AI on Chip 2024

Final - MobilenetV3 Accelerator

REPORT

Student name: \_胡家豪、蔡明翰、王文楷、陳冠穎、黃雍翔

Student ID: N26122246\_\_\_\_\_\_\_\_

目錄

[一. Software 2](#_Toc169641944)

[1. Model introduction 2](#_Toc169641945)

[2. Model Training 4](#_Toc169641946)

[3. Quantization and Hardware mapping 4](#_Toc169641947)

[4. Hardware mapping 7](#_Toc169641948)

[5. Golden generation 7](#_Toc169641949)

[6. Software Result 8](#_Toc169641950)

[二. Hardware 8](#_Toc169641951)

[1. System Overview 8](#_Toc169641952)

[2. System Architecture 9](#_Toc169641953)

[3. Network on Chip 9](#_Toc169641954)

[三. Verification 10](#_Toc169641955)

[四. Share your thoughts 10](#_Toc169641956)

1. Software
2. Model introduction
3. MobileNetV3 overview

本次實作中，我們以mobilenetV3 為目標進行實作，其主要是由pointwise convolution進行運算，並且使用了SE layer與h swish增加精確度。

一張含有 電子藍, 行, 圖表, 貨櫃 的圖片

自動產生的描述

在原有的架構中，一個Mobilenet V3 block的流程為：pointwise conv -> depthwise conv -> SE layer -> pointwise conv。

原始的MobileNet V3中，總共含有11個這樣的block ，並且每一層這樣的block都可以自由選擇是否要用H-swish或是ReLU作為activation function以及是否需要SE layer。

1. Pointwise and Depthwsie

作者使用了Pointwise 加上 Depthwise來取代原有的Convolution，下方左圖為Depthwise conv之示意圖、右側則為Pointwise之示意圖

一張含有 鮮豔, 螢幕擷取畫面, 圖形, 正方形 的圖片

自動產生的描述 一張含有 螢幕擷取畫面, 鮮豔, 圖表, 像素 的圖片

自動產生的描述

作者透過使用Depthwise 作為提取特徵、使用pointwise 作為特徵混和，以此來取代原本的Convolution。

1. SE layer

一張含有 圖表, 行, 方案, 工程製圖 的圖片

自動產生的描述

SE layer可以看做是圖片的attention 機制，將原有的圖片藉由訓練乘上一個權重來得知哪個channel比較重要。

1. H-swish

一張含有 文字, 行, 繪圖, 字型 的圖片

自動產生的描述

Hswish是本篇用於取代原有ReLU的activation function，用於解決ReLU在負數部分消失的狀況，以此增加精確度。

一張含有 文字, 螢幕擷取畫面, 字型, 數字 的圖片

自動產生的描述

1. 一張含有 文字, 螢幕擷取畫面, 字型, 數字 的圖片

   自動產生的描述Model Training

由於原本的Mobilenet在模型建立上有較高的自由性，為了硬體的設計方便，我們將自己設計了一個Mobilenet V3 Like 的模型，與原有模型的差異在於我們的block每一層的內容都是固定的，在架構比較固定的狀況下比較好映射上硬體。

右側是我們的block架構，與原本的mobilenet block接近，差異是我們直接固定這個架構在每一層，而不能在特定層決定activation與要不要SE

一張含有 文字, 螢幕擷取畫面, 字型 的圖片

自動產生的描述

上圖是我們整體架構，我們主要使用了7個block進行模型建立。

1. Quantization and Hardware mapping
2. BN Fold

在模型訓練完畢，已經在inference的階段，由於BN層的參數均已經固定，所以我們可以將其透過下列公式併入前一層的Conv layer

一張含有 圖表 的圖片

自動產生的描述一張含有 文字 的圖片

自動產生的描述

在實作上，我們建立了一個專門用來folding BN的function，並且遍歷我們的model來進行BN fold

一張含有 文字, 螢幕擷取畫面, 字型, 軟體 的圖片

自動產生的描述一張含有 文字, 螢幕擷取畫面, 軟體 的圖片

自動產生的描述

1. Normal Quantization vs Our Quantization

在Inference-only這篇的PTQ說道，其量化可以透過以下方式進行

透過上述公式，可以將其圖像如下：

一張含有 文字, 圖表, 方案, 工程製圖 的圖片

自動產生的描述一張含有 文字, 螢幕擷取畫面, 軟體 的圖片

自動產生的描述

他可以這樣做是因為一般模型都是使用ReLU進行，而ReLU在正的部分其實就是Linear的。

如果要將此方法應用到我們的模型上並不能直接應用，原因是因為hswish其實並不是一個線性函數，因此我們可以將PTQ的量化公式寫成如下形式：

由此可知，Inference-only的步驟我們必須先進行DeQuant，經過hswish後，再進行ReQuant

一張含有 螢幕擷取畫面, 圖表, 文字, 設計 的圖片

自動產生的描述流程如左圖表示。

1. SE layer Quantization

在SE layer中，由於有相乘與pooling的運算，所以也必須特別處理，下圖是我們的處理方式，在pool的後方與相乘的後方加入fake Quant(或稱Requant)，而中間的Linear則使用一般的QLinear

