Software and Hardware Implementation of Pose Estimation Algorithm Based on 4-bit Quantization

基於4位元量化姿態辨識演算法之軟硬體實作

組別: B147　　組員: 蔡惠芸、吳書磊、張博閔、陳泰融　指導教授: 呂仁碩

I. 摘要

在深度學習的發展中，深度學習模型訓練與推論所需要花費的運算資源以及時間都變得越大量，如何減少這運算資源以及時間是重要的課題。本專題針對姿態辨識演算法HRNet的推論部分進行運算資源優化以及計算過程加速。

本研究為達成上述目的，嘗試將推論所需運算量化，並且因為推論的核心計算為矩陣乘法，為了能降低記憶體存取的時間和能源消耗，來縮減計算資源的花費，我們利用在硬體端實作脈動式陣列來做矩陣運算。

最後本專題在軟體方面實作了三種不同的量化版本，其中權衡辨識效能及儲存空間、推論時間等因素後選取4位元版本作為硬體實作之規格。而硬體實作上以Verilog硬體描述語言進行模擬，主體採用的是脈動式陣列搭配其他控制電路，達成功能正確以及可使用Verilog模擬之效果。未來如果有機會能克服硬體空間大小的限制，期許能實際將模型量化之資料放進去脈動式陣列進行運算或是使用FPGA板實際演示本專題之硬體電路功能。

II. 軟體實作

訓練

本專題以COCO資料集先進行單精度浮點數(FP32)版本之訓練，於國網中心(TWCC)使用兩張NVIDIA Tesla V100 GPU訓練210個週期(epoch)收斂得到浮點數版本之模型。接著再持續使用一張 GPU將浮點數版本模型繼續以QAT模式進行量化，訓練若干週期後收斂得到量化版本之模型。本專題實作8bits、6bits及4bits三種量化版本與浮點數版本進行比較，並以4bits版本進行後續硬體實作。

推論

訓練完畢之量化版本模型將存有所有卷積層量化後之浮點數(w_{DQi})及所有輸入特徵圖之running_max、running_min，於推論時一層卷積層之運算可使用以下方式。

$$\sum w_{DQi}x_i = \sum\left(w_{Qi}S_w + r_{min,w}\right)\left(x_{Qi}S_x + \text{running\_min}\right)$$

$$= \left(\sum w_{Qi}x_{Qi}\right)(S_wS_x) + \left(\sum w_{Qi}\right)(S_w \cdot \text{running\_min})$$

$$+\left(\sum X_{Qi}\right)\left(S_x r_{min,w}\right) + \sum r_{min,w} \cdot \text{running\_min} \qquad (2.1)$$

其中$\sum w_{Qi}x_{Qi}$、$\sum w_{Qi}$及$\sum X_{Qi}$為進行整數之運算，計算完成後再分別與(S_wS_x)、$(S_w \cdot \text{running_min})$、$\left(S_x r_{min,w}\right)$進行浮點數相乘，最終將所有運算結果再與定值$\sum r_{min,w} \cdot \text{running_min}$加總得出輸出特徵圖(output feature map)之結果。而於硬體中之實作可將四個浮點數(S_wS_x)、$\left(S_x r_{min,w}\right)$、$(S_w \cdot \text{running_min})$、$\sum r_{min,w} \cdot \text{running_min}$做以下轉換。

$$n_{int} = \text{int}(n_{fp} \times 2^M) \qquad (2.2)$$

將所有浮點數放大適當2^M倍後取近似整數進行運算，式(2.1)即可近似為式(2.2)。

$$\sum w_{DQi}x_i = \left[\left(\sum w_{Qi}x_{Qi}\right)(S_wS_x)_{int} + \left(\sum w_{Qi}\right)(S_w \cdot \text{running\_min})_{int} + \left(\sum X_{Qi}\right)\left(S_x r_{min,w}\right)_{int} + \left(\sum r_{min,w} \cdot \text{running\_min}\right)_{int}\right] \gg M \qquad (2.3)$$

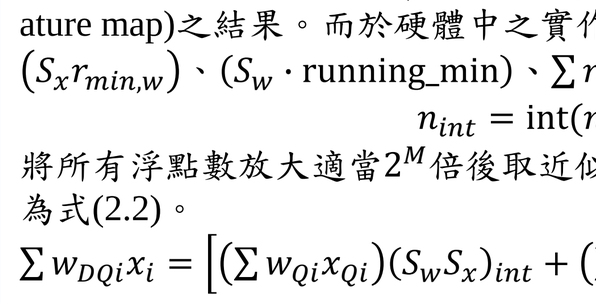
分析4bits量化版本所有卷積層之四種浮點數後我們將M選定為8，使近似後之整數運算誤差不致過高。由上述之運算方式即可利用量化版本之參數將所有浮點數運算轉換成整數運算，以便在硬體上加速模型推論時間。

