Designing Approximated Machine Learning Models in Python for Homomorphic Evaluations

Lando Momo Paul Junior (UBa23EP021)

Supervisor: Miss AIKATA
Co-Supervisor: Pr. FOUOTSA Emmanuel

Department of Applied Cryptology and Cybersecurity
National Higher Polytechnic Institute, The University of Bamenda

June 2025

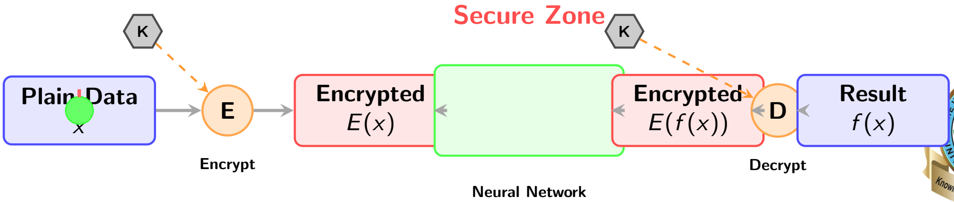
Overview

Introduction: Why Privacy Matters

- Every day, we share sensitive data: medical records, bank details, personal photos.
- Machine learning can help us diagnosing diseases, catching fraud but it needs our data.
- Problem: Sharing data risks leaks or misuse. How do we protect it?

Solution: Homomorphic Encryption

A way to compute on encrypted data without unlocking it like a locked safe you can still work with!



Problem Statement: The Privacy Challenge

- Data breaches are rising: 2.6 billion personal records exposed in 2021–2024.
- Machine learning needs data, but neural networks use complex math (e.g., ReLU) that doesn't work with encrypted data.
- Current solutions either lose accuracy or are too slow for real-world use.

Why It Matters

Insecure ML processing exposes millions of sensitive records to breaches, violating healthcare privacy laws and data protection regulations while undermining patient trust.

Applebaum, A. (2024). *2023 Data Breach Report*. Cybersecurity Insights Journal.

What’s the Issue?

Challenges with Neural Networks

1. Non-linear functions (ReLU: $\max(0, x)$) can’t be computed on encrypted data.
2. Polynomial approximations are inaccurate or slow.
3. Errors pile up in deep networks.

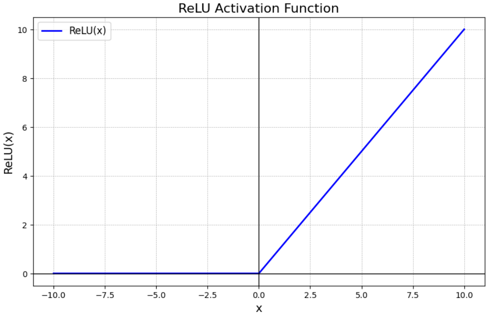
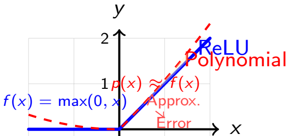


Figure: ReLU function (Jadon, 2025).



Objectives: What We Aimed to Do

Main Goal

Build neural networks that work on encrypted data with good accuracy and speed.

- Create better polynomial approximations for ReLU.
- Control errors in encrypted computations.
- Optimize performance for practical use.
- Test on real-world data (MNIST dataset).

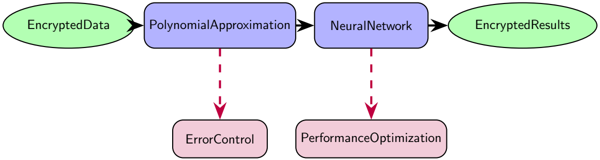


Figure: Secure neural network processing workflow.

Python Implementation: How We Built It

- TenSEAL & OpenFHE: Homomorphic encryption with CKKS scheme
- PyTorch: Neural network design & training
- NumPy: Efficient numerical computations
- Custom Approximations: Optimized ReLU polynomials

