

SUPPLEMENTARY MATERIAL FOR SUCCINCT NETWORK CHANNEL AND SPATIAL PRUNING VIA DISCRETE VARIABLE QCQP

Anonymous authors

Paper under double-blind review

A QCQP FORMULATION ON NONSEQUENTIAL CONVOLUTIONS

A.1 SKIP ADDITION

We define *output channel activation* $u^{(t)} \in \{0, 1\}^{C_t}$ and *input channel activation* $v^{(t)} \in \{0, 1\}^{C_t}$ to each indicate remaining output channels of the t -th convolution layer and remaining input channels of the $t+1$ -th layer, respectively. Note that similar to Section 3.2, the definition of $u^{(t)}$ induces constraints between the shape column activations ($= q^{(t)}$) and output channel activation variables ($= u^{(t)}$). Concretely, the constraints are given as $u_j^{(t)} \leq \sum_{a,b} q_{j,a,b}^{(t)}$ and $q_{j,a,b}^{(t)} \leq u_j^{(t)} \forall a, b$.

We now discuss the constraints between the input and output channel activation variables under four possible scenarios depending on the architectural implementations of skip additions. Here, we denote the set of layer index pairs $\{(s, t)\}$ which have skip additions as \mathcal{P} . Concretely, $(s, t) \in \mathcal{P}$ if and only if the input feature map of the $s+1$ -th convolution layer is added to the output feature map of the t -th layer, forming the input feature map of the $t+1$ -th layer. Also, let $T = \{t \mid (s, t) \in \mathcal{P}\}$. For a layer t , we formulate the channel activation constraints for each possible connection scenarios separately:

- (i) If there is no skip addition incoming to the t -th layer ($t \notin T$), then we force $u^{(t)} = v^{(t)}$.
- (ii) For a skip addition pair $(s, t) \in \mathcal{P}$ with matching channel dimensions ($C_s = C_t$), the input feature map of the $s+1$ -th convolution layer is directly added to the output feature map from the t -th layer as illustrated in Figure 1a. In this case, we can formulate the constraints as $u^{(t)} \preceq v^{(t)} \preceq u^{(t)} + v^{(s)}$.
- (iii) For a skip addition pair $(s, t) \in \mathcal{P}$ with mismatching channel dimensions ($C_s < C_t$), the skip addition can utilize zero padding (iii-a) or 1×1 convolutions (iii-b) to resolve the mismatch (He et al., 2016). We define the *augmented channel activation* $\tilde{v}^{(s)} \in \{0, 1\}^{C_t}$ and formulate the constraints for both cases as below.
 - (iii-a) A $(C_t - C_s)$ -dimensional zero-valued feature map is padded to the end of the $s+1$ -th convolution layer’s input feature map. We define $\tilde{v}^{(s)} = [v^{(s)}, 0_{C_t - C_s}]$, as illustrated in Figure 1b.
 - (iii-b) 1×1 convolution is applied to the $s+1$ -th layer’s input feature map to match the larger channel dimension ($= C_t$). Since the number of FLOPs and weights in a 1×1 convolution is negligible compared to the total number of FLOPs and weights, we assume all of the output channels of a 1×1 convolution are activated and define $\tilde{v}^{(s)} = 1_{C_t}$ as illustrated in Figure 1c.

We now summarize the constraints for the four cases discussed above to Equation (1).

$$\begin{aligned}
 (i) \quad & v^{(t)} = u^{(t)} \quad \forall t \notin T & (ii) \quad & u^{(t)} \preceq v^{(t)} \preceq u^{(t)} + v^{(s)} \quad \forall (s, t) \in \mathcal{P} \text{ and } C_t = C_s \\
 (iii) \quad & \forall (s, t) \in \mathcal{P} \text{ and } C_t > C_s, \\
 (iii-a) \quad & u_j^{(t)} \leq v_j^{(t)} \leq u_j^{(t)} + v_j^{(s)} \quad \forall j \leq C_s \text{ and } v_j^{(t)} = u_j^{(t)} \quad \forall j > C_s \\
 (iii-b) \quad & u^{(t)} \preceq v^{(t)}.
 \end{aligned} \tag{1}$$

Proposition 2 shows that the constraints of Equation (1) prevent inactive weights from remaining in the pruned network with skip additions.

Proposition 2. *Optimizing over the input and output channel activation variables $u^{(0:L)}, v^{(0:L)}$ and shape column activation variables $q^{(1:L)}$ under the constraints in Equation (1) prevents the existence of any inactive weights in the pruned network guaranteeing exact computation of 1) resource usage and 2) the sum of the importance of active weights in the pruned network.*

Proof. See supplementary D. □

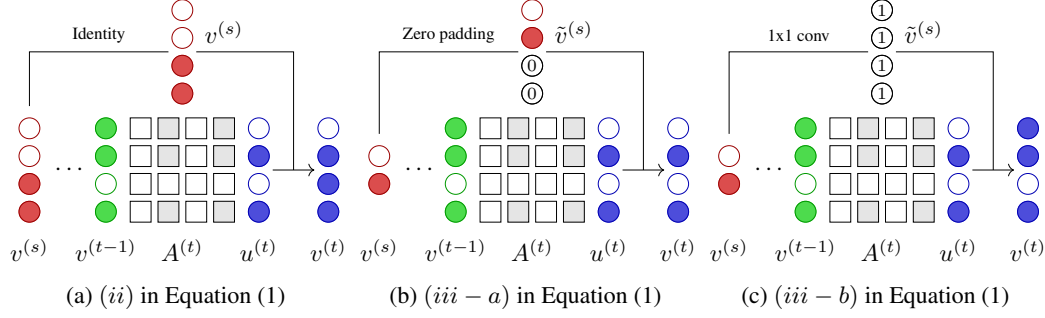


Figure 1: Illustration of the skip addition procedure for each skip addition scenarios.

