Efficient CNN Building Blocks for Encrypted Data

Authors:

Nayna Jain (International Institute of Information Technology, Bangalore and IBM Systems, US)
Karthik Nandakumar (Mohammad Bin Zayed University of Artificial Intelligence)
Nalini Ratha (University of Buffalo, SUNY)
Sharath Pankanti (Microsoft)
Uttam Kumar (International Institute of Information Technology, Bangalore)

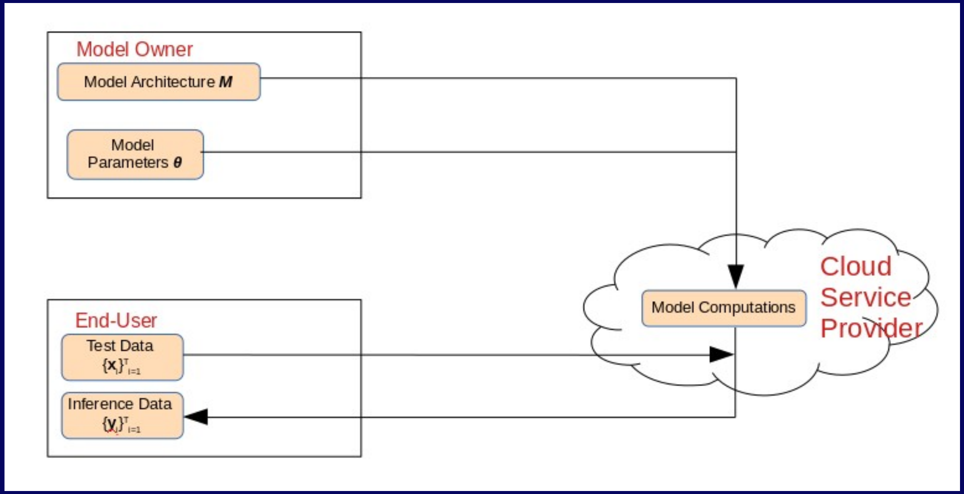
Agenda

- Problem Statement
- State of the art
- Proposed Approach
- Experimental Results

Problem Statement

MLaaS – Machine Learning as a Service

Enabled the broadened use of deep learning techniques like Convolution Neural Networks(CNN) by enabling offloading of the resource intensive operations to the Cloud.



Data
End-User/Devices
Like IoT/Mobile

Training
Generate Model

Deploy Model
For Inference

The owners for each of these steps can be different

CONCERNS

Loss of Privacy of user
Government compliance for eg. General Data Protection Regulation(GDPR)

Key Question

How do I extract insights from the data without affecting user privacy and while maintaining government compliance ?

State of the art

Privacy Preserving Machine Learning

Privacy of the input, output and model
Correctness of the result

Popular Techniques

Properties\ Techniques	Fully Homomorphic Encryption(FHE)	Secure Multiparty Computation (SMPC)
Definition	*Form of encryption that allows to perform arbitrary calculations on encrypted data without decrypting it first*	*Multiple distrusting parties jointly compute a function over their inputs while keeping those inputs private.*
Technique	Ring Learning With Errors (RLWE) based	Shamir Secret Sharing/Garbled Circuits
Computation Example	$A + B = C$ $E(a) + E(b) = E(C)$	$F(x,y,z) = max(x,y,z)$
Bottleneck	Computation	Communication (Network)
Use Cases	Outsourcing to Cloud	Joint computations
Involved Party	Offline	Online
Design	Centralized	Distributed
Security Assumptions	Minimal	Minimal

