A Low-Power Arithmetic Element for Multi-Base Logarithmic Computation on Deep Neural Networks

Jiawei Xu[1], Yuxiang Huan[2], Li-Rong Zheng[1], Zhuo Zou[1]
[1]State Key Laboratory of ASIC and System, Fudan University, Shanghai, China;
[2]KTH Royal Institution of Technology, Stockholm, Sweden
E-mail: jwxu16@fudan.edu.cn, lrzheng@fudan.edu.cn, zhuo@fudan.edu.cn

***Abstract*—Computational complexity and memory intensity are crucial in deep convolutional neural network algorithms for deployment to embedded systems. Recent advances in logarithmic quantization has manifested great potential in reducing the inference cost of neural network models. However, current base-2 logarithmic quantization suffers from performance upper limit and there is few work that studies hardware implementation of other bases. This paper presents a multi-base logarithmic scheme for Deep Neural Networks (DNNs). The performance of Alexnet is studied with respects to different quantization resolutions. Base-$\sqrt{2}$ logarithmic quantization is able to raise the ceiling of top-5 classifying accuracy from 69.3% to 75.5% at 5-bit resolution. A segmented logarithmic quantization method that combines both base-2 and base-$\sqrt{2}$ is then proposed to improve the network top-5 accuracy to 72.3% in 4-bit resolution. The corresponding arithmetic element hardware has been designed, which supports base-$\sqrt{2}$ logarithmic quantization and segmented logarithmic quantization respectively. Evaluated in UMC 65nm process, the proposed arithmetic element operating at 500MHz and 1.2V consumes as low as 120 μW. Compared with 16-bit fixed point multiplier, our design achieves 58.03% smaller in area, with 73.74% energy reduction.**

I. Introduction

Recently, Deep Neural Network (DNN) based machine learning algorithms have achieved significant progress and drawn great attention in various areas, and have ranked among the most promising and powerful techniques in handling complex tasks, such as visual processing, speech recognition and so on [1]–[5]. Due to the computational complexity and memory intensity of current DNN algorithms, compression techniques are required for these trained large networks to perform on embedded systems and mobile systems. Low power consumption is also a growing demand for DNN algorithms as it is a precondition for embedding artificial intelligence into the ubiquitous electronic platform to achieve the vision of Internet of Things (IoT) to achieve connectivity among any object and to extent the current computer-based internet to a more generalized cyber-physical system.

As DNNs typically manifest considerable system resilience, algorithm-level optimization can be used to remove redundancy and facilitate hardware implementation, such as fixed-point implementation [6], pruning [7], BinaryNet [8], and LogNet [9]. Logarithmic quantization has been considered as a preferable method to relax model complexity and profit hardware efficiency. The results of LogNet show that logarithmic encoding of weights and activations is preferred over linear encoding at low-bit resolution without retraining [10]. However, there is a clear upper limit for the performance of base-2 logarithmic arithmetic, as the network will reach its upper limit at low-precise resolution and will not improve with the increase of bit width. To our best knowledge, there is few work that studies feasibility and hardware realization of logarithmic arithmetic with other bases.

In order to address the above-mentioned problem of base-2 logarithmic quantization, our work explores the base-$\sqrt{2}$ logarithmic arithmetic on weight quantization, and further improves the network performance in low-precision representation through segmented logarithmic representation. Evaluation results on Alexnet show that base-$\sqrt{2}$ logarithmic quantization achieves 6.2% higher in top-5 upper limit and 3.9% higher in top-1 upper limit than base-2 logarithmic quantization, and segmented logarithmic quantization improves top-5 accuracy by 3% and top-1 accuracy by 2.8% in 4-bit resolution. An arithmetic element hardware compatible for both base-$\sqrt{2}$ logarithmic quantization and segmented logarithmic quantization is designed and simulated in UMC 65nm process. Our design outperforms 16-bit fixed point multiplier by 58.03% smaller in area and 73.74% less in energy consumption.

II. Logarithmic Quantization

Compared to traditional uniform quantization method, logarithmic quantization not only allows networks to run more efficiently on hardware, but also achieves comparative performance in low-precise representation.

A. Logarithmic Base-2 Quantization

The dot production between weights and inputs in both convolutional layer and fully-connected layer are computed through multiply-add operations in hardware. With the use of logarithmic base-2 ($\log_2$) encoding, the multiplication can be replaced by shifting operation to achieve a more energy-efficient inference.

Due to the difference of the data range and distribution in convolutional and fully-connected layers, layer-wise quantization will be performed on every layer. After N-bit quantization using base-2 encoding, the weights or activations will be represented by a discrete element in a codebook of size 2^N. The full-scale range is defined as $FSR = round(\log_2(\max - \min))$, where $\max$ is the maximum absolute value, and $\min$ is the minimum absolute value. For the weight or activation x, $x\_\log = \log_2(|x|)$ is defined for notational convenience.

