We thank all four reviewers' time, effort, and valuable suggestions. We will (I) add the suggested experiments and comparisons, (II) explain more intuition behind various design, (III) do our best to proofread our paper in revision.

To Reviewer #2 (R2)

Q2.1: Compare results with/without KD. We have compared TAS and other basic baselines with/without KD in Table 1 in our paper, which shows KD can improve about 2% accuracy when pruning ResNet-32 on CIFAR-100. Additionally, we compare advanced methods: SFP[15] 68.4% (with) VS 69.1% (without); FPGM[16] 68.5% (with) VS 69.3% (without).

Q2.2: Is the applied sampling strategy the best? The sampling procedure aims to largely reduce the memory cost and training time to an acceptable amount by only back-propagating gradients of the sampled architectures instead of all architectures. Compared to sampling via a uniform distribution, the applied sampling method (sampling based on probability) can weaken the gradients difference caused by per-iteration sampling after multiple iterations.

Q2.3: Effect of selecting different numbers of architecture samples $|\mathbb{I}|$. As suggested, we compare different numbers of selected channels in the table on the right. The searching time and the GPU memory usage will increase linearly to $|\mathbb{I}|$. When $|\mathbb{I}|$=1, since the re-normalized probability in Eq.(4) becomes a constant scalar of 1, the gradients of parameters α will become 0 and the searching failed. When $|\mathbb{I}|$>1, the performance for different $|\mathbb{I}|$ is similar.

Table 1: Results of different configurations when prune ResNet-32 on CIFAR-10 with one V100 GPU. "#SC" indicates the number of selected channels. "H" indicates hours.

#SC	Search Time	Memory	Train Time	FLOPs	Accuracy
$\|\mathbb{I}\|$=1	2.83 H	1.5GB	0.71 H	23.59 MB	89.85%
$\|\mathbb{I}\|$=2	3.83 H	2.4GB	0.84 H	38.95 MB	92.98%
$\|\mathbb{I}\|$=3	4.94 H	3.4GB	0.67 H	39.04 MB	92.63%
$\|\mathbb{I}\|$=5	7.18 H	5.1GB	0.60 H	37.08 MB	93.18%
$\|\mathbb{I}\|$=8	10.64 H	7.3GB	0.81 H	38.28 MB	92.65%

Q2.4: Does different channel-wise interpolation (CWI) affect the performance? and its formulation. As pointed in L-140∼L-142, the proposed CWI is a general operation to align feature maps with different sizes. We use adaptive average pool (AAP) [13] in our experiments, and its formulation is $\widehat{\mathbf{O}}_{i,h,w}$=mean($\mathbf{O}_{s:e\text{-}1,h,w}$), where $s = \lfloor \frac{i \times C}{C_{out}} \rfloor$ and $e = \lceil \frac{(i+1) \times C}{C_{out}} \rceil$. $\mathbf{O} \in \mathbb{R}^{CHW}$ and $\widehat{\mathbf{O}} \in \mathbb{R}^{C_{out}HW}$ are input and output tensors of CWI. We did try other forms of CWI, such as bilinear and trilinear interpolation methods. They can obtain similar performance but are much slower than our choice. It is interesting to analyze other CWI operations, which will be explored in future work.

To Reviewer #3 (R3)

Q3.1: Why only search depth and width? Researches on network architecture mainly focus on two aspects, i.e., network topology and network size. Most NAS methods automated the design of network topology, which outperforms manually designed topology. Most pruning methods manually set the network size. However, there is not an efficient way to automate the tuning procedure of network size. In our paper, we target on automating the network size design, which is challenging and helps to further boost performance. We are the first to search for the network size in a differentiable way.

Q3.2: What is the intuition behind CWI? We apply CWI to align feature map fragments with different channel sizes to the *same* channel size. In this way, we can combine the fragments together and optimize the sampling probability for each architecture. Please see Q2.4 for the specific case of CWI used in our experiments.

Q3.3: What is the intuition of $|\mathbb{I}|$=2? As analyzed in Q2.3 and L-204 in the paper, we choose $|\mathbb{I}|$=2 because it costs the minimum searching time and GPU memory usage to find suitable width and depth for a network.

To Reviewer #4 (R4)

Many thanks for your valuable comments. We have carefully proofread the paper following your suggestion.

Q4.1: Difference with [6](CVPR'19). Even though [6] and our paper both use similar technique to differentiate the sampling procedure by Gumbel Softmax, we target on different problems: our TAS focuses on network pruning by searching width and depth, while [6] focuses on searching CNN topology in a general way.

Q4.2: Compare with EA/RL with speedup gain. Given the short period of rebuttal, we can only implement co-variance matrix adaptation evolution strategy (CMA-ES) to optimize parameters α, which represents the width and depth selection. The estimated searching cost is over 1000 GPU hours. We would add the results once the CMA-ES algorithm finished. In sum, our TAS is much faster than CMA-ES (3.8H VS 1000H) and Random Search (3.8H VS 20H).

To Reviewer #5 (R5)

Q5.1: Clarify CWI. 'CHI' is a typo and will be replaced by 'CWI'. Please find its formulation in Q2.4.

Q5.2: Clarify the cost. As indicated in L172, we use FLOPs to represent one network cost. $F(\mathbb{A})$ indicates FLOPs of a network, whose width and depth are derived from $\arg\max(\mathbb{A})$ (explained in L179∼181). $\mathbb{E}_{cost}(\mathbb{A})$ is the weighted sum of FLOPs for all candidate networks, where the weight is the sampling probability of the corresponding candidate.

Q5.3: Clarify the distillation. The unpruned network is trained on the whole training set. The new searched network is derived from variable $\mathbb{A}$, which selects the best width/depth as L179∼L182. This network is trained from scratch using the whole training set, with KD loss to distill knowledge from the pre-trained unpruned network.

Q5.4: Clarification on validation. During searching, we split 50% of training set as $\mathbb{D}_{train}$ (the training set in Alg.1) and the rest 50% training set as $\mathbb{D}_{val}$ (the validation set in Alg.1) following the standard NAS setting [3,6,28]. After obtaining the searched network (Step 7 in Alg.1), we randomly initialize this searched network and train it using $\mathbb{D}_{train} \cup \mathbb{D}_{val}$. The reported accuracy is evaluated on the test set, which is never used during searching or training.

Q5.5: Why not constrain FLOP in Fig.3? As suggested, we did 4 experiments in Fig.3 with the computation cost constraint $\mathcal{L}_{cost}$ to prune about 50% FLOPs of ResNet-110. We observe that with $\mathcal{L}_{cost}$, different cases have similar relative performances but lower computation costs. Given the limited space, we can not include the new figure in rebuttal. These results and more ablation/case studies suggested in Q2.3/Q4.2 will be added in the next revision.