This is a sample of the report, but applicable for all homework. [113062575] [徐義鈞] This is for double verification.

Don't copy the problem statement, just write the answer. Please write down the question number in unit of sub-question.

Please write down the sub-question number even if you don't know how to solve it.

# Part1

(1.3.1)

=================================================================================

Layer (type:depth-idx) Output Shape Param #

=================================================================================

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Net | | [1, 10] |  | -- |
| ├─Sequential: 1-1 | | [1, 6, 28, 28] | -- |  |
| │ └─Conv2d: 2-1 | | [1, 6, 28, 28] | 150 |  |
| │ └─ReLU: 2-2 | | [1, 6, 28, 28] | -- |  |
| ├─Sequential: 1-2 | | [1, 6, 14, 14] | -- |  |
| │ └─MaxPool2d: 2-3 | | [1, 6, 14, 14] | -- |  |
| ├─Sequential: 1-3 | | [1, 16, 10, 10] | -- |  |
| │ | └─Conv2d: 2-4 | [1, 16, 10, 10] | 2,400 | |
| │ | └─ReLU: 2-5 | [1, 16, 10, 10] | -- | |
| ├─Sequential: 1-4 | | [1, 16, 5, 5] | -- | |
| │ └─MaxPool2d: 2-6 | | [1, 16, 5, 5] | -- | |
| ├─Sequential: 1-5 | | [1, 120, 1, 1] | -- | |
| │ └─Conv2d: 2-7 | | [1, 120, 1, 1] | 48,000 | |
| │ └─ReLU: 2-8 | | [1, 120, 1, 1] | -- | |
| ├─Sequential: 1-6 | | [1, 84] | -- | |
| │ └─Linear: 2-9 | | [1, 84] | 10,080 | |
| │ └─ReLU: 2-10 | | [1, 84] | -- | |
| ├─Sequential: 1-7 | | [1, 10] | -- | |
| │ └─Linear: 2-11 | | [1, 10] | 840 | |

=================================================================================

Total params: 61,470

Trainable params: 61,470

Non-trainable params: 0

Total mult-adds (Units.MEGABYTES): 0.42

=================================================================================

Input size (MB): 0.00 Forward/backward pass size (MB): 0.05 Params size (MB): 0.25

