & 0 \\ 0 & w_{n22}=1 & 0 & \dots & 0 \\ 0 & 0 & w_{n33}=1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & w_{ndd}=1 \end{bmatrix} \tag{4}$$

Weight matrices $\mathbf{A}$ and $\mathbf{B}$ are trained by using the back-propagation algorithm. In the initial iterations of training, only the diagonal non-zeros weights, shown in Eqs. (3) and (4), are updated. In the later iterations of training, all of the weights in $\mathbf{A}$ and $\mathbf{B}$ are updated. This two-stage training strategy are formulated in Eq. (5) and (6), where the parameter γ is set as 0 in the intial stage and as 1 in the final stage.

$$h_j = \sum_{i=1, A_{ij} \in \bar{\mathbf{W}}}^{|\mathbf{fc}|} A_{ij}^T \cdot fc_i + \gamma \sum_{i=1, A_{ij} \notin \bar{\mathbf{W}}}^{|\mathbf{fc}|} A_{ij}^T \cdot fc_i \tag{5}$$

Where $\bar{\mathbf{W}} = \mathbf{W_p} \leftarrow \sum_{k=1}^{p-1} \mid \mathbf{l_k} \mid < i \leq \sum_{k=1}^{p} \mid \mathbf{l_k} \mid$.

$$y_j = \frac{exp(\sum_{i=1, B_{ij}=w_{jpp}}^{|\mathbf{h}|} B_{ij}^T \cdot h_i + \gamma \sum_{i=1, B_{ij} \neq w_{jpp}}^{|\mathbf{h}|} B_{ij}^T \cdot h_i))}{\sum_{k=1}^{d} exp(\sum_{i=1, B_{ik}=w_{ipp}}^{|\mathbf{h}|} B_{ik}^T \cdot h_i + \gamma \sum_{i=1, B_{ik} \neq w_{ipp}}^{|\mathbf{h}|} B_{ik}^T \cdot h_i)} \tag{6}$$

Where $p \in \{1, 2, 3 \dots, d\}$, and d is the number of labeled classes.

IV. Simulation and Results

In this section, the performance of our blockchain-based cooperative deep learning architecture is evaluated by considering two scenarios. In each scenario, the application initiator starts a data-driven tasks. The authors would like to mention that, because of the significant latency for the mining process for all the transactions in both scenarios, the main Ethereum network and Testnest, we decided to use a local private chain to develop the test system for our work. To do so, we use our oracle in the private chain, in which we use Ethereum Bridge in broadcast mode [43] and create an additional account as the Oraclize API linking into our smart contract. Additionally, the Truffle development suite is used for compiling and deploying the DApp developed for our work. One example of the DApp User Interface (UI) used by the verification contributors is shown in Fig. 5.

A. Scenario I

In this scenario, we consider that the computing contributors develop their own deep learning models, by using the local data for the given task. After training the deep learning models sufficiently, the computing contributors publish the

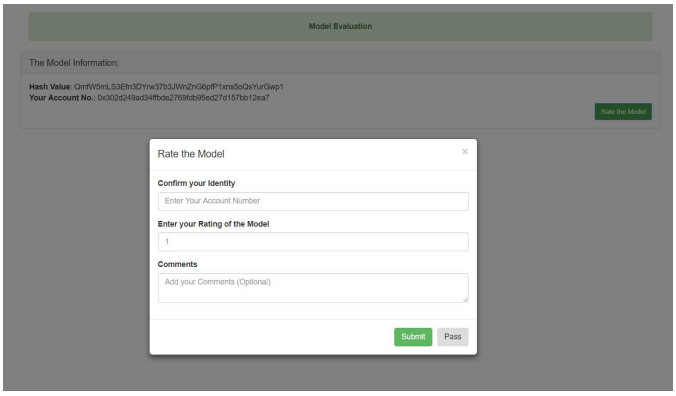

Fig. 5. The evaluation decentralized application UI.

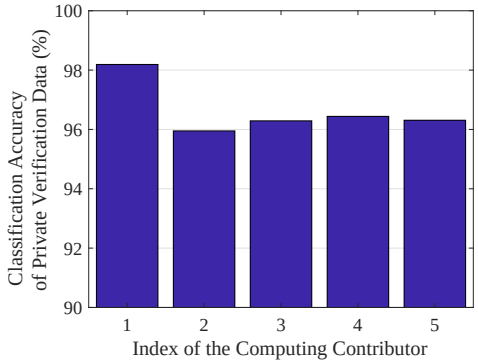

Fig. 6. Verification accuracy obtained by each by using their local verification data.