State of the art

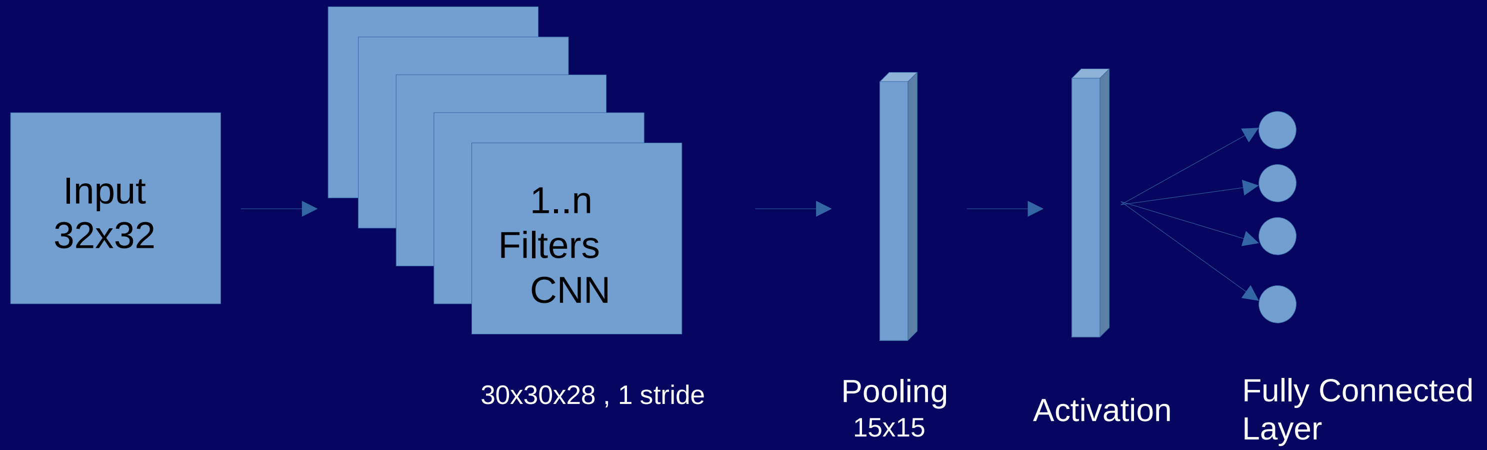
Paper	Technique	Machine Learning Algorithm	Model	Inference/Training
Machine learning classification over Encrypted Data	Additive Homomorphic Encryption	Hyperplane Decision, Naive Bayes, Decsion Trees	Encrypted Data/ Encrypted Model	Inference
SecureML, A system for scalable privacy preserving machine learning	Secure Multi-party computation (Two-party)	Logistic Regression, Linear Regression and Neural Network (Fully connected)	Encrypted Data. Model is generated as it addresses training.	Training
Faster CryptoNets	Homomorphic Encryption(BFV)	Convolution Neural Network	Encrypted Data/ Plaintext Model.	Inference
SHE:A fast and accurate deep neural networks for encrypted data	Homomorphic Encryption (TFHE)	Convolution Neural Network	Encrypted Data/ Plaintext Model	Inference
LoLa: Low Latency Privacy Preserving Inference	Homomophic Encryption (BFV) – Different packing scheme than FCN	Convolution Neural Network	Encrypted Data/ Plaintext Model	Inference
Towards Deep Neural Network Training on Encrypted Data	Homomorphic Encryption (BGV)	Neural Network	Encrypted Data. Model is generated	Training
Privacy Enchanced Decision Tree Inference	Homomorphic Encryption (CKKS)	Decision Trees	Encrypted Data/ Plaintext Model	Inference
Efficient CNN Building Blocks for Encrypted Data	Homomorphic Encryption (CKKS)	Convolution Neural Network	Encrypted Data / Encrypted Model	Inference

Contributions

- Proposed MLaaS Scenario for convolution neural networks where both data and model are encrypted
- Presented use of CKKS Scheme for Convolution Neural Network thereby avoiding need of quantization
- Detailed analysis of impact of homomorphic encryption security parameters over convolution operators
- Exploring multithreading strategy for performance improvement

Convolution Neural Network (Key Operators)

