Proof-of-Learning: a Blockchain Consensus Mechanism based on Machine Learning Competitions

Felipe Bravo-Marquez*[†], Steve Reeves[†] and Martín Ugarte[‡][§]
*Department of Computer Science, University of Chile, Chile
[†]Department of Computer Science, University of Waikato, New Zealand
[‡]Pontificia Universidad Católica, Chile
[§]Millennium Institute for Foundational Research on Data, Chile

***Abstract*—This article presents WekaCoin, a peer-to-peer cryptocurrency based on a new distributed consensus protocol called Proof-of-Learning. Proof-of-learning achieves distributed consensus by ranking machine learning systems for a given task. The aim of this protocol is to alleviate the computational waste involved in hashing-based puzzles and to create a public distributed and verifiable database of state-of-the-art machine learning models and experiments.**

***Index Terms*—distributed consensus, blockchain, machine learning, incentive mechanisms, algorand**

I. Introduction

Popular cryptocurrencies such as Bitcoin [1] and Ethereum rely on hashing-based puzzles for getting transactions validated (e.g. Hashcash, Ethcash) in a distributed ledger represented by a blockchain. This computation (commonly referred to as "mining") is expensive, environmentally unfriendly and not used for anything apart from validating transactions. As discussed in [2], Bitcoin mining costs $15M/day in energy.

We present WekaCoin, a blockchain-based cryptocurrency that works similarly to BitCoin but uses Proof-of-Learning instead of hashing-based puzzles as Proof-of-Work. The currency is named after Weka, a widely used open source machine learning framework implemented in Java [3].

Proof-of-learning is a proposed mechanism for validating blocks of transactions in a distributed ledger inspired by machine learning competitions such as the ones hosted in Kaggle and Codalab. These competitions help improving the state-of-the-art in many relevant tasks such as image recognition, recommender systems, HIV research, etc.

Machine learning has become pervasive in science as well as the decision-making process of businesses and governments. Building a state-of-the-art machine learning system usually requires a combination of human talent, computational power, and access to large datasets. This has led to the centralization of state-of-the-art machine learning knowledge by very few institutions (e.g., Google, Facebook, Amazon). We believe that having a decentralized and open repository of machine learning models and experiments will be beneficial to the whole of society.

WekaCoin nodes operate in a peer-to-peer network where a group of nodes called "trainers" submit machine learning models for tasks that were previously published by other nodes in the network called "suppliers". These models are executed on data that was not observed by the trainers during training and ranked in a distributed fashion according to a performance metric determined by the task supplier. The "validators" are randomly selected nodes of the network in charge of ranking models and proposing new blocks to the chain. The trainer owning the best model is rewarded by the supplier with a certain amount of WekaCoins. The validators are also evenly compensated with a transaction fee paid by the task supplier as well as with new WekaCoins. This whole process is used to validate blocks of transactions in a blockchain. The resulting blockchain is a distributed and verifiable database of transactions plus an open repository of machine learning models and experiments used to verify them.

Proof-of-Learning establishes a symbiotic relationship between two complex and unrelated tasks: 1) validate transactions in a distributed ledger, and 2) storing machine learning models and experiments in a distributed database. The idea of aligning two different tasks is inspired by reCAPTCHA [4], a mechanism that allows detecting whether a web user is human (for security reasons) while assisting to digitize printed material.

II. Background

A. Cryptocurrencies and Blockchain Technology

Bitcoin [1] is a distributed cryptocurrency in which all transactions are published in a public ledger represented by a cryptographic data structure called a blockchain. The ledger is replicated across a network of users in a peer-to-peer network. Bitcoin was the first distributed cryptocurrency to solve the double-spending problem.

A Bitcoin blockchain B is a list of blocks $b_0, \ldots, b_t$ where each block b_i contains three items (tr_i, h_{i-1}, s_i), tr_i contains the transaction data represented as a Merkle Tree, h_{i-1} is a cryptographic hash to the previous block $(hash(b_{i-1}) =$

[†] Steve Reeves was funded by the New Zealand government via Science for Technological Innovation: http://www.sftichallenge.govt.nz

h_{i-1}), and s_i is an arbitrary nonce number that ensures the validity of the block, as explained below.