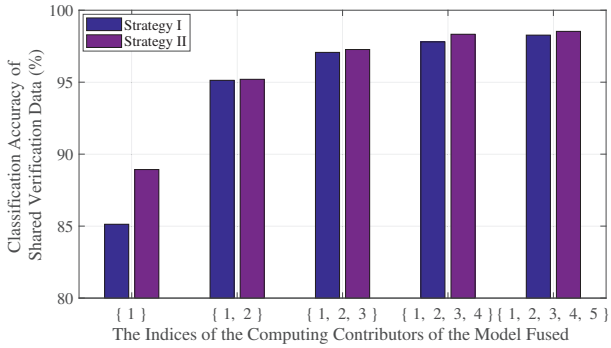

Fig. 7. Comparison of average evaluation accuracy reported by the verification contributors by using Strategies I and II versus the indices of the computing contributors of the fused models.

models to the verification contributors. Based on the evaluations reported by the verification contributors, the initiator provides the computing contributors with financial rewards and accesses to the computing models, shared by the rewarded computing contributors. In the simulations, the performance of the cooperative deep learning mechanism is evaluated by considering the two strategies introduced in Section III-B.

In the simulation, a cooperative deep learning task is considered for classifying the 10 image classes provided by the MNIST handwritten digit database [44]. It is assumed that the deep learning models used by the individual computing contributors are 3-layer convolutional neural networks (CNNs)-based classifiers. Table I summarizes the structures of the CNN models and the training parameters used by the individual computing contributors. The models are designed as TensorFlow graphs [45]. The serialized graph files are shared with the verification contributors through IPFS. In the simulation, it is assumed that there are 5 computing contributors, each of which owns 500 local training data, 200 local testing data and 500 verification data. The initiator provides a set of data to the verification contributors for a verification task. All verification contributors receive the same data-set, shared by the initiator. The distribution of data amongst the contributors is summarized in Table II. The accuracy of the CNN models developed by the individual computing contributors for classifying their local verification data is shown in Fig. 6. We assume that all of the five computing contributors publish their models to the verification contributors.

The average evaluation results, obtained by the verification contributors, by using Strategies I and II, respectively, are shown in Fig. 7. In the simulation, we assume that the verification contributors receive the computing models announced by Computing Contributors 1 to 5 in an ascending order. When a verification contributor receives a new computing model, it fuses the model with currently available models. From Fig. 7, we can observe that the accuracy of verification data increases as the number of fused models increases for both strategies, and thus, all the models are considered to have contributions to the given task, by using both strategies. Therefore, all these five computing contributors get financial rewards from the initiator, according to the smart contract. From Fig. 7, we can also get that the fused model accuracy achieved by using Strategy II is marginally better than that obtained by using Strategy I.

B. Scenario II

In this scenario, the initiator recommends the deep learning model for training. The computing contributors participate in cooperative deep learning by training the recommended model. After reaching a predetermined threshold accuracy, the computing model is automatically shared with the verification contributors, whose role remains the same as that in the scenario I. In the simulation, it is considered that there are 5 computing contributors. Data distribution amongst the computing and verification contributors is detailed in Table II. The deep learning model recommended by the initiator in this scenario is considered to be similar to the model used by Computing Contributor 1 in Table I.

The accuracy of the CNN models, developed by the individual computing contributors, for classifying their local verification data, is shown in Fig. 8. We assume that all of the five computing contributors publish their models to the verification contributors. The average evaluation results obtained by the verification contributors by using Strategy II, are shown in Fig. 9. In the simulation, we assume the verification contributors receive the computing models announced by Computing Contributors 1 to 5 in an ascending order. When

