VLSI ARCHITECTURE DESIGN
PROJECT

AN IMPROVED LOGARITHMIC MULTIPLIER FOR ENERGY-EFFICIENT NEURAL COMPUTING

Ajitesh, Chinmay, Chirag,
Shannon, Sreyas, Vedant

OVERVIEW

- Introduction
- Multipliers
- Proposed Multiplier
- Neural Network Application
- Accuracy Metrics
- Hardware Metrics
- Conclusion

INTRODUCTION

- Focus: Energy-efficient multiplication for neural networks (NNs)
- Key Contribution: Improved Logarithmic Multiplier (ILM) rounds inputs to nearest powers of two using a novel Nearest-One Detector (NOD)
- Applications: Evaluated on MNIST (MLP) and CIFAR-10 (AlexNet) datasets
- Significance: Leverages NN error tolerance for hardware efficiency

CONVENTIONAL VS APRROXIMATE MULTIPLIER

Conventional Multipliers

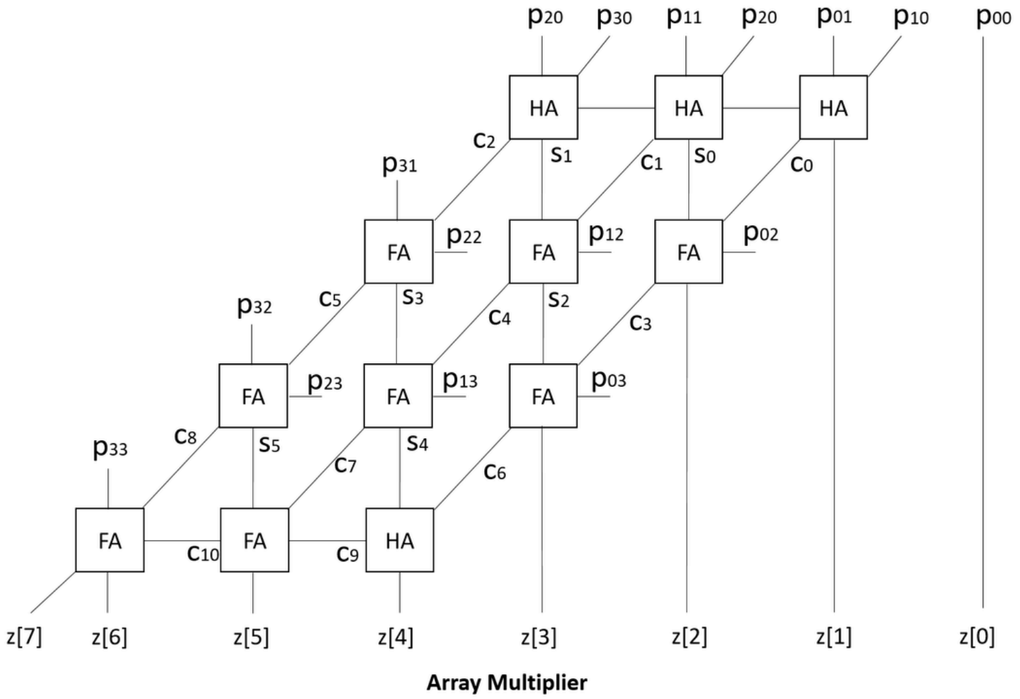
- Perform accurate and deterministic computations
- Used in precision-critical applications (e.g., scientific computing, cryptography)
- Higher area, power, and delay overhead
- Complex hardware with full adder trees
- Less tolerant to errors or approximations

Approximate Multipliers

- Trade off accuracy for efficiency
- Used in error-resilient applications (e.g., machine learning, signal processing)
- Reduced power consumption, area, and latency
- Simplified logic or truncated partial products
- Higher Throughput in energy-constrained systems