A blockchain B is valid if and only if all its blocks are valid. A block b_i is valid if and only if: 1) each transaction in tr_i is valid; 2) it contains the cryptographic hash of the previous block; and 3) the hash of the entire block (the concatenation of tr_i and s_i) is below a certain threshold w[1]. The value of w defines the difficulty in getting a block validated and is adjusted by the network to ensure that blocks are validated at an average pace of one every 10 minutes. A transaction is valid if and only if it has sufficient funds and is digitally signed by the sender.

The Hashcash method is the process by which miners, which are nodes of the network, validate a block by solving a hash puzzle. This puzzle consists of randomly finding a nonce number s that satisfies the third condition in the definition of a valid block as given above. The cryptographic hash function used in Bitcoin, SHA256, exhibits the *puzzle friendliness* security property [5], which tells us that it is unlikely that a strategy better than random guessing will be found for finding the target nonce number. The miner who solves the puzzle first is allowed to add a special transaction to the block in which new coins are created and transferred to them. Afterwards, the new validated block is added to the blockchain and broadcast to the network. Since finding a correct nonce gives a monetary reward, all miners invest computational resources to solve the puzzle, and the whole validation process is very expensive in terms of energy consumption.

The other members of the network proceed to verify if the new blockchain B is valid (by validating all the blocks in it). In the case where members receive two contradictory chains, the protocol suggests accepting the longest one.

The process described above is the distributed consensus protocol of Bitcoin. It provides statistical guarantees that a malicious node would need to have at least 51% of the mining power of the entire network in order to tamper with the blockchain. The malicious node would have to create a new blockchain at least as long as the valid one, and this will require too much computational power.

A more environmentally friendly protocol is Proof-of-Stake [6], in which miners are selected according to a criterion referred to as their "stake". Stake can be represented by a combination of the number of coins held by the miner and the *age* of those coins. Intuitively, the miner holds a certain number of coins for a period of time, and the chances of being selected to validate a block is proportional to the number and age of the coins staked by the miner.

In Proof-of-Stake a malicious miner can force a fork in the blockchain by announcing one block to the network while presenting a different block to some isolated users. This problem has been solved by periodically signing the correct branch of the chain by a trusted authority. This solution, however, is against the trustless spirit of many cryptocurrencies.

[1]This is equivalent to requiring the hash value to be smaller than a given value.

Algorand is a Proof-of-Stake based protocol that avoids forks in the ledger without relying on trusted authorities [7]. It implements a Byzantine agreement protocol in which a committee of voters is selected via weighted random selection using cryptographic sortition (the weights are based on wealth). Each candidate runs a verifiable random function (VRF) using its private key. A candidate is chosen to be part of the committee if the output of the VRF is below a certain value. This can be verified by anyone in the network. The consensus about a new block is reached by the committee in steps through a gossip protocol.

Blockchain technology is not suitable for storing big data files because nodes (i.e. the computers of people participating in the protocol) are expected to contain a full copy of the chain. Hence, storing large files would increase network latency. The Interplanetary File System [8] is a protocol for distributing files in a decentralized network. Large files are broken up into blocks that are arranged in a Merkle tree. Files are identified by the cryptographic hash of the Merkle tree root, which serves as a content address that can be referenced from a block within a blockchain. Any node in the IPFS network can add, request, and seed files. Nodes are identified by the hash of its public key. The data published on the network can be digitally signed using the private key of a node, allowing any member of the network to verify the data with the corresponding public key.

FileCoin [9] is a cryptocurrency that incentivizes nodes to store data on IPFS by receiving FileCoin tokens in exchange. Clients ask miners to store their files in the network and miners receive a payment if they can show via cryptographic proofs that the data is being continuously stored. This distributed consensus approach is referred to as "Proof-of-Storage". The chance of getting a reward is proportional to the amount of storage contributed by the miner to the network. This protocol is based on two sub-protocols: "Proof-of-Replication" and "Proof-of-SpaceTime". In Proof-of-Replication nodes show that they are hosting the data and in Proof-of-SpaceTime nodes show that the hosting is done over a specific time period.