We now formulate the network channel and spatial pruning optimization problem handling non-sequential convolutions with the input channel, output channel, and shape column activation variables below in Equation (2).

$$\begin{aligned}
 & \underset{u^{(0:L)}, v^{(0:L)}, q^{(1:L)}}{\text{maximize}} && \sum_{t=1}^L \langle I^{(t)}, A^{(t)} \rangle \\
 & \text{subject to} && \sum_{t=0}^L a_t \|u^{(t)}\|_1 + \sum_{t \in T} a_t \|v^{(t)}\|_1 + \sum_{t=1}^L b_t \|A^{(t)}\|_1 \leq M \\
 & && u_j^{(t)} \leq \sum_{a,b} q_{j,a,b}^{(t)} \quad \text{and} \quad q_{j,a,b}^{(t)} \leq u_j^{(t)} \quad \forall t, j, a, b \\
 & && A^{(t)} = v^{(t-1)} \otimes q^{(t)} \quad \forall t \\
 & && u^{(t)}, v^{(t)} \in \{0, 1\}^{C_t} \quad \text{and} \quad q^{(t)} \in \{0, 1\}^{C_t \times K_t \times K_t} \quad \forall t \in [L] \\
 & && (i) \quad v^{(t)} = u^{(t)} \quad \forall t \notin T \\
 & && (ii) \quad u^{(t)} \preceq v^{(t)} \preceq u^{(t)} + v^{(s)} \quad \forall (s, t) \in \mathcal{P} \quad \text{and} \quad C_t = C_s \\
 & && (iii) \quad \forall (s, t) \in \mathcal{P} \quad \text{and} \quad C_t > C_s, \\
 & && \quad (iii - a) \quad u_j^{(t)} \leq v_j^{(t)} \leq u_j^{(t)} + v_j^{(s)} \quad \forall j \leq C_s \quad \text{and} \quad v_j^{(t)} = u_j^{(t)} \quad \forall j > C_s \\
 & && \quad (iii - b) \quad u^{(t)} \preceq v^{(t)}
 \end{aligned} \tag{2}$$

Concretely, Equation (2) reduces to the optimization problem for sequential convolution networks when $u^{(t)} = v^{(t)}$, $q_{j,a,b}^{(t)} = u_j^{(t)}$, and $\mathcal{P} = \emptyset \quad \forall t, j, a, b$.

A.2 SKIP CONCATENATION

Skip concatenation, which is a crucial feature of the well-known DenseNet (Huang et al., 2017), requires different techniques from skip addition. Concretely, the skip concatenation of a layer pair (p, q) means the p -th layer's feature map is concatenated with the q -th layer's feature map before the $q+1$ -th convolution. To handle the possible skip concatenations, we utilize the fact that when (p, q) is the skip concatenation pair, $q+1$ -th convolution operation on the q -th layer can be thought as separate convolution operations on the p -th layer and the q -th layer, respectively. In this regard, we first assume there are convolution operations between every pair of layers. Then, we define

$W^{(p,q)} \in \mathbb{R}^{C_p \times C_q \times K_{p,q} \times K_{p,q}}$ as the convolution weights between p -th layer and q -th layer where $p < q$. If there is no skip concatenation from p -th layer to $q-1$ -th layer, we regard there is no convolution operation between p -th layer and q -th layer and set $W^{(p,q)} = 0_{K_{p,q} \times K_{p,q}}$. Also, we introduce the corresponding shape column activation variables, $q^{(p,q)} \in \{0, 1\}^{C_q \times K_{p,q} \times K_{p,q}}$, for the convolution operation from p -th layer to q -th layer. Then, we extend the optimization problem for skip concatenation as

$$\begin{aligned}
& \underset{r^{(0:L)}, q^{(1:L, 1:L)}}{\text{maximize}} && \sum_{p=1}^L \sum_{q=1}^L \langle I^{(p,q)}, A^{(p,q)} \rangle \\
& \text{subject to} && \sum_{p=0}^L a_p \|r^{(p)}\|_1 + \sum_{p=1}^L \sum_{q=1}^L b_{p,q} \|A^{(p,q)}\|_1 \leq M \\
& && r_j^{(q)} \leq \sum_{a,b} q_{j,a,b}^{(p,q)} \quad \text{and} \quad q_{j,a,b}^{(p,q)} \leq r_j^{(q)} \quad \forall p, q, j, a, b \\
& && A^{(p,q)} = r^{(p)} \otimes q^{(p,q)} \quad \forall p, q \\
& && r^{(p)} \in \{0, 1\}^{C_p} \text{ and } q^{(p,q)} \in \{0, 1\}^{C_q \times K_{p,q} \times K_{p,q}} \quad \forall p, q \in [L].
\end{aligned} \tag{3}$$

B COORDINATE DESCENT STYLE OPTIMIZATION

In this section, we provide a block coordinate descent-style optimization algorithm for solving Equation (2) in Algorithm 1. Note that Equation (2) is the generalized version of Equation (3) in Section 3.2.

Algorithm 1 Succinct channel and spatial pruning optimization via QCQP

Input : $B, M, \gamma, a_l, a'_l, b'_l, I^{(l)}, \forall l$
Initialize $u^{(0:L)}, v^{(0:L)}$.
 $M := M/\gamma$.
for $n = 1, \dots, \text{MAXITER}$ **do**
 for $i = 0, \dots, L - B + 1$ **do**
 $z = [u^{(i:i+B-1)}, v^{(i:i+B-1)}]$
 $\tilde{f}(z) = f(z; \text{rest of the variables in } u, v \text{ fixed})$.
 Optimize $\tilde{f}(z)$ with respect to z under the constraints in Equation (2).
 end for
end for
 $M := \gamma M$
Optimize Equation (2) with respect to $q^{(1:L)}$ while fixing $u^{(0:L)}$ and $v^{(0:L)}$.
Output : $u^{(0:L)}, v^{(0:L)}, q^{(1:L)}$

We first set all shape column activation variables to $(q_{j,a,b}^{(t)} = u_j^{(t)} \forall t, j, a, b)$. Then, we optimize over the input and output channel activation variables $(u^{(0:L)}, v^{(0:L)})$ in a block coordinate descent fashion with the resource constraint M/γ where γ is the average spatial sparsity smaller than 1. Then, we optimize over the shape column activation variables $q^{(1:L)}$, fixing the input and output channel activation variables. In all experiments using Algorithm 1, γ is decreased from ‘ $M/(\text{Resource requirement of the initial network})$ ’ to 1.0 with a step size of 0.1. After the pruning procedure, we employ one round of finetuning on the pruned network. Note that in Algorithm 1, we denote the objective function in Equation (2) as $f(\cdot)$ when the shape column activations are all forced to match the output channel activations $(q_{j,a,b}^{(t)} = u_j^{(t)} \forall t, j, a, b)$. z denotes the concatenated variables of input and output channel activation in the target block.