Estimated Total Size (MB): 0.30

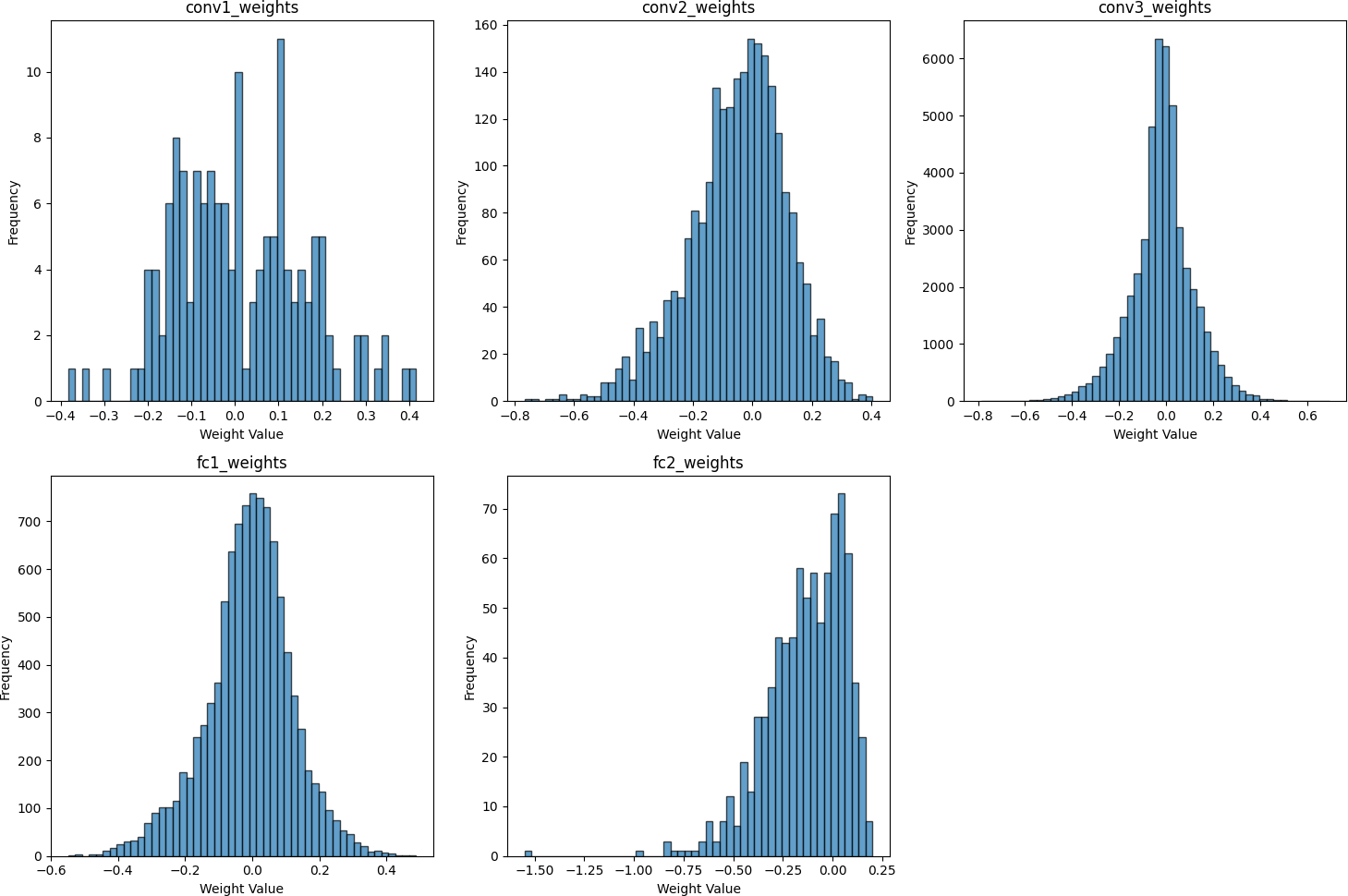
================================================================================= (1.3.2)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **type** | **input activation**  **size** | **output activation**  **size** | **activation**  **function** |
| conv1 | convolution | 1\*32\*32 | 6\*28\*28 | ReLU |
| maxpool2 | pooling | 6\*28\*28 | 6\*14\*14 |  |
| conv3 | convolution | 6\*14\*14 | 16\*10\*10 | ReLU |
| maxpool4 | pooling | 16\*10\*10 | 16\*5\*5 |  |
| conv5 | convolution | 16\*5\*5 | 120\*1\*1 | ReLU |
| fc6 | fully-connected | 120 | 84 | ReLU |
| output | fully-connected | 84 | 10 |  |

(1.3.3)

可以，藉由修改input / ouput activation size 即可正常運作。但是會使參數量顯著提升，影響模型效能，對於影像識別的表現可能會下降。

(2.1.1)



(2.2.1)

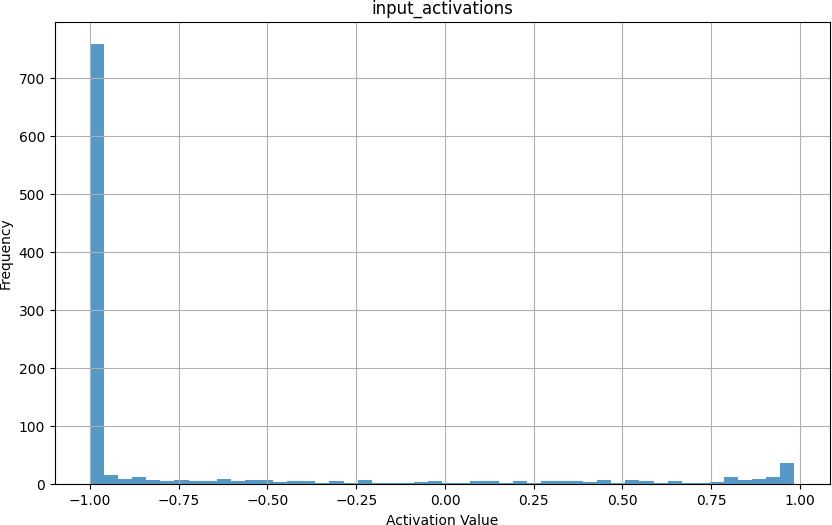
我使用symmetric quantization，具體計算為:max(abs(weights)) / 255