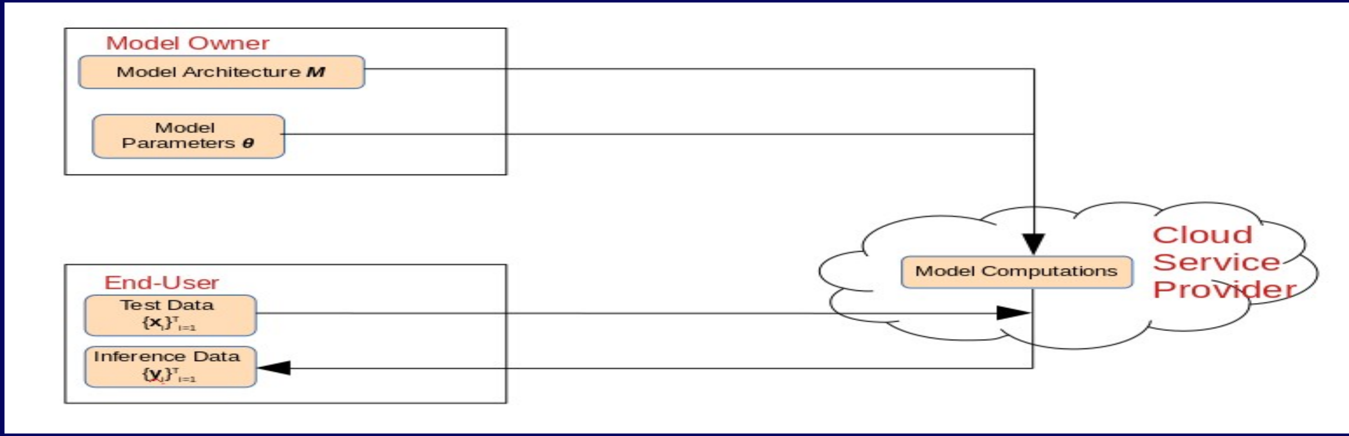
- Convolution – Inner product of input with kernels that can map to different features
- Non-Linear Activation – Adds non-linearity to the output of the previous layer
- Pooling – Reduces the dimension of the data