The codebook is defined as:

$$P\_Log = \{FSR - 2^N + 1, FSR - 2^N + 2, \cdots, FSR - 1, FSR\} \quad (1)$$

The threshold set is described as:

$$T\_Log = \{FSR - 2^N + \frac{1}{2}, FSR - 2^N + \frac{3}{2}, \cdots, FSR + \frac{1}{2}\} \quad (2)$$

Base-2 logarithmic representation of weight or activation x can be viewed from Fig.1, and is described by:

$$\mathrm{Log\_Quant}(x) = \begin{cases} Sign(x) \cdot 2^{\tilde{x}} & x \neq 0 \\ 0 & x = 0 \end{cases} \quad (3)$$

for $i = 1, 2, \cdots, 2^N$

$$\tilde{x} = \begin{cases} P\_Log(1) & x\_\log < T\_Log(1) \\ P\_Log(i) & T\_Log(i) \leq x\_\log < T\_Log(i+1) \\ P\_Log(2^N) & x\_\log \geq T\_Log(2^N + 1) \end{cases} \quad (4)$$

B. Logarithmic Base-$\sqrt{2}$ Quantization

Research shows that weights of larger absolute value are more important to the accuracy than those of smaller absolute value [7]. To place more code around large values, base-$\sqrt{2}$ is adopted for the logarithmic quantization.

With the change in the base, the FSR, $x\_\log$ and the $\mathrm{Log\_Quant}$ equation will change accordingly. And P_Log, T_log and $\tilde{x}$ will be decided by the equation (1), (2) and (4) shown above.

$$FSR = round(\log_{\sqrt{2}}(\max - \min))$$

$$x\_\log = \log_{\sqrt{2}}(|x|)$$

$$\mathrm{Log\_Quant}(x) = \begin{cases} Sign(x) \cdot \sqrt{2}^{\tilde{x}} & x \neq 0 \\ 0 & x = 0 \end{cases}$$

When switching to base-$\sqrt{2}$, the multiplication cannot be simply replaced by shifting operation, the corresponding hardware design is proposed and will be described in the next section.

It is noticed in the Fig. 1 that with the maximum code fixed to FSR, under the condition of same bit width, the dynamic range between the lower threshold and the upper threshold in base-2 logarithmic quantization will be larger than that in base-$\sqrt{2}$ logarithmic quantization. In low-precise situation, the gap between zero and the lower threshold may be fatal to accuracy. To improve the performance in low-precise situation, the segmented logarithmic quantization method is proposed.

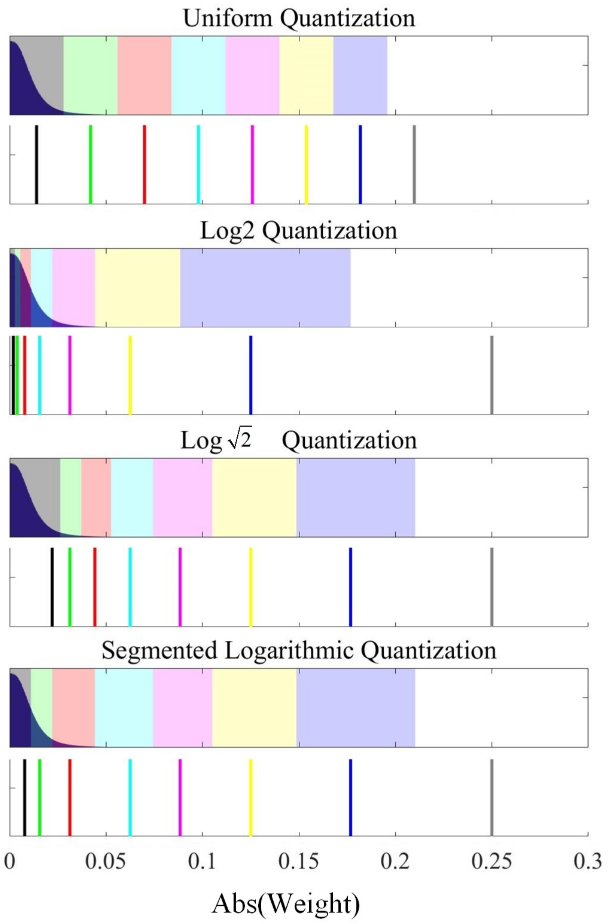


Fig. 1: The threshold and codebook of weights of fc8 layer in Alexnet after different quantization methods. Thresholds are indicated by the colored region on the original distribution. The codes are shown as colored lines below. Values in the colored reigions will be quantified to the corresponding colored code after quantization.