ARRAY MULTIPLIER

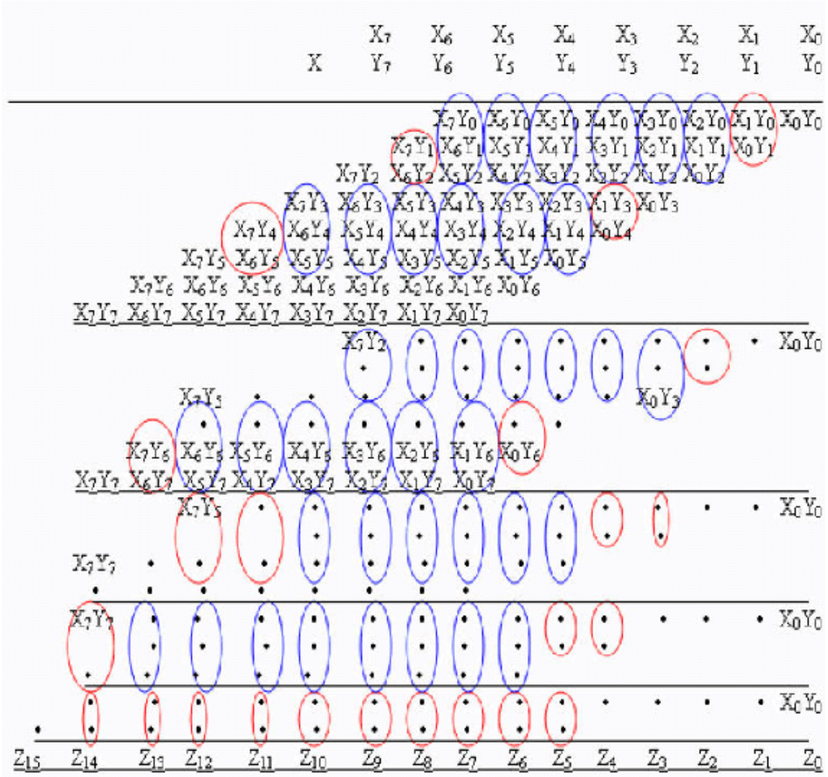
$$
\begin{array}{cccccccc}
 & & a_3 & a_2 & a_1 & a_0 & & \\
 & x & b_3 & b_2 & b_1 & b_0 & & \\
\hline
 & & p_{30} & p_{20} & p_{10} & p_{00} & & \\
 & p_{31} & p_{21} & p_{11} & p_{01} & x & & \\
p_{32} & p_{22} & p_{12} & p_{02} & x & x & & \\
p_{33} & p_{23} & p_{13} & p_{03} & x & x & x & \\
\hline
z_7 & z_6 & z_5 & z_4 & z_3 & z_2 & z_1 & z_0
\end{array}
$$



Array Multiplier

Array multiplier is similar to how we perform multiplication with pen and paper i.e. finding a partial product and adding them together. It is simple architecture for implementation.