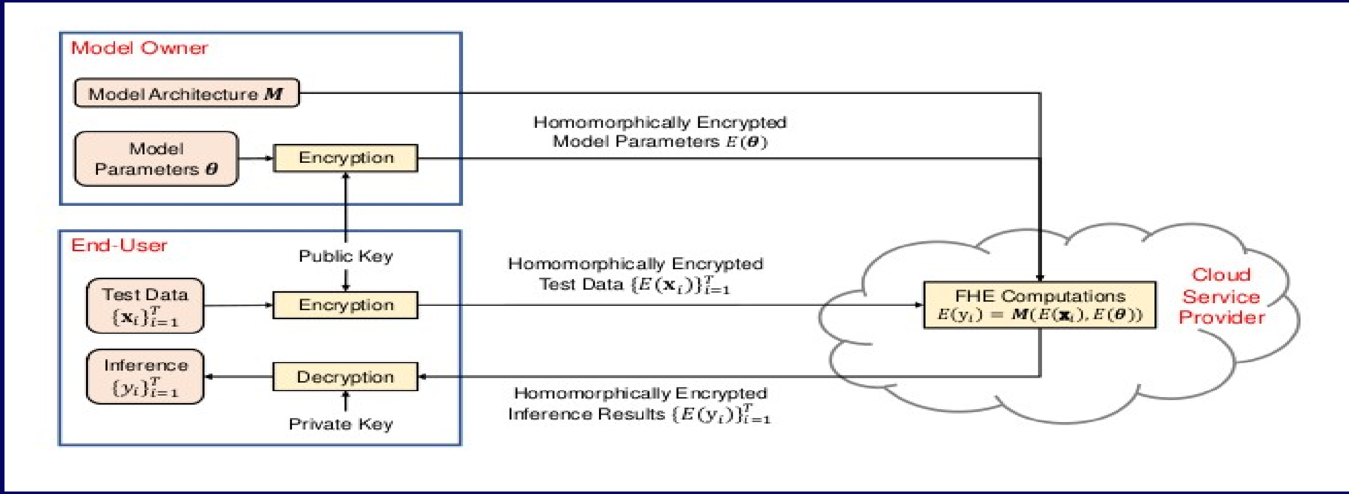
Proposed Approach

MLaaS Revisited

Unencrypted Domain



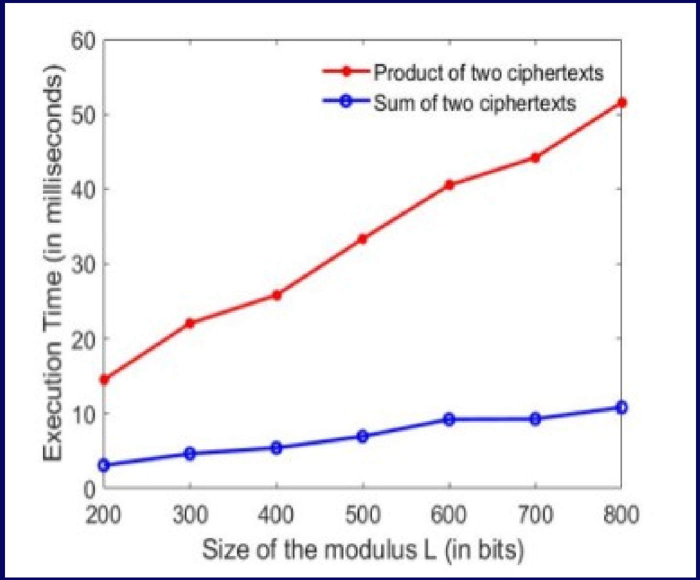
Encrypted Domain



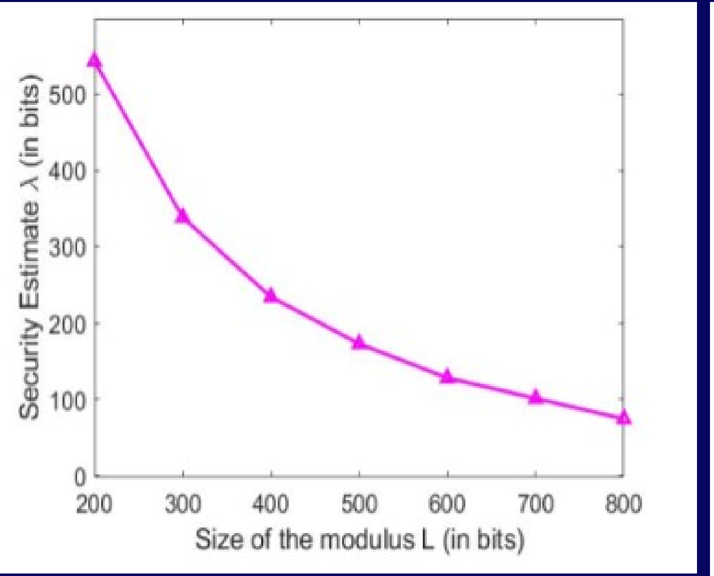
CKKS (Jung Hee Cheon, Andrey Kim, Miran Kim, Yongsoo Song)

- Only scheme to support approximate numbers than integers
- Plaintext space is complex numbers
- Scheme is based on treating encryption noise as part of errors in the approximate computations
- Currently, not all libraries support bootstrapping in CKKS
- Security Level depends on modulus of the cyclotomic ring(m), multiplicative depth(L) and computation precision(r)
- Supports SIMD parallelization where slots for multiple data is defined by modulus of the cyclotomic ring

