Appendix

A Algorithm Procedure

A full workflow of channel number search with the proposed transitionary APS is shown in Algorithm 1. The overall procedure consists of two stages. In the first stage, we fix the controller $\pi(\theta)$ and initialize Φ of affine parameter sharing with maximal value, so that meta weights $\boldsymbol{\mathcal{W}}$ can be efficiently optimized. During this stage architectures are uniformly sampled from the search space $\mathcal{C}$ and are thus equally updated. In the second stage, we gradually anneal Φ via Equation 5 so as to transit the sharing level Φ. We alternatively update meta weights $\boldsymbol{\mathcal{W}}$ and controller parameter θ based on architectures sampled from the controller.

Algorithm 1 RL-based CNS algorithm with Transitionary APS

Input:
Training data $\mathcal{D}_{tr}$, validation data $\mathcal{D}_{val}$;
Base network with meta weights $\boldsymbol{\mathcal{W}}$, transformation matrices $\boldsymbol{\mathcal{P}}$, $\boldsymbol{\mathcal{Q}}$
RL controller $\pi(\theta)$ and channel search space $\mathcal{C}$.
Output:
Optimal channel configurations.

// Stage 1: fast optimization of meta-weights $\boldsymbol{\mathcal{W}}$
Initialize $\boldsymbol{\mathcal{P}}$, $\boldsymbol{\mathcal{Q}}$ with maximal Φ;
for $t = 1, ..., T_1$ **do**
  Sample the architecture a uniformly from $\mathcal{C}$;
  Update $\boldsymbol{\mathcal{W}}$ via gradient descent with a on $\mathcal{D}_{tr}$;
end for
// Stage 2: transitionary affine parameter sharing
for $t = 1, ...T_2$ **do**
  Sample the architecture from controller $a \sim \pi(\theta)$;
  Update $\boldsymbol{\mathcal{W}}$ via gradient descent with a on $\mathcal{D}_{tr}$;
  Update $\theta^{t+1} = \theta^t + \eta \mathbb{E}_a \left[\nabla_\theta \log p(a) \mathcal{R}\right]$ on $\mathcal{D}_{val}$;
  Anneal Φ by updating $\boldsymbol{\mathcal{P}}$, $\boldsymbol{\mathcal{Q}}$ in Equation (5);
end for

B Proof for Theorem 3.1

Theorem. *For $\forall i \leq \tilde{i}$ and $\forall o \leq \tilde{o}$, the overall level Φ of APS is maximized if $\mathcal{R}\big(\mathbf{Q}^i\big) \subseteq \mathcal{R}\big(\mathbf{Q}^{\tilde{i}}\big)$ and $\mathcal{R}\big(\mathbf{P}^o\big) \subseteq \mathcal{R}\big(\mathbf{P}^{\tilde{o}}\big)$. Φ is minimized if $\mathcal{R}\big(\mathbf{Q}^i\big) \subseteq \mathcal{R}^\perp\big(\mathbf{Q}^{\tilde{i}}\big)$ and $\mathcal{R}\big(\mathbf{P}^o\big) \subseteq \mathcal{R}^\perp\big(\mathbf{P}^{\tilde{o}}\big)$.*

To prove Theorem 3.1, we first show the case of two candidate decision (i, o) and $(\tilde{i}, \tilde{o})$, after which we can combine the pairwise optimal conditions together to yield Theorem 3.1. Without loss of generality, suppose $c_{\tilde{i}} > c_i$ and $c_{\tilde{o}} > c_o$, we have the following lemma:

Lemma 1. *Given candidate decisions (i, o) and $(\tilde{i}, \tilde{o})$, $\phi(i, o; \tilde{i}, \tilde{o})$ is maximized if $\mathcal{R}(\mathbf{Q}^i) \subseteq \mathcal{R}(\mathbf{Q}^{\tilde{i}})$ and $\mathcal{R}(\mathbf{P}^o) \subseteq R(\mathbf{P}^{\tilde{o}})$; $\phi(i, o; \tilde{i}, \tilde{o})$ is minimized if $\mathcal{R}(\mathbf{Q}^i) \subseteq \mathcal{R}^\perp(\mathbf{Q}^{\tilde{i}})$ or $\mathcal{R}(\mathbf{P}^o) \subseteq R^\perp(\mathbf{P}^{\tilde{o}})$.*