C. Segmented Logarithmic Quantization

To narrow the gap between zero and the lower threshold as well as attach greater importance to larger values, the segmented logarithmic quantization adopts mixed bases. The larger part uses base-$\sqrt{2}$, and the smaller part uses base-2 as shown in Fig. 1. Weights or activations in either part will be represented by the discrete element in the codebook of size 2^{N-1}.

For the two parts, scale range SR_1 (base-2) and SR_2 (base-$\sqrt{2}$) are defined.

$$SR\_1 = round(\log_2(\max - \min)) - 2^{N-2}$$

$$SR\_2 = 2 \times round(\log_2(\max - \min))$$

Codebooks for two parts are described as P_1 and P_2.

$$P\_1 = \{SR\_1 - 2^{N-1} + 1, SR\_1 - 2^{N-1} + 2, \cdots, SR\_1 - 1, SR\_1\}$$

$$P\_2 = \{SR\_2 - 2^{N-1} + 1, SR\_2 - 2^{N-1} + 2, \cdots, SR\_2 - 1, SR\_2\}$$

The threshold sets for the two parts are defined as T_1 and T_2.

$$T\_1 = \{SR\_1 - 2^{N-1} + \frac{1}{2}, SR\_1 - 2^{N-1} + \frac{3}{2}, \cdots, SR\_1 - \frac{1}{2}\}$$

$$T\_2 = \{SR\_2 - 2^{N-1} + \frac{1}{2}, SR\_2 - 2^{N-1} + \frac{3}{2}, \cdots, SR\_2 + \frac{1}{2}\}$$

For notational convenience, $xl\_1 = \log_2(|x|)$ and $xl\_2 = \log_{\sqrt{2}}(|x|)$ are defined. The separation line for the two parts is $T\_2(1)$, and the logarithmic representation of weight or activation x is described by:

$$\text{Log\_Quant}(x) = \begin{cases} 0 & x = 0 \\ Sign(x) \cdot 2^{\tilde{x}\_1} & 0 < |x| < \sqrt{2}^{T\_2(1)} \\ Sign(x) \cdot \sqrt{2}^{\tilde{x}\_2} & |x| \geq \sqrt{2}^{T\_2(1)} \end{cases}$$

for $i = 1, 2, \cdots, 2^{N-1} - 1$

$$\tilde{x}\_1 = \begin{cases} P\_1(1) & xl\_1 < T\_1(1) \\ P\_1(i) & T\_1(i) \leq xl\_1 < T\_1(i+1) \\ P\_1\left(2^{N-1}\right) & xl\_1 \geq T\_1\left(2^{N-1}\right) \end{cases}$$

for $i = 1, 2, \cdots, 2^{N-1}$

$$\tilde{x}\_2 = \begin{cases} P\_2(1) & xl\_2 < T\_2(1) \\ P\_2(i) & T\_2(i) \leq xl\_2 < T\_2(i+1) \\ P\_2\left(2^{N-1}\right) & xl\_2 \geq T\_2\left(2^{N-1} + 1\right) \end{cases}$$

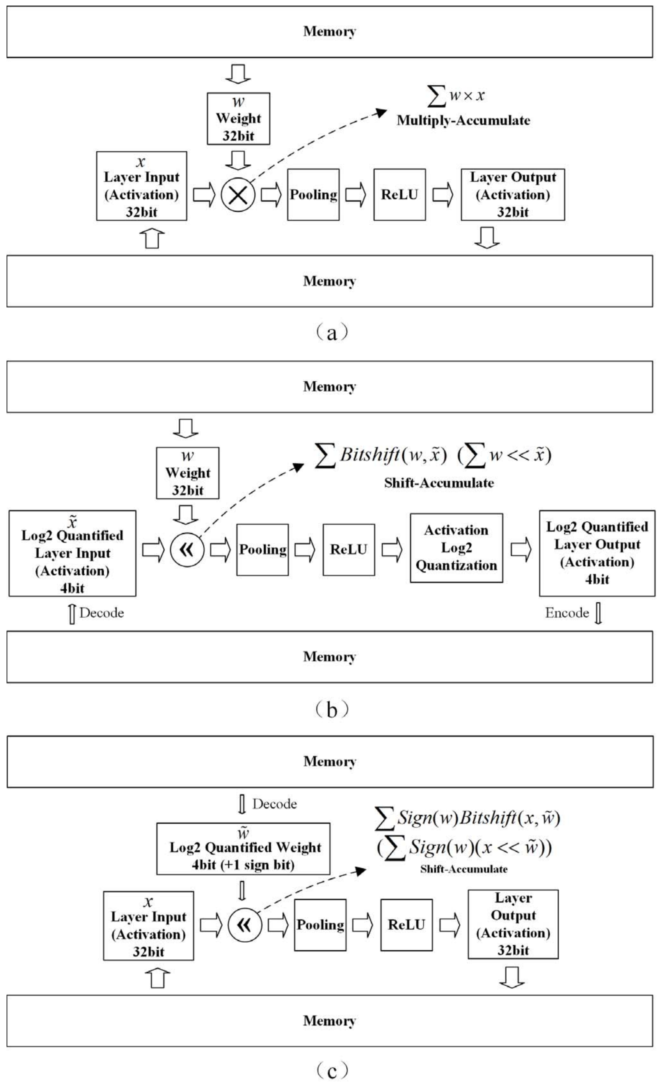
Due to the mixed base in the segmented logarithmic quantization, the hardware design in the two parts will differ with the base used. The corresponding hardware implementation for segmented logarithmic quantization can be effectively realized using the arithmetic element we proposed in the next section.