B. Machine Learning Competitions

A machine learning competition or data science challenge is an exercise in which a machine learning task is crowd-sourced from a supplier to voluntary participants. The process starts with a host publishing a task on a centralized and trusted platform (e.g., Kaggle). Publishing a task involves providing training data, testing data, a submission deadline, a performance metric, and a reward.

A training dataset is usually a list of training instances where each instance is a vector of attribute-value pairs [3]. An attribute can be of various types such as: real, nominal, binary, ordinal, string, or multi-valued (e.g., a time-series, video, images).

The most common type of task used in competitions is a predictive task in which one target variable must be predicted. A predictive task training dataset contains one special attribute referred to as "label" or "target" that usually corresponds to the last attribute of their instances. The two most common types of predictive tasks are: 1) classification, where there is one nominal target variable (e.g., classifying images according to some predefined categories, classifying emails as spam or ham) and 2) regression, where there is one numeric target variable (e.g., weather forecasting).

Predictive tasks are generally solved by means of supervised machine learning algorithms. These algorithms learn a hypothesis function f_h from the training dataset mapping the independent attributes $x_1, \ldots, x_n$ into the target one y.

The learning process attempts to optimize a performance metric such as minimizing training error. There exist many machine learning algorithms for training such functions, such as decision trees, logistic regression, support vector machines, neural networks, many of which are described in [3]. In many cases the independent attributes are transformed before being fed into the training algorithm in order to improve performance. These transformations can be performed manually in a process called feature engineering, or they can be done automatically. Deep learning (or representation learning) algorithms are neural network-based techniques capable of jointly learning to transform and perform predictions. Deep neural networks have pushed the state-of-the-art in many predictive tasks such as image classification, speech recognition and machine translation.

The performance of a machine learning model is evaluated according to a performance metric such as accuracy, F-score, Kappa statistics or area under the ROC curve for classification; and root mean square error (RMSE) or mean absolute error (MAE) for regression [3]. An over-fitted model is a model that exhibits good performance on the training data but poor performance on unseen data. Because many machine learning models are prone to over-fit the training data, estimating the performance of a model using the same data on which the model was trained can lead to optimistic results. The goal of machine learning is to achieve generalization: obtain a model able to perform well on unseen data.

A procedure called hold-out can be employed to measure the generalization abilities of a model. In this approach, the data is split into a training set and a testing set. The model is trained only on the training set with the objective of generating predictions for the testing set. The estimate of the performance metric is then naturally computed by comparing the predictions and the real values on the testing set.

In a machine learning competition participants are asked to train a model using the training data provided. The task is usually predictive. Participants also receive a testing dataset formed by unlabeled examples i.e., examples where the label values are hidden. A submission contains the predictions generated by the model for the unlabeled examples. Since the real or “gold” values of the test instances are not known by the participant, there is little space for cheating. Valid submissions must be submitted before the submission deadline.

The gold values of the testing instances are sent by the supplier to the centralized platform on which the task is hosted. Hence, the centralized platform can compute the performance metric for the test data and rank all the received models. A leaderboard is usually published after the submission deadline, and the winning system can be monetarily rewarded.

III. WekaCoin and Proof-of-Learning

A WekaCoin blockchain B is a list of blocks where each block b_i is formed by three items (tr_i, h_{i-1}, c_i), where tr_i (transactions) and h_{i-1} (hash pointer to previous block) are analogous to the Bitcoin blocks and c_i contains metadata about the machine learning competition used to validate the block. A WekaCoin blockchain is valid if and only if all its nodes are valid.

The process by which WekaCoin transactions are validated is called Proof-of-Learning and is illustrated in Figure 1. There are three types of actors involved in this process: 1) **suppliers** who host machine learning competitions; 2) **trainers** who train and submit models for any task available; and 3) **validators** who evaluate the models on the test data, reach consensus about the winning system, and propose new blocks to the chain.