WALLACE TREE MULTIPLIER

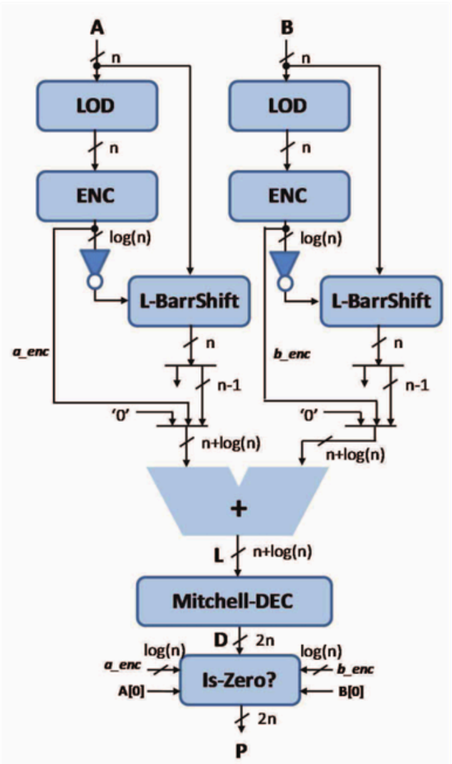


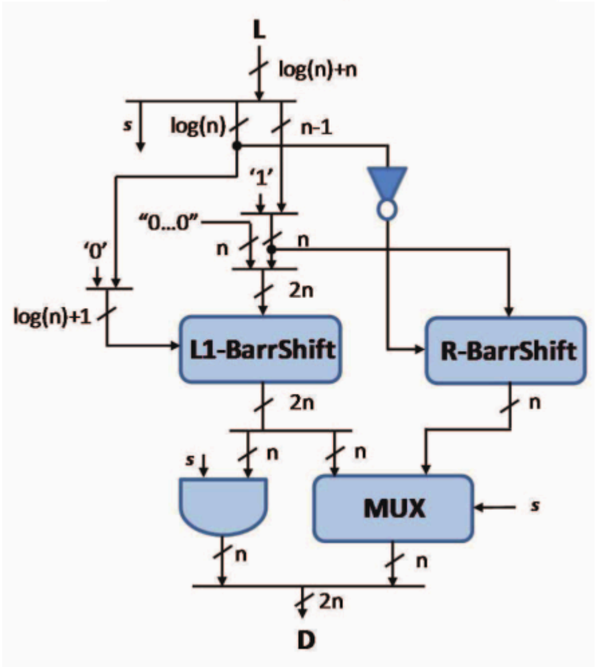
MITCHELL

$$P' = 2^{k_1+k_2}(1 + x_1 + x_2) \qquad x_1 + x_2 < 1$$

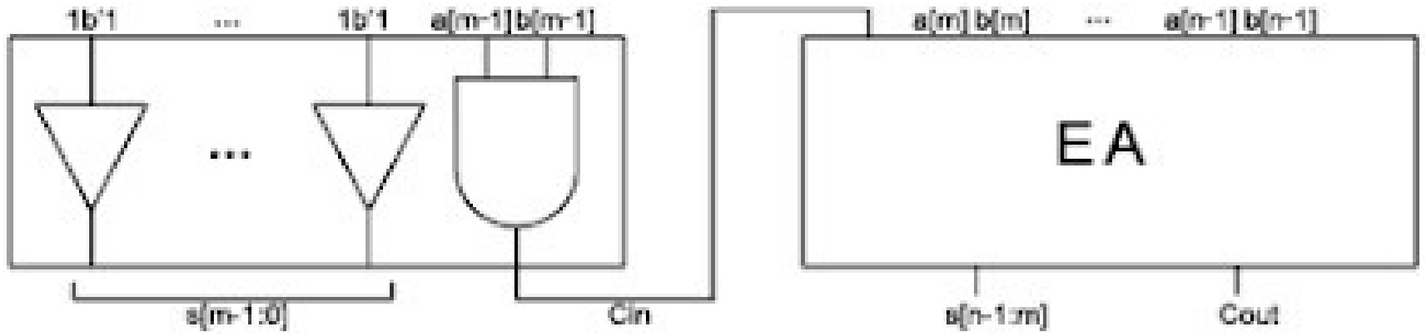
$$P' = 2^{1+k_1+k_2}(x_1 + x_2) \qquad x_1 + x_2 \geq 1.$$

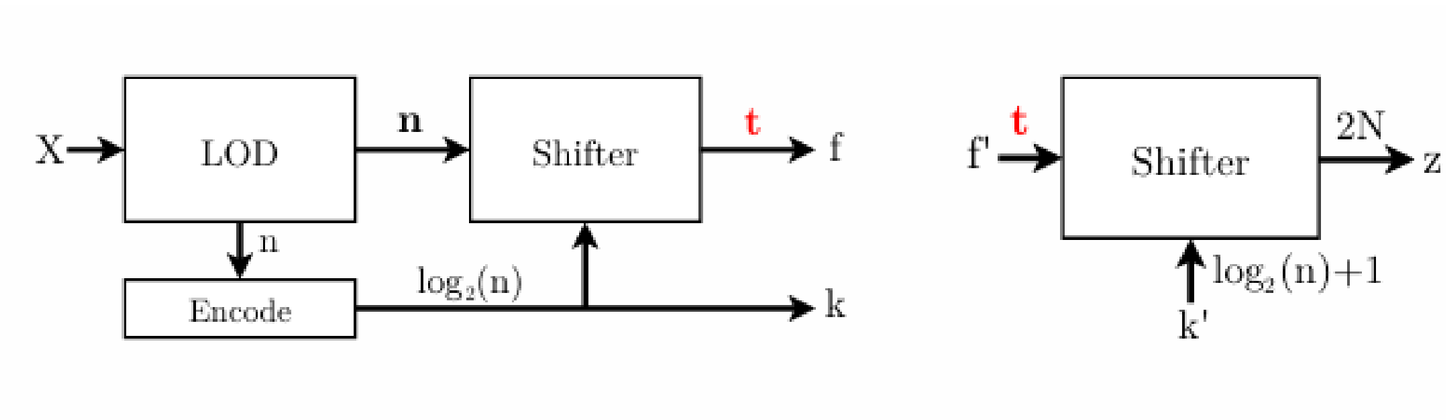
- Approximates multiplication using logarithms which turns multiplication into addition.
- Steps:
  - Convert operands to log domain via shifts and leading-one detection.
  - Add logs.
  - Use antilog (bit shift) to get the product.
- Hardware: Uses adders and shifters making it lightweight and efficient.