For the two quantization methods we proposed, logarithmic base-$\sqrt{2}$ arithmetic improves the upper limit of classifying accuracy, and segmented logarithmic arithmetic further improves network performance in low-precise situation. Detailed evaluation results of different quantization methods can be viewed in Section IV. As hardware implementation for the two quantization methods cannot be simply realized by shifter-adders, an arithmetic element compatible for both base-$\sqrt{2}$ and segmented logarithmic quantization is designed in the next section.

III. Hardware

In this section, we first compared the hardware design for conventional multiply-accumulate, logarithmic quantization of activation and logarithmic quantization of weight, and then the compatible arithmetic element for base-$\sqrt{2}$ and segmented logarithmic quantization is proposed and presented using approximate computing.

As the two operand of the dot product, the weight or the input (also known as the activation from the previous layer) can be quantified through logarithmic arithmetic. To realize logarithmic representation of activations, an additional quantization part is needed in the circuit to map the intermediate data of activation results to log-domain as shown in Fig. 2(b). While weight quantization can be done in external device before inference process. With weights quantified and mapped to low-precise logarithmic representations, the memory required for static weight data can be reduced drastically. Therefore, weight quantization is favored over activation quantization in hardware implementation, and is adopted in this paper.

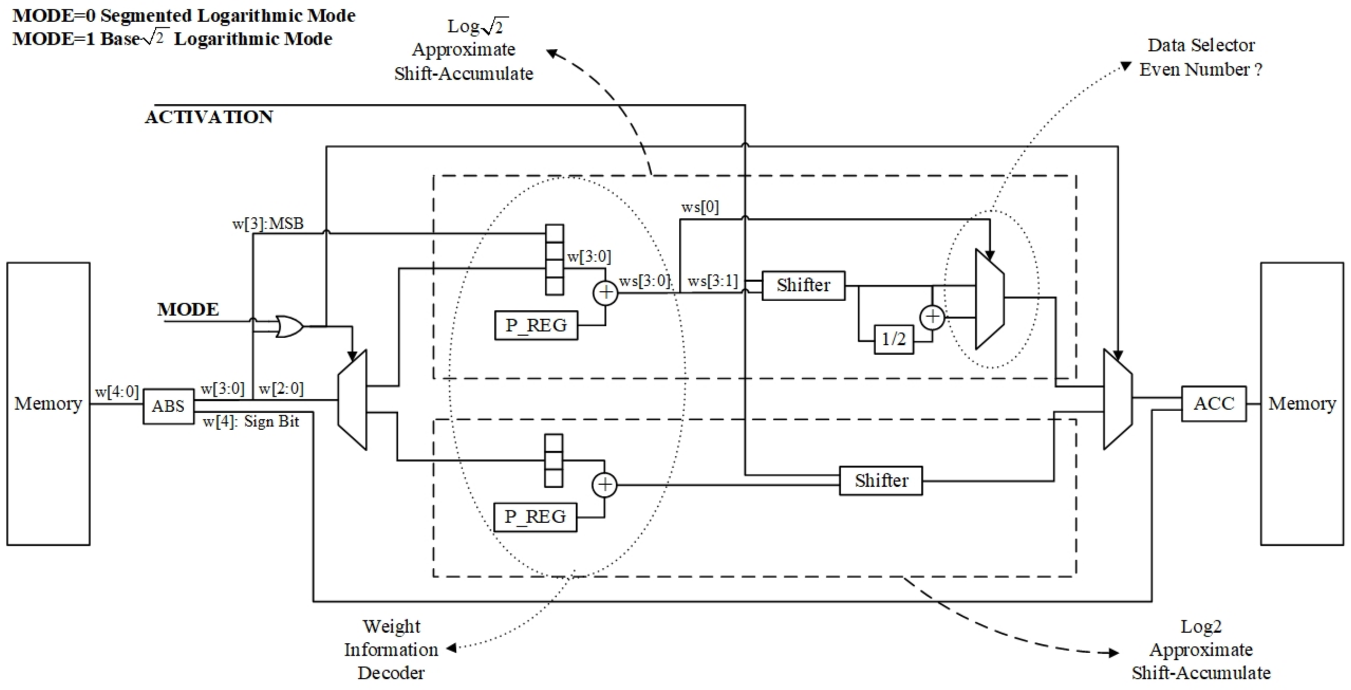


Memory
w Weight 32bit
$\sum w \times x$ Multiply-Accumulate
x Layer Input (Activation) 32bit
Pooling
ReLU
Layer Output (Activation) 32bit
Memory
(a)
Memory
w Weight 32bit
$\sum Bitshift(w, \tilde{x})$ $(\sum w << \tilde{x})$
Shift-Accumulate
$\tilde{x}$ Log2 Quantified Layer Input (Activation) 4bit
Pooling
ReLU
Activation Log2 Quantization
Log2 Quantified Layer Output (Activation) 4bit
Decode
Encode
Memory
(b)
Memory
Decode
$\sum Sign(w) Bitshift(x, \tilde{w})$
$(\sum Sign(w)(x << \tilde{w}))$
Shift-Accumulate
$\tilde{w}$ Log2 Quantified Weight 4bit (+1 sign bit)
x Layer Input (Activation) 32bit
Pooling
ReLU
Layer Output (Activation) 32bit
Memory
(c)

Fig. 2: (a) The conventional multiply-accumulate design. (b) The flow-chart for activation quantization using base-2 logarithmic representation. (c) The flow-chart for weight quantization using base-2 logarithmic algorithm.

A. Existing Designs

In conventional multiply-add design, the dot products are performed using floating or fixed point representation as shown