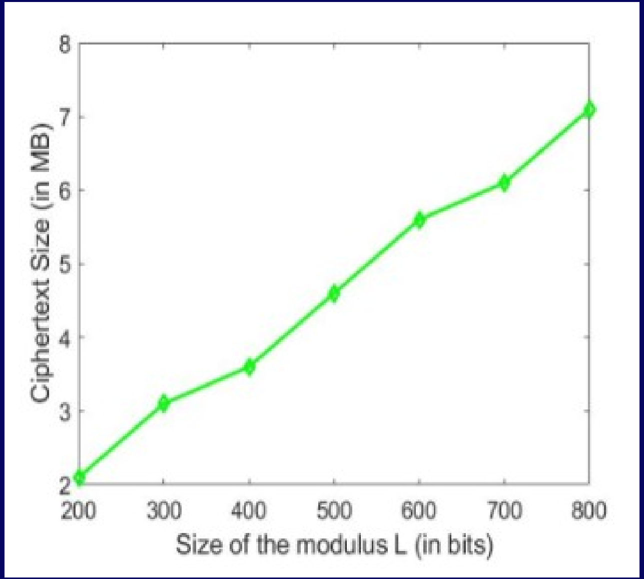
Impact of Depth



Computational Time



Security Level



Ciphertext Size

Multiplicative depth vs Accuracy

Activation Function - Relu

$$ReLU(a) = max(0, a) = \begin{cases} 0, & \text{if } a \leq 0 \\ a, & \text{if } a > 0. \end{cases}$$

Polynomial Approximation - Relu

$$g(u) = 0.47 + 0.50u + 0.09u^2, u \in [-\sqrt{2}, \sqrt{2}].$$

Max Pooling

$$f(a, b, c, d) = max(a, max(b, max(c, d)))$$

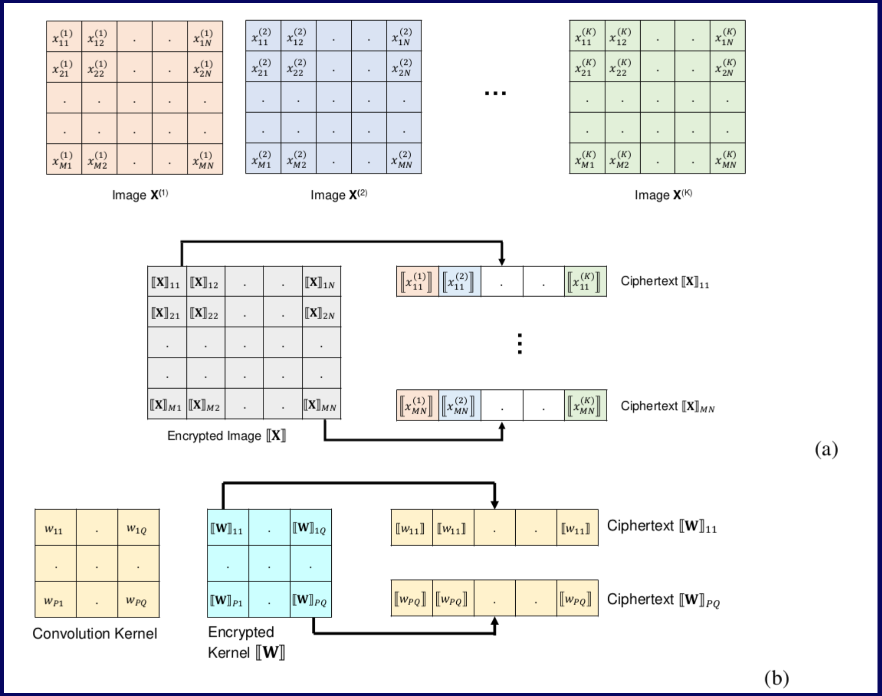
Mean Pooling

$$meanpool(a, b, c, d) = (g(a) + g(b) + g(c) + g(d))/4,$$

Depth Requirements for max() function is much higher compared to polynomial approximation or mean pooling

Depth is inversely proportional to accuracy

Packing Schemes



- Use SIMD parallelization through different packing schemes
- Multiple image packing has advantage of throughput over latency. It is optimized for batched inference
- Single image packing has advantage of latency over throughput. It is optimized for single inference
- Ensure minimum interaction between slots