ALM-SOA





USES BOTH THE SET-ONE ADDER AS THE MANTISSA ADDER AND A TRUNCATED BINARY LOGARITHMIC CONVERTER (TLBC).

ILM

TABLE 1
The Proposed versus Conventional Full Adder

a	b	c_{in}	Conventional FA		Proposed FA	
			sum	c_{out}	sum	c_{out}
0	0	0	0	0	0	0
0	0	1	1	0	1	0
0	1	0	1	0	1	0
0	1	1	0	1	0	1
1	0	0	1	0	1	0
1	0	1	0	1	N/A	N/A
1	1	0	0	1	0	1
1	1	1	1	1	N/A	N/A
Conventional	$sum = \overline{a}\overline{b}c_{in} + \overline{a}b\overline{c_{in}} + a\overline{b}\overline{c_{in}} + abc_{in}$ $c_{out} = ab + ac_{in} + bc_{in}$					
Proposed	$sum = \overline{b}c_{in} + \overline{a}b\overline{c_{in}} + a\overline{b}$ $c_{out} = ab + bc_{in}$					

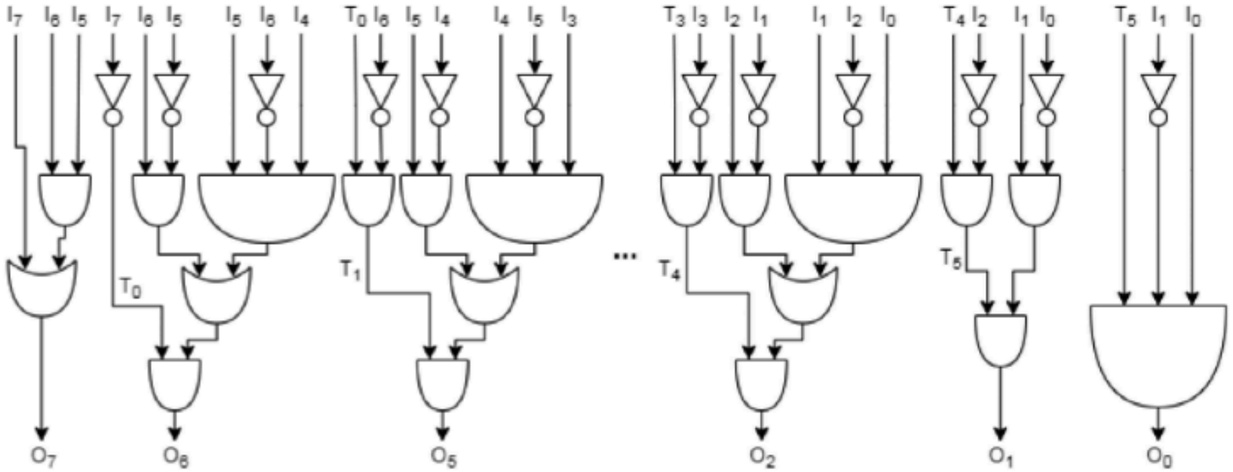
Algorithm 1. Proposed Approximation for $log_2 N$

$N = 2^k(1+x) = 2^{k+1}(1-y)$
if $N - 2^k < 2^{(k+1)} - N$ **then** ▷ use underestimate
  $x = N/2^k - 1$
  $log_2 N \approx k + x$
else ▷ use overestimate
  $y = 1 - N/2^{k+1}$
  $log_2 N \approx k + 1 - y$
end if

