Brainlock - Encrypted Inference on any ML Model

Abstract

Fully Homomorphic Encryption (FHE) enables encrypted inference on machine learning (ML) models, allowing computations to be performed on encrypted data without revealing sensitive information. This paper outlines the architecture of a Bittensor Subnet designed to incentivize speed enhancements for FHE inference.

1. Introduction

The increasing use of machine learning in various industries requires handling sensitive data, often involving personally identifiable information (PII) or confidential records. Privacy concerns and regulatory requirements have made it essential to find solutions that allow data processing without compromising confidentiality.

Fully Homomorphic Encryption is a form of encryption that permits computations on encrypted data, with the resulting output also being in encrypted form. Traditional ML models require plaintext data for training and inference, which creates potential privacy issues when the data is sensitive.

Mathematically, if $E(m)$ represents the encryption of a message m, and f is the ML model function, FHE allows us to compute $E(f(m))$ without needing to decrypt m at any point, as shown in figure 1.

$$E(f(m)) = f(E(m)) \tag{1}$$

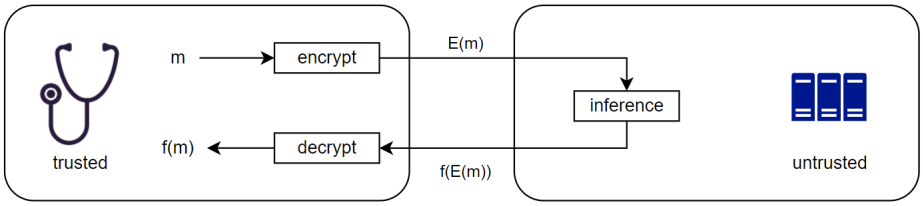

Figure 1: No decryption on the untrusted side

This property ensures that data remains confidential throughout the computation process.

A major issue with using FHE is that it is much more computationally intensive than regular inference, resulting in longer processing times and higher computational requirements.

2. Implementation

To implement FHE-encrypted inference, we use a library from Zama named "Concrete-ML" that is open sourced under the BSD-3 license, which allows for research. Profitable uses require a commercial license. This library is the current state of the art (SoTA). For the time being, inference will be ran on Visual Geometry Group (VGG) model with 9 layers, which is the standard used by Zama in their SoTA papers, to allow for an easy comparison of the improvements. The miner and validator interaction proceeds as follows:

1. **Data Encryption**: The validator encrypts the input data, or in our current case, an image for the VGG9 model. The source of the image is fetched randomly from a HuggingFace dataset. Miners cannot hash or memorize these inputs as they would never get the same image encrypted with the same key more than once, ensuring that they never know what image they are given. Synthetic images will quickly be added into the mix.

2. **Encrypted Inference**: The encrypted data is sent to an VGG9 model hosted by the miner. Switching between models is trivial, and in the future, we plan to support Large Language Models (LLMs). As miners' speeds improve, this approach can be scaled to include smaller sections of LLMs called attention heads, allowing for faster iteration and optimization. Eventually, additional or all layers of attention of a small size LLM will be used for inference.