一張含有 文字, 圖表, 方案, 工程製圖 的圖片

自動產生的描述

一張含有 文字, 螢幕擷取畫面, 字型 的圖片

自動產生的描述

1. Hardware mapping

完成量化後，就可以開始針對量化flow將需要的運算映射到硬體上面，參考下面的圖，說明在圖片右側：

一張含有 文字, 圖表, 平行, 字型 的圖片

自動產生的描述

Software preprocess：由於我們的量化有zero point，所以會在軟體先把需要與zero point操作的部分先做好，例如Q\_wieght與Q\_input，input\_scale與weight\_scale的相乘，1/output\_scale等

PE array：在PE array中主要是執行Convolution與psum還有bias的相加

Postprocess：主要執行De\_quant與activation function與requant。

* 首先Convolution的運算會輸入量化為8 bits的權重與輸入，其結果為16 bits
* 用24 bits累加psum與bias
* 進行Dequant ，也就是乘上dequant scale，在這邊我們將scale表示乘一個8 bits的fix point，其小數點點在最左邊，並且記錄需要的位移。Dequant的輸出是一個24bit整數與8bit小數的fixed point，這個數往右上面的位移數就是結果
* 經過swish與requant，由於1/output\_scale通常是整數，所以直接乘就可以
* 最後的結果truncate掉小數部分即為解答。

對於dequant的部分圖像化為下圖：

一張含有 圖表, 行, 螢幕擷取畫面, 文字 的圖片

自動產生的描述

1. Golden generation

完成上述部分後，就可以將model的input以及weight轉換成硬體所需要的形式，我們的硬體input 與 weight存放在SRAM當中，由於使用的32 位元CPU支援LH等指令，所以需要一個Byte一個Byte的存。下面左圖是量化前的輸入、中圖是量化後的輸入、右圖則是將輸入轉換成硬體可以讀取的十六進位形式。

一張含有 文字, 字型, 螢幕擷取畫面, 功能表 的圖片

自動產生的描述 一張含有 文字, 字型, 螢幕擷取畫面, 功能表 的圖片

自動產生的描述 一張含有 文字, 字型, 螢幕擷取畫面, 設計 的圖片

自動產生的描述

1. Software Result

下面是我們的軟體結果，可以看到我們自己搭建的模型得到了88.1%的精確度；BN\_fold的模型也獲得同樣的精確度，以此驗證我們的BN 方法是正確的、最後是我們的模型經過PTQ的精確度，由於模型較深且沒有用QAT的方式train，所以最後量化誤差會不斷累積導致精確度下降。因此量化後的模型精確度為70.1%，還在堪用的範圍內。

|  |  |
| --- | --- |
| **Model** | **accuracy** |
| Our-mobilenet | 88.1% |
| BN\_fold\_our\_mobilenet | 88.1% |
| Q\_BN\_fold\_our\_mobilenet | 70.1% |

1. Hardware
2. System Overview

一張含有 文字, 螢幕擷取畫面, 便利貼, Rectangle 的圖片

自動產生的描述

1. System Architecture

一張含有 文字, 螢幕擷取畫面, Rectangle, 正方形 的圖片

自動產生的描述

1. Network on Chip

一張含有 螢幕擷取畫面, 文字, 正方形, 設計 的圖片

自動產生的描述

PE array size: 16\*16

|  |  |  |  |
| --- | --- | --- | --- |
| Layer | Convolution type | Ofmap Size | Utilization |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

1. Verification
2. Share your thoughts
3. 胡家豪

經過了本次專題我學習到了很多，主要我是負責軟體的部分，包括模型搭建與量化。除了對量化的flaw更加認識以外，我也理解到其實在做量化有一部份就是在設計model。必須不斷的與硬體端溝通到底要做那些運算，還必須考慮到位寬到底夠不夠，會不會溢位等。雖然這次專題很累，但是收獲也很大！

1. 蔡明翰
2. 王文楷
3. 陳冠穎
4. 黃雍翔