Proof. As $\phi(i, o; \tilde{i}, \tilde{o})$ is defined as the squared sum of pairwise correlation between $\mathbf{W}^{i,o}$ and $\mathbf{W}^{\tilde{i},\tilde{o}}$, we can explicitly write it out as:

$$\begin{aligned}\phi(i, o; \tilde{i}, \tilde{o}) &= \sum_{x=1}^{c_i} \sum_{y=1}^{c_o} \sum_{\tilde{x}=1}^{c_{\tilde{i}}} \sum_{\tilde{y}=1}^{c_{\tilde{o}}} \left[\mathrm{Cov}_{x,y,\tilde{x},\tilde{y}}(\mathbf{W}^{i,o}, \mathbf{W}^{\tilde{i},\tilde{o}})\right]^2 \\ &= \sum_{x=1}^{c_i} \sum_{y=1}^{c_o} \sum_{\tilde{x}=1}^{c_{\tilde{i}}} \sum_{\tilde{y}=1}^{c_{\tilde{o}}} \left[\mathbb{E}\big(W^{i,o}_{x,y} W^{\tilde{i},\tilde{o}}_{\tilde{x},\tilde{y}}\big) - \mathbb{E}\big(W^{i,o}_{x,y}\big)\mathbb{E}\big(W^{\tilde{i},\tilde{o}}_{\tilde{x},\tilde{y}}\big)\right]^2. \end{aligned} \tag{6}$$

Note that the second term can be removed since $\mathbb{E}\big(W^{i,o}_{x,y}\big) = \mathbb{E}\big((\mathbf{q}^i_x)^\top \boldsymbol{\mathcal{W}} \mathbf{p}^o_y\big) = (\mathbf{q}^i_x)^\top \mathbb{E}(\boldsymbol{\mathcal{W}}) \mathbf{p}^o_y = 0$ and similarly $\mathbb{E}\big(W^{\tilde{i},\tilde{o}}_{\tilde{x},\tilde{y}}\big) = 0$. Therefore $\phi(i,o;\tilde{i},\tilde{o})$ can be simplified as

$$\begin{aligned}
\phi(i,o;\tilde{i},\tilde{o}) &= \sum_{x=1}^{c_i}\sum_{y=1}^{c_o}\sum_{\tilde{x}=1}^{c_{\tilde{i}}}\sum_{\tilde{y}=1}^{c_{\tilde{o}}} \mathbb{E}^2\Big[W^{i,o}_{x,y}\cdot W^{\tilde{i},\tilde{o}}_{\tilde{x},\tilde{y}}\Big] \\
&= \sum_{x,y}\sum_{\tilde{x},\tilde{y}} \mathbb{E}^2\Big[(\mathbf{q}^i_x)^\top \boldsymbol{\mathcal{W}} \mathbf{p}^o_y \cdot (\mathbf{q}^{\tilde{i}}_{\tilde{x}})^\top \boldsymbol{\mathcal{W}} \mathbf{p}^{\tilde{o}}_{\tilde{y}}\Big] \\
&= \sum_{x,y}\sum_{\tilde{x},\tilde{y}} \mathbb{E}^2\Big[(\mathbf{q}^i_x)^\top \boldsymbol{\mathcal{W}} \mathbf{p}^o_y \cdot (\mathbf{p}^{\tilde{o}}_{\tilde{y}})^\top \boldsymbol{\mathcal{W}}^\top \mathbf{q}^{\tilde{i}}_{\tilde{x}}\Big] \\
&= \sum_{x,y}\sum_{\tilde{x},\tilde{y}} \Big((\mathbf{q}^i_x)^\top \mathbb{E}\Big[\boldsymbol{\mathcal{W}} \mathbf{p}^o_y \ (\mathbf{p}^{\tilde{o}}_{\tilde{y}})^\top \boldsymbol{\mathcal{W}}^\top\Big] \mathbf{q}^{\tilde{i}}_{\tilde{x}}\Big)^2 .
\end{aligned} \tag{7}$$