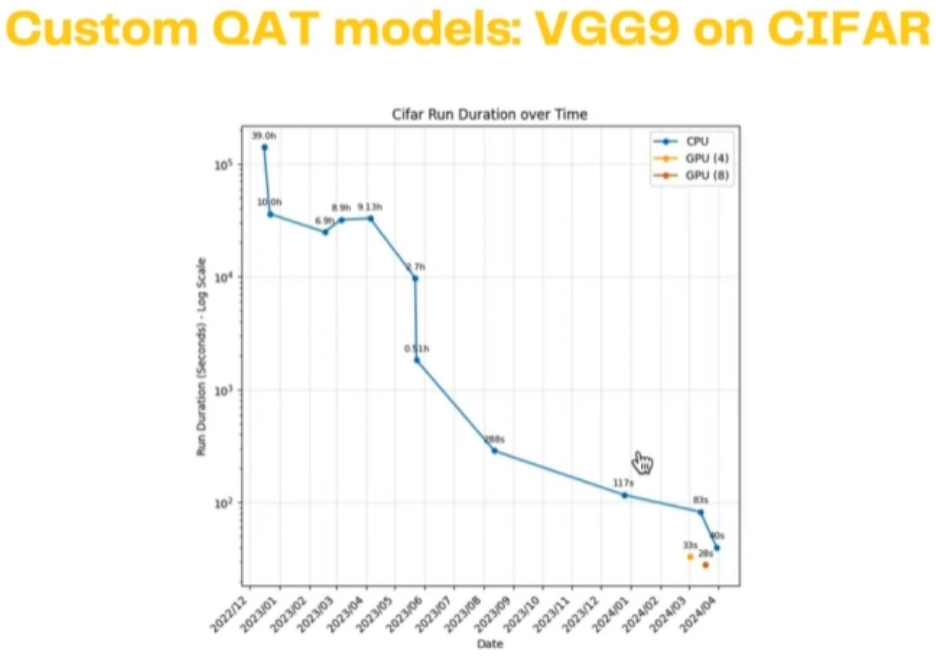

Figure 2: Performance improvements of VGG9 on CIFAR datasets over time using Custom QAT models.

3. **Return of Encrypted Results**: The miner streams the encrypted output back to the validator. In our current system, we have opted not to use axons, instead using Targon's Epistula Protocol. Its multi-language support can help miners accelerate performance by allowing logic to be rewritten in lower-level languages.

4. **Decryption and Verification**: The client decrypts the miner's output to obtain the inference result. Simultaneously, a validator runs a simulated version of Fully Homomorphic Encryption (FHE) without encryption to generate a baseline output for comparison. For models such as VGG, a direct comparison suffices, while for large language models (LLMs), the validator assesses the miner's decrypted output using cosine similarity to evaluate the reasonableness of the response against the prompt.

The miner's performance is evaluated based on a combination of inference speed (time per encrypted inference) and accuracy relative to the simulated output. Miners are ranked by their comparative scores. In practical applications, queries sent to miners include both synthetic queries (used to score miners) and organic customer queries. To maintain high scores, miners must provide honest and accurate responses to all queries, as discrepancies in synthetic query responses negatively impact their ranking.

3. Discussion

The primary advantage of FHE-encrypted inference is the ability to perform computations on sensitive data without ever exposing it in plaintext. This capability is crucial in sectors such as healthcare, finance, and legal, where data privacy is paramount. For example, medical institutions can use FHE to run predictive models on encrypted patient data, ensuring compliance with privacy regulations such as HIPAA.

However, as mentioned earlier, the computational cost of FHE remains a significant limitation. FHE operations are currently much slower (100-1000x) than traditional inference due to the complexity of working with encrypted data. This additional latency makes FHE challenging to deploy in real-time applications. While this added delay is a trade-off for privacy, the cost of maintaining privacy is far outweighed by the growing number of cybersecurity incidents. Sending unencrypted data to a third party not only risks your own information but also makes you vulnerable to breaches involving that third party.

Leveraging a Bittensor subnet presents a promising solution to address this challenge. Bittensor's incentive structures have demonstrated significant improvements in inference speeds across other subnets. By creating a subnet specifically for FHE-encrypted inference, miners could be incentivized to optimize Zama's FHE framework operations. This would not only help overcome performance hurdles but also provide a valuable revenue opportunity in a space without any provider.

Miners on the subnet will be scored based on their inference speed and response accuracy on encrypted images, and eventual prompts. The incentive structure encourages miners to develop or use faster algorithms, approximations for non-linear operations or hardware solutions to reduce the computational overhead associated with FHE. As miners achieve faster inference speeds, they will be rewarded using a Pareto-based system to ensure continuous improvement and prevent stagnation.

Eventually, advanced incentive mechanisms like progressive bounties would be implemented. As miners achieve new performance milestones in encrypted inference (e.g., reducing inference latency by a further 10%), they will receive additional rewards from

the subnet owners. When possible, the optimizations will be shared with Zama's team, which developed and maintains the Concrete-ML library.

The ultimate achievement of the subnet is to provide inference on encrypted inputs using encrypted model weights in a reasonable inference time. This approach provides maximum security as neither the model nor the data is exposed.