適用動畫超解析之卷積神經網路處理器設計

Convolutional Neural Network Processor for Animation Super Resolution

組員： 張嘉祐、林俊曄、陳昭霖

指導教授：黃朝宗　　組別： A49

1. Abstract

在這個科技發達的世代，很多人使用手機或平板看動畫，但在網路不好的狀況下，傳輸速率變差，導致畫質下降，降低了我們的觀看品質。如果先行下載高畫質的影片至手機內，不僅要等待下載的時間而且會占掉許多寶貴的記憶體空間。因此在這個專題研究中，我們使用Pytorch訓練出一個針對海賊王動畫風格的超解析之CNN網路，讓畫質較差的圖片經過這個網路後解析度能提高，並以Verilog實作出一個CNN硬體加速器，最後以TSMC 40nm製程完成APR。最終目標為將此處理器應用於手機或平板這種方便攜帶的電子產品上，讓我們只需以少量的記憶體空間存放Parameters，便能隨時觀賞高畫質的動畫。

2. Implementation

I. Anime-ResNet Model

- 我們分別訓練圖1這些Model，以20張不在Data Set內的圖片測試，平均後得出其相對應的PSNR，並衡量Performance與該Model實作在硬體上所需的記憶體空間。我們最終決定採用ResBlock*4 和nFearture=24的架構，如表1所示。

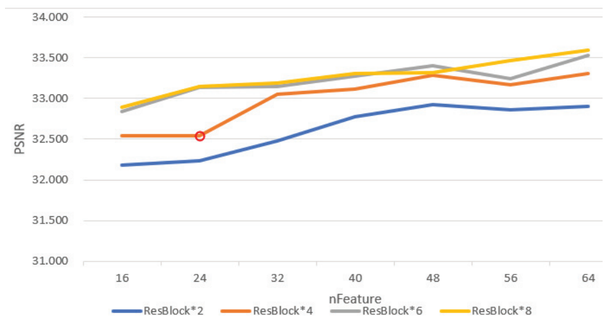

圖1 不同規格Model的比較 (紅圈是我們的Model)

- Model 的架構如圖2所示，包含四個ResBlock unit和兩個Upsampler，因此Output的圖片大小為Input的16倍。

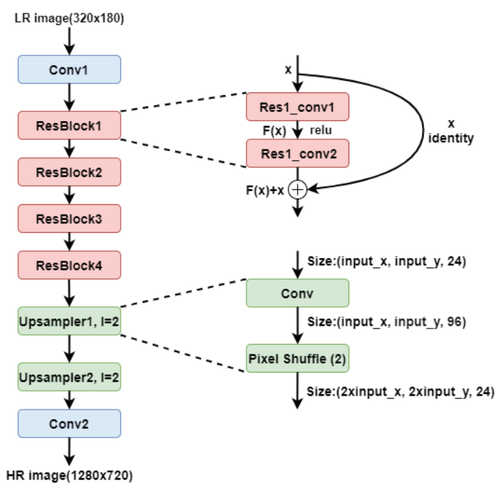

圖2 Anime-ResNet Model Structure

Parameter	nFeat	nResblock	nEpochs
Value	24	4	200

表1 Anime-ResNet Model Specification

- 利用Dynamic Fixed Point 可以將硬體的資源最大化。在硬體上不適合用32bits的浮點數運算，因此我們分析每層Parameters跟Activation的分布情形，再擷取最佳範圍，將每個bit的效用發揮到最大。

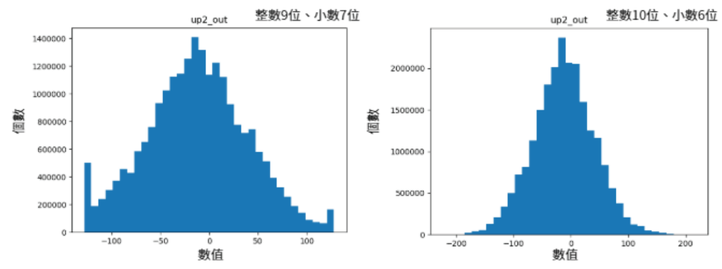

圖3 Dynamic Fixed Point Operation

- 如圖3所示，左圖是還未經過Dynamic Fixed Point調整的Upsampler 2 Activation分布圖，可觀察到有Saturation的現象，因此我們將小數部分的1個bit 分給整數部分，以完整表達所有整數部分。

II. 硬體架構設計

- 在數位電路設計中，為了使Throughput能達到動畫等級的規格(每秒輸出24張圖)，我們使用864個乘法器，一個Cycle可以輸出四個值存進SRAM。運算過程我們使用兩組相同規格的SRAM(A、B)交互存放Activation。

Input	320x180 image
Output	1280x720 image
乘法器數量	864個
SRAM for activation	45MB x2個 (SRAM_A&SRAM_B)
SRAM for weight	88KB x1個
SRAM for bias	0.4KB x1個