We additionally conducted an experiment to check the CPLEX performance of Algorithm 1 compared to direct optimization on Equation (2), which we denote as Algorithm 0. However, Algorithm 0 is not scalable even in ResNet-20 on CIFAR-10. Therefore we compare Algorithm 0 and 1 for the first eight

layers of ResNet-56. We set $B = 2$ and adjust γ with a step size of 0.1. Algorithm 1 succeeds in increasing the objective value to 100.16 in 12 minutes, while Algorithm 0 reaches 106.27 in 1 hour. Also, Algorithm 1 requires only 3 hours and 4GB memory for pruning ResNet-50 on ImageNet.

C IMPLEMENTATION DETAILS

For ResNet experiments, we mostly follow the implementation from FPGM (He et al., 2019). We apply batch normalization and remove bias weight in every convolution layer. Zero padding and 1×1 convolution are used as the downsampling technique in CIFAR-10 (Krizhevsky et al., 2009) and ImageNet (Russakovsky et al., 2015), respectively.

For CIFAR-10 experiments on ResNet (He et al., 2016) architectures, we finetune the pruned model from the pretrained network given in He et al. (2019) and follow the protocol of He et al. (2019) for fair comparison. We finetune the pruned network for 200 epochs with batch size 128 and initial learning rate 0.01. Then, we adjust the learning rate at 60, 120, and 160 epoch by multiplying 0.2 each time. We use SGD optimizer with momentum 0.9, weight decay 5×10^{-4} , and Nesterov momentum. For CIFAR-10 experiments on the DenseNet-40 (Huang et al., 2017) architecture, we finetune the pruned network for 300 epochs with batch size 128 and initial learning rate 0.1. We adjust the learning rate at 150 and 225 epoch by multiplying 0.1. Here, we also use SGD optimizer with momentum 0.9, weight decay 5×10^{-4} , and Nesterov momentum.

For ImageNet experiments on ResNet architectures, we follow the protocol of FPGM and start from the pretrained network provided by PyTorch (Paszke et al., 2017). We finetune the pruned network for 80 epochs on ImageNet with batch size 384 and the initial learning rate of 0.015. Then, we adjust the learning rate at 30 and 60 epoch by multiplying 0.1. Here, we use SGD optimizer with momentum 0.9 and weight decay 10^{-4} . For ImageNet experiments on VGG-16 (Simonyan & Zisserman, 2015), we follow the protocol of Molchanov et al. (2017) for network training. We start from the pretrained network from Pytorch. We finetune the pruned network using SGD optimizer with a constant learning rate 10^{-4} and batch size 32 for 5 epochs. We run all ImageNet experiments on a machine with Intel Xeon E5-2650 CPU and 4 TITAN XP GPUs.

D PRUNING CONSISTENCY

Pruning operation that removes weights through output channel direction leads to *inactive* weights during the pruning procedure and prevent the exact modeling of the hard resource constraints (FLOPs and network size). In previous channel pruning methods based on the greedy approach, the pruned network requires post-pruning procedures to eliminate the remaining inactive weights. However, our formulation guarantees the exclusion of inactive weights from the pruned network.

D.1 PRELIMINARY

We assume each pruning methods outputs a pruning mask $A^{(t)} \in \{0, 1\}^{C_{t-1} \times C_t \times K_l \times K_l}$. Then, we denote the pruned weights as $\hat{W}^{(t)} = W^{(t)} \odot A^{(t)}$. In the pruned network with pruned weights $\hat{W}^{(1:L)}$, we denote the input feature map of the $t+1$ -th convolution as $V^{(t)} \in \mathbb{R}^{C_t \times H_t \times W_t}$. Also, we denote the output feature map of the t -th convolution as $U^{(t)} \in \mathbb{R}^{C_t \times H_t \times W_t}$. To avoid notation clutter, we ignore batch normalization and nonlinear activation function in this section. Then, $U^{(t)} = g^{(t)}(V^{(t-1)}; \hat{W}^{(t)})$, where $g^{(t)} : \mathbb{R}^{C_{t-1} \times H_{t-1} \times W_{t-1}} \rightarrow \mathbb{R}^{C_t \times H_t \times W_t}$, represents the convolution operation. In particular,

$$U_j^{(t)} = \sum_{i=1}^{C_{t-1}} g_{i,j}^{(t)} \left(V_i^{(t-1)}; \hat{W}_{i,j}^{(t)} \right), \quad (4)$$

where $g_{i,j}^{(t)} : \mathbb{R}^{H_{t-1} \times W_{t-1}} \rightarrow \mathbb{R}^{H_t \times W_t}$ is a 2-D convolution operation with $\hat{W}_{i,j}^{(t)}$. Also, for ResNet, we formulate the relationship between the output feature map of a layer and the input feature map of

the subsequent layer as

$$\begin{aligned}
 (i) \quad & V^{(t)} = U^{(t)} \quad \forall t \notin T \\
 (ii) \quad & V^{(t)} = U^{(t)} + V^{(s)} \quad \forall (s, t) \in \mathcal{P} \text{ and } C_t = C_s \\
 (iii) \quad & \forall (s, t) \in \mathcal{P} \text{ and } C_t > C_s, \\
 (iii-a) \quad & V_j^{(t)} = U_j^{(t)} + V_j^{(s)} \quad \forall j \leq C_s \text{ and } V_j^{(t)} = U_j^{(t)} \quad \forall j > C_s \\
 (iii-b) \quad & V^{(t)} = U^{(t)} + \tilde{V}^{(s)} \quad \text{where } \tilde{V}^{(s)} \text{ is } V^{(s)} \text{ after } 1 \times 1 \text{ convolution.}
 \end{aligned} \tag{5}$$

D.2 INACTIVE WEIGHTS

Before we specify inactive weights, we first define two important terms (*trivially zero* and *meaningless*). A feature map is *trivially zero* if the feature map is zero for any input, $V^{(0)}$. A feature map is *meaningless* if the values in the feature map do not have any effect on the final layer output feature map, $U^{(L)}$. Concretely, we state the definitions of trivially zero and meaningless in a cascading fashion.