MODE=0 Segmented Logarithmic Mode
MODE=1 Base-$\sqrt{2}$ Logarithmic Mode
Log$\sqrt{2}$ Approximate Shift-Accumulate
Data Selector Even Number ?
ACTIVATION
w[3]:MSB
MODE
Memory
w[4:0]
ABS
w[3:0]
w[2:0]
w[4]: Sign Bit
w[3:0]
ws[3:0]
ws[0]
ws[3:1]
Shifter
1/2
P_REG
P_REG
Shifter
ACC
Memory
Weight Information Decoder
Log2 Approximate Shift-Accumulate

Fig. 3: The proposed arithmetic element: The input MODE will decide the operation mode (base-$\sqrt{2}$ mode or segmented mode). In base-$\sqrt{2}$ mode, only log$\sqrt{2}$ approximate shift-accumulate part will be used; and in segmented mode, either log$\sqrt{2}$ approximate shift-accumulate part or log2 approximate shift-accumulate part will be used according to MSB.

in Fig. 2(a).

In base-2 logarithmic quantization, by transferring weight w to its base-2 logarithmic representation $\tilde{w}$, the multiplication $x \times w$ can be transformed to shifting operation, which is to shift the value x by an integer $\tilde{w}$. The flowchart is shown in Fig. 2(c). After the quantified weight information transfers from memory to the computing unit, it will map to the corresponding integer b in $Bitshift\,(a, b)$ through the decoding unit, and then enter into the shifter.

B. The Proposed Arithmetic Element

The arithmetic element supporting base-$\sqrt{2}$ and segmented logarithmic algorithm is illustrated in Fig. 3. There are two main computing units in the design, one is the log$\sqrt{2}$ approximate shift-accumulate part, the other is the log2 approximate shift-accumulate part. This arithmetic element has two operation mode, base-$\sqrt{2}$ mode and segmented mode. In base-$\sqrt{2}$ mode, only log$\sqrt{2}$ approximate shift-accumulate part will be used, and in segmented mode, both log$\sqrt{2}$ approximate shift-accumulate and log2 approximate shift-accumulate part will be used.

1) Base-$\sqrt{2}$ Mode: Logrithmic base-$\sqrt{2}$ quantization

Although base-$\sqrt{2}$ algorithm cannot be directly converted to shifting operation, it is able to be performed with base-2 using approximation $\sqrt{2} \approx 2^0 + 2^{-1} - 2^{-4} - 2^{-6} - 2^{-7} \cdots$.

If $\tilde{w}$ is an even integer, $\sqrt{2}^{\tilde{w}}$ can be represented by $2^{\frac{\tilde{w}}{2}}$. The multiplication $x \times w$ can be transformed to shifting the value x by an integer $\frac{\tilde{w}}{2}$. And if $\tilde{w}$ is an odd number, the base-$\sqrt{2}$ algorithm can be simply converted to base-2 algorithm by:

$$\sqrt{2}^{\tilde{w}} = 2^{\frac{\tilde{w}-1}{2}} \times \sqrt{2} \approx 2^{\frac{\tilde{w}-1}{2}} \times \left(2^0 + 2^{-1} - 2^{-4} - 2^{-6} - 2^{-7} \cdots\right)$$

