zk-SNARK for Machine Learning Integrity

Alexander Baron

Problem

- Challenge
  - LLM’s are trained on sensitive data
  - Organizations need to prove their model’s outputs are correct
  - Without revealing training data or model weight
- Current Approaches
  - Expose proprietary weights
  - Reveal training data
  - No cryptographic guarantees
- Medicine
  - Zero-Knowledge Proofs
  - “I can prove I know the password without revealing it”

Zero-Knowledge Proofs

- Properties
  - Completeness - Valid proofs always verify
  - Soundness - Can't forge fake proofs
  - Zero-Knowledge - Reveals nothing except validity
- zk-SNARK
  - Zero-Knowledge - No leaked information
  - Succinct - Small Proof Sizes
  - Non-Interactive - One message, no back and forth
  - Argument of Knowledge - Cryptographically sound

Transformer Challenge

- LLM’s often contain billions of parameters, with the modest of them reaching millions.
- The challenge
  - Each Parameter + Operation = Circuit Constraints
  - Operations such as softmax and ReLU are expensive
  - This results in billions of constraints making zk-SNARK on LLM’s computationally infeasibly
- Our Approach
  - Vocabulary - 100 Tokens
  - Hidden Dimensions - 32
  - Context Length - 8 Tokens
  - No Activation Functions
  - Linear Attention
  - Resulting in ~12,500 parameters

Architecture

- 18-bit Fixed-Point Conversion
- Circom Circuit (163K constraints)
- Witness Generation
- Gorth16 Proving System
- Proof (743 Bytes)
- Verification
- Components
  - Input Token Ids : [12,3,89,1]
  - Private : Model Weights, input tokens
  - Public : Predicted output token
  - Commitment: Poseidon hash of weights

Results

- 163K total Constraints
- 15K private inputs (weights + tokens)
- 1 Public Input (Predicted Output)
- The program on the first run took over an hour to produce a proof and verify.
- But we were able to get our proof, with it only being 742 bytes.