Expanding the expectation $\mathbb{E}[\boldsymbol{\mathcal{W}} \mathbf{p}^o_y (\mathbf{p}^{\tilde{o}}_{\tilde{y}})^\top \boldsymbol{\mathcal{W}}^\top]$ elementwisely, we have

$$\mathbb{E}\begin{bmatrix} \boldsymbol{w}_1 \mathbf{p}^o_y (\mathbf{p}^{\tilde{o}}_{\tilde{y}})^\top \boldsymbol{w}_1^\top & \cdots & \boldsymbol{w}_1 \mathbf{p}^o_y (\mathbf{p}^{\tilde{o}}_{\tilde{y}})^\top \boldsymbol{w}_c^\top \\ \vdots & \ddots & \vdots \\ \boldsymbol{w}_c \mathbf{p}^o_y (\mathbf{p}^{\tilde{o}}_{\tilde{y}})^\top \boldsymbol{w}_1^\top & \cdots & \boldsymbol{w}_c \mathbf{p}^o_y (\mathbf{p}^{\tilde{o}}_{\tilde{y}})^\top \boldsymbol{w}_c^\top \end{bmatrix} = \begin{bmatrix} (\mathbf{p}^o_y)^\top \mathbf{p}^{\tilde{o}}_{\tilde{y}} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & (\mathbf{p}^o_y)^\top \mathbf{p}^{\tilde{o}}_{\tilde{y}} \end{bmatrix} \tag{8}$$

where we have used the fact that $\mathbb{E}\Big[\boldsymbol{w}_m \mathbf{p}^o_y (\mathbf{p}^{\tilde{o}}_{\tilde{y}})^\top \boldsymbol{w}_n^\top\Big] = (\mathbf{p}^o_y)^\top \mathbb{E}\Big[\boldsymbol{w}_m^\top \boldsymbol{w}_n\Big] \mathbf{p}^{\tilde{o}}_{\tilde{y}} = (\mathbf{p}^o_y)^\top \mathbf{p}^{\tilde{o}}_{\tilde{y}}$ if $m = n$, and 0 otherwise. With Equation 8, Equation 7 can be simplified to

$$\begin{aligned}
\phi(i,o;\tilde{i},\tilde{o}) &= \sum_{x,\tilde{x}}\sum_{y,\tilde{y}} \Big[(\mathbf{q}^i_x)^\top \mathbf{q}^{\tilde{i}}_{\tilde{x}} \cdot (\mathbf{p}^o_y)^\top \mathbf{p}^{\tilde{o}}_{\tilde{y}}\Big]^2 \\
&= \sum_{x,\tilde{x}}\sum_{y,\tilde{y}} \Big[\sum_{m=1}^{c} q^i_{m,x} q^{\tilde{i}}_{m,\tilde{x}} \cdot \sum_{n=1}^{c} p^o_{n,y} p^{\tilde{o}}_{n,\tilde{y}}\Big]^2 .
\end{aligned} \tag{9}$$

