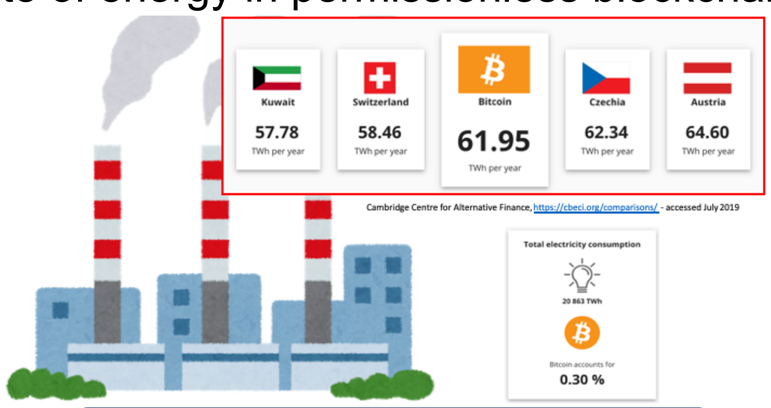
Towards Useful Proof of Work: combining Blockchain and Machine Learning

Andrea Merlina
Supervised by Roman Vitenberg* and Vinay Setty**

Motivation

Waste of energy in permissionless blockchain



Waste of computation

Proof of Work is a computation-intensive process which does not provide any useful side product besides securing the blockchain.

This can be regarded as waste of computation.

Promoting healthy competition

In a blockchain whose work is based on solving Machine Learning tasks, miners compete between each other to produce in a bounded amount of time a model that minimizes the errors.

Because of this healthy competition, miners strive to achieve the optimum balance between training efficiency and quality of the produced model.

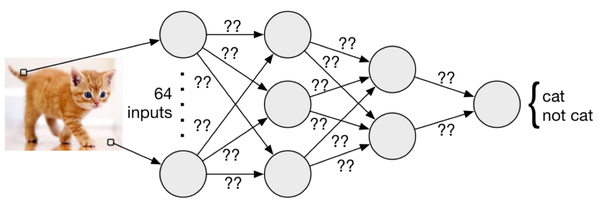
Machine Learning task

Machine Learning tasks are classified into several categories. We mostly focus on supervised learning, where an algorithm builds a mathematical model that maps an input to an output based on examples of input-output pairs.

One method to implement the mapping function is using Artificial Neural Network (ANN), as the one showed in the picture.

The process is divided in two phases. The first phase builds (trains) the model feeding it with a collection of input-output pairs. The second phase evaluates the performance of the trained model in terms of accuracy on test data which is separate from the training data.

A good model is able to correctly predict the test data.



Source: https://homes.cs.washington.edu/~bornholt/post/nnsmt.html

The loss function is a common metric for performance evaluation as it quantifies the difference between true and predicted outputs.

Problem definition

Assumptions

We assume permissionless access, the same network communication, threat model and failure model as in Bitcoin

Goal

The goal is replace PoW with the training of Machine Learning tasks,

where the tasks executed by miners

This provides the additional property of:

Usefulness. The puzzle solution provides some useful application aside securing the blockchain.

Challenges

- PoW is required to satisfy several properties.
- Input-output domain and data need to be continuously supplied as part of the mining process.
- Both training and test data might have considerable sizes, impacting communication latency and blockchain storage.
- Supervised learning is divided in two phases, the training and the evaluation, which need to be kept separated also in the solution approach.
- The protocol must assure safety and incentivize participants to not deviate from the default behavior by setting appropriate fees and rewards, for example in the case of a colluding Miner and Supplier.
- Miners that compete to solve a machine learning task must work on the same data in order to correctly rank the solution and for the competition to be meaningful.
- The competition in solving machine learning tasks provides models which are not perfect per se but selected as winning only in relation to the other ones proposed. As a consequence, validation requires all the competing models, and becomes expensive while opening to problems such as where to store the models and for how long.