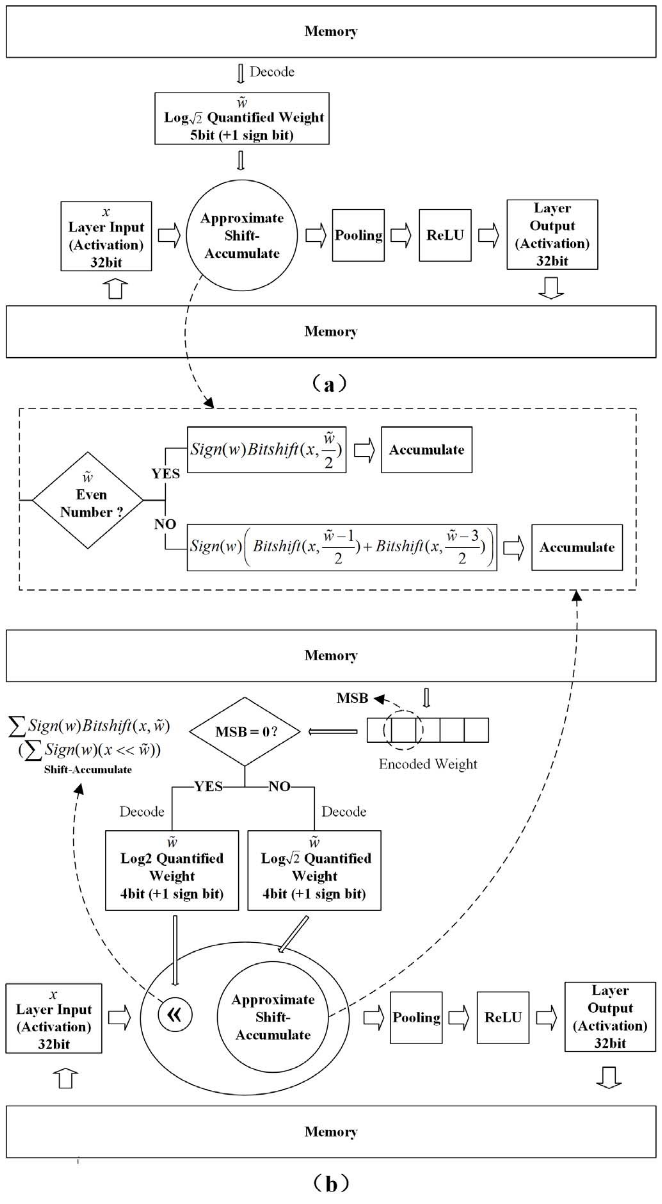
Taking the approximation $\sqrt{2} \approx 2^0 + 2^{-1}$ as an example, the multiplication $x \times w$ can be transferred to shift-add operations as:

$$Bitshift\left(x, \frac{\tilde{w}-1}{2}\right) + Bitshift\left(x, \frac{\tilde{w}-3}{2}\right)$$

Where $Bitshift\,(a, b)$ is a function that shifts the value a by an integer b. The corresponding flowchart is shown in Fig. 4(a).

2) Segmented Mode: Segmented logarithmic quantization

The segmented logarithmic algorithm adopts mixed bases, and in the proposed design for segmented logarithmic algorithm, the two main computing units, namely log$\sqrt{2}$ approximate shift-accumulate part and log2 approximate shift-accumulate part can be used through a digital selector. All the weights in the network will be represented by the discrete element in the codebook $\{P\_1, P\_2\}$, with P_1 sized 2^{N-1} and P_2 sized 2^{N-1}. Therefore, the selector is easy to realize using the Most Significant Bit (MSB). If MSB is 0, base-2 part is selected, else base-$\sqrt{2}$ part is selected. Fig. 4(b) shows the flowchart of segmented logarithmic mode.



Memory
Decode
$\tilde{w}$
Log$\sqrt{2}$ Quantified Weight 5bit (+1 sign bit)
x
Layer Input (Activation) 32bit
Approximate Shift-Accumulate
Pooling
ReLU
Layer Output (Activation) 32bit
Memory
(a)
$Sign(w)Bitshift(x, \frac{\tilde{w}}{2})$
Accumulate
YES
$\tilde{w}$ Even Number ?
NO
$Sign(w)\left(Bitshift(x, \frac{\tilde{w}-1}{2}) + Bitshift(x, \frac{\tilde{w}-3}{2})\right)$
Accumulate
Memory
MSB
$\sum Sign(w)Bitshift(x, \tilde{w})$
$(\sum Sign(w)(x << \tilde{w}))$
Shift-Accumulate
MSB = 0?
Encoded Weight
YES
NO
Decode
Decode
$\tilde{w}$
Log2 Quantified Weight 4bit (+1 sign bit)
$\tilde{w}$
Log$\sqrt{2}$ Quantified Weight 4bit (+1 sign bit)
x
Layer Input (Activation) 32bit
«
Approximate Shift-Accumulate
Pooling
ReLU
Layer Output (Activation) 32bit
Memory
(b)

Fig. 4: (a) The flowchart for base-$\sqrt{2}$ logarithmic algorithm. (b) The flowchart for segmented logarithmic algorithm.