Without loss of generality, we take $\mathbf{Q}^{\tilde{i}}$ and $\mathbf{P}^{\tilde{o}}$ as standard orthogonal basis, i.e. $q^{\tilde{i}}_{\tilde{x},\tilde{x}} = 1$ and $q^{\tilde{i}}_{m,x} = 0$ for $m \neq \tilde{x}$ and $\tilde{x} \in \{1,...,c_{\tilde{i}}\}$, and similarly for $\mathbf{P}^{\tilde{o}}$. Thus Equation 9 can be further reduced to

$$\sum_{x,\tilde{x}}\sum_{y,\tilde{y}} \Big[q^i_{\tilde{x},x} \cdot p^o_{\tilde{y},y}\Big]^2 = \sum_{x=1}^{c_i}\sum_{y=1}^{c_o} \Big(\sum_{\tilde{x}=1}^{c_{\tilde{i}}} (q^i_{\tilde{x},x})^2\Big) \cdot \Big(\sum_{\tilde{y}=1}^{c_{\tilde{o}}} (p^o_{\tilde{y},y})^2\Big) \leq \sum_x \sum_y 1 = c_i c_o . \tag{10}$$

The equality holds if $q^i_{m,x} = 0$ for $m > c_{\tilde{i}}$ and $p^o_{n,y} = 0$ for $n > c_{\tilde{o}}$. Therefore the maximum is attained when orthogonal basis of $\mathbf{Q}^i$ and $\mathbf{P}^o$ lie in the span of those in $\mathbf{Q}^{\tilde{i}}$ and $\mathbf{P}^{\tilde{o}}$ respectively, i.e, $\mathcal{R}(\mathbf{Q}^i) \subseteq \mathcal{R}(\mathbf{Q}^{\tilde{i}})$ and $\mathcal{R}(\mathbf{P}^o) \subseteq R(\mathbf{P}^{\tilde{o}})$.

Conversely, minimum for Equation 9 is attained if $q^i_{m,x} = 0$ for $m \leq c_{\tilde{i}}$ or $p^o_{n,y} = 0$ for $n \leq c_{\tilde{o}}$, which is equivalent to $\mathcal{R}(\mathbf{Q}^i) \subseteq \mathcal{R}^\perp(\mathbf{Q}^{\tilde{i}})$ or $\mathcal{R}(\mathbf{P}^o) \subseteq \mathcal{R}^\perp(\mathbf{P}^{\tilde{o}})$. □

Finally, to prove Theorem 3.1, we only need to extend Lemma 1 to the case of multiple candidate decisions. The maximum and minimum of $\Phi = \sum_{i,\tilde{i}} \sum_{o,\tilde{o}} \phi(i,o;\tilde{i},\tilde{o})$ can be achieved when each $\phi(i,o;\tilde{i},\tilde{o})$ attains its maximum and minimum respectively. This corresponds to the intersection of optimal conditions in Lemma 1, which is exactly Theorem 3.1.

C Implementation Details and Hyper-parameters

Design of RL controller We follow ENAS [25] to take the RL-based algorithm to illustrate channel number search, and the overall formulation is given in Equation 1. Here we cover more details in the design. For the RL controller, we take a two-layer LSTM with 100 hidden units, and channel decisions are generated auto-regressively. The RL states contain the previous layer width decision, the one-hot layer index encoding, and available FLOPs left. Naively, one could simply set the accuracy

as the reward for controller training. However, this may be not reasonable under budget-constrained search. Instead we follow MNasNet [30] and design the reward $\mathcal{R}$ as:

$$\mathcal{R} = Acc(a) \times [\frac{\mathcal{B}(a)}{B}]^{\gamma}, \quad \text{where } \gamma = \begin{cases} \alpha & \text{if } \mathcal{B}(a) < B \\ \beta & \text{otherwise} \end{cases}, \tag{11}$$

where $a \sim \pi(\theta)$ is the layerwise channel decision, $Acc(\cdot)$ is the accuracy function, and α, β are coefficients of FLOPs penalty. Such design is shown to approximate the pareto-optimal solutions [30]. We adopt policy gradient to maximize the reward function. To prevent the RL controller $\pi(\theta)$ from getting stuck in local-minimal, we follow [25] to add an entropy regularization.