表2 Hardware Specification

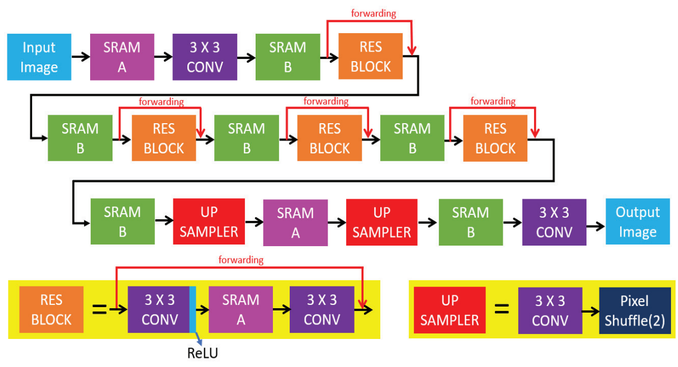

圖4 Data Flow

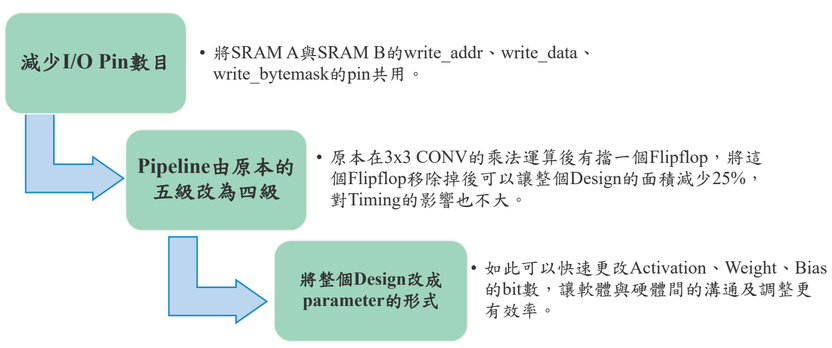

圖5 優化過程

3. Result

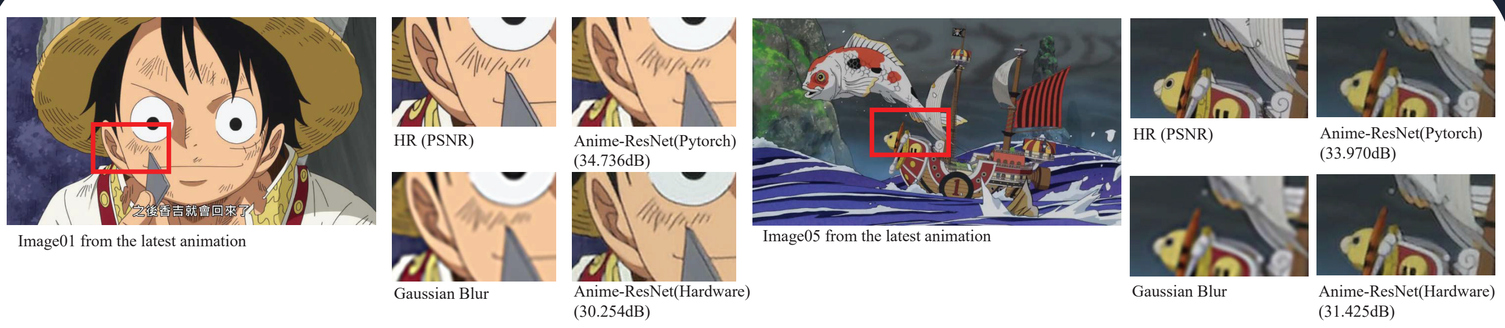

圖6 軟體硬體成果比較

Activation: 20bits	Synthesis	APR (Utilization 0.5)
Timing	3.8ns	5.4ns
Area	$844568\mu m^2$	$936079\mu m^2$
Power(dynamic):	$56.046mW$	$50.96mW$
Power(leakage):	$3.57\times10^3\mu W$	$4.46\times10^3\mu W$
Total Cycles	10,715,029個	10,715,029個
Throughput	24.56 image/sec	17.28 image/sec

表3 合成與APR之結果

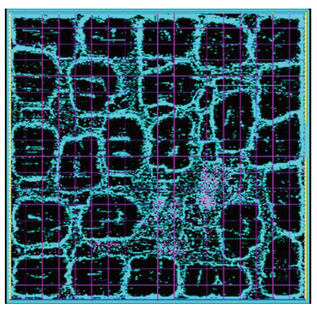

圖7 APR Layout

4. Contribution

- 軟體的Quantization，除了Parameter，將 Activation 也做Dynamic Fixed Point的處理，可以讓整體PSNR再提高0.3dB。
- 硬體設計處理Resblock Forwarding運算，使用先讀後寫，利用Pipeline錯開SRAM讀值與寫值的時間，將Forwarding的值從SRAM_B讀出，處理完加法後再寫回SRAM_B同一位置，如此雖然會有多一個加法運算的Overhead，但可以節省大量的Cycle、不必多開一組SRAM去存值。