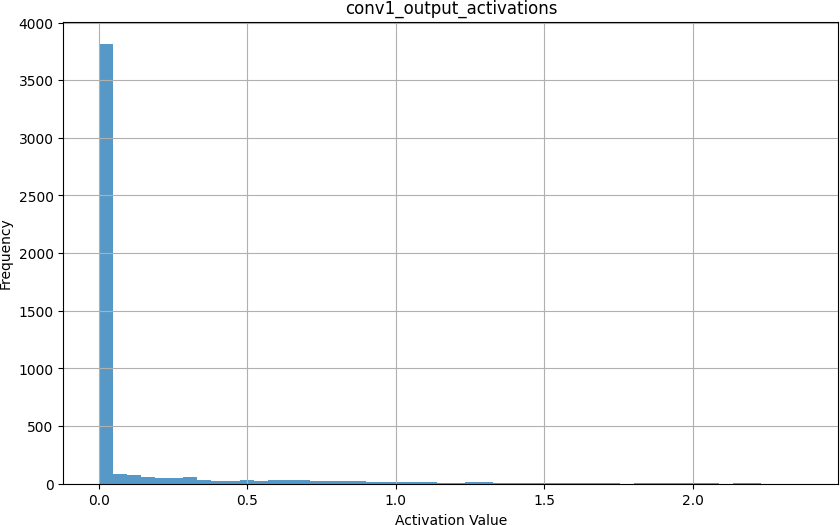
(2.2.2)