Calculating the cosine similarity of gradients Given two different candidates (i, o) and $(\tilde{i}, \tilde{o})$ of one convolutional layer, we describe the steps to calculate the alignment of gradients ($\cos(\mathbf{g}, \tilde{\mathbf{g}})$) on the meta-weight. We change the channel configuration in one layer and fix all the other layers. Specifically, for $a = (a_1, a_2, ..., a_L)$, we vary $a_l \in \mathcal{A}$ and fix all the rest a_j for $j \neq l$. Varying $a_l \in \mathcal{A}$ gives A different gradients on $\boldsymbol{\mathcal{W}}$. We thus compute and average the pairwise cosine similarities of these gradients at different values of Φ, each of which involves $A(A-1)/2$ combinations. Then these averaged cosine values are summed over all layers to produce Figure 3(a). During the transition of Φ, we repeat the above procedure at each training epoch to plot the cosine values in Figure 4.

Calculating the norm of coupled gradients To collect the norm of coupled gradients, we maintain an accumulator for each candidate. During the searching process at time step t, each gradient term $\mathbf{Q}^t\big(\nabla_{\mathbf{W}}\mathcal{L}(\mathbf{W}^t)\big)(\mathbf{P}^t)^\top$ from a_l is added to its own accumulator. The corresponding coupled gradients can be computed by summing over all candidate accumulators except for a_l and then multiply $\mathbf{Q}^t$ and $\mathbf{P}^t$. Similar to the above subsection, the norm of coupled gradients across different layers are then averaged. The accumulators are reset every epoch to fully show the level of coupling change with respect to the parameter sharing transition. We follow such procedure at different values of Φ to produce Figure 2(b). To plot the variation of coupled gradients in Figure 4, we repeat this process at every training epoch, and clean accumulators before the start of next epoch. We also empirically observe that the step-wise change of learning rates affects the observation of coupling. Thus we set constant learning rate as 1e-2 during the collection of coupled gradients.

Efficient optimization of Φ The form of objective function Φ w.r.t to $\boldsymbol{\mathcal{P}}, \boldsymbol{\mathcal{Q}}$ can be readily obtained by vectorizing the element-wise summation in Equation 9. Direct optimization of the sharing level Φ could be computationally expensive. Instead, we alternatively update $\boldsymbol{\mathcal{P}}, \boldsymbol{\mathcal{Q}}$ to minimize Φ with lower frequencies. For instance, we fix $\boldsymbol{\mathcal{P}}$ to optimize $\boldsymbol{\mathcal{Q}}$ for ten iterations, where the product terms of $\boldsymbol{\mathcal{P}}$ are computed in advance and can be reused them for multiple turns. In this way the computational burden can be effectively reduced. Furthermore, one can update $\boldsymbol{\mathcal{P}}, \boldsymbol{\mathcal{Q}}$ separately for each layer, which is an equivalent implementation but with less memory overhead.

Hyper-parameter settings Finally, the detailed hyper-parameters for the RL based CNS algorithms on both CIFAR-10 and ImageNet datasets are shown in Table 3. Given the searched architecture, we follow default settings of the base models to train stand-alone models from scratch.

D Visualization of Channel Configurations

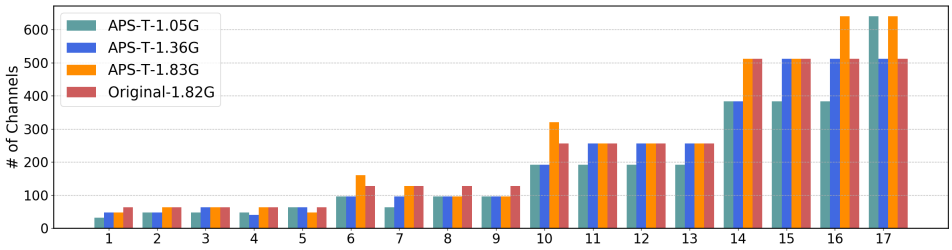
We visualize the searched channel configurations under different FLOPs constraint of ResNet-18 and MobileNet-v2 in Figure 10. Note that for MobileNet-v2, we omit the output channels of the depth-wise convolution in each block since it equals to the input channels. It can be observed that for both ResNet-18 and MobileNet-v2 under various FLOPs constraint, APS-T tends to find configurations with less channels in front layers and more channels in deep layers.

