CVPDL HW2

洪郡辰
R11944050

April 18, 2023

HW2

R11944050 沈昭翰

P1.1

$$Y[x,y,s]=\sum_{i=1}^{k}\sum_{j=1}^{k}\sum_{c=1}^{c_{in}}|X[x+i,y+j,c]-W[i,j,c,s]|$$

在原本使用的 $S(\cdot,\cdot)$ 是透过 Cross-Correlatin measure similarity

在加法取代 $S(\cdot,\cdot)$ 是透过 L1-distance measure similarity，取絕對值便正負偏差都視為誤差

P1.2

Convolution	addition	multiplication
normal	$(k-1)(k-1)\cdot(c_{in}-1)$	$k\cdot k\cdot c_{in}$
faster	$(k-1)(k-1)(c_{in}-1)+k\cdot k\cdot c_{in}$	0

因為每個 pixel 在 convolution 時計算量都一樣，表格中為對一個 pixel convolution 的次數

P1.3

$$\frac{\partial Y[x,y,s]}{\partial W[i,j,c,s]}=\begin{cases}1, & \text{if } X[x+i,y+j,c]-W[i,j,c,s]>0\\ 0, & \text{if } X[x+i,y+j,c]-W[i,j,c,s]=0\\ -1, & \text{if } X[x+i,y+j,c]-W[i,j,c,s]<0\end{cases}$$

$$\frac{\partial\sqrt{(X-W)^2}}{\partial W}=-1\cdot\frac{2\cdot 1\cdot(X-W)}{2\sqrt{(X-W)^2}}=-1\cdot\frac{W-X}{\sqrt{(X-W)^2}}=-1\cdot\frac{(W-X)|X-W|}{|X-W||X-W|}=-1\cdot\frac{(W-X)\cdot|X-W|}{(X-W)^2}$$

$$=1\cdot\frac{-|X-W|}{(X-W)}=\frac{|X-W|}{(X-W)}\rightarrow\begin{cases}1 & \text{if } X-W>0\\ 0 & \text{if } X-W=0\\ -1 & \text{if } X-W<0\end{cases}$$

P1.4 P1.3中的 partial derivative 可視為 signSGD，它會導致在 optimization 中因為只沿著 sign direction 優化而收斂差及訓練不穩定，忽略了 gradient magnitude 的影響。

P1.5 The variance of W should be $\frac{1}{k^2c_{in}}$

$$Var[Y]=\sum_{i=0}^{k}\sum_{j=0}^{k}\sum_{c=0}^{c_{in}}Var[X]\cdot Var[W]$$

$$=k^2c_{in}\,Var[X]\cdot Var[W]$$

$$\therefore Var[W]=\frac{1}{k^2c_{in}}$$

1-6.

據P1-5, normal convolution使 output的 variable = input的 variance。faster convolution 中是 $Y = X \cdot W$ 而不是 $Y = X \cdot W$。無法透過 $Var(Y) = Var(X) \cdot Var(W)$ 限制 $Var(Y)$，因此 Var(Y) from faster convolution is much larger than from normal convolution。

1-7. 因為在 faster 中 gradient 小，filter update 速度慢，因此會希望以較大的 learning rate 學習，但又能發現不同 layer 的 gradient 差異很大，所以需要透過 adaptively adjusting learning rate 讓每個 filter 以相近的速度學習。

設 ΔW_l 為第 l 層的 update，r 為 global learning rate，B_l 為 l 層的 local learning rate

$\Delta L(W_l)$ 為第 l 層的 gradient。

$$\Delta W_l = r \cdot B_l \cdot \Delta L(W_l)$$

$$B_l = \frac{\eta \sqrt{k}}{\|\Delta L(W_l)\|_2}$$

k 為在 l 層的 elements 數量，η 為控制 layer learning rate 的 hyper parameter。

Problem 2

2.1

在這三種 pruning 技術中最大的差異是 pruning 對象的不同，weight pruning 依照 weight 大小進行移除、neuron pruning 依照 neuron 對 mdoel 的貢獻進行移除、filter prunuing 依照 filter 對 model 的貢獻進行移除。三種 pruning 技術的主要優點都在於減少 model 的儲存大小及計算量，讓 model 能在較少計算及儲存資源上的硬體也能運作。

2.2

我選擇的 paper 是 Filter Pruning via Geometric Median for Deep Convolutional Neural Networks Acceleration，在這篇論文中它透過改變 pruning criterion 優化 pruning 效果，它將“smaller-norm-less-important”的 norm-based criterion 改成 geometric median-based criterion，這個方法的優點是不受 norm-based criterion’s constraints，分別為:

- the norm deviation should be large
- the minimum of the norm should be arbitrarily small

在 geometric median-based criterion 下，它選擇在同一層中與其他 filter 的 geometric median 最小的 filter 進行 pruning，它認為這個 filter 代表 the common information of all the filters within the single layer，在 fine-tuning 後，能夠輕鬆地回復原有的 performance，因為被 pruning 的 filter 中所擁有的資訊也存在其他 filter 中。透過這個方法能夠避免掉以上提到的 norm-based criterion’s constraints，讓 pruning model 仍有良好的表現。

2.3

在 weight pruning 後會將某些 weight 值設為 0，這些設為 0 的 weight 不會進行運算及儲存，會減少 MACs 的次數及被儲存的參數數量。再參數部分要多考慮 bias vector。

以下是 pruning 前的 model:

- MACs:512*1024=524288
- number of parameters:512*(1024+1)=524800

以下是 pruning 後的 model:

- MACs:round(512*1024*0.7)=367002
- number of parameters:512*1024*0.7+512=367514

P3.1 $X \in R^{H\times W\times C}$, $Y \in R^{H\times W\times C}$, $W \in R^{K\times K\times C\times C}$

	MACs	# of parameters	# of memory access
regular	$H\cdot W\cdot k^2\cdot C^2$	$k^2\cdot C^2+C$	$2HWk^2C^2+HWC$
DW	HWk^2C	$k^2\cdot C+C$	$2HWk^2C+HWC$
PW	HWC^2	C^2+C	$2HWC^2+HWC$

regular:

① 考虑一个 pixel

MACs: $k\cdot k\cdot C$

共有 $H\cdot W\cdot C$ 个 pixel 产生

∴ MACs: $H\cdot W\cdot C\cdot k\cdot k\cdot C = HWk^2C^2$

② 共有 k^2C^2 个 parameters, 再加上 bias C 个

③ 考虑 1 个 pixel

Read: $k\cdot k\cdot C + k\cdot k\cdot C = 2k^2C$

(X) (kernel)

共有 $H\cdot W\cdot C$ 个

Read: $H\cdot W\cdot C\cdot (2k^2C) = 2HWk^2C^2$

Write: $H\cdot W\cdot C$

Depthwise:

① 考虑一个 pixel

MACs: $k\cdot k\cdot 1$

共有 $H\cdot W\cdot C$ 个 pixel 产生

∴ MACs: $H\cdot W\cdot C\cdot k^2 = HWk^2C$

② 共有 $k^2\cdot C$ 个 parameter, 再加上 bias C 个

③ 考虑 1 个 pixel

Read: $k\cdot k\cdot 1 + k\cdot k\cdot 1 = 2k^2$

(X) (kernel)

共有 HWC 个

Read: $H\cdot W\cdot C\cdot 2k^2 = 2HWk^2C$

Write: HWC

Pointwise: (k=1)

① HWC^2

② C^2+C

③ $2HWC^2+HWC$

P3.2

	MACs	parameters	memory access
regular	HWk^2c^2	k^2c^2+c	$2HWk^2c^2+HWc$
depthwise separable	HWk^2c + HWc^2	k^2c+c + c^2+c	$2HWk^2c+HWc$ + $2HWc^2+HWc$

$HWc(k^2c)$ $\quad c\cdot(k^2c+1)$ $\quad 2HWc(k^2c+\frac{1}{2})$

$HWc(k^2+c)$ $\quad c\cdot(k^2+c+2)$ $\quad 2HWc(k^2+c+1)$

可以從三種 metrics 看出在 kernel size 越大及 kernel Map 數量越多的情況下，能夠節省越多的計算成本。

它的優勢在於計算成本較低，在計算量及 memory access 上都比 regular convolution 低，可以應用在較少運算資源的裝置如手機等。或是用在有較嚴格的時間限制的應用上。在儲存成本上相較於 regular convolution 也較低。這種優勢在計算成本越高時越明顯。

P3.3

MACs	parameters	memory access
$HW4c^2$ + $HW9\cdot4c$ + $HW4c^2$ $=8HWc^2+36HWc$	$4c^2+4c+9\cdot4c+4c$ $+4c^2+c$ $=8c^2+45c$	$8HWc^2+HW4c$ $+16HW4c+HW4c$ $+8HWc^2+HWc$ $=16HWc^2+81HWc$

相較原本的設計，計算成本因為在一開始先升維使其上升許多，模型複雜度也隨之上升，也會造成 model 泛化能力下降。