Why Python? **Flexibility + Rich Libraries = Secure & Fast AI**

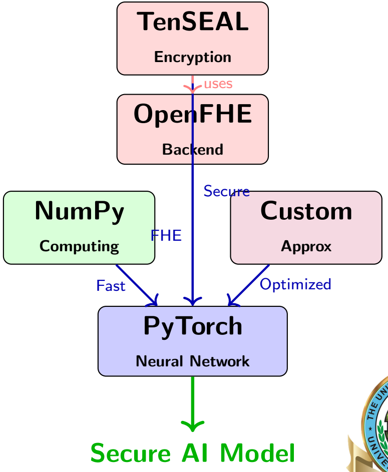
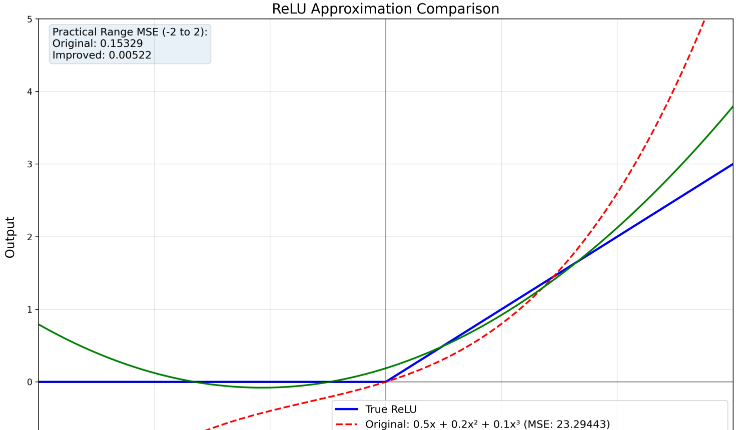


Figure: Architecture flow of Python components

Results: What We Achieved

- Accuracy: 90% on MNIST dataset, 100% agreement between plain and encrypted models.
- Improved ReLU: $12.12\times$ better approximation (MSE $= 0.00522$).
- Security: Maintained 128-bit security with CKKS scheme.



Key Numbers

Metric	Original	Our Work
MSE (ReLU)	0.15329	0.00522
Accuracy	-	90%
Processing Time	-	77.2s/sample

Table: Performance improvements.

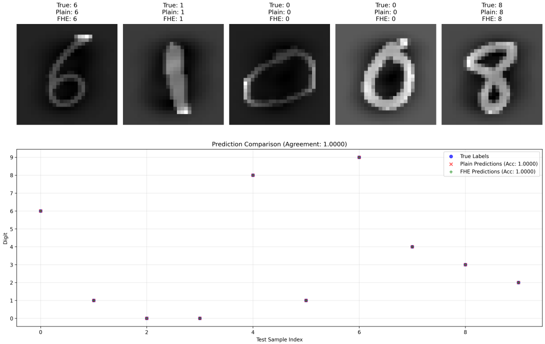


Figure: Learning convergence

Challenges: What We Faced

- Slow Processing: 77.2 seconds per sample 8.5 days for 10,000 samples!
- Small Scale: Tested on 10 samples; larger datasets need validation.
- Error Buildup: Approximation errors can grow in deeper networks.

Why It's Tough

Homomorphic encryption adds noise and computational overhead.

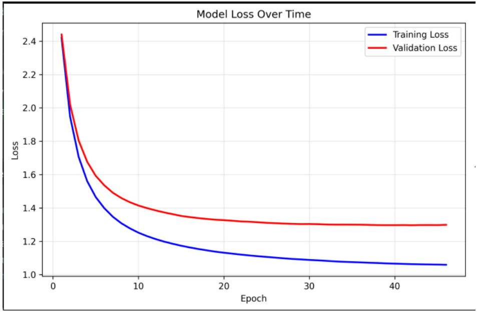


Figure: (a) Training loss showing error buildup

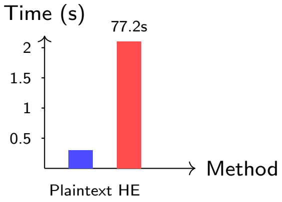


Figure: (b) Processing time comparison.

Conclusion: What We Learned

- We built a neural network that works on encrypted data with 90% accuracy.
- Our ReLU approximation is $12.12\times$ better than older methods.
- It's secure (128-bit) and works on real data (MNIST).

Impact

This could protect sensitive data in healthcare, finance, and more!

Perspectives: What's Next?

- Faster Processing: Use GPUs or FPGAs to cut down time.
- Bigger Tests: Try larger datasets and deeper networks.
- Real-World Use: Apply to medical diagnosis or fraud detection.

Examples

Imagine hospitals sharing encrypted patient data for AI diagnostics safe and secure!

References

Applebaum, A. (2023). *2023 Data Breach Report*. Cybersecurity Insights Journal.

Cheon, J. H., et al. (2017). Homomorphic encryption for arithmetic of approximate numbers. *ASIACRYPT*, 409–437.

Jadon, S. (2025). Different Activation Functions and their Graphs. (Source from dissertation).

The End