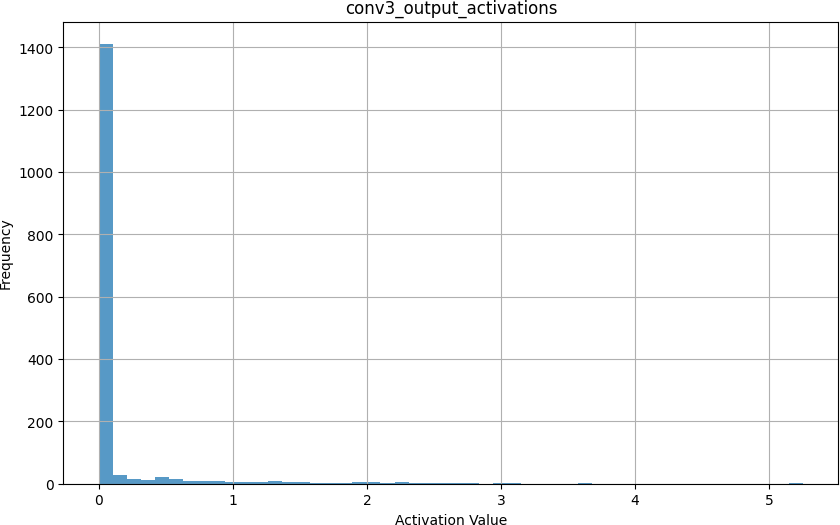


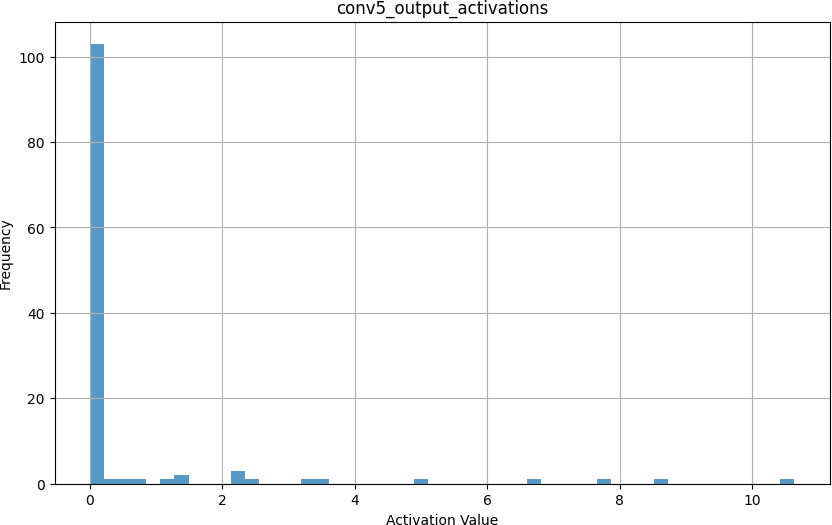
Accuracy degradation is 0.01%

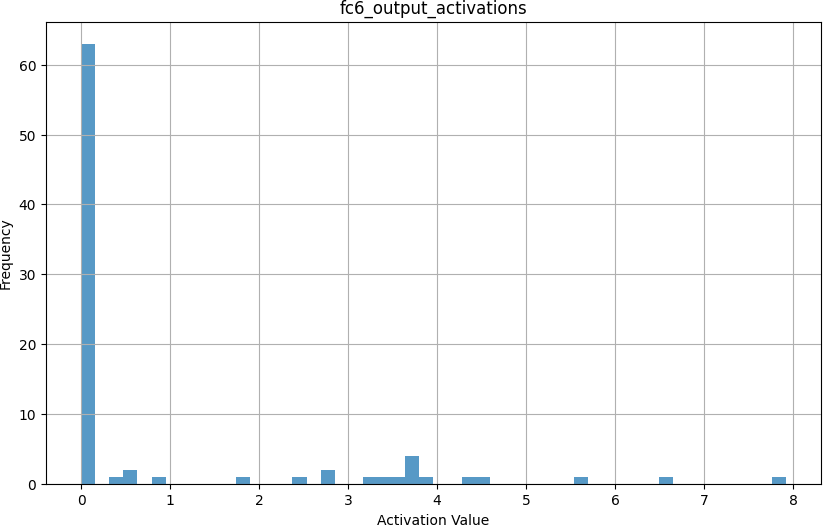
(2.3.1)

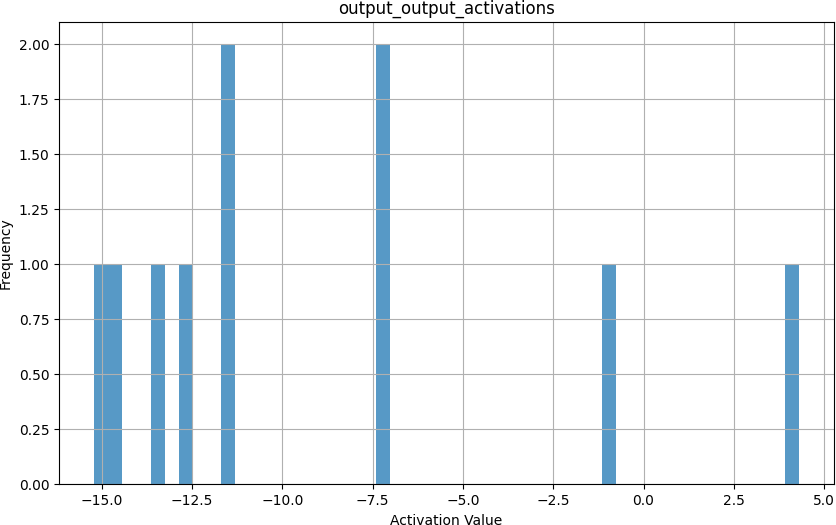












(2.4.1)

𝑆𝐼, 𝑆𝑊𝑐𝑜𝑛𝑣1, and 𝑆𝑂𝑐𝑜𝑛𝑣1 這三者的計算方式概念都一樣，以𝑆𝐼為例：

max(abs(inputactivations)) / 255，其他兩者依此類推。

(2.4.2)

𝑂𝑐𝑜𝑛𝑣1𝑞= SWconv1 ∗ SI

SOconv1

∗ (𝑊𝑐𝑜𝑛𝑣1𝑞 \* 𝐼𝑞)

𝑀1

SWconv1∗ SI

=

SOconv1

(2.4.3)

