IOC 5009 Accelerator Architectures for Machine Learning
Lab II

November, 2021

1 Benchmarking Deep Neural Networks

People often estimate the number of parameters and MAC operations to take insight of one neural network model. Results of this benchmarking help people to optimize the computation of neural networks and neural network hardware designs.

This lab requires you to implement each neural network layer of VGG 16 model through high level programming languages such as C/C++/python(numpy and scipy) etc.. You don't allow to using any external DNN libraries such as cuDNN, MKL-DNN or DNN frameworks such as pyTorch, TensorFlow and Keras in your implementation. You can follow VGG 16 model architecture model (See Table 1) to complete your **forward pass** implementation. You only require to run your implementation on the CPU. Furthermore, you also need to calculate the memory size of inputs, the number of parameters, and the number of MAC operations in each layer. The batch size of this VGG 16 model is 1. The activation function in CONV layer is "ReLU". The size of the initial input is 224 x 224 x 3 and the input's values are randomly generated. Finally, you need to fill your results in the VGG16 Benchmark Table (See Table 2) and turn your codes and completed table in.

Table 1: VGG 16 Model Architecture

	Input	Filter Size	# of channel	# of filter	Pool Size	stride	Activation
INPUT	224 x 224		3	3			
CONV	224 x 224	3 x 3	64	64			ReLU
CONV	224 x 224	3 x 3	64	64			ReLU
MAXPOOL					2 x 2	2	
CONV	112 x 112	3 x 3	128	128			ReLU
CONV	112 x 112	3 x 3	128	128			ReLU
MAXPOOL					2 x 2	2	
CONV	56 x 56	3 x 3	256	256			ReLU
CONV	56 x 56	3 x 3	256	256			ReLU
CONV	56 x 56	3 x 3	256				ReLU
MAXPOOL					2 x 2	2	
CONV	28 x 28	3 x 3	512	512			ReLU
CONV	28 x 28	3 x 3	512	512			ReLU
CONV	28 x 28	3 x 3	512	512			ReLU
MAXPOOL					2 x 2	2	
CONV	14 x 14	3 x 3	512	512			ReLU
CONV	14 x 14	3 x 3	512	512			ReLU
CONV	14 x 14	3 x 3	512	512			ReLU
MAXPOOL					2 x 2	2	
FC 4096							
FC 4096							
FC 1000							

Table 2: VGG 16 Benchmarking Table

	Memory Size	# of parameter	# of MAC operations
INPUT	224 x 224 x 3 = 150K	0	0
CONV			
CONV			
MAXPOOL	112 x 112 x 64 = 800 K	0	
CONV			
CONV			
MAXPOOL		0	
CONV			
CONV			
CONV			
MAXPOOL		0	
CONV			
CONV			
CONV			
MAXPOOL		0	
CONV			
CONV			
CONV			
MAXPOOL		0	
FC 4096			
FC 4096			
FC 1000			

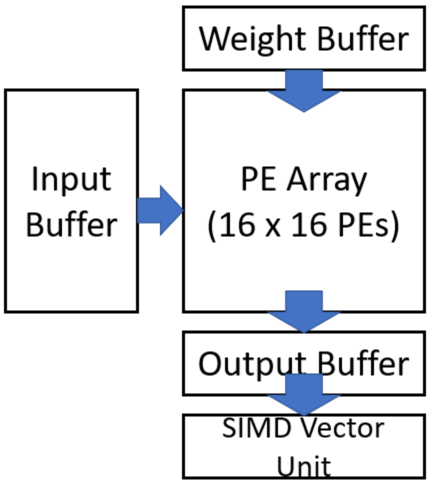

Figure 1: The prototype of a systolic array accelerator

2 Systolic Array Architecture

The systolic array accelerator tailors for operations of DNN models. The accelerator contains a 2D array of processing elements (PE) to calculate matrix multiplication and convolution, an unified multi-bank buffer decomposed into input, weight and output buffers, and a SIMD vector unit for POOL, ACT, normalization, and etc..

This lab requires you to implement a systolic array accelerator by using verilog. Figure 1 presents the prototype of the systolic array accelerator. The specification of a systolic array accelerator is shown as follows:

1. The size of PE array is 16 $\times$ 16.
2. Each Processing element (PE) takes one cycle to calculate one MAC operation.
3. The PE array can proceed the convolution and fully-connected operation through the systolic

Table 3: TinyML Model Architecture

	Input	Filter Size	# of channel	# of filter	Pool Size	stride	Activation
INPUT	16 x 16		3	3			
CONV	16 x 16	2 x 2	4	16			ReLU
MAXPOOL					2 x 2	2	
CONV	8 x 8	3 x 3	1	8			ReLU
FC 8							

Table 4: TinyML Benchmarking Table

	cycles	Max PE utilization
INPUT	0	0
CONV		
MAXPOOL		0
CONV		
FC 8		

execution manner.
4. No limit for buffer size and DRAM.
5. There are 16 SIMD vector lanes. Each SIMD lane can complete POOL and ACT operation in one cycle.

You need to run a TinyML model shown in the table 3 on this systolic array accelerator and complete the table 4. Note that PE utilization indicates the max number of PE used by each layer.

Finally, you need to fill your results in the TinyML Benchmark Table (See Table 4) and turn your codes and completed table in.