Trivially zero We define trivially zero in the ascending order of t . Concretely, we define trivially zero in the following order $V^{(0)} \rightarrow U^{(1)} \rightarrow V^{(1)} \rightarrow U^{(2)} \rightarrow \dots \rightarrow V^{(L-1)} \rightarrow U^{(L)}$.

1. $V_j^{(0)}$ in the input feature map is not trivially zero for all j .
2. $U_j^{(t)}$ is trivially zero if and only if $A_{i,j}^{(t)} = 0_{K_t, K_t}$ or $V_i^{(t-1)}$ is trivially zero for all $i \in [C_{t-1}]$ due to Equation (4).
3. In case of $V_j^{(t)}$, we divide the cases according to Equation (5).
 - (i) $V_j^{(t)}$ is trivially zero if and only if $U_j^{(t)}$ is trivially zero.
 - (ii) $V_j^{(t)}$ is trivially zero if and only if $U_j^{(t)}$ is trivially zero and $V_j^{(s)}$ is trivially zero.
 - (iii-a) If $j \leq C_s$, the condition is the same with (ii). Otherwise, the condition is the same with (i).
 - (iii-b) We suppose the output feature map of the 1×1 convolution, $\tilde{V}_j^{(s)}$, is not trivially zero. Therefore, $V_j^{(t)}$ is not trivially zero.

Meaningless We define meaningless in descending order of t . Concretely, we define meaningless in the following order $U^{(L)} \rightarrow V^{(L-1)} \rightarrow U^{(L-1)} \rightarrow V^{(L-2)} \rightarrow \dots \rightarrow U^{(1)} \rightarrow V^{(0)}$.

1. $U_j^{(L)}$ in the final feature map is not meaningless for all j .
2. $V_i^{(t)}$ is meaningless if $A_{i,j}^{(t+1)} = 0_{K_{t+1}, K_{t+1}}$ or $U_j^{(t+1)}$ is meaningless for all $j \in [C_{t+1}]$ due to Equation (4).
3. $U_i^{(t)}$ is meaningless if and only if $V_i^{(t)}$ is meaningless.

We now move on define *active weight* and *inactive weight* in Definition 1 with trivially zero and meaningless.

Definition 1 (Active weight, inactive weight). A weight $W_{i,j,a,b}^{(t)}$ is an **inactive weight** if 1) the weight is pruned ($A_{i,j,a,b}^{(t)} = 0$) or 2) the corresponding input channel feature map ($V_i^{(t-1)}$) is trivially zero or 3) the corresponding output channel feature map ($U_j^{(t)}$) is meaningless. Conversely, a weight $W_{i,j,a,b}^{(t)}$ is an **active weight** if 1) the weight is not pruned ($A_{i,j,a,b}^{(t)} = 1$) and 2) the corresponding input channel feature map ($V_i^{(t-1)}$) is not trivially zero and 3) the corresponding output channel feature map ($U_j^{(t)}$) is not meaningless.

Note that only active weights should account for computation of the resource usage and the sum of the importance of weights. In this next subsection, we show that the inactive weights are provably excluded from the network pruned with our formulation.

D.3 PRUNING CONSISTENCY IN OUR FORMULATION

In our method, discrete variables $u^{(0:L)}$, $v^{(0:L)}$, and $q^{(1:L)}$ satisfy the constraints in Equation (6). We assume at least one of channel activation is set for each layer. Concretely, $\|u^{(t)}\|_1 \geq 1$ and $\|v^{(t)}\|_1 \geq 1 \quad \forall t$.

$$\|u^{(t)}\|_1 \geq 1 \text{ and } \|v^{(t)}\|_1 \geq 1 \quad \forall t \quad (6a)$$

$$u_j^{(t)} \leq \sum_{a,b} q_{j,a,b}^{(t)} \quad \text{and} \quad q_{j,a,b}^{(t)} \leq u_j^{(t)} \quad \forall t, j, a, b \quad (6b)$$

$$\begin{aligned} (i) \quad & v^{(t)} = u^{(t)} \quad \forall t \notin T & (ii) \quad & u^{(t)} \preceq v^{(t)} \preceq u^{(t)} + v^{(s)} \quad \forall (s, t) \in \mathcal{P} \text{ and } C_t = C_s \\ (iii) \quad & \forall (s, t) \in \mathcal{P} \text{ and } C_t > C_s, & & \\ & (iii-a) \quad u_j^{(t)} \leq v_j^{(t)} \leq u_j^{(t)} + v_j^{(s)} \quad \forall j \leq C_s \text{ and } v_j^{(t)} = u_j^{(t)} \quad \forall j > C_s \\ & (iii-b) \quad u^{(t)} \preceq v^{(t)} \end{aligned} \quad (6c)$$

Lemma 1. For $l \in [L]$, if $v_j^{(l-1)} = 1$, then $V_j^{(l-1)}$ is not trivially zero.

Proof. We prove by mathematical induction with respect to l .