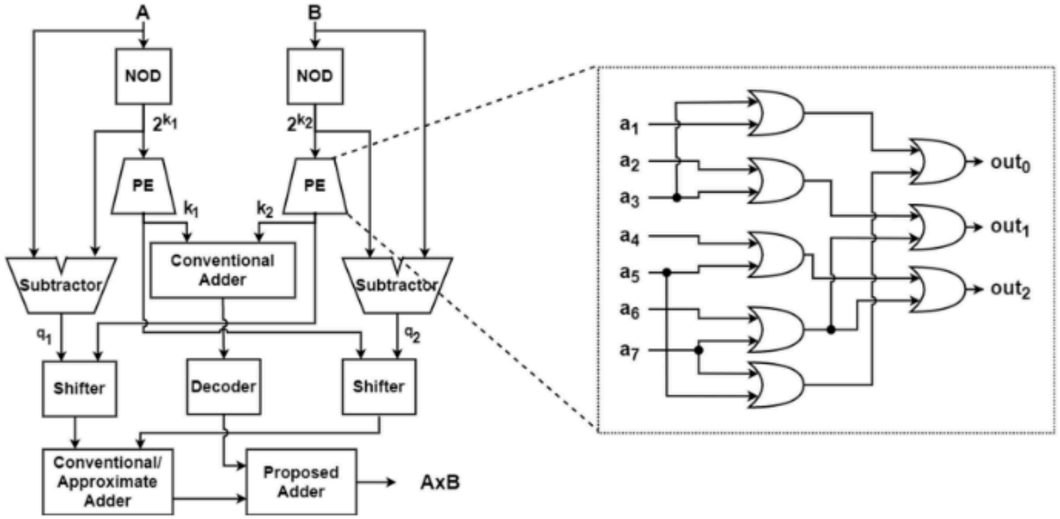
Algorithm 2. Proposed Logarithmic Multiplication

procedure MA, B
  A, B: inputs, γ: approximate output
  $2^{k_1} \leftarrow \text{NOD}(A)$,
  $k_1 \leftarrow \text{PE}(2^{k_1})$,
  $q_1 \leftarrow A - 2^{k_1}$, ▷ for steps 3-5 see (14)
  $2^{k_2} \leftarrow \text{NOD}(B)$,
  $k_2 \leftarrow \text{PE}(2^{k_2})$,
  $q_2 \leftarrow B - 2^{k_2}$, ▷ for steps 6-8 see (15)
  $q_1 2^{k_2} \leftarrow q_1 << k_2$,
  $q_2 2^{k_1} \leftarrow q_2 << k_1$,
  $2^{k_1+k_2} \leftarrow \text{DEC}(k_1 + k_2)$,
  $\gamma \leftarrow 2^{k_1+k_2} + q_2 2^{k_1} + q_1 2^{k_2}$. ▷ see (16)

ILM



(b) Proposed nearest-one detector (NOD) Circuit.



(a) Block diagram of the ILM and the priority encoder [35].

NOVELTY:

1. **ESTIMATION(UNDER AND OVER)**

2. **PROPOSED ADDER**

NEURAL NETWORK

BASICS OF NEURAL NETWORK

- **LAYERED STRUCTURE:** NNS CONSIST OF AN INPUT LAYER, ONE OR MORE HIDDEN LAYERS (E.G. CONVOLUTIONAL, POOLING, FULLY-CONNECTED), AND AN OUTPUT LAYER.
- **NEURON MODEL & WEIGHTS**: EACH NEURON MULTIPLIES ITS INPUTS BY LEARNED WEIGHTS, SUMS THEM, APPLIES AN ACTIVATION, AND PASSES THE RESULT FORWARD.
- **TRAINING VIA WEIGHT UPDATES**: DURING TRAINING (BACK-PROPAGATION), WEIGHTS ARE ITERATIVELY ADJUSTED SO THE NETWORK LEARNS TO PERFORM TASKS LIKE CLASSIFICATION OR PATTERN RECOGNITION.