TABLE I
PARAMETERS OF THE CNNS

CNN parameters					
	Computing Contributor 1	Computing Contributor 2	Computing Contributor 3	Computing Contributor 4	Computing Contributor 5
Inputs	28x28 images				
Convolution layer 1	32 5x5 kernels	64 5x5 kernels	32 5x5 kernels	16 10x10 kernels	16 10x10 kernels
Pooling layer 1	2x2 maximum pooling	2x2 maximum pooling	2x2 maximum pooling	2x2 maximum pooling	2x2 maximum pooling
Convolution layer 2	16 5x5 kernels	32 5x5 kernels	32 10x10 kernels	16 10x10 kernels	16 5x5 kernels
Pooling layer 2	2x2 maximum pooling	2x2 maximum pooling	2x2 maximum pooling	2x2 maximum pooling	2x2 maximum pooling
Convolution layer 3	8 2x2 kernels	16 2x2 kernels	16 4x4 kernels	12 5x5 kernels	8 5x5 kernels
Reshaped vector (Convolution layer 3 output are flatten as a vector of size)	7x7x8=392	7x7x16=784	7x7x16=784	7x7x12=588	7x7x8=392
hidden layer	Fully connected hidden layer with size 500				
Outout	10 labels with softmax activation				
Training method	Adam Optimizer				
Learning rate	0.0001				
Maximum number of iterations	100000				

TABLE II
SUMMARY OF DATA DISTRIBUTION AMONGST THE COMPUTING CONTRIBUTORS

Contributor	Set of Labels	No. Training Data	No. Testing Data	No. Verification Data
Verifiers	{0, 1, 2, 3, 4, 5, 6, 7, 8, 9}	2000	500	1000
Computing Contributor 1	{0, 1, 2, 3, 4}	500	200	500
Computing Contributor 2	{0, 1, 2, 8, 9}	500	200	500
Computing Contributor 3	{0, 6, 7, 8, 9}	500	200	500
Computing Contributor 4	{0, 1, 5, 7, 8, 9}	500	200	500
Computing Contributor 5	{5, 6, 7, 8, 9}	500	200	500

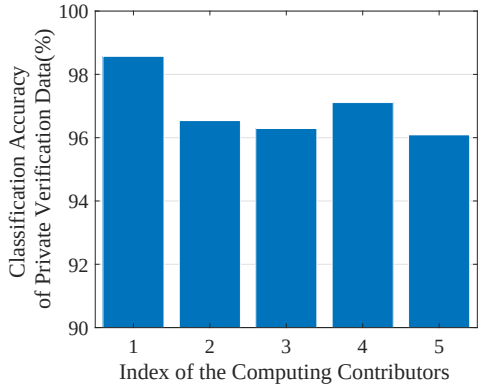

Fig. 8. Verification accuracy obtained by each computing contributor by using the local verification data.

a verification contributor receives a new computing model, it fuses the model with currently available models. From Fig. 9, we can observe that the accuracy of verification data increases as the number of fused models increases. Thus, all the models are considered to have contributions to the given task, by using both strategies. Therefore, all these five computing contributors get financial rewards from the initiator according to the smart contract. Furthermore, as shown in Fig. 9, the final accuracy obtained with Strategy II is 98.6%, which is an improvement from the accuracy obtained by using strategy I 98.4%.

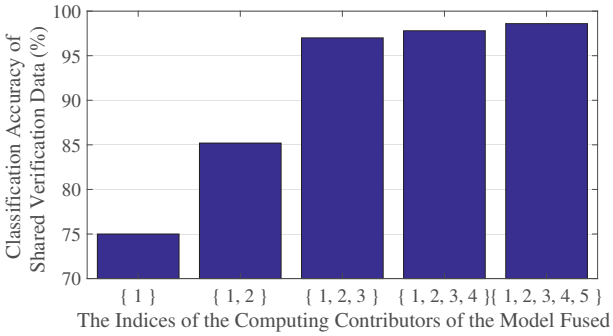

Fig. 9. Evaluation accuracy reported by the verification contributors versus the indices of the computing contributors of the fused models.