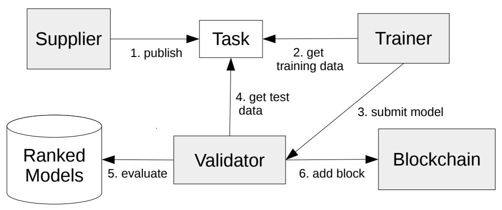

Fig. 1. Proof-Of-Learning Workflow.

WekaCoin implements three types of transactions tr: 1) a **standard transaction** which consists of a payment expressed in WekaCoins from one user to another, 2) a **task publication transaction** in which a supplier proposes a machine learning competition, and 3) a **model transaction** in which a trainer proposes a solution for a particular task.

WekaCoin is a trustless network in which anyone can either host a competition, propose a model, or participate as a validator. The protocol is further detailed in the following subsections.

A. Task Publication

A supplier S, can publish a machine learning task ta at any time by publishing a task publication transaction on the blockchain. We use block height (the length of the blockchain) to establish a consented temporal order of events in our protocol. A task publication transaction contains the following information:

- A publication timestamp pt;
- A training dataset trd: a set of training records according to the description given in Section II-B. This dataset is published on IPFS;
- A reward r expressed in WekaCoins for the competition winner;
- A performance metric pm used to evaluate models;
- A testing release block height tbh. This is the time (measured by block height) when the test dataset ted will

be released. The testing dataset will have the same format as the training dataset;

- An evaluation script es which is a program that takes two inputs: the gold labels and the predictions made by a model. It computes a value for pm;
- A baseline model $bmod$ which is a program that takes any dataset with the format of trd in which the target labels (y) are not necessarily given, and produces predictions for all instances in the data $\hat{y}$. Baseline models are usually very simple models, such as a majority vote for classification and the mean of the target variable for regression. This model is published on IPFS;
- A baseline performance score $bscore$ which is the output of the evaluation script applied to the labels from the training dataset and the predictions made by $bmod$ of the same dataset ($es(trd(y), bmod(trd))$);
- The cryptographic hash of the testing dataset ted. This is a dataset with the same form as the training set that will be used to evaluate the models. This file will remain hidden until the testing release block height tbh, when S will publish it on IPFS, but its hash value will be known at the time of publication of the task.

The task publication transaction also includes a standard transaction in which the supplier transfers the reward r and a task hosting fee thf to an empty address. Consequently, if a supplier does not release the testing data after the releasing block height it will lose those tokens.

The task publication transaction is digitally signed by the supplier and can be verified with the supplier's public key spk. A task is valid if and only if: 1) the value of $bmod$ is correct: $es(trd(y), bmod(trd)) = bscore$; 2) the task supplier S has enough funds to pay the reward r and a task hosting fee thf; and 3) the difference between the testing release block height tbh and the current block height is positive and larger than a minimum training period established by the network.

As will be discussed later, once a competition has finished, the reward is paid to the competition winner and the transaction fee is evenly distributed to the validators. The validators will also forge a new WekaCoin that is evenly distributed among them.

A possibly malicious strategy that a supplier can adopt is to participate in its own competition and submit a perfect model based on the knowledge obtained from the testing data. As will be discussed later, the supplier will need a lot of control of the network to get a reward from this behaviour. Even if it wins its own competition, it will have to pay the reward to itself and the transaction fee to the validators. So in order to recover its investment it will need to control a significant number of the validators. The randomness involved in the selection of validators will make the chances of profiting from this behaviour low.

B. Training

A trainer can pick any machine learning task ta published in the blockchain that is still in its training phase (current block height $< tbh$). Then it can proceed to train a model from the training dataset trd. We expect trainers to prioritize tasks with higher rewards.

A model is a program implementing a mathematical function (e.g., a neural network) or a set of executable rules (e.g., a decision tree) mapping elements from x into y. The program can be executed by any member of the network. There will be a consensus about the ML libraries and versions that every node should support (e.g., Sci-kit learn, TensorFlow, Pytorch, Weka). We expect nodes to have a virtual environment on their machines with the supported libraries and versions.