NEURAL NETWORK

WHY USE APPROXIMATE MULTIPLIERS IN NNS

- **MULTIPLIERS DOMINATE COST:** IN HARDWARE, FIXED-POINT OR FLOATING-POINT MULTIPLIERS CONSUME MOST AREA, DELAY AND ENERGY IN THE MULTIPLY-ACCUMULATE OPERATIONS OF EACH NEURON.
- **LOGARITHMIC MULTIPLIER ALTERNATIVE:** BY CONVERTING OPERANDS TO A (TRUNCATED) BASE-2 LOGARITHM, MULTIPLICATION REDUCES TO SHIFTS + ADDS, GREATLY CUTTING HARDWARE COMPLEXITY.
- **CONTROLLED, BOUNDED ERROR:** TRUNCATING THE LOG DOMAIN INTRODUCES AN ERROR, WHICH IN A WELL-DESIGNED PIECE-WISE OR MITCHELL-BASED SCHEME CAN BE KEPT SMALL AND UNBIASED (TWO-SIDED), AVOIDING SYSTEMATIC DRIFT.
- **ENERGY–ACCURACY TRADE-OFF:** APPROXIMATE MULTIPLIERS LET YOU BALANCE INFERENCE ACCURACY AGAINST SIGNIFICANTLY LOWER AREA AND POWER—CRITICAL FOR ENERGY-EFFICIENT, HIGH-THROUGHPUT NN ACCELERATORS.

WHY ONLY INFERENCE ON APPROX MULTIPLIER

Training - Conventional Multiplier

Evaluation - Approximate Multiplier

Gradient Descent

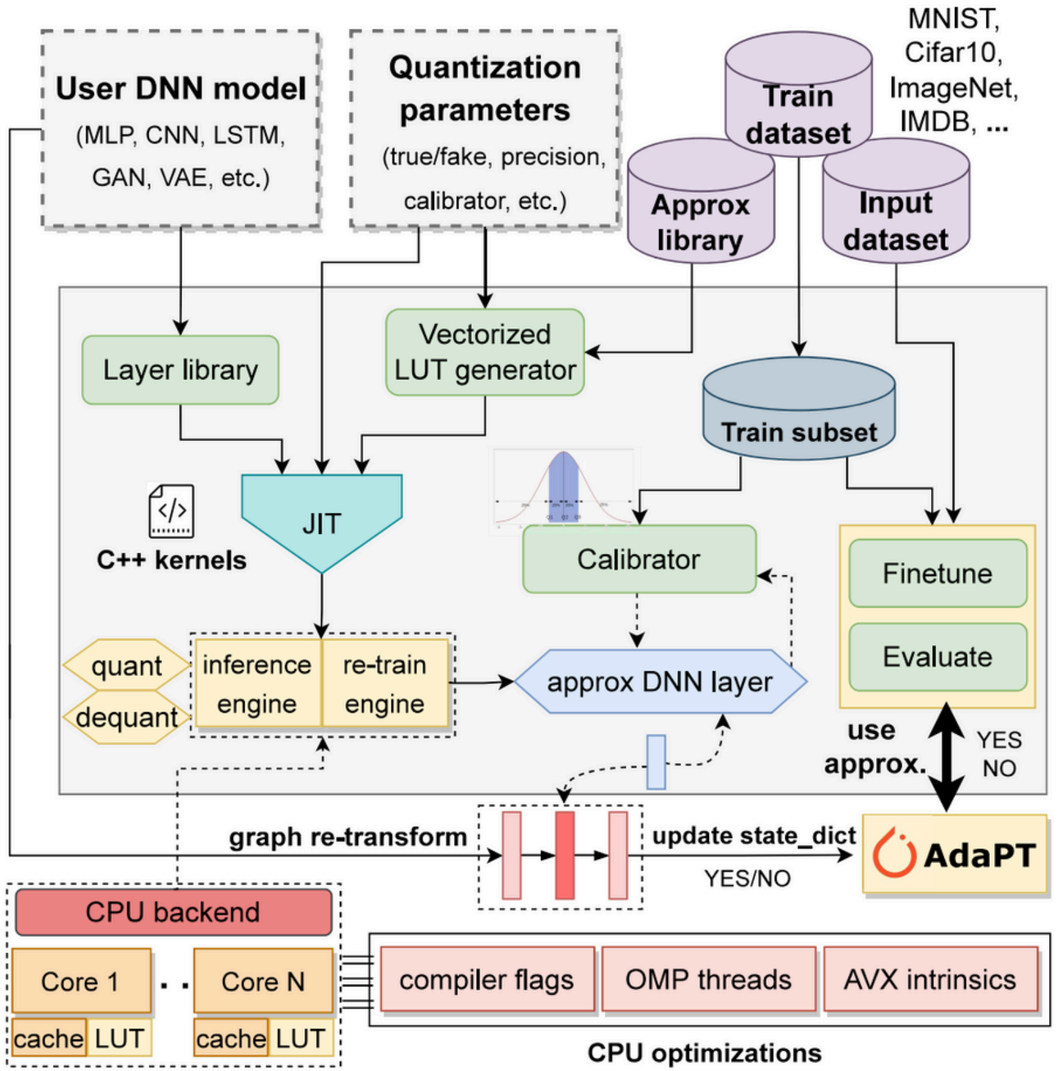
$$\theta_t = \theta_{t+1} - \eta \cdot \nabla\theta_{t+1}$$

