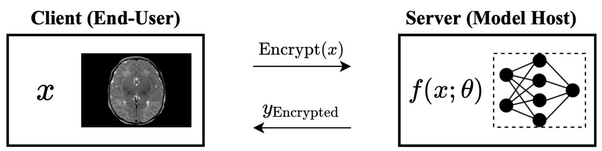
Sisyphus: A Cautionary Tale of Using Low-Degree Polynomial Activations in Privacy-Preserving Deep Learning

Karthik Garimella, Nandan Kumar Jha, Brandon Reagen

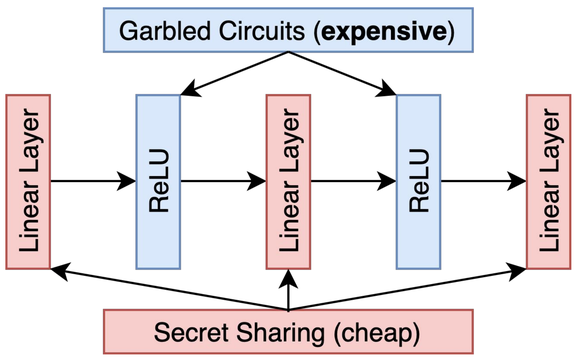
New York University

Introduction

Privacy concerns in client-server machine learning have given rise to private inference (PI), where neural inference occurs directly on encrypted inputs [1]:



Linear layers are implemented using a combination of Homomorphic Encryption and Secret-Sharing, while Garbled Circuits implement non-linear functions such as ReLU (max(0,x)).



In private inference, Garbled Circuits for ReLU dominate latency and storage costs (e.g. ~10 GB storage, ~10 sec latency for ResNet-18 on CIFAR-10) [2]. In this work, we attempt to replace *all* ReLUs with low-degree polynomials.

Solution: Drop-And-Replace

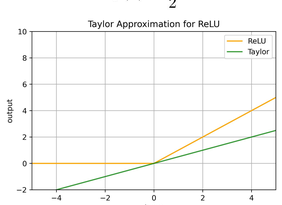
Taylor Approximation

Replace ReLU with it's Taylor Series expansion around *a* = 0:

$$f(x) \approx \sum_{n=0}^{\infty} \frac{f^{(n)}(a)}{n!}(x-a)^n$$

Higher order derivative terms for ReLU vanish, leaving:

$$f(x) = \frac{1}{2}x$$



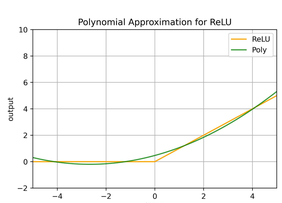
Each neural network collapses to a linear model.

Polynomial Regression

Fit a polynomial function to ReLU and find coefficients by minimizing mean squared error:

$$E(\vec{w}) = \int_{-a}^{a} \left(\sum_{d=0}^{D} w_d x^d - \text{ReLU}(x) \right)^2 dx$$

Bayesian Optimization is employed to optimally choose both *a* and *d*.

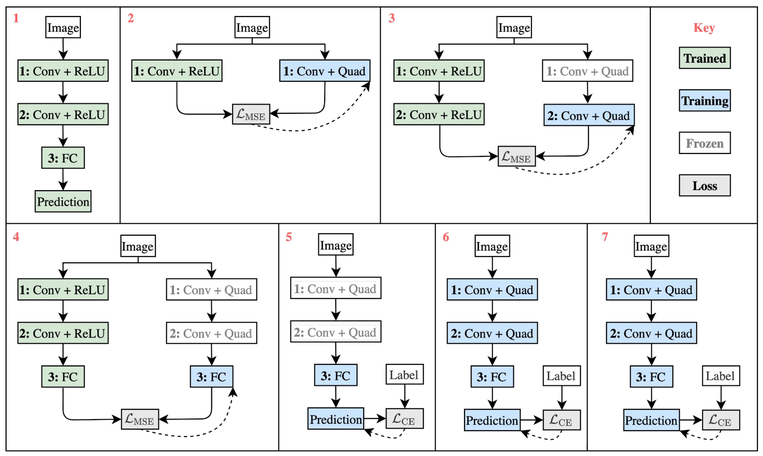


Non-linear polynomial activation functions introduce *escaping activations* during inference.