A model is submitted to the WekaCoin network by publishing a model transaction on the blockchain. This transaction is formed by the model mod hash, a timestamp, and the performance score for the training data. The training score will not be considered for ranking but would help validators to validate model submissions.

Trainers could potentially submit many models for a task, making the task of ranking computationally expensive. Therefore, the model transaction contains a payment equivalent to a participation fee from the trainer to an empty address. This is going to be a very small fee, but will prevent malicious nodes from performing DoS attacks by submitting useless models or trainers creating many identities and then submitting the same model many times. This cost will redeemed to trainers whose models performed above a certain percentile in the final ranking.

The model transaction is digitally signed and can be verified using the trainer's public key. It is important to remark that WekaCoin trainers publish entire models in contrast to centralized machine learning competition platforms where submissions are restricted to task predictions. This is because the ranking needs to be computed in a distributed and verifiable fashion. Moreover, the aim of WekaCoin is to create a decentralized repository of machine learning models that anyone can execute. However, trainers just submit the hash value of their models at this stage. Once the test data has been released, trainers are expected to upload their models to IPFS. This will prevent trainers from plagiarizing other trainers' models and to incentivize task suppliers to release the test data in order to have access to the models.

A malicious trainer could wait until the test data is released and then submit a model trained on the test data. This is avoided because the protocol establishes that model transactions are not added to the blockchain if the current blockchain is longer than the release block height of the task associated with the submitted model.

When the training period for a task has elapsed (current block height $\geq tbh$), the task supplier proceeds to upload the testing dataset ted to IPFS, which is an unseen dataset with the same form as the training set. Afterwards, trainers proceed to upload their models IPFS identified by the previously published hash address mod.

C. Block Proposal and Ranking

The process of adding a new block to the chain is done by a committee vc of randomly selected validators who need to reach consensus about three things: 1) the transactions to include in the proposed block, 2) the task used for validating the block, and 3) the best model for the selected task. The consensus is reached in various steps using a variation of the Algorand Byzantine Agreement approach [7].

The members of the committee are selected using a cryptographic sortition method implemented with a verifiable random function (VRF). Each node is weighted using a Proof-of-Storage approach, similarly to FileCoin. The likelihood of being selected is proportional to the amount of data (this includes datasets and models) related to WekaCoin competitions stored on IPFS. This will encourage nodes to host models and datasets from previous competitions and turn the WekaCoin blockchain into a distributed database of machine learning models and datasets.

Each validator builds a candidate block from its pool of transactions that have not been incorporated into the blockchain yet. This step requires verifying the set of selected transactions. The protocol establishes that the task used to validate the whole block must meet the following conditions: 1) it has not been used to validate a previous block in the chain; 2) its training period has concluded (tbh) a minimum number of blocks ago[2]; 3) it has the largest number of model transactions (according to the consented blockchain).

The validator proceeds to evaluate all the models published in the blockchain for the selected task by downloading each model from IPFS using the hashing value mod given in the corresponding transaction. A submitted model is valid if and only if: 1) it can be successfully downloaded, 2) it can be executed on the testing dataset, and 3) the performance score reported for the training data is the same as the value reported in the corresponding transaction.

The validator will then calculate the performance metric pm obtained for each model on the testing data ted using the evaluation script. Based on this, the validator builds a candidate ranking cr, which is an ordered list of pairs with the model and its performance score for the testing data. The list is sorted in ascending or descending order depending on the metric. The network establishes a time limit for evaluating all the models in a validation round. Models that cannot be evaluated on time are scored with a special "unevaluted" value in cr. Likewise, invalid models receive a special "invalid" score.

The proposed block together with the candidate ranking is sent to the other validators using a gossiping protocol. A validator can behave maliciously in many ways, such as including invalid transactions in the proposed block, ignore some models in the ranking to benefit a particular trainer, or send different rankings or blocks to different members of the committee. Our Algorand-based Byzantine agreement protocol mitigates those actions using a voting mechanism.