III. 軟體實作結果

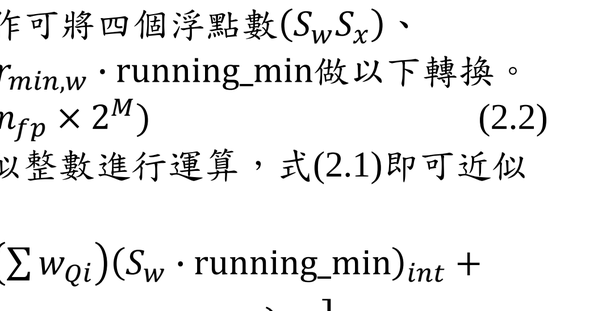
表一 各版本模型測試結果(COCO test dataset)

單位：mAP

FP32(Paper)	FP32(Our)	8bits	6bits	4bits
0.758	0.759	0.756	0.753	0.711



圖三 FP32版本



圖四 4bits版本

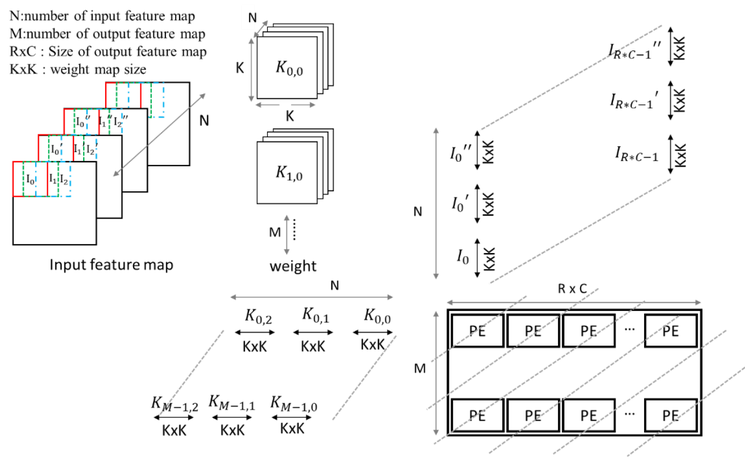
衡量mAP指標之跌幅及硬體實作上之時間與空間，我們選擇以4bits版本進行硬體實作。