PoW Properties

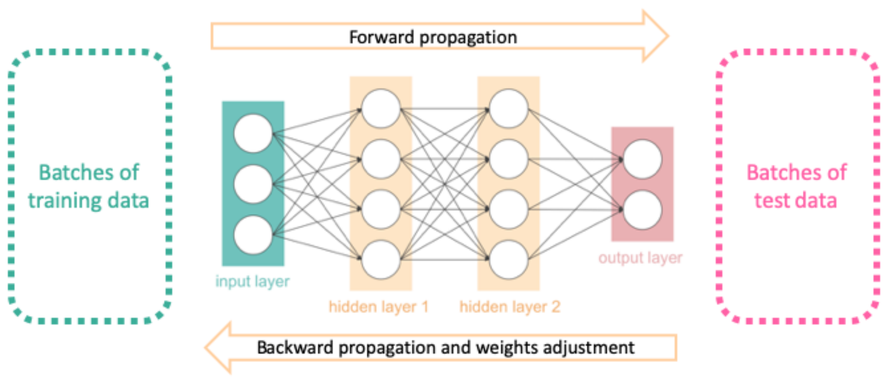
1. **Hardness**.
   - Finding a correct solution for the puzzle necessitates actual work.
2. **Adjustability**.
   - The difficulty of the puzzle can be tuned to modify the average block interval.
3. **Efficiency**.
   - The solution must be efficient to verify by validators.
4. **Completeness**.
   - A solution for the puzzle is assured to pass verification.
5. **Soundness**.
   - An invalid solution for the puzzle is very unlikely to pass verification.
6. **Changes sensitivity**.
   - The work is tied to a specific block and set of transactions. If some value of the block changes, the nodes must be able to detect it.
7. **Creator free**.
   - Finding a solution for a puzzle does not provide any advantage in solving any other.
8. **Chance to win**.
   - Every participant has non-negligible probability to be the one proposing the next block.

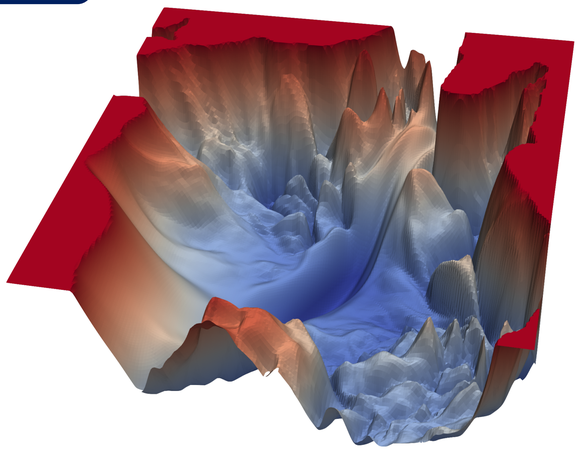
Solution background

Neural Networks are trained adjusting the weights of the network so that the true and the predicted output coincide. The adjustment is calculated using a Gradient Descent algorithm that minimizes the loss function (a 3D representation is showed in the picture).

Training starts with a forward propagation of the input data. The adjustment is calculated with Gradient Descend and the weights are updated accordingly, using a Backpropagation algorithm.

As its name suggests, the Backpropagation algorithm is applied first to the output layer of the network and then backward, iterating over the other layers back to the first.



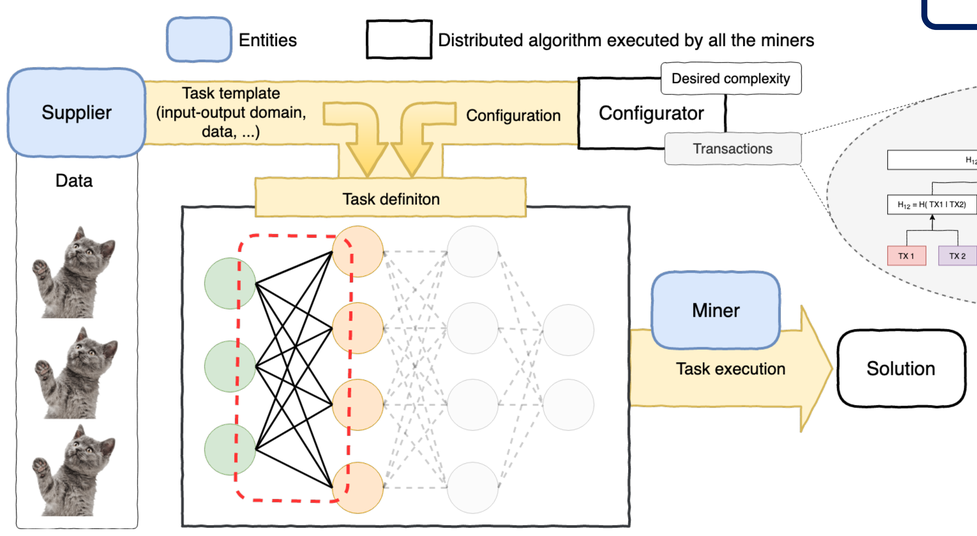


Loss function of a 56 layers deep neural network trained on the CIFAR-10 dataset (https://www.cs.umd.edu/~tomg/projects/landscapes/)

In the state of the art, the weights of an untrained Neural Network are initialized with values that have been empirically demonstrated to provide a more effective training, for example mitigating the vanishing gradient problem.