E More Results on Unconstrained Search

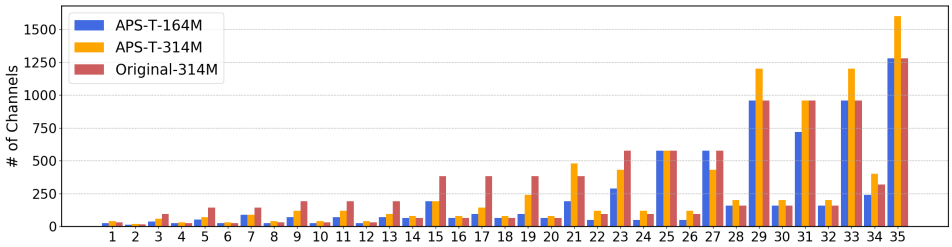
We provide more results on the architecture discrimination without FLOPs constraint in Figure 11. The hyper-parameters are consistent to those used Figure 5. It can be found that both APS-O and APS-T are relatively robust in the searched results. However, APS-I ends with large architecture discrimination but sometimes converges to sub-optimal configurations. We conjecture it is possibly due to early convergence of the controller as a result of insufficient training of meta weights $\boldsymbol{\mathcal{W}}$.

Table 3: Hyper-parameters for different base models on CIFAR-10 and ImageNet.

Hyper-parameters	CIFAR-10		ImageNet	
	ResNet-20	ResNet-56	ResNet-18	MobileNet-v2
Channel Number $\mathcal{C}$	[4,8,16,32,64]	[4,8,16,32,64]	[32,48,64,80]	[8,12,16,20]
Width Multipliers	1	1	2	default
Max Channel Width c	208	208	128	32
Batch Size Per GPU	256	256	256	256
Init. Learning Rate of $\boldsymbol{\mathcal{W}}$	1e-1	1e-1	1e-1	5e-2
Learning Rate Decay	Stepwise	Stepwise	Cosine	Cosine
Optimizer	SGD	SGD	SGD	SGD
Momentum	0.9	0.9	0.9	0.9
Nestrov	False	False	True	True
Learning Rate of $\boldsymbol{\mathcal{P}}$, $\boldsymbol{\mathcal{Q}}$	1e-3	1e-3	1e-3	1e-3
Learning Rate of θ	1.6e-4	1.6e-4	1.6e-4	1.6e-4
FLOPs Penalty α, β	0, -0.1	0, -0.06	0, -0.1	0,-0.1
Entropy Regularization	4e-3	4e-3	5e-1	4e-1
Weight Decay	2e-4	2e-4	1e-4	4e-5
Warmup Epochs	200	200	80	80
Max Epochs	600	600	160	160

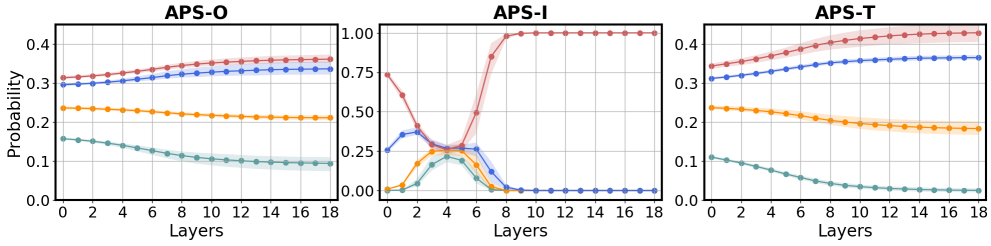


(a) ResNet-18



(b) MobileNet-v2

Figure 10: Channel configurations of ResNet-18 and MobileNet-v2 under different FLOPs constraint.