Solution: Replace-And-Retrain

Quadratic Imitation Learning (QuaIL)

Quadratic Imitation Learning (QuaIL) iteratively builds and trains a neural network with polynomial activations by mimicking the intermediate representation values of a trained ReLU network.

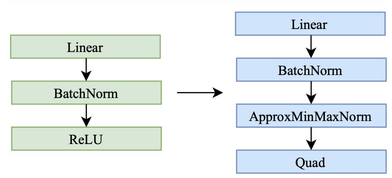


An overview of the QuaIL setup for a simple three-layer network. **1)** Train a baseline network with the ReLU activation function. **2)** Clone the first layer of the ReLU network, copy over the trained weights, and replace ReLU with Quad. Minimize the M.S.E. loss between the first-layer intermediate representations of the two networks and backpropagate through the Quad network. **3-4)** Repeat this process for each subsequent layer (while freezing the previous layers) until the full baseline network is cloned. **5-7)** Fine-tune the Quad network by gradually unfreezing layers and training with standard C.E. Loss.

Approximate Min-Max Normalization (AMM)

Even with the QuaIL training procedure, small differences in the intermediate representations between the Quad and ReLU networks causes activation values to rapidly increase for deeper networks, leading to unstable training. This illustrates the need to bound pre-activation values during training time.

$$\hat{x} = \alpha \frac{x - \min x}{\max x - \min x} - \beta$$



Before each Quad activation, an Approx Min-Max Norm layer is placed in order to bound pre-activation values. This mitigates the escaping activation problem at training time. During training, approximations of both the minimums and maximums are calculated and stored using a weighted moving average of the true minimums and maximums (we use a smoothing factor of 1/10). When performing inference, these stored approximations are then used to perform approximate normalization.

References

[1] R Gilad-Bachrach et al, "Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy." ICML 2016

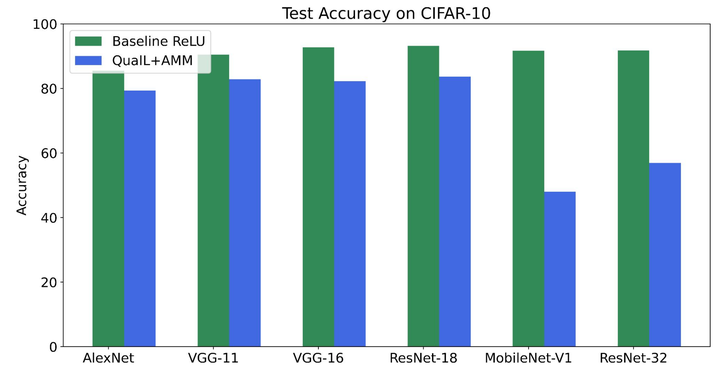
[2] P Mishra et al, "DELPHI: A Cryptographic Inference Service for Neural Networks." USENIX Security 2020

CODE

Results

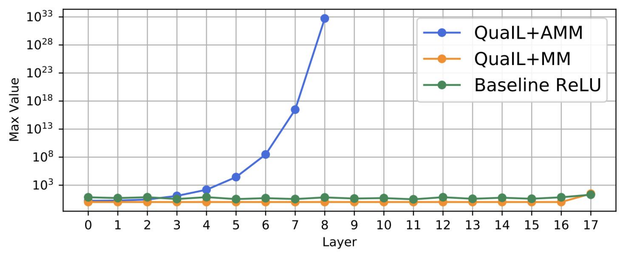
Combining QuaIL and Approx Min-Max Norm

Training networks with the Quad activation using *only* QuaIL quickly leads to escaping activations for even VGG-16. By combining the QuaIL training method *with* Approximate Min-Max Normalization, we are able to guarantee bounds on pre-activation values during training



QuaIL+AMM achieves reasonable test accuracy up to ResNet-18, but fails to generalize for even deeper networks due to escaping activations for CIFAR-10. As dataset complexity increases to CIFAR-100 and TinyImageNet, the generalization gap between the baseline ReLU networks and the QuaIL+AMM networks increases significantly.

Escaping Activations



The maximum forward activation values after each nonlinear layer at inference time for using Quail+AMM rapidly increase for deeper networks such as ResNet-18. When using the true minimum and maximum at inference time (QuaIL+MM), the maximum forward activation values are similar to a baseline ReLU network. Thus a true bounding function is required at test time to guarantee bounds on pre-activation values.

Key Takeaway

Low degree polynomial activation functions introduce *escaping activations* which lead to unstable inference and training for deep neural networks.