A Privacy Preserving Machine Learning System
Alexander Wu, Hank O’Brien, Joseph Gonzalez, John Kubiatowicz

System Overview

Overview
We use the browser’s built-in iframe isolation policy to enforce the untrusted app/UTE barrier and utilize a message passing interface to communicate between our iframes and the untrusted app. The TTC is a separate product from a third-party infrastructure provider and in its most secure form would use secure enclaves to provide privacy.

Network Critical Path
1. App Developer creates a new ML model, encrypts* it, and uploads it to one or more KV Stores
2. App Developer creates a new application with an embedded UTE
3. User uploads user data to the UTE
4. UTE returns a DataDescriptor to the App
5. App developer tells the UTE to publish the user data to a particular KV store and to prepare it for a particular TTC
6. Data is encrypted by the UTE and uploaded to the KV store
7. Application initiates a serve request with a model handle and data handle
8. TTC downloads model handle and decrypts from potentially different KV Stores and decrypts both, and runs the model on this data
9. TTC re-encrypts the inference result and uploads it to a KV store
10. Application receives a response handle and asks UTE to receive the corresponding result
11. UTE downloads and decrypts the inference result and renders it to the user

Cryptography
Cryptography hardening primarily relies upon the Signal Protocol, a well-established asynchronous double ratchet protocol. The protocol establishes a confidential channel between the UTE and TTC. Here, we use the abstraction that each machine is assigned its own identity key. This public component of the key can be signed by a certificate authority in less secure settings. In more secure settings, remote attestation can be used to prove that the key originated from an enclave, and that was in place to prevent the abuse of sensitive user data. We propose a machine learning inference system which provides an end-to-end preservation of privacy for both a machine learning model developer and user. Our system aims to minimize its constraints on the expressiveness and accuracy of machine learning models. Our system achieves this by utilizing trusted computation, with a trust-performance tradeoff which extends to a cryptographic proof that data is not tampered with.

Future Work
Our first priority in the near term is finishing the Signal integration and fixing our ONNX issues with GPUs. On a more medium term however, we think adding features like supporting ML models that accept multiple data points and running the ML models in containers with heavily restricted I/O permissions to provide further isolation and protection in this scheme. There is also a good deal of long-term future work in two areas, 1) developing a new UTE for desktop and mobile applications, and 2) extending the security of the TTC to support remote attestation and verifiable computing from secure enclave technologies like Intel SGX. Furthermore, there is a good deal more work we could do to further improve overall system latency, such as by initiating the model fetch and the data fetch in parallel when the model is not already on the TTC, or some form of dynamic data replication where the more a model is used in a particular TTC, the more likely it becomes that the data for this model is located nearer this TTC geographically. This will significantly reduce the model fetch time, which is on the inference critical path.