V. CONCLUSION

Availability of computation power and data are two of the main reasons for the success of deep learning in a variety of application areas. However, both acquisition of processing power and data can be expensive. Also, data-privacy is a growing concern, that discourages sharing of data. Therefore, mechanisms are required to process private data in suitable decentralized ways for deep learning. In this paper, we proposed a privacy-preserving, decentralized blockchain-based architecture for data-driven, deep learning applications. In this work, an initiator initiates a data-driven, deep-learning application. A set of computing contributors train a private deep learning model with private data for the initiated data-driven application. The trained models are shared and a set of verification contributors fuse the trained models and verify whether a shared model contributes to improving the overall accuracy of a cooperatively learned deep learning model. One contribution of the proposed method is that the computing contributors can both train a model provided by the initiator or design their own

deep learning model for the data-driven application. In order to realize the cooperative learning mechanism as an incentive compatible market, we incorporate the blockchain technology in the form of Ethereum blockchain. In the simulation results, we have shown the effectiveness of the proposed decentralized cooperative deep learning method, considering two scenarios. Computing contributors design their own convolutional neural networks (CNNs), deep learning structures in the first scenario and in the second scenario, computing contributors utilize a CNN structure provided by the initiator.

ACKNOWLEDGMENT

This research work was supported by NASA under Grant 80NSSC17K0530.

REFERENCES

[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[2] G. Hinton, S. Osindero, and Y. Teh, "A fast learning algorithm for deep belief nets," *Neural Computation*, vol. 18, pp. 1527–1554, 2006.

[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[4] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng, "Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations," in *Proceedings of the 26th Annual International Conference on Machine Learning*. ACM, 2009, pp. 609–616.

[5] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Acoustics, speech and signal processing (icassp), 2013 ieee international conference on*. IEEE, 2013, pp. 6645–6649.

[6] Y. Zhang, M. Pezeshki, P. Brakel, S. Zhang, C. L. Y. Bengio, and A. Courville, "Towards end-to-end speech recognition with deep convolutional neural networks," *arXiv preprint arXiv:1701.02720*, 2017.

[7] O. Abdel-Hamid, A. r. Mohamed, H. Jiang, L. Deng, G. Penn, and D. Yu, "Convolutional neural networks for speech recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 10, pp. 1533–1545, Oct 2014.

[8] T. Young, D. Hazarika, S. Poria, and E. Cambria, "Recent trends in deep learning based natural language processing," *arXiv preprint arXiv:1708.02709*, 2017.

[9] R. Socher, A. Perelygin, J. Y. Wu, J. Chuang, C. D. Manning, A. Y. Ng, and C. Potts, "Recursive deep models for semantic compositionality over a sentiment treebank," in *Proceedings of the conference on empirical methods in natural language processing (EMNLP)*, vol. 1631. Citeseer, 2013, p. 1642.

[10] Y.-L. Hsu, C.-P. Lin, B.-C. Lin, H.-C. Kuo, W.-H. Cheng, and M.-C. Hu, "Deepsheet: A sheet music generator based on deep learning," in *Multimedia & Expo Workshops (ICMEW), 2017 IEEE International Conference on*. IEEE, 2017, pp. 285–290.

[11] E. J. Humphrey, J. P. Bello, and Y. LeCun, "Moving beyond feature design: Deep architectures and automatic feature learning in music informatics." in *ISMIR*. Citeseer, 2012, pp. 403–408.

[12] A. Esteva, B. Kuprel, R. A. Novoa, J. Ko, S. M. Swetter, H. M. Blau, and S. Thrun, "Dermatologist-level classification of skin cancer with deep neural networks," *Nature*, vol. 542, no. 7639, p. 115, 2017.

[13] Y. Tang and R. R. Salakhutdinov, "Learning stochastic feedforward neural networks," in *Advances in Neural Information Processing Systems*, 2013, pp. 530–538.

[14] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[15] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*, 2016, vol. 1.

[16] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural networks*, vol. 61, pp. 85–117, 2015.

[17] https://www.netflixprize.com/index.html.

[18] A. Narayanan and V. Shmatikov, "Robust de-anonymization of large sparse datasets," in *Security and Privacy, 2008. SP 2008. IEEE Symposium on*. IEEE, 2008, pp. 111–125.

[19] X. Xu, C. Pautasso, L. Zhu, V. Gramoli, A. Ponomarev, A. B. Tran, and S. Chen, "The blockchain as a software connector," in *2016 13th Working IEEE/IFIP Conference on Software Architecture (WICSA)*, April 2016, pp. 182–191.

[20] R. Dennis and G. Owen, "Rep on the block: A next generation reputation system based on the blockchain," in *Internet Technology and Secured Transactions (ICITST), 2015 10th International Conference for*. IEEE, 2015, pp. 131–138.