1. When $l = 1$, the statement is true since input data is not trivially zero.
2. Suppose the statement is true for $l = 1, \dots, t$.
3. For j such that $v_j^{(t)} = 1$, we can think of three possible cases according to Equation (6c)
 - (i) If $t \notin T$, $u_j^{(t)} = v_j^{(t)} = 1$. First, $\exists i$, $v_i^{(t-1)} = 1$ since $\|v^{(t-1)}\|_1 \geq 1$ from Equation (6a). By the induction hypothesis, $V_i^{(t-1)}$ is not trivially zero. On the other hand, $\exists a, b$ $q_{j,a,b}^{(t)} = 1$ since $u_j^{(t)} \leq \sum_{a,b} q_{j,a,b}^{(t)}$ from Equation (6b). Then, $A_{i,j,a,b}^{(t)} = v_i^{(t-1)} q_{j,a,b}^{(t)} = 1$. By the second condition of trivially zero (2), $U_j^{(t)}$ is not trivially zero since $V_i^{(t-1)}$ is not trivially zero and $A_{i,j,a,b}^{(t)} = 1$. Also, by the third definition of trivially zero (3 - (i)), $V_j^{(t)}$ is not trivially zero since $U_i^{(t)}$ is not trivially zero.
 - (ii) If $\forall (s, t) \in \mathcal{P}$ and $C_t = C_s$, $u_j^{(t)} + v_j^{(s)} \geq v_j^{(t)} = 1$. Then, $u_j^{(t)} = 1$ or $v_j^{(s)} = 1$. If $u_j^{(t)} = 1$, $U_j^{(t)}$ is not trivially zero as in (i). If $v_j^{(s)} = 1$, $V_j^{(s)}$ is not trivially zero by the induction hypothesis. By the definition of trivially zero (3 - (ii)), $V_j^{(t)}$ is not trivially zero since $V_j^{(s)}$ or $U_j^{(t)}$ is not trivially zero.
 - (iii) $\forall (s, t) \in \mathcal{P}$ and $C_t > C_s$,
 - (iii-a) If $j \leq C_s$, the proof is the same with (ii). Otherwise, the proof is the same with (i).
 - (iii-b) By the definition of trivially zero (3 - (iii-b)), $V_j^{(t)}$ is not trivially zero. In every possible cases, $V_j^{(t)}$ is not trivially zero and the statement is true for $l = t + 1$.

By mathematical induction, for $l \in [L]$, if $v_j^{(l-1)} = 1$, then $V_j^{(l-1)}$ is not trivially zero. \square

Lemma 2. For $l \in [L]$, if $u_i^{(l)} = 1$, then $U_i^{(l)}$ is not meaningless.

Proof. We prove by mathematical induction with respect to l .

1. When $l = L$, the statement is true since $U^{(L)}$ is not meaningless.

2. Suppose the statement is true for $l = t+1, \dots, L$.
3. For i such that $u_i^{(t)} = 1, v_i^{(t)} = 1$ since $v_i^{(t)} \geq u_i^{(t)}$. Then, $\exists j \quad u_j^{(t+1)} = 1$ since $\|u^{(t+1)}\|_1 \geq 1$ from Equation (6a). By the induction hypothesis, $U_j^{(t+1)}$ is not meaningless. On the other hand, $\exists a, b \quad q_{j,a,b}^{(t+1)} = 1$ since $u_j^{(t+1)} \leq \sum_{a,b} q_{j,a,b}^{(t+1)}$ from Equation (6b). Then, $A_{i,j,a,b}^{(t+1)} = v_i^{(t)} q_{j,a,b}^{(t+1)} = 1$. By the definition of meaningless (2), $V_i^{(t)}$ is not meaningless. By the definition of meaningless (3), $U_i^{(t)}$ is not meaningless. The statement is true for $l = t$.

By mathematical induction, for $l \in [L]$, if $u_i^{(l)} = 1$, then $U_i^{(l)}$ is not meaningless. \square

Proposition 2. *Optimizing over the input and output channel activation variables $u^{(0:L)}, v^{(0:L)}$ and shape column activation variables $q^{(1:L)}$ under the constraints in Equation (6) prevents the existence of any inactive weights in the pruned network guaranteeing exact computation of 1) resource usage and 2) the sum of the importance of active weights in the pruned network.*

Proof. Weight $W_{i,j,a,b}^{(l)}$ is not pruned if $A_{i,j,a,b}^{(l)} = 1$. If $A_{i,j,a,b}^{(l)} = 1$, then $u_j^{(l)} = 1$ and $v_i^{(l-1)} = 1$. Then, by Lemma 1 and Lemma 2, $V_i^{(l-1)}$ is not trivially zero and $U_j^{(l)}$ is not meaningless. By Definition 1, the weight $W_{i,j,a,b}^{(l)}$ is active. All the remaining weights in the network pruned with our method are active, which guarantees the exact specification of resource usage and sum of the importance of active weights in Equation (2). \square

Proposition 1. *Optimizing over the input and output channel activation variables $r^{(0:L)}$ and shape column activation variables $q^{(1:L)}$ under the constraints in Equation (8) prevents the existence of any inactive weights in the pruned network guaranteeing exact computation of 1) resource usage and 2) the sum of the importance of active weights in the pruned network.*

Proof. Proposition 1 is the special case of Proposition 2 when $u^{(t)} = v^{(t)}$ ($:= r^{(t)}$), $q_{j,a,b}^{(t)} = u_j^{(t)}$, and $\mathcal{P} = \emptyset \quad \forall t, j, a, b$. \square

E QCQP FORMULATION

$$\begin{aligned}
 & \underset{r^{(0:L)}}{\text{maximize}} \quad \sum_{l=1}^L \langle I^{(l)}, A^{(l)} \rangle \\
 & \text{subject to} \quad \sum_{l=0}^L a_l \|r^{(l)}\|_1 + \sum_{l=1}^L b_l \|A^{(l)}\|_1 \leq M \\
 & \quad A^{(l)} = r^{(l-1)} r^{(l)\top} \otimes J_{K_l} \quad \forall l \in [L] \\
 & \quad r^{(l)} \in \{0, 1\}^{C_l}
 \end{aligned} \tag{7}$$