由圖三及圖四可見，實際演示時以肉眼分辨沒有太大差別，僅少數細節有些微不同。

IV. 硬體設計

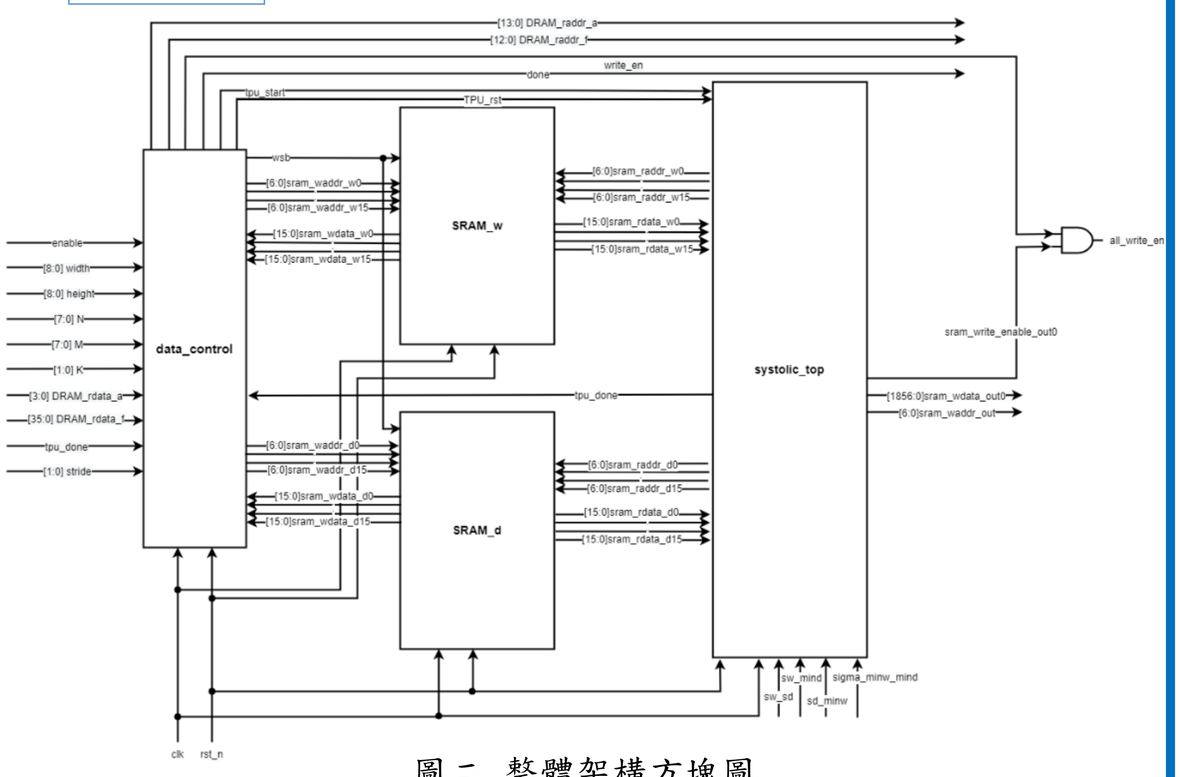
理論介紹

在模型中最常出現的運算是卷積層運算，所以我們的目標是實現卷積層的運算，依據脈動陣列的資料流排序以及將卷積運算可透過重新排列資料順序轉成矩陣乘法運算，圖一所示，N張輸入特徵圖與M組N張大小為KxK的權重圖，會輸出M張大小為RxC之輸出特徵圖。輸入特徵圖由上方送入，每一行由有同樣receptive filed的輸入特徵圖依序排列，而權重由左方送入，每一列為同一組權重依序排列，並將矩陣排列成平行四邊形形狀。另外，因為脈動陣列的特性，每一組輸出完成，會從左上角的對角線(灰色虛線)開始完成，往右下每個cycle完成一個對角線的運算單元，等所有資料通過脈動陣列，取出運算單元中的partial sum register的數值，便能得到輸出矩陣的數值，便可以完成卷積層的運算。



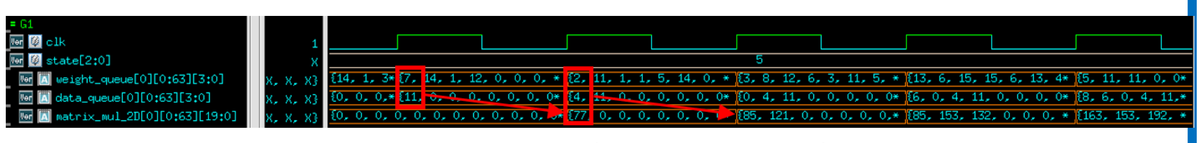
圖一 使用脈動陣列執行卷積運算的資料排列

整體架構



圖二 整體架構方塊圖

V. 硬體實作結果



圖五 脈動陣列中的運算單元$PE_{0,0}$計算波型圖

由於模型資料龐大難以測試，此報告的實做結果以10x10x14的輸入特徵圖以及65組權重做為輸入，測試實做結果是否與目標預期相同。
在第一筆輸出資料，脈動矩陣的結果加上前述軟體計算的量化常數，送入輸出SRAM中，之後利用testbench與軟體的結果比較是否相同，圖五顯示功能正確。

VI. 結論

本專題在軟體方面實作了三種不同的量化版本，其中權衡辨識效能及儲存空間、推論時間等因素後選取4位元版本作為硬體實作之規格。而硬體實作上以Verilog硬體描述語言進行模擬，主體採用的是脈動式陣列搭配其他控制電路，達成功能正確以及可使用Verilog模擬之效果。未來如果有機會能克服硬體空間大小的限制，期許能實際將模型量化之資料放進去脈動式陣列進行運算或是使用FPGA板實際演示本專題之硬體電路功能。

[1] Ke Sun, Bin Xiao, Dong Liu, Jingdong Wang, “Deep High-Resolution Representation Learning for Human Pose Estimation”, *CVPR*, Feb. 2019.
[2] CS231n: Convolutional Neural Networks for Visual Recognition, https://cs231n.github.io/convolutional-networks/
[3] TPU: Tensor-Processing-Unit- https://github.com/leo47007/TPU-Tensor-Processing-Unit
[4] Vivienne Sze, Yu-Hsin Chen, Tien-Ju Yang, Joel S, “EmerEfficient Processing of Deep Neural Networks: A Tutorial and Survey”, IEEE, Dec. 2017
[5] Zhijie Yang, Lei Wang, Dong Ding, Xiangyu Zhang, Yu Deng, Shiming Li, and Qiang Dou, “Systolic Array Based Accelerator and Algorithm Mapping for Deep Learning Algorithms”, NPC 29, Nov. 2018