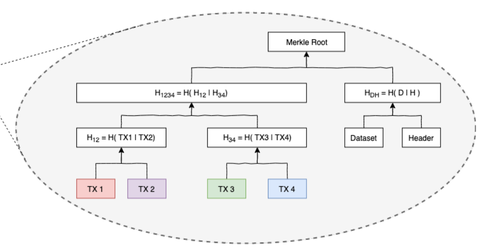
The authors in [2] argument that it is faster to train a suitably initialized depth-100 network than it is to train a depth-10 network, showing how accuracy depends on weight initialization.

Solution approach



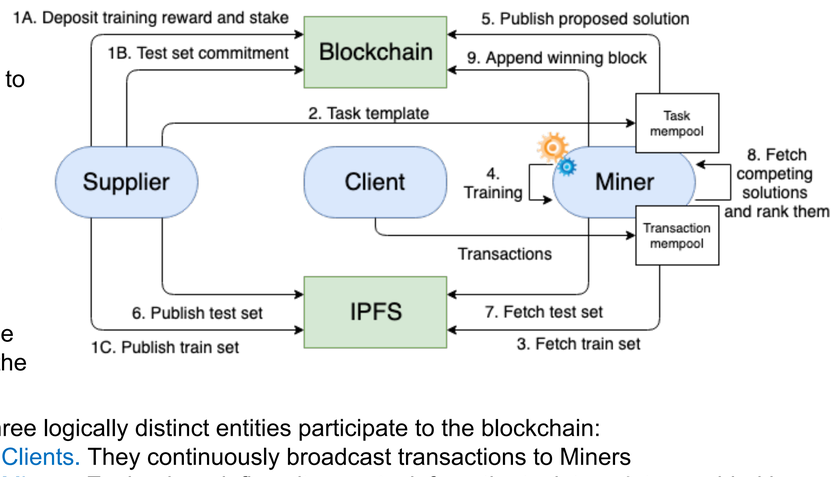
Task template and configuration define a task that miners have to solve to propose a valid block.

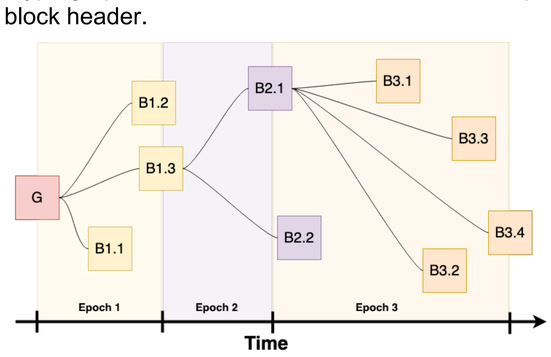
The weights of the first layer of the Neural Network are set based on a surjective mapping function that takes as input the modified Merkle tree root. This process is called encoding. Weights are kept fixed during the training phase so that a change in any transaction of the winning block modifies the model in a detectable way.



Suppliers send task templates to miners, with references to:
- A deposited training reward
- A deposited stake
- A train set
- A commitment to its test set

A Merkle tree is built with transactions as leaves.
In addition, the tree is modified because the root contains the cryptographic hash of the dataset of the current epoch and the block header.





Three logically distinct entities participate to the blockchain:
- Clients. They continuously broadcast transactions to Miners
- Miners. Each miner defines its own task from the task template provided by Suppliers and its locally computed configuration. Miners propose and rank solutions to establish the next-to-be-appended block to the blockchain.
- Suppliers. They are centralized non-trusted entities which provide data and pay for the training cost.

The blockchain grows in time, where each block is a trained model that won the competition for the specific epoch.

References

1. Ball, Marshall, et al. "Proofs of Useful Work." *IACR Cryptology ePrint Archive* 2017 (2017): 203.
2. Hanin, Boris & Rolnick, David. (2018). How to Start Training: The Effect of Initialization and Architecture.
3. Boyd C., Carr C. (2018) Valuable Puzzles for Proofs-of-Work. In: Garcia-Alfaro J., Herrera-Joancomartí J., Livraga G., Rios R. (eds) Data Privacy Management, Cryptocurrencies and Blockchain Technology. DPM 2018, CBT 2018.
4. Faye Loe, Angelique & A. Quaglia, Elizabeth. (2018). Conquering Generals: an NP-Hard Proof of Useful Work. 54-59. 10.1145/3211933.3211943.
5. Bravo-Marquez, Felipe & Reeves, Steve & Ugarte, Martin. (2019). Proof-of-Learning: a Blockchain Consensus Mechanism based on Machine Learning Competitions.

* UiO : University of Oslo
** University of Stavanger