𝑂𝑐𝑜𝑛𝑣3𝑞= SWconv3 ∗ SOconv1

SOconv3

∗ (𝑊𝑐𝑜𝑛𝑣3𝑞 \* 𝑂𝑐𝑜𝑛𝑣1𝑞)

(2.4.4)

𝑀𝑛 = SWn ∗ SI

SOn

, 𝑖𝑓 𝑛 = 1

SWn ∗ SOn−1

=

SOn

, otherwise

(2.4.6)

使用floor在硬體層面比較容易實作，相比於round還需要去設計硬體考慮carry-in carry- out

(2.4.7)

由於output\_scale為小於1的浮點數，x \* output\_scale為浮點數乘法運算較難實現。

而 round(1/output\_scale) 為整數，則x/round(1/output\_scale)為整數除法運算，硬體上較容易實現

(2.5.1)

Sβ = 𝑆𝑊 ∗ 𝑆𝑂𝑛−1

(3.1.1)

QAT訓練時就考量quantization，模型學會在該條件下調整權重減少quantization error，故相比於PTQ訓練後才去做quantization，能有更高的accuracy

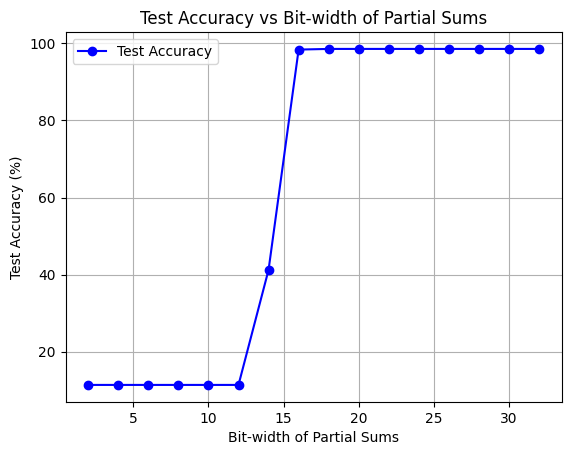
(3.1.2)

quant：將 floating-point 的 weight 轉成 quantized values (e.g.,8-bit integer values)

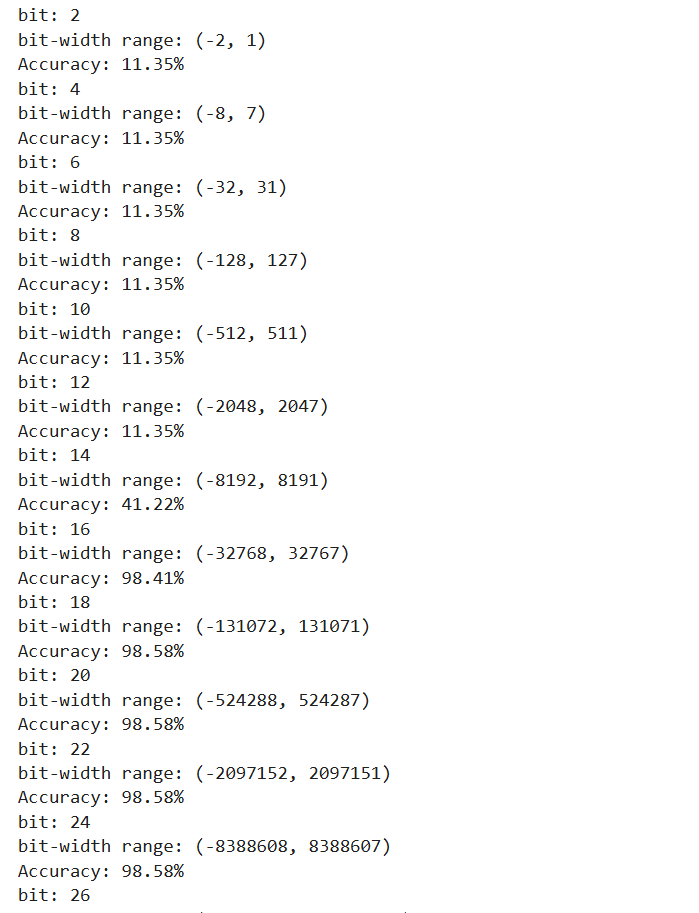
dequant：將 quantized values 轉回 floating-point 以供後續的layer運算。