Variance in each epoch of gradient descent

Exact Multiplier **0**

Approx Multiplier $\eta^2 \cdot T \cdot \sigma^2$

ADAPT FRAMEWORK



Accuracy Metrics

$$\mathrm{AE} = \frac{1}{N}\sum_{i=1}^{N}|A_i - E_i|$$

- AE (Average Error): The mean error across all test cases.

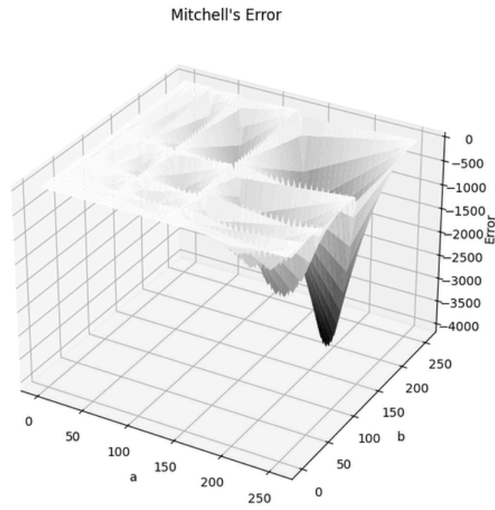
$$\mathrm{MRED} = \frac{1}{N}\sum_{i=1}^{N}\frac{|A_i - E_i|}{A_i}$$

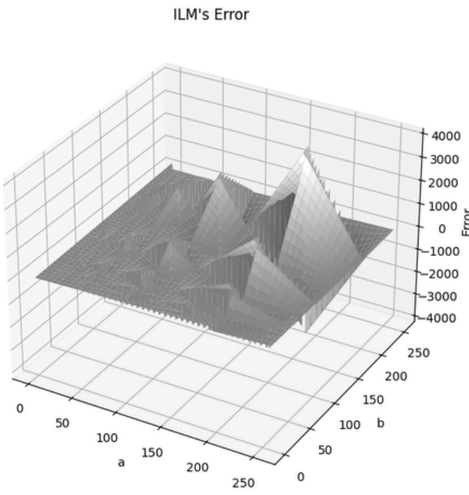
- MRED (Mean Relative Error Distance): Average Relative Error, relative error is error/exact value. Expresses average relative deviation from the correct result

$$\mathrm{NMED} = \frac{\sum_{i=1}^{N}|A_i - E_i|}{N \times Error_{max}}$$

.

- NMED (Normalized Mean Error Distance): Average Error relative to the maximum achieved error





Accuracy Metrics

- All the trends match.
- ILM-0 delivers the best overall accuracy across both distributions, balancing very low absolute and normalized errors.

Distribution	Multiplier Type	|AE|	MRED	NMED
Uniform	Mitchell	607.15	0.0379	0.0093
	ALM-SOA-3	607.52	0.0377	0.0093
	ALM-SOA-5	599.94	0.0373	0.0092
	ILM-0	0.2662	0.0288	0.0065
	ILM-5	4.3083	0.0351	0.0066
	ILM-9	23.1226	0.1771	0.0353
Normal	Mitchell	96.04	0.0358	0.0015
	ALM-SOA-3	173.73	0.0360	0.0027
	ALM-SOA-5	171.91	0.0358	0.0026
	ILM-0	3.3567	0.0394	0.0012
	ILM-5	9.6572	0.1152	0.0018
	ILM-9	21.7361	0.0334	0.0049