IV. Simulation

A. Different Quantization Methods

To evaluate the performance of the quantization strategies proposed, the accuracies after logarithmic quantization methods with different bases are tested and compared with linear quantization. The published AlexNet [11] model from Matconvnet [12] is tailored, and the classification accuracy (top-5 and top-1) is used as the main performance metric. The original network achieves 76.8% in top-5 accuracy and 53.8% in top-1 accuracy. We quantize every layer of AlexNet without retraining, and the results are shown in Table IV.

TABLE I: Top-5 and Top-1 accuracies after different quantization methods on every layer's weights without retraining

Top-5 (Original 76.8%)						
Method	Bit Width					
	8	7	6	5	4	3
Uniform	76.1%	71.7%	49%	18.6%	4.4%	0.6%
Log2	69.3%	69.3%	69.3%	69.3%	69.3%	40.6%
Log$\sqrt{2}$	75.5%	75.5%	75.5%	75.5%	63.7%	0.3%
Segmented Log	75.5%	75.5%	75.5%	75.6%	72.3%	2.8%
Top-1 (Original 53.8%)						
Method	Bit Width					
	8	7	6	5	4	3
Uniform	52.4%	47.2%	28.6%	6.7%	1.4%	0%
Log2	46.9%	46.9%	46.9%	46.9%	46.9%	21.3%
Log$\sqrt{2}$	50.8%	50.8%	50.8%	50.8%	41.3%	0%
Segmented Log	50.8%	50.8%	50.8%	51%	49.7%	0.9%

TABLE II: Top-5 and Top-1 accuracies after $\sqrt{2}$ approximation with different approximate qualities

Top-5 (Original 76.8%)				
Method	Approximation	Bit Width		
		6	5	4
Log$\sqrt{2}$	$\sqrt{2}$	75.5%	75.5%	63.7%
	2^0+2^{-1}	73.9%	73.9%	64.3%
	$2^0+2^{-1}-2^{-4}$	75.1%	75.1%	64.5%
	$2^0+2^{-1}-2^{-4}-2^{-6}$	75.2%	75.2%	63.7%
Segmented Log	$\sqrt{2}$	75.5%	75.6%	72.3%
	2^0+2^{-1}	73.9%	73.6%	72.8%
	$2^0+2^{-1}-2^{-4}$	75.1%	74.9%	72.6%
	$2^0+2^{-1}-2^{-4}-2^{-6}$	75.2%	75.4%	72.6%
Top-1 (Original 53.8%)				
Method	Approximation	Bit Width		
		6	5	4
Log$\sqrt{2}$	$\sqrt{2}$	50.8%	50.8%	41.3%
	2^0+2^{-1}	51.1%	51.1%	41.8%
	$2^0+2^{-1}-2^{-4}$	50.7%	50.7%	41.6%
	$2^0+2^{-1}-2^{-4}-2^{-6}$	50.6%	50.7%	41.4%
Segmentd Log	$\sqrt{2}$	50.8%	51%	49.7%
	2^0+2^{-1}	51.1%	51%	48.2%
	$2^0+2^{-1}-2^{-4}$	50.7%	50.7%	49.2%
	$2^0+2^{-1}-2^{-4}-2^{-6}$	50.6%	50.8%	49.3%

Compared with uniform quantization, all three logarithmic quantization methods perform better below 6-bit. All three logarithmic quantization methods have clear upper limits for both top-5 and top-1 accuracies. The base-2 logarithmic quantization reaches its upper limit at 4-bit with 69.3% top-5 accuracy and 46.9% top-1 accuracy, while base-$\sqrt{2}$ logarithmic quantization achieves its best accuracy at 5-bit with 75.5% top-5 accuracy and 50.8% top-1 accuracy. The accuracy upper limit of segmented logarithmic algorithm is the same as that of base-$\sqrt{2}$ logarithmic algorithm, with a slightly rise at 5-bit. As can

TABLE III: Comparison with 16-bit fixed point multiplier

Design	Technology	Power(mW)	Area(μm^2)	Number of Equivalent Logic Gates
16-bit fixed point multiplier	65nm	0.457	2031.12	1762.50
The proposed arithmetic element	65nm	0.120	852.48	984.00

TABLE IV: Top-5 and Top-1 accuracies after different quantization methods on every layer's weights without retraining