Proposition 3. *Equation (7) is a QCQP problem.*

Proof. We define the *importance of 2-D filter*, which is the sum of the importance of weights in the filter as $F^{(l)} \in \mathbb{R}_+^{C_{l-1} \times C_l} \quad \forall l$. Concretely, $F_{i,j}^{(l)} = \sum_{a,b} I_{i,j,a,b}^{(l)} \quad \forall i, j$. We wish to express the objective function and constraints in Equation (7) with respect to $r^{(0:L)}$. Note that $\|A^{(l)}\|_1 = K_l^2 \|r^{(l-1)}\|_1 \|r^{(l)}\|_1$ and $\langle I^{(l)}, A^{(l)} \rangle = r^{(l-1)\top} F^{(l)} r^{(l)}$. To express Equation (7) in a standard QCQP form, we denote $[r^{(0)}, r^{(1)}, \dots, r^{(L)}]$ as $\mathbf{r} \in \{0, 1\}^N$ where $N = \sum_{l=0}^L C_l$. Standard QCQP

form of Equation (7) is

$$\begin{aligned} & \underset{\mathbf{r} \in \{0,1\}^N}{\text{maximize}} \quad \frac{1}{2} \mathbf{r}^\top P_0 \mathbf{r} \\ & \text{subject to} \\ & \quad \frac{1}{2} \mathbf{r}^\top P_1 \mathbf{r} + q_1^\top \mathbf{r} \leq M, \end{aligned}$$

where

$$\begin{aligned} P_0 &= \begin{pmatrix} 0 & F^{(1)} & 0 & \cdots & 0 & 0 \\ F^{(1)\top} & 0 & F^{(2)} & \cdots & 0 & 0 \\ 0 & F^{(2)\top} & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 0 & F^{(L)} \\ 0 & 0 & 0 & \cdots & F^{(L)\top} & 0 \end{pmatrix}, \\ P_1 &= \begin{pmatrix} 0 & b_1 K_1^2 J_{C_1} & 0 & \cdots & 0 & 0 \\ b_1 K_1^2 J_{C_1} & 0 & b_2 K_2^2 J_{C_2} & \cdots & 0 & 0 \\ 0 & b_2 K_2^2 J_{C_2} & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 0 & b_L K_L^2 J_{C_L} \\ 0 & 0 & 0 & \cdots & b_L K_L^2 J_{C_L} & 0 \end{pmatrix}, \\ & \text{and } q_1 = [\underbrace{a_0, \dots, a_0}_{C_0}, \underbrace{a_1, \dots, a_1}_{C_1}, \dots, \underbrace{a_L, \dots, a_L}_{C_L}]. \end{aligned}$$

□

$$\begin{aligned} & \underset{r^{(0:L)}, q^{(1:L)}}{\text{maximize}} \quad \sum_{l=1}^L \langle I^{(l)}, A^{(l)} \rangle \\ & \text{subject to} \quad \sum_{l=0}^L a_l \|r^{(l)}\|_1 + \sum_{l=1}^L b_l \|A^{(l)}\|_1 \leq M. \\ & \quad r_j^{(l)} \leq \sum_{a,b} q_{j,a,b}^{(l)} \quad \text{and} \quad q_{j,a,b}^{(l)} \leq r_j^{(l)} \quad \forall l, j, a, b \\ & \quad A^{(l)} = r^{(l-1)} \otimes q^{(l)} \quad \forall l \\ & \quad r^{(l)} \in \{0,1\}^{C_l} \text{ and } q^{(l)} \in \{0,1\}^{C_l \times K_l \times K_l} \quad \forall l \in [L] \end{aligned} \tag{8}$$

Proposition 4. Equation (8) is a QCQP problem.

Proof. To prove Equation (8) is a QCQP problem, we show that objective function $\sum_{l=1}^L \langle I^{(l)}, A^{(l)} \rangle$, and the constraint $\sum_{l=0}^L a_l \|r^{(l)}\|_1 + \sum_{l=1}^L b_l \|A^{(l)}\|_1$, are sum of quadratic and linear terms of $r^{(0:L)}$ and $q^{(1:L)}$. Note that

$$\begin{aligned} \langle I^{(l)}, A^{(l)} \rangle &= \sum_{i=1}^{C_{l-1}} \sum_{j=1}^{C_l} \sum_{a=1}^{K_l} \sum_{b=1}^{K_l} I_{i,j,a,b}^{(l)} r_i^{(l-1)} q_{j,a,b}^{(l)} \\ \|r^{(l)}\|_1 &= \sum_{i=1}^{C_l} r_i^{(l)} \\ \|A^{(l)}\|_1 &= \sum_{i=1}^{C_{l-1}} \sum_{j=1}^{C_l} \sum_{a=1}^{K_l} \sum_{b=1}^{K_l} r_i^{(l-1)} q_{j,a,b}^{(l)}. \end{aligned}$$

Clearly, the objective function and all the constraints in Equation (8) can be expressed as the sum of quadratic and linear terms of $r^{(0:L)}$ and $q^{(1:L)}$. Therefore, Equation (8) is a QCQP problem with discrete variables, $r^{(0:L)}$ and $q^{(1:L)}$. \square

Proposition 5. Equation (2) is a QCQP problem.

Proof. To prove Equation (2) is a QCQP problem, we show that $\sum_{t=1}^L \langle I^{(t)}, A^{(t)} \rangle$ and $\sum_{t=0}^L a_t \|u^{(t)}\|_1 + \sum_{t \in T} a_t \|v^{(t)}\|_1 + \sum_{t=1}^L b_t \|A^{(t)}\|_1$ are sum of quadratic and linear terms of $u^{(0:L)}$, $v^{(0:L)}$ and $q^{(1:L)}$. Note that

$$\begin{aligned} \langle I^{(t)}, A^{(t)} \rangle &= \sum_{i=1}^{C_{t-1}} \sum_{j=1}^{C_t} \sum_{a=1}^{K_t} \sum_{b=1}^{K_t} I_{i,j,a,b}^{(t)} v_i^{(t-1)} q_{j,a,b}^{(t)} \\ \|u^{(t)}\|_1 &= \sum_{i=1}^{C_t} u_i^{(t)} \\ \|v^{(t)}\|_1 &= \sum_{i=1}^{C_t} v_i^{(t)} \\ \|A^{(t)}\|_1 &= \sum_{i=1}^{C_{t-1}} \sum_{j=1}^{C_t} \sum_{a=1}^{K_t} \sum_{b=1}^{K_t} v_i^{(t-1)} q_{j,a,b}^{(t)}. \end{aligned}$$