Table 2: Error Metrics of the LMs for General Input Distributions

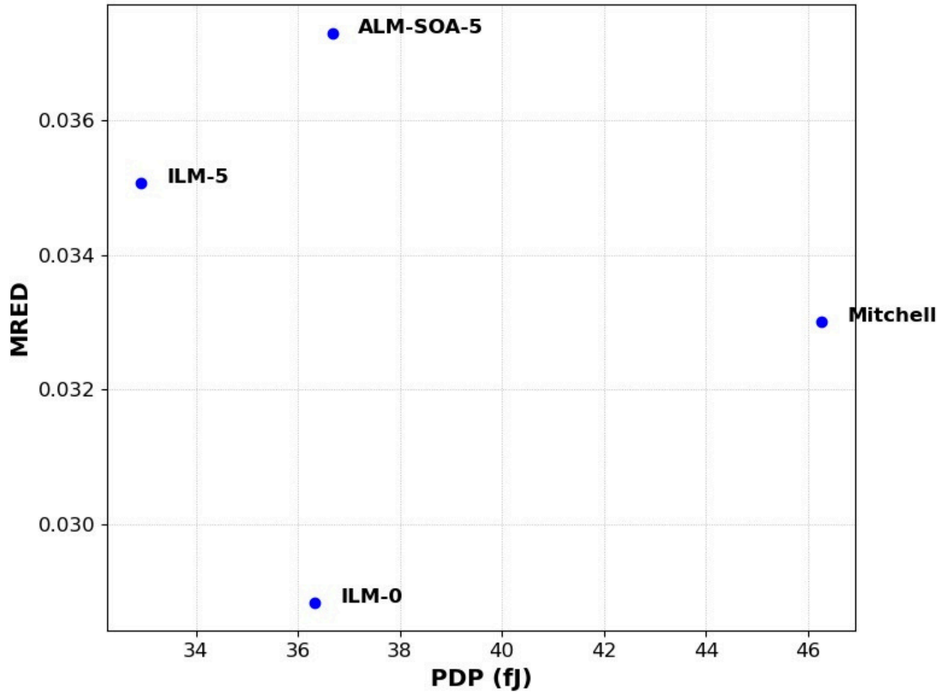
Multiplier	|AE|	MRED	NMED
ILM0	0.1177	0.0870	0.2564
ALM3	0.7526	0.4803	0.3355
Mitchell	7.1602	3.6273	0.3619

Table 3: Error Metrics of the LMs for the NN Workloads

Hardware Metrics

Design	Power (μW)	Delay (ns)	Area (μm^2)	PDP (fJ)
Array	18.399	3.055	499.374	56.2089
Wallace	19.590	2.878	489.060	56.3800
Mitchell	12.960	3.570	431.138	46.2672
ALM-SOA-5	10.983	3.340	413.492	36.6832
ILM-0	9.982	3.640	492.315	36.3345
ILM-5	9.122	3.610	413.491	32.9304

Table 1: Hardware Metrics of the Logarithmic Multipliers



ILM-0	1261.98	4.521	45.5826	206.0789346
ILM-5	1221.47	4.321	45.5826	196.9624146

- ILM-0 provides the best balance of low error and energy, ILM-5 is best if absolute energy minimization is the priority, and Mitchell is least energy-efficient.
- PDP (Power-Delay product): This measures the energy used per computation by the multiplier
- All the trends match.

CONCLUSION

- **Implementation**: Successfully implemented the Improved Logarithmic Multiplier (ILM), specifically the ILM-5 variant (approximating 5 LSBs), based on the paper's novel log2N approximation method.
- **Hardware Efficiency**: Verified the anticipated hardware benefits of ILM-5, demonstrating reductions in power consumption and area.
- **Neural Network Integration**: Integrated the ILM-5 multiplier into benchmark Neural Network (NN) architectures.
- **Performance Validation (NNs)**: Confirmed the paper's key findings within the NN context: Achieved superior energy efficiency compared to NNs using other multipliers. Obtained higher classification accuracy on the CIFAR-10 dataset, outperforming both other logarithmic multipliers and traditional exact multipliers.

THANK YOU

FOR THE ATTENTION