A validator V_i will receive the proposed block and the candidate rankings from the other validators V_j (where $i, j \leq |vc|$, and $i \neq j$). A candidate ranking cr_j submitted by another validator V_j will be considered invalid by V_i if it includes a model that is not part of the consented blockchain or assigns a different score to a valid model included in their own candidate ranking cr_i.

Regarding "invalid" and "unevaluated" models, if they receive valid scores in the candidate rankings of at least k validators (and all of them agree on the score obtained by this model), then the model is considered genuine. The value of k should be a fixed value determined by the network. For example, if k is $\lfloor \frac{|vc|}{2} \rfloor + 1$, then a model needs to be validated and evaluated by the majority of the nodes to be be considered as genuine. If there is disagreement between two validators for a certain model, the local validator will consider as genuine the score assigned by the majority of the other validators, and the candidate rankings associated with the minority score will be considered invalid. A candidate ranking sent by another validator is genuine if and only if all their models are genuine.

There is trade-off for the value of k. If k is too small, there is a chance that malicious models, included by malicious validators, are considered in the final ranking and win the competition. In contrast, if k is too large, there is a chance that a genuine model gets discarded only because of network latency.

Once all genuine candidate rankings from the other validators have been identified, the validator V_i will proceed to create a meta-ranking by merging all the genuine candidate rankings (candidate rankings with illegitimate models are discarded). The validator will also proceed to merge its candidate block with the ones sent by the genuine validators. It will also discard any invalid transaction from the other candidate blocks and transactions that could potentially be added to the blockchain.

This consensus ranking and consensus block will be sent to all the validators that submitted a genuine candidate ranking. Since only the genuine validators will receive this information, we expect that all consensus blocks and rankings will be the same after a certain number of iterations. This is a strong incentive for validators to only send genuine blocks and rankings.

D. Block Commit

Once all genuine validators have reached consensus about the proposed block and the ranking, they add the following transactions to the block:

1) A transaction transferring the task reward r from the task supplier S to the owner of the top performing model mod according to the consented ranking. Ties will be broken in favour of the model with the smallest size. This is inspired by the principle known as Occam's razor applied to a machine learning context and it states that simple models should be preferred. If ties are still found they should be broken by random selection (e.g., picking the trainer with the lowest value for the product of the public key and the submitted model hash value);

[2]This is to give the trainers enough time to upload their models to IPFS after the test data is released.

2) A transaction transferring the task hosting fee from the task supplier S to all the valid validators. The fee is evenly divided among all valid validators;
3) A transaction redeeming the participation fee to all trainers whose models were ranked in the first quartile (or a different percentile to be defined by the network);
4) A transaction transferring the participation fee of all trainers whose models were either discarded in the ranking consensus process or ranked below the threshold (as above) to all valid validators. This fee is evenly distributed;
5) A new WekaCoin divided among all the valid validators.

The new block of transactions b_i includes a metadata object c_i containing information about the the cryptographic sortition process, the chosen task, and the rankings (each of those digitally signed by the corresponding validator). This allows anyone to verify the validation process of the block.

IV. Related Work

The most related work to ours is the Proof of eXercise (PoX) protocol proposed in [10]. In this protocol miners solve matrix-based scientific computation problems provided by employers (e.g., matrix product, determinant, eigenvectors) to validate transactions in a blockchain network. The employer pays for the hosting of the matrix using a hosting credit system. Miners bid for receiving a problem to solve which is then refunded if the problem is successfully solved. In contrast to Proof-of-Learning, there is no need to rank solutions here, because the task has an exact solution i.e., the result of the matrix operation. This is a strong difference between PoX and Proof-of-Learning. The solutions are validated by auditors. However, since the only way to perform an exact validation of the solution is to recompute the whole matrix multiplication process again, a probabilistic verification scheme is employed in which auditors verify random parts of the matrix problem.