[21] H. Watanabe, S. Fujimura, A. Nakadaira, Y. Miyazaki, A. Akutsu, and J. J. Kishigami, "Blockchain contract: A complete consensus using blockchain," in *2015 IEEE 4th Global Conference on Consumer Electronics (GCCE)*, Oct 2015, pp. 577–578.

[22] R. Shokri and V. Shmatikov, "Privacy-preserving deep learning," in *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*. ACM, 2015, pp. 1310–1321.

[23] J. Konečnỳ, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," *arXiv preprint arXiv:1610.05492*, 2016.

[24] J. Konečnỳ, H. B. McMahan, D. Ramage, and P. Richtárik, "Federated optimization: Distributed machine learning for on-device intelligence," *arXiv preprint arXiv:1610.02527*, 2016.

[25] B. McMahan and D. Ramage, "Federated learning," https://research.googleblog.com/2017/04/federated-learning-collaborative.html.

[26] Y. Kokkinos and K. G. Margaritis, "Confidence ratio affinity propagation in ensemble selection of neural network classifiers for distributed privacy-preserving data mining," *Neurocomputing*, vol. 150, pp. 513–528, 2015.

[27] P. Knez, V. L. Smith, and A. W. Williams, "Individual rationality, market rationality, and value estimation," *The American Economic Review*, vol. 75, no. 2, pp. 397–402, 1985.

[28] R. B. Myerson, "Incentive compatibility and the bargaining problem," *Econometrica: journal of the Econometric Society*, pp. 61–73, 1979.

[29] M. Swan, *Blockchain: Blueprint for a new economy*. " O'Reilly Media, Inc.", 2015.

[30] E. community, *Ethereum Blockchain*, 2013. [Online]. Available: https://www.ethereum.org/

[31] A. Kosba, A. Miller, E. Shi, Z. Wen, and C. Papamanthou, "Hawk: The blockchain model of cryptography and privacy-preserving smart contracts," in *Security and Privacy (SP), 2016 IEEE Symposium on*. IEEE, 2016, pp. 839–858.

[32] V. Vielzeuf, S. Pateux, and F. Jurie, "Temporal multimodal fusion for video emotion classification in the wild," in *Proceedings of the 19th ACM International Conference on Multimodal Interaction*. ACM, 2017, pp. 569–576.

[33] S. E. Kahou, C. Pal, X. Bouthillier, P. Froumenty, Ç. Gülçehre, R. Memisevic, P. Vincent, A. Courville, Y. Bengio, R. C. Ferrari *et al.*, "Combining modality specific deep neural networks for emotion recognition in video," in *Proceedings of the 15th ACM on International conference on multimodal interaction*. ACM, 2013, pp. 543–550.

[34] G. Ye, D. Liu, I.-H. Jhuo, and S.-F. Chang, "Robust late fusion with rank minimization," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 3021–3028.

[35] N. Neverova, C. Wolf, G. Taylor, and F. Nebout, "Moddrop: adaptive multi-modal gesture recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 8, pp. 1692–1706, 2016.

[36] USA Patent provisional 62/663,287, 2018.

[37] T. Suite, *Truffle Ethereum developement Framework*, 2014. [Online]. Available: https://github.com/trufflesuite/truffle

[38] O. D. Team, *Oraclize API*, 2015. [Online]. Available: http://www.oraclize.it/

[39] W. D. Team, *Web3 API*, 2014. [Online]. Available: https://github.com/ethereum/wiki/wiki/JavaScript-API

[40] J. Benet, "Ipfs-content addressed, versioned, p2p file system," *arXiv preprint arXiv:1407.3561*, 2014.

[41] V. Buterin *et al.*, "A next-generation smart contract and decentralized application platform," *white paper*, 2014.

[42] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[43] E. B. D. Team, *Oraclize-Ethereum Bridge*, 2016. [Online]. Available: https://github.com/oraclize/ethereum-bridge

[44] Y. LeCun, "The mnist database of handwritten digits," *http://yann. lecun. com/exdb/mnist/*, 1998.

[45] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, "Tensorflow: A system for large-scale machine learning." in *OSDI*, vol. 16, 2016, pp. 265–283.