Clearly, the objective function and all the constraints in Equation (2) can be expressed by the sum of quadratic and linear terms of $u^{(0:L)}$, $v^{(0:L)}$ and $q^{(1:L)}$. Therefore, Equation (2) is a QCQP problem with discrete variables, $u^{(0:L)}$, $v^{(0:L)}$, and $q^{(1:L)}$. \square

F SPATIAL PATTERN APPEARING IN PRUNED NETWORK

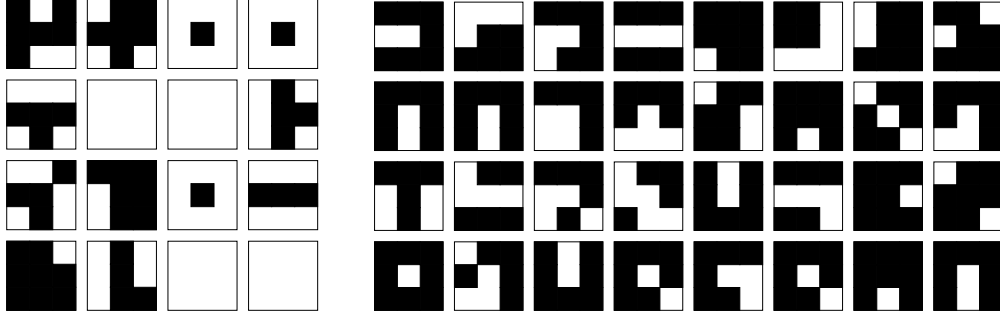


Figure 2: Visualization of shape column activations in the first convolution layer with $q^{(1)} \in \{0, 1\}^{16 \times 3 \times 3}$ (left) and shape column activations in the eighth convolution layer with $q^{(8)} \in \{0, 1\}^{32 \times 3 \times 3}$ (right) in ResNet-20 after pruned by ‘ours (c+s)’. Black area indicates that the shape column activation is set.

‘ours-cs’ discovers diverse spatial patterns in convolution weights, as illustrated in Figure 2.

G EXPERIMENTS ON MOBILENETV2

We additionally apply our pruning method on MobileNetV2 (Sandler et al., 2018). We start from the pretrained network with accuracy 71.88 provided by Pytorch. Then, we finetune the pruned network for 150 epochs using SGD optimizer with weight decay 0.00004, momentum 0.9, and batch size 256. For the training schedule, we apply a learning rate warm-up for the initial five epochs, which steps up from 0 to 0.05. Then, we use the cosine learning rate decay for the remaining epochs.

Network	Method	Top1 Pruned Acc \uparrow	Top1 Acc drop \downarrow	FLOPs(%) \downarrow
MobileNetV2	NetAdapt (Yang et al., 2018)	70.9	0.9	70
	AMC (He et al., 2018)	70.8	1.0	70
	MetaPruning (Liu et al., 2019)	71.2	0.6	69
	ours-c	70.8	1.0	67
	ours-cs	70.2	1.6	67
	ours-c (tuned)	71.0	0.8	67
	ours-cs (tuned)	70.9	0.9	67

Table 1: Top1 pruned accuracy and accuracy drop from the baseline network at given FLOPs on MobileNetV2 architecture at ImageNet

We show our experiment results with three recent pruning baselines NetAdapt (Yang et al., 2018), AMC (He et al., 2018), and MetaPruning (Liu et al., 2019) in Table 1. ‘(tuned)’ indicates that the normalizing factor, γ_i , is tuned with grid search. A fixed value is used otherwise. Our method shows performance competitive to the other baselines, NetAdapt, AMC and MetaPruning. However, we note that our method is much more efficient than NetAdapt, AMC, and MetaPruning since NetAdapt requires repetitive finetuning steps for the proposed networks, AMC requires repetitive trial and error steps to train DDPG (Lillicrap et al., 2016) agent, and MetaPruning trains PruningNet of which network size is at least 30 times bigger than that of original model.

H ABLATION STUDY

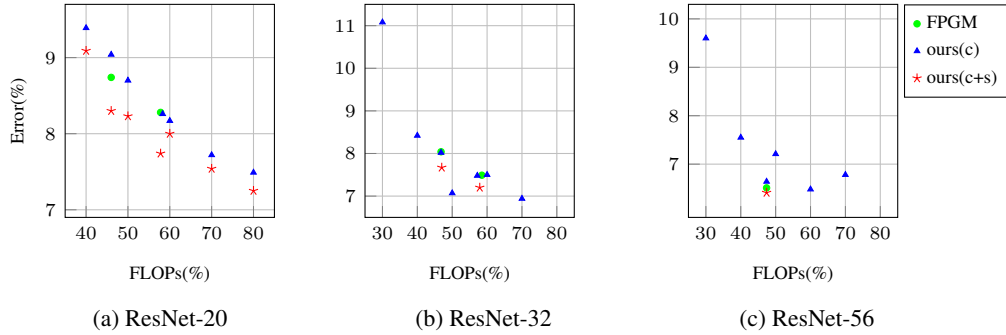


Figure 3: The plots of classification error (%) versus FLOPs (%) on various ResNet architectures.

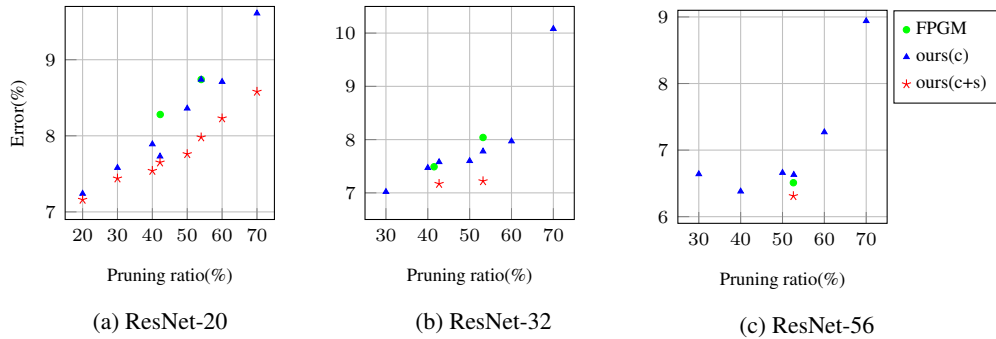


Figure 4: The plots of classification error (%) versus pruning ratio (%) on various ResNet architectures.