To a certain extent, our protocol generalizes PoX to a more general task in which the quality of the solution is measured with a performance metric. Platforms such as Kaggle and Codalab suggest that there are many people willing to crowd-source their machine learning tasks. To the best of our knowledge, there are not many equivalent services for matrix-based computations. Moreover, machine learning models exhibit a property that is desirable for a proof-of-work consensus mechanism: they are hard to solve but easy to verify. This is because executing a machine learning model is much faster than training it. Finally, our protocol serves an additional purpose: it creates an open repository of machine learning models and datasets.

There are a couple of other related projects aiming at hosting machine learning competitions in a decentralized way such as Cerebrum[3] and Danku[4]. Both of them rely on the existing Ethereum blockchain infrastructure (including its hash-based proof-of-work method) via smart contracts.

Danku relies on the Ethereum Virtual Machine (EVM) for training and deploying machine learning models. This is a strong barrier for implementing state of the art machine learning models, since modern ML libraries are not implemented for the EVM. In contrast, WekaCoin enables using general-purpose machine learning libraries, since models are locally trained and run by each node. Cerebrum transactions, on the other hand, only include task predictions instead of executable models.

[3]https://cerebrum.world/
[4]https://github.com/algorithmiaio/danku

V. Conclusions and Future Work

This article presented Proof-of-Learning, a new distributed consensus protocol for validating blockchain transactions using machine learning competitions. The main benefit of this protocol is that the energy involved in the validation process is used to solve useful tasks while creating an open repository of machine learning models and datasets.

There are still many open questions that need to be addressed in future work. The first question is how to obtain a continuous supply of tasks in order to validate transactions at acceptable rates. It is plausible to believe that many research institutions will be willing to crowd-source their machine learning problems to the WekaCoin network. If data privacy is a concern, suppliers could anonymize the datasets by renaming, reordering, or even transforming the attribute space. An additional solution involves having trusted nodes generating synthetic machine learning tasks during periods of supply shortage.

Another problem that we intend to study further is how to prevent collusion between the three main actors of the network: suppliers, trainers, and validators. The accurate randomization of the ranking committee is crucial to avoid this behaviour. We need to ensure that suppliers have little chance to rank their own tasks. However, we still need to encourage nodes for hosting data on IPFS. In future work, we will further investigate the incentives of these three actors (e.g., task hosting fees, task rewards, mining participation costs, forging new coins), such that their expected utilities are negative when malicious behaviour is exhibited.

References

[1] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008.
[2] M. Swan, *Blockchain: Blueprint for a new economy.* " O'Reilly Media, Inc.", 2015.
[3] I. H. Witten, E. Frank, M. A. Hall, and C. J. Pal, *Data Mining: Practical machine learning tools and techniques.* Morgan Kaufmann, 2016.
[4] L. Von Ahn, B. Maurer, C. McMillen, D. Abraham, and M. Blum, "recaptcha: Human-based character recognition via web security measures," *Science*, vol. 321, no. 5895, pp. 1465–1468, 2008.
[5] A. Narayanan, J. Bonneau, E. Felten, A. Miller, and S. Goldfeder, *Bitcoin and cryptocurrency technologies: a comprehensive introduction.* Princeton University Press, 2016.
[6] S. King and S. Nadal, "Ppcoin: Peer-to-peer crypto-currency with proof-of-stake," *self-published paper, August*, vol. 19, 2012.
[7] Y. Gilad, R. Hemo, S. Micali, G. Vlachos, and N. Zeldovich, "Algorand: Scaling byzantine agreements for cryptocurrencies," in *Proceedings of the 26th Symposium on Operating Systems Principles*, ser. SOSP '17. New York, NY, USA: ACM, 2017, pp. 51–68.
[8] J. Benet, "Ipfs-content addressed, versioned, p2p file system," *arXiv preprint arXiv:1407.3561*, 2014.
[9] P. Labs, "Filecoin: A decentralized storage network," Tech. Rep., 2017.
[10] A. Shoker, "Sustainable blockchain through proof of exercise," in *Network Computing and Applications (NCA), 2017 IEEE 16th International Symposium on.* IEEE, 2017, pp. 1–9.