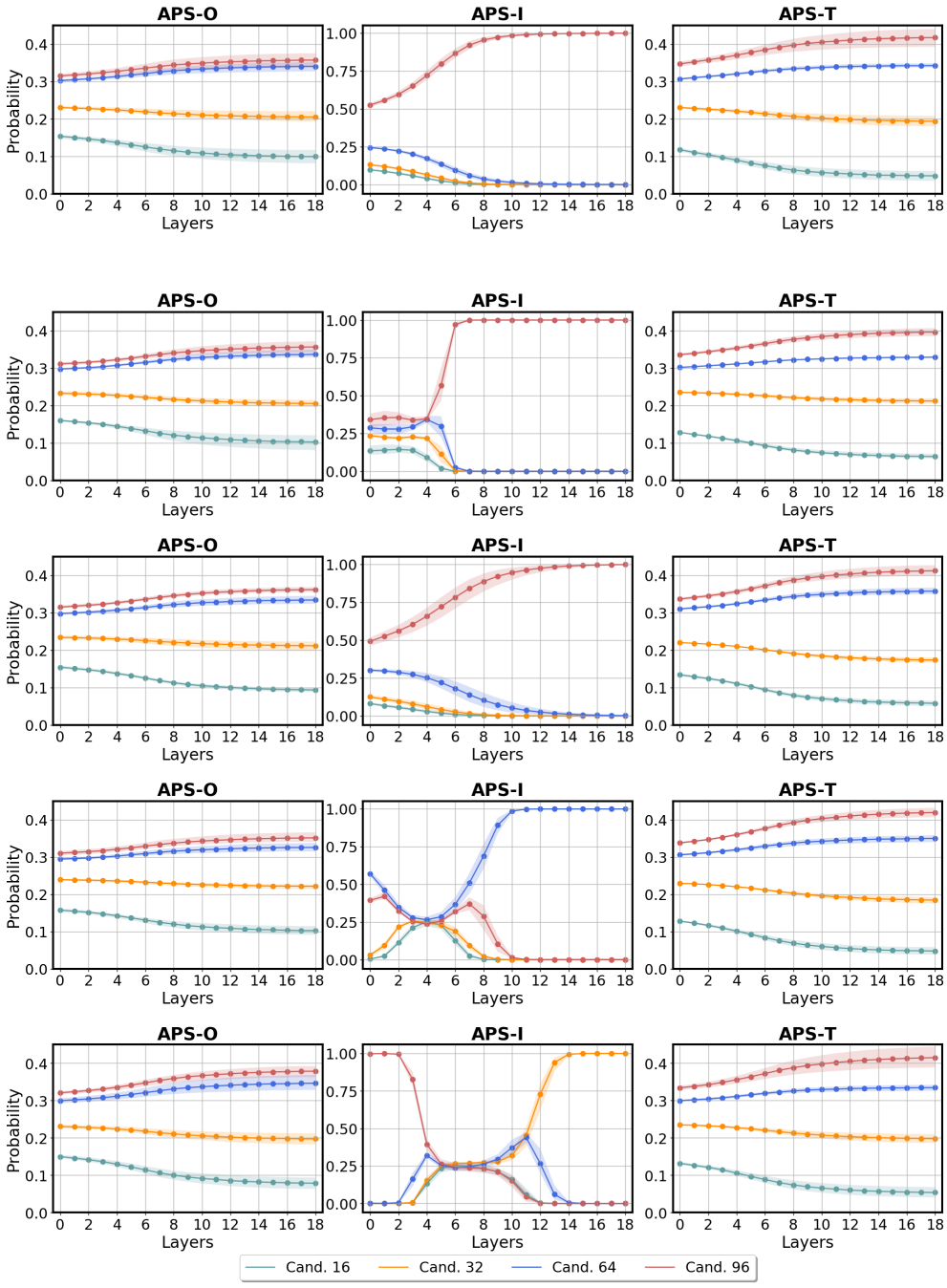


Figure 11: More searching results of APS-O, APS-I and APS-T without FLOPs constraint.