Top-5 (Original 76.8%)				
Method	Bit Width			
	6	5	4	3
Uniform	49%	18.6%	4.4%	0.6%
Log2	69.3%	69.3%	69.3%	40.6%
Log$\sqrt{2}$	75.5%	75.5%	63.7%	0.3%
Segmented Log	75.5%	75.6%	72.3%	2.8%
Top-1 (Original 53.8%)				
Method	Bit Width			
	6	5	4	3
Uniform	28.6%	6.7%	1.4%	0%
Log2	46.9%	46.9%	46.9%	21.3%
Log$\sqrt{2}$	50.8%	50.8%	41.3%	0%
Segmented Log	50.8%	51%	49.7%	0.9%

be viewed from Table IV, when bit width is restricted to 4-bit, segmented logarithmic quantization achieves the best top-5 and top-1 accuracies, which are almost 3% ahead the closest after.

B. Hardware Evaluation

$\sqrt{2}$ approximation is an important enabler in the proposed design, and approximate quality will affect the accuracy. The error brought by $\sqrt{2}$ approximation is shown in Table II. Higher approximate quality will result in more complicated hardware design. The proposed arithmetic element is evaluated under $\sqrt{2} \approx 2^0 + 2^{-1}$ approximation and 4-bit weight quantization. The arithmetic element is synthesized in UMC 65nm Low Leakage Process by Synopsys Design Compiler. Operating at 500MHz and 1.2V, the entire block consumes 0.12 mW. Table III compares this work with a 16-bit fixed point multiplier. 16-bit fixed point multiplier is widely used in current DNN accelerators. It can be viewed that the proposed arithmetic element is 58.03% smaller in area and exhibits 73.74% energy reduction compared with 16-bit fixed point multiplier.

V. Conclusion

In this paper, we describe the method to improve the accuracy in efficient inference process for DNNs using base-$\sqrt{2}$ and segmented logarithmic representation. Base-$\sqrt{2}$ logarithmic arithmetic is adopted to raise the ceiling of traditional base-2 arithmetic. Segmented logarithmic quantization is proposed to further improve the performance in low-precise situation. The simulation results show that base-$\sqrt{2}$ logarithmic quantization achieves 6.2% higher in top-5 upper limit and 3.9% higher in top-1 upper limit than base-2 logarithmic quantization. Segmented logarithmic quantization improves top-5 accuracy by 3% and top-1 accuracy by 2.8% in 4-bit situation. A aithmetic element supporing both algorithms are designed and simulated in UMC 65nm process. The proposed arithmetic element is 58.03% smaller in area compared with 16-bit fixed point multiplier. And our design achieves 73.74% reduction in the energy of executing the task.

VI. Acknowledgement

This work was supported in part by the Shanghai Pujiang Program under Grant 17PJ1400800, in part by NSFC under Grant 61571137, in part by the Shanghai Science and Technology Innovation Program under Grant 17JC1401400, and in part by Shanghai Institute of Intelligent Electronics & Systems.

References

[1] G. Indiveri and S. C. Liu, "Memory and information processing in neuromorphic systems," *Proceedings of the IEEE*, vol. 103, no. 8, pp. 1379–1397, 2015.

[2] T. Serre, L. Wolf, and T. Poggio, "Object recognition with features inspired by visual cortex," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, 2005, pp. 994–1000 vol. 2.

[3] C. Dan, U. Meier, and J. Schmidhuber, "Multi-column deep neural networks for image classification," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 3642–3649.

[4] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[5] M. Ge, "Statistical parametric speech synthesis using deep neural network," *University of Edinburgh*, 2013.

[6] V. Vanhoucke and M. Z. Mao, "Improving the speed of neural networks on cpus," *Deep Learning & Unsupervised Feature Learning Workshop Nips*, 2011.

[7] S. Han, J. Pool, J. Tran, and W. Dally, "Learning both weights and connections for efficient neural network," in *Advances in neural information processing systems*, 2015, pp. 1135–1143.

[8] M. Courbariaux, I. Hubara, D. Soudry, R. El-Yaniv, and Y. Bengio, "Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1," *arXiv preprint arXiv:1602.02830*, 2016.

[9] E. H. Lee, D. Miyashita, E. Chai, B. Murmann, and S. S. Wong, "Lognet: Energy-efficient neural networks using logarithmic computation," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2017, pp. 5900–5904.

[10] D. Miyashita, E. H. Lee, and B. Murmann, "Convolutional neural networks using logarithmic data representation," *arXiv preprint arXiv:1603.01025*, 2016.

[11] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *International Conference on Neural Information Processing Systems*, 2012, pp. 1097–1105.

[12] A. Vedaldi and K. Lenc, "Matconvnet: Convolutional neural networks for matlab," in *Proceedings of the 23rd ACM international conference on Multimedia*. ACM, 2015, pp. 689–692.