# Part2

(2.1.1)

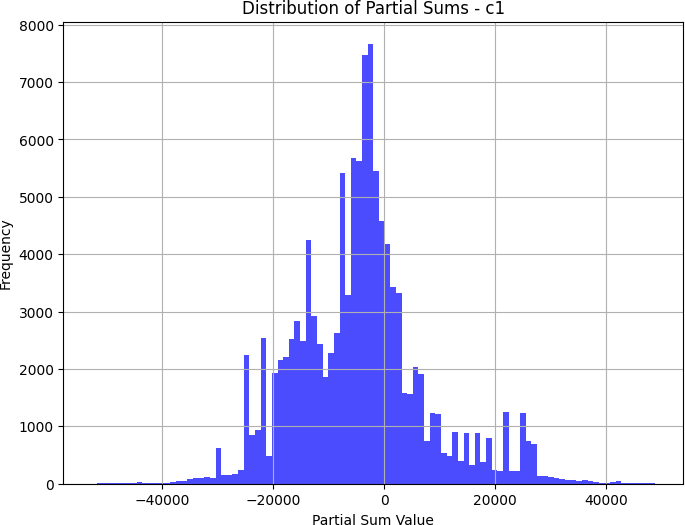


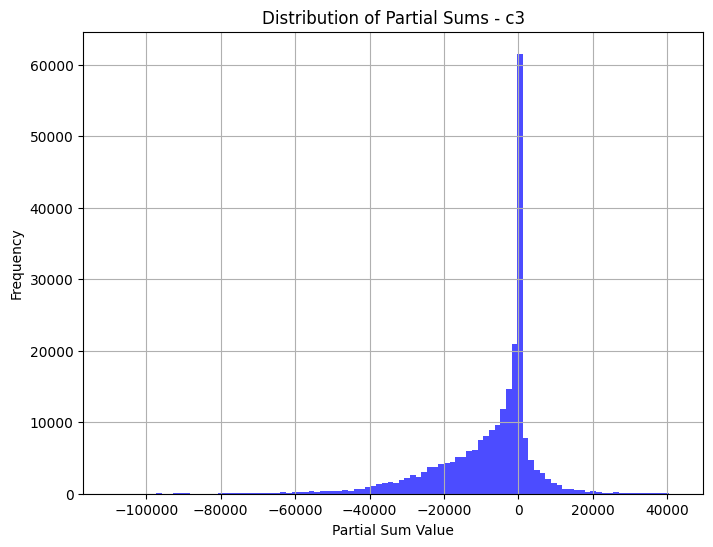
(2.1.2)

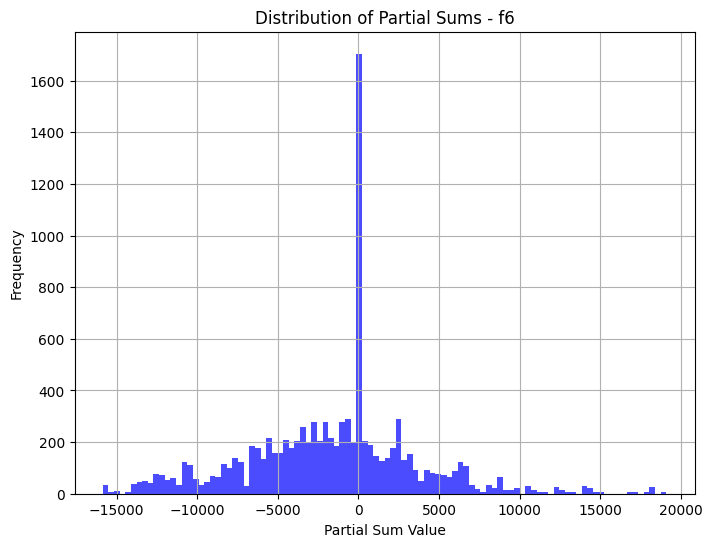
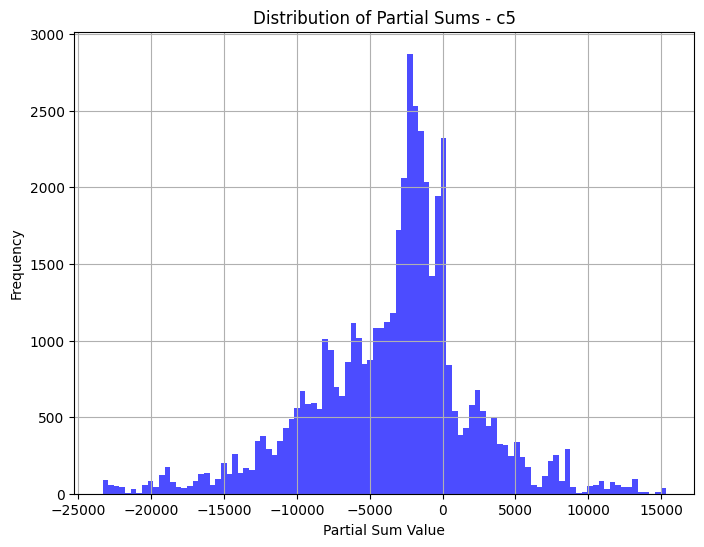