H.1 EXPERIMENTS ON VARIOUS FLOPS CONSTRAINT

In Figure 3, we prune and finetune the ResNet architectures under various FLOPs constraints with ‘ours-c’, ‘ours-cs’, and FPGM. ‘ours-cs’ outperforms ‘ours-c’ and FPGM under almost all FLOPs constraints.

H.2 EXPERIMENTS ON VARIOUS NETWORK SIZE CONSTRAINT

In Figure 4, we prune and finetune the ResNet architectures under various network size constraints with ‘ours-c’, ‘ours-cs’, and FPGM. ‘ours-cs’ outperforms ‘ours-c’ and FPGM on almost all network size constraints.

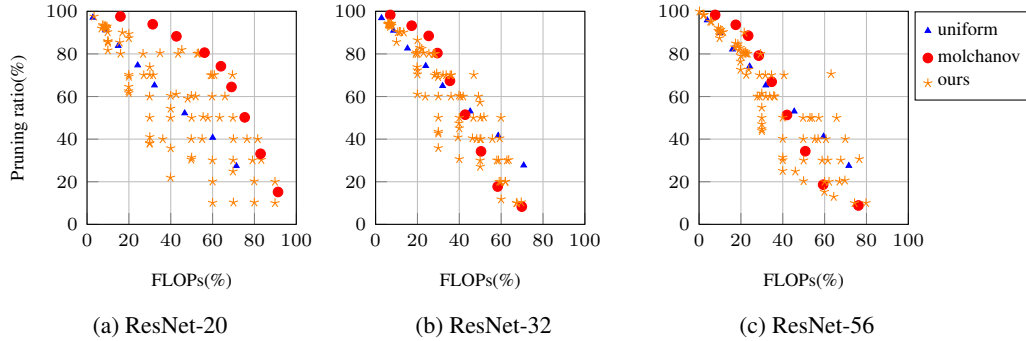


Figure 5: The possible pairs of resource usage (FLOPs and pruning ratio) from three different pruning criteria : (1) ‘uniform’ which greedily prunes channels layer-wise, (2) ‘molchanov’ which greedily prunes the channels from all layers, and (3) ‘ours’ which prunes via QCQP.

H.3 ABLATION STUDY ON THE POSSIBLE PAIRS OF RESOURCE USAGE (FLOPS AND PRUNING RATIO)

In the real-world, the resource constraint for network pruning may vary significantly in terms of how much each resource is available. In some cases, we may allow high FLOPs but strictly limit the network size, while in other cases, low FLOPs are much more important. However, when we prune the channels greedily, possible pairs of resource usages are limited and nonadjustable. In contrast, our method can target any resource budget pairs. Figure 5 shows the possible pairs (FLOPS and pruning ratio) from three different pruning methods: ‘uniform’, which greedily prunes channels layer-wise, ‘molchanov’, which greedily prunes the channels from all layers, and ‘ours’, which prunes via QCQP. ‘ours’ results in diverse pairs of target resources which cover the pairs of ‘uniform’ and ‘molchanov’.

I EXPERIMENTS ON FCN-32S FOR SEGMENTATION

We apply our pruning method on FCN-32s (Long et al., 2015) for segmentation on PASCAL Visual Object Classes Challenge 2011 dataset. Then, we evaluate the segmentation performance with a widely-used measure, mean Intersection over Union (mIoU).

I.1 IMPLEMENTATION DETAILS

We use SGD optimizer with weight decay 0.0005, momentum 0.99, and batch size 1. For the original network to be pruned, we train FCN-32s for 12 epochs with a constant learning rate 10^{-11} . Then, we prune the original network of which mIoU 62.58 (%) and finetune the pruned network for 25 epochs with a constant learning rate 6×10^{-11} .

I.2 EXPERIMENT RESULTS

We show our experiment results in Figure 6. The pruned network reduces the FLOPs by 27% with 0.15 (%) mIoU drop for ‘ours-c’ and 0.09 (%) mIoU drop for ‘ours-cs’.

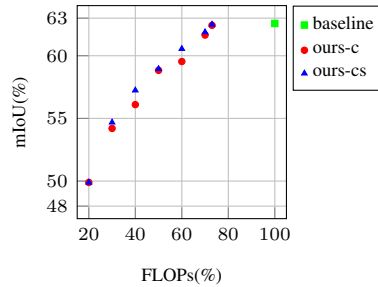


Figure 6: The plot of mIoU (%) versus FLOPs (%) on FCN-32s.

REFERENCES

- Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.
- Yang He, Ping Liu, Ziwei Wang, Zhilan Hu, and Yi Yang. Filter pruning via geometric median for deep convolutional neural networks acceleration. In *CVPR*, 2019.
- Yihui He, Ji Lin, Zhijian Liu, Hanrui Wang, Li-Jia Li, and Song Han. Amc: Automl for model compression and acceleration on mobile devices. In *ECCV*, 2018.
- Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *CVPR*, 2017.
- Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. In *Tech Report*, 2009.
- Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. In *ICLR*, 2016.
- Z. Liu, H. Mu, X. Zhang, Z. Guo, X. Yang, K. Cheng, and J. Sun. Metapruning: Meta learning for automatic neural network channel pruning. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.
- J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.
- Pavlo Molchanov, Stephen Tyree, Tero Karras, Timo Aila, and Jan Kautz. Pruning convolutional neural networks for resource efficient inference. In *ICLR*, 2017.
- Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017.
- Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. In *IJCV*, 2015.
- Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *CVPR*, 2018.
- Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.
- Tien-Ju Yang, Andrew Howard, Bo Chen, Xiao Zhang, Alec Go, Mark Sandler, Vivienne Sze, and Hartwig Adam. Netadapt: Platform-aware neural network adaptation for mobile applications. In *ECCV*, 2018.