ZK-friendly ML model explorations - Milestone 3

Saeyoon Oh

March 2024

This article outlines the progress of milestone 3 for acceleration program by Ethereum PSE team. Article first restates the goals made for the report, concluding with the progress made and future works.

1 Goals

The goal of milestone 3 is to provide source code and documentations of how one can make classification using decision tree given heart failure dataset. We provide source code to prove decision tree prediction using EZKL, and mimic proof generation to obtain resource required using Circom, and compare the number of constraints with neural network.

- Functionality: Provide source code to train and prove decision tree output using EZKL.
- Functionality: Provide approach to prove trained balanced decision tree using Circom.
- Analysis: Compare proving approach using EZKL and Circom, analyze the constraints that are required to prove decision tree and compare them with neural network.

2 Experiment Conditions

All of the experiments were performed on Apple M3 Pro with 18GB of RAM. Unless metioned, we use decision tree of max depth 3 as shown in Figure 3.3.2 which consists of 7 classification nodes. For multi-layer perceptron (MLP), we use MLP with 1 hidden layer having hidden dimension of 8. This results in 38 trainable parameters.

3 Progress

The progress made are as follows.

3.1 Generate script for automatic Circom key generation

While previous steps required users to manually generate Circom keys, we added a script that allows automatic key generation using snarkjs. Users can appropriately set the constant depending on the circuit size the user is trying to prove. You can find the file here.

3.2 Proving decision tree using EZKL

We first provide pipeline to train scikit learn decision tree and prove its output using EZKL. Since EZKL currently supports automatic proving scikit learn decision tree, this is straightforward. We provide end-to-end code for proving decision tree in the repository. The results are as follows:

- setting_time: 0.016
- calibration_time: 0.343
- compile_time: 0.002
- get_srs_time: 0.003
- witness_generation_time: 0.189
- setup_time: 4.975
- proof_generation_time: 49.212
- verification_time: 0.124
- total: 276
- correct: 230
- zk_total: 276.0
- zk_correct: 230.0
- Accuracy: 83.33
- ZK Accuracy: 83.33

We also note that for EZKL we need to use larger max_logrows (13) compared to max_logrows used for neural network verification. Although we set max depth of decision tree to 3, which forbids the tree to grow in complex manner, it yields accuracy of 83.33% which is significantly larger than scores we observed using multilayer-perceptrons. Yet, decision tree shows much longer proof generation time compared to multilayer-perceptrons. (The proving time for MLP were at most times under 1 second.) We conjecture that this is due to different protocol used for proving neural network and decision tree. While MLP can be proven by only using only matrix multiplication proof system and lookup argument, decision tree proof using EZKL also requires private indexing argument along with matrix multiplications. More information can be found in this link.

3.3 Proving trained decision tree using Circom

3.3.1 Proof generation analysis for ZKDT

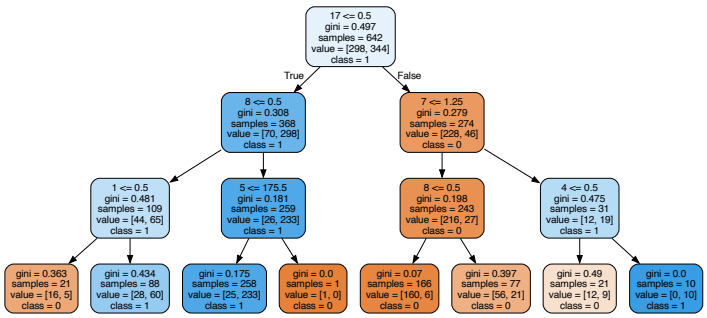
In this section, we outline the process we have gone through for proving decision tree using ZKDT protocol. We tried using previously implemented source codes such as ZK-DTP or ZKDT_release but was not able to resolve dependency issue nor make appropriate changes due to deprecated packages. Therefore, we used ZKDT repository to prove decision tree.

ZKDT protocol is consisted of two parts. 1) Proving permutation of inputs (which is required within proving protocol) and 2) Path validation, which proves that the path within the decision tree used for classification is indeed true by using decision tree commitment made earlier. Currently only the second part is implemented, therefore requiring users to manually permute the data depending on the decision tree.

We observe that this protocol works only if each classification node uses different features. For example, this proof cannot be used if two classification nodes try to use the same features for classification. (Which was the main reason why I could not transform trained decision tree to desired format.)

While the protocol itself accepts unbalanced decision tree, the implementation of path validation (which is really similar to Merkle Tree proof) assumes balanced binary tree. We plan to further extend the implementation in the future.

Specifically, the decision tree that can be proven need to be balanced, and each tree nodes need to use different feature. This is unlike the decision tree that we trained as in Figure 3.3.2, generated by scikit learn decision tree classifier.



Therefore, we mimic a decision tree in the implementation-wise desired format so that it can be proven using the ZKDT source code. This allows us to check the time for proving and number of constraints and compare the results with EZKL. ZKDT also contains deprecated Circom codes, so we provide another repository forked from ZKDT to enable proving. We also provide script to prove decision tree. Using the same sized decision tree, we calculate the time for proving. The results are as follows:

- proof generation time: 1.142
- verification time: 0.208

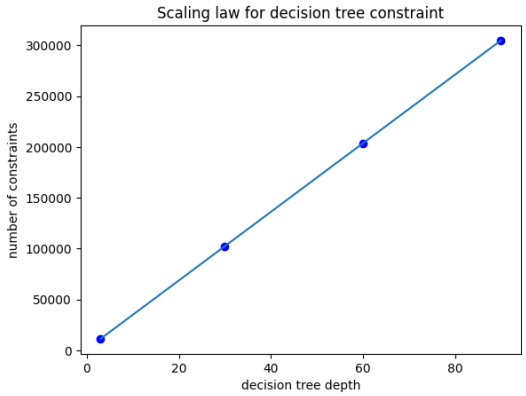
In terms of number of constraints, MLP model Circom circuit uses 14800 constraints, while decision tree contained 135936 constraints. Again we conjecture that a merkle-tree-like proof results in more complex proving system and therefore requiring greater number of constraints.

The proof generation time was 1.06 in our previous experiment, resulting in slight lower proving time. Detailed results for MLP Circom circuits is contained in progree report 1. The result can be reproduced by following the modified ZKDT repository.

We note that the proof generation time is much shorter when proving in Circom compared to EZKL. Yet, one thing to remind is that current Circom proving system not only implement the protocol partially, is constrained to proving decision trees such that each node uses different data feature.

3.3.2 Scaling law for decision tree in Circom

We calculate the number of constraint in the decision tree proving circuit by increasing the number of depth in decision tree. We observe a linear increase in the number of constraint per depth of decision tree, which is also reported in the ZKDT paper.



3.4 Conclusion

For EZKL and Circom, decision tree provided higher accuracy while being more time inefficient. Comparing EZKL and Circom, Circom showed faster proof generation, but was restricted in different manners.

Therefore if neural networks were to show higher accuracy, they can be seen as a better option for building ZK-ML applications. While this may be alleviated by increasing the size of MLP, we leave that as future work. We also

note that this result may not apply generally to all other tasks. We believe that the dataset we are using is a good fit for decision tree, where features do not have to be interacted implicitly. Therefore we believe different results can be made for other datasets that require machine learning models to learn the relationship between features.

4 Future Works

For milestone 4, we plan to do the followings.

- Cleanup and refactor source code for better user experience.
- Build proof generation pipeline for k-means clustering using EZKL and possibly by Leo using library built for k-means clustering using leo.
- Analyze the constraints and time complexity for generating proof for k-means clustering, comparing with decision tree and neural networks.