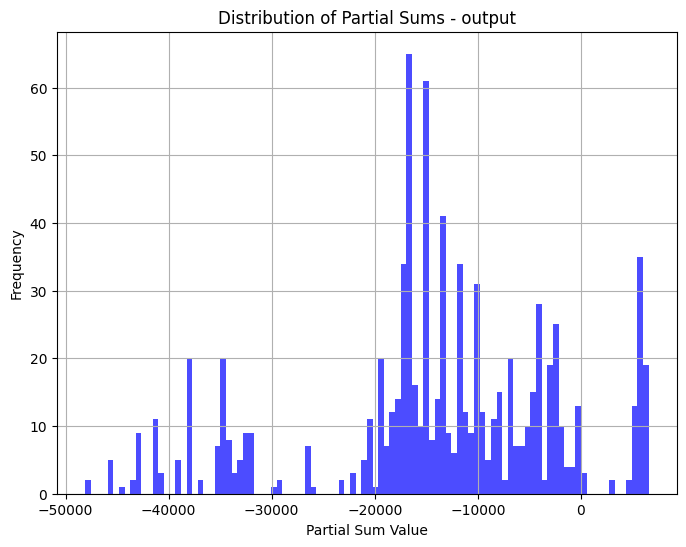
smallest bit-width of partial sums that maintains the same accuracy = 18

(2.2.1)









|  |  |  |  |
| --- | --- | --- | --- |
|  | **min** | **max** | **standard deviation** |
| C1 | -52685 | 48834 | 11930.04150 |
| C3 | -109360 | 42057 | 13442.25206 |
| C5 | -23325 | 15341 | 5500.06614 |
| F6 | -15866 | 19112 | 5324.82541 |
| Output | -48103 | 6569 | 11984.27439 |

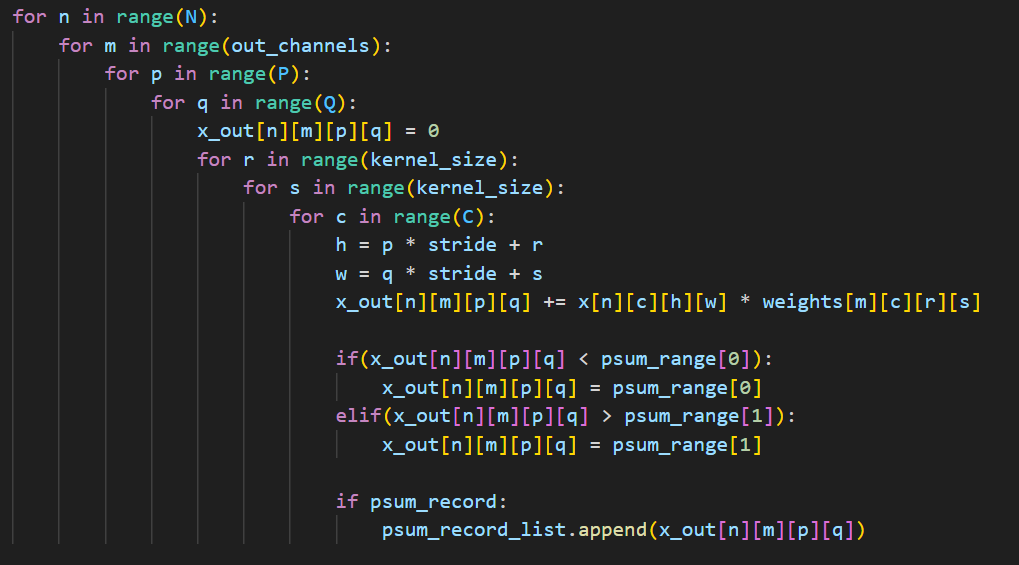
(2.2.2)