Multithreading

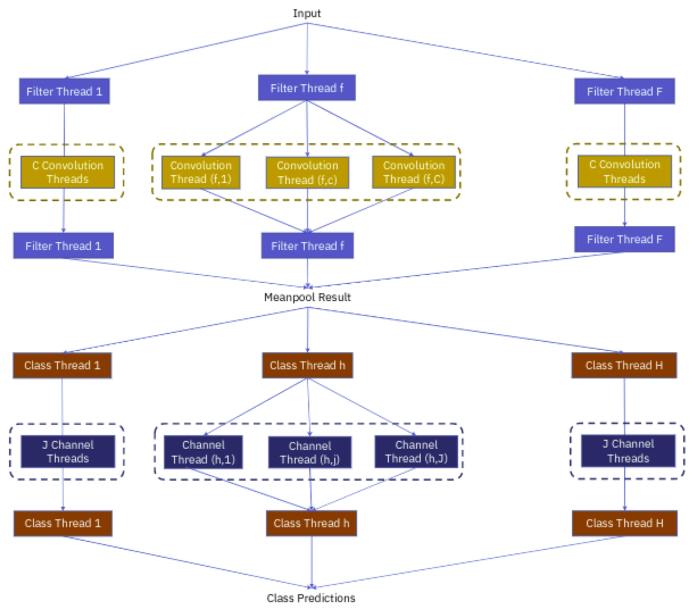
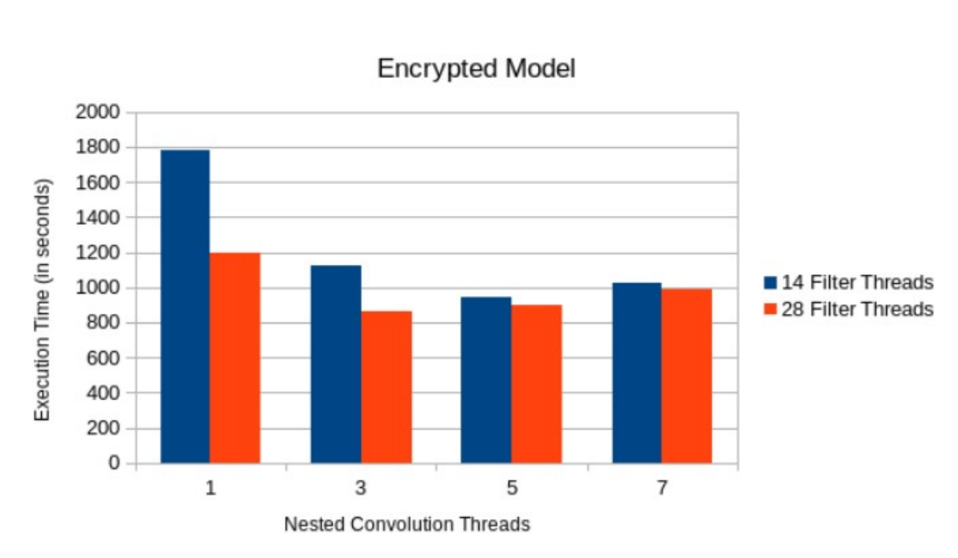


Figure 6: Proposed nested multi-threading strategy.

- Number of Filter threads – (F = 14 vs 28)
- Number of Nested threads – For each filter thread, 7 convolution threads and 10 class threads
- Avoid locking by ensuring data segregation between different threads
- Ensure minimum sharing of resources by dedicated cores

Experimental Results
Dataset – MNIST

Multithreading Analysis



As the number of threads increases performance improves, but if they start contending for same computing resource, it results in adverse effect on the benefit of multithreading.

28 * 3 = 84 Threads
28 * 5 = 140 Threads
28 * 7 = 196 Threads

140 and 196 threads perform poorly than 84 threads.

Inference Results

Operation	Execution Time (in seconds) for Single Filter and Single Thread
Convolution	487.4
Approximate ReLU	102.1
Mean pooling	16.9
Fully Connected	123.4
Total (including overhead)	812.6

With multithreading, the final result we had is – 561 seconds with 70-80 threads. This is 40x improvement over total single threaded strategy.

Amortized time taking benefit of SIMD = Time Taken / Number of images that can be packed

= 561 / 16384

= 0.034 = 34 milliseconds

Single Image Packing Reference Time = 8.8 seconds

Future Work

- Improve multithreading strategy
- Experiment different packing strategies
- Explore further optimization techniques

THANKS !!

nayna.jain@iiitb.org/naynjain@ibm.com,Karthik.nandakumar@mbzuai.ac.ae, nratha@buffalo.edu, sharat.pankanti@gmail.com, uttam@iiitb.ac.in