|  |  |
| --- | --- |
|  | minimum bit-width |
| C1 | 18 |
| C3 | 18 |
| C5 | 16 |
| F6 | 16 |
| Output | 16 |



(3.1.1)

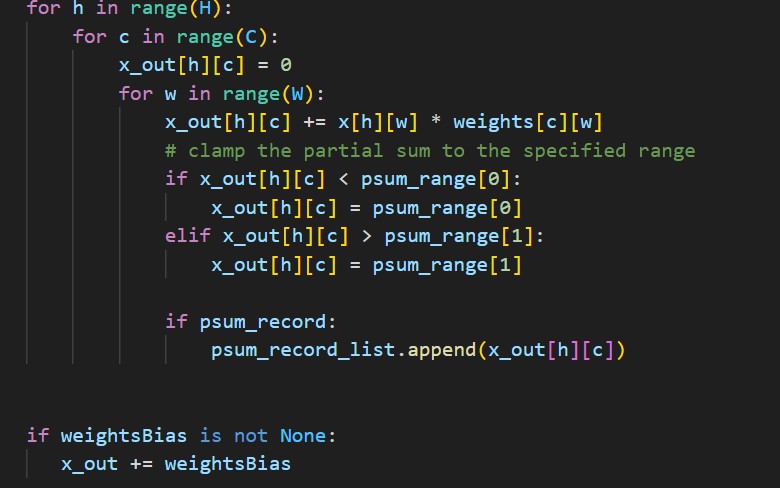
根據程式碼 Batch Size = 4



Conv2d 迴圈次數 N\*M\*P\*Q\*R\*S\*C

最內層 3 次 mul , 3 次 add

C1 = 4\*6 \*28 \*28\*5\*5\*1 = 470400 C3 = 4\*16\*10\*10\*5\*5\*6 = 960000 C5 = 4\*120\*1\*1\*5\*5\*16 = 192000



Linear 迴圈次數 H\*C\*W F6 = 4\*84\*120 = 40320

Output = 4\*10\*84 = 3360最內層 1 次 mul, 1 次 add H\*C 個 elements 要加 bias

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Nmul | Nadd | Bmul | smul | Badd | sadd | Ew |
| C1 | 1411200 | 1411200 | 8 | 64 | 18 | 18 | 115718400 |
| C3 | 2880000 | 2880000 | 8 | 64 | 18 | 18 | 236160000 |
| C5 | 576000 | 576000 | 8 | 64 | 18 | 18 | 47232000 |
| F6 | 40320 | 40320 | 8 | 64 | 18 | 18 | 3306240 |
| Output | 3360 | 3400 | 8 | 64 | 18 | 18 | 276240 |

# Overall Ew = 402,692,880

(3.1.2)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Nmul | Nadd | Bmul | smul | Badd | sadd | Ew |
| C1 | 1411200 | 1411200 | 8 | 64 | 18 | 18 | 115718400 |
| C3 | 2880000 | 2880000 | 8 | 64 | 16 | 16 | 236160000 |
| C5 | 576000 | 576000 | 8 | 64 | 16 | 16 | 46080000 |
| F6 | 40320 | 40320 | 8 | 64 | 16 | 16 | 3225600 |
| Output | 3360 | 3400 | 8 | 64 | 16 | 16 | 269440 |

**Overall Ew = 401,453,440**