Supplementary Material for Learning to Learn by Jointly Optimizing Neural Architecture and Weights

Yadong Ding[1] Yu Wu[2] Chengyue Huang[1] Siliang Tang[1*]
Yi Yang[1] Longhui Wei[3] Yueting Zhuang[1] Qi Tian[4]
[1]Zhejiang University [2]Princeton University
[3]University of Science and Technology of China [4]Huawei Cloud & AI
{dyadongcs, hcyue, siliang, yangyics, yzhuang}@zju.edu.cn , yuwu@princeton.edu
longhuiwei@pku.edu.cn, tian.qi1@huawei.com

1. Detail searching settings and searched architectures

To avoid the computational cost of Hessian matrix, the first-order DARTS [11] and the first-order approximation of MAML [3] are employed for searching meta-learners. As for the inner-learners of ϕ and θ, we use the vanilla SGD with inner learning rate $\alpha_{inner} = 1 \times e^{-2}$ for optimizing ϕ, while a inner learning rate $\beta_{inner} = 0.1$ for training θ. In the meta-learner of ϕ, an Adam [6] optimizer is employed for updating, with an initial learning rate $\alpha_{meta} = 1 \times e^{-3}$ and a weight decay of 3×10^{-4}. A similar Adam without weight decay is applied to training the meta-learner of θ. We choose $M = 5$ as the inner update step. The searching is executed on both Omniglot and Mini-Imagenet under the setting of 5-way, 5-shot. For each dataset, we sample 1200 tasks from $\mathcal{D}_{meta-train}$ for training and 600 tasks from $\mathcal{D}_{meta-test}$ for evaluation. On Omniglot, we prune the architecture every three epochs from the fifth epoch, while we do it every five epoch from ninth epoch in Mini-Imagenet. All search and adaptation experiments are carried out on NVIDIA RTX 2080TI GPUs. The whole search process requires about 0.6 GPU days on Mini-Imagenet. The searched architectures is visualized in Fig.1 and Fig.2.

2. Complete experimental comparison

In this section, we make a complete experimental comparison of our methods with the methods utilizing the pretrained model in Table 1. There are some methods [12, 14, 16] obtaining better performance with more complex architectures and pretrained models. P-MAML [17] tries to learn a good initialization from ResNet18 through knowledge distillation. However, the results are not promising. Code is available at this http URL[1].

[*]Siliang Tang is the corresponding author.
[1]https://github.com/bansheng/CAML

3. Heatmap of the connection parameters

We illustrate the heatmap of connection parameters when we do pruning in Fig.3. It is evident that without CAML (treat connection parameters and network weights as the same kind of parameters), we will find a sub-optimal architecture, which contains more convolution layers. Without progressive connection consolidation, the searched architecture cannot cooperate better with the kept weights in the supernet than random initialization.

4. 5-Way accuracy results on Omniglot dataset

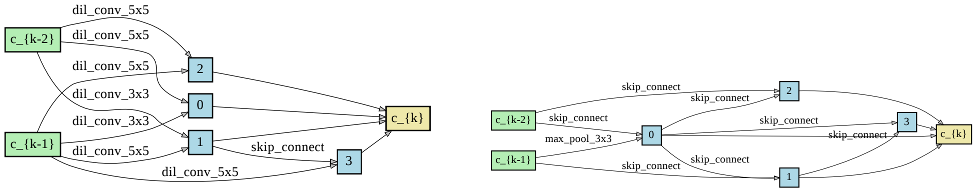
We illustrate the results of 5-way 1-shot and 5-way 5-shot on Omniglot dataset in Tab.2. We can observe that CAML++ achieves state-of-the-art performance among existing NAS-based methods.

5. Dataset splits

In few-shot learning, the dataset is composed by train, validation and test classes. Under N-way K-shot setting, we sample N classes, of which each contains K examples as one task. Tasks sampled from train classes is denoted $D_{\text{meta-train}}$. So as $D_{\text{meta-val}}$ and $D_{\text{meta-test}}$. Each of them is divided into two subset: support set $\mathcal{T}^{\text{s}}$ and query set $\mathcal{T}^{\text{q}}$. The former is used for updating the inner-learners, while the later is for the meta-learners. In our experiments, we split the $D_{\text{meta-train}}$ into two part. $D_{\text{meta-train-split1}}$ is used for optimizing the connection parameters; the other is for updating the network weights.

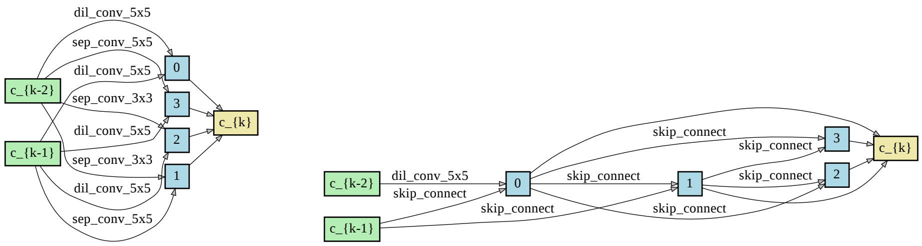
6. Results on CIFAR-10 and ImageNet

We also perform evaluation of the searched architecture on Mini-Imagenet on standard NAS benchmarks. The results are demonstrated in Tab.?? and Tab.??. We can observe that CAML can achieve comparable performance with less



(a) Normal cell (b) Reduction cell

Figure 1. Architecture searched in 5-way 5-shot setting of Mini-imagenet.



(a) Normal cell (b) Reduction cell

Figure 2. Architecture searched in 5-way 5-shot setting of FC100.

parameters on NAS benchmarks.

References

[1] Antreas Antoniou, Harrison Edwards, and Amos Storkey. How to train your MAML. In *International Conference on Learning Representations*, 2019. 4

[2] Harrison Edwards and Amos Storkey. Towards a neural statistician. In *International Conference on Learning Representations*, 2017. 4

[3] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning*, pages 1126–1135, 2017. 1, 4

[4] Łukasz Kaiser, Ofir Nachum, Aurko Roy, and Samy Bengio. Learning to remember rare events. *arXiv preprint arXiv:1703.03129*, 2017. 4

[5] Jaehong Kim, Sangyeul Lee, Sungwan Kim, Moonsu Cha, Jung Kwon Lee, Youngduck Choi, Yongseok Choi, Dong-Yeon Cho, and Jiwon Kim. Auto-meta: Automated gradient based meta learner search. *arXiv preprint arXiv:1806.06927*, 2018. 4

[6] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR (Poster)*, 2015. 1

[7] Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. Siamese neural networks for one-shot image recognition. In *ICML deep learning workshop*, volume 2. Lille, 2015. 4

[8] Zhenguo Li, Fengwei Zhou, Fei Chen, and Hang Li. Meta-sgd: Learning to learn quickly for few-shot learning. *arXiv preprint arXiv:1707.09835*, 2017. 4

[9] Dongze Lian, Yin Zheng, Yintao Xu, Yanxiong Lu, Leyu Lin, Peilin Zhao, Junzhou Huang, and Shenghua Gao. Towards fast adaptation of neural architectures with meta learning. In *International Conference on Learning Representations*, 2020. 4

[10] Chenxi Liu, Barret Zoph, Maxim Neumann, Jonathon Shlens, Wei Hua, Li-Jia Li, Li Fei-Fei, Alan Yuille, Jonathan Huang, and Kevin Murphy. Progressive neural architecture search. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 19–34, 2018. 4, 5

[11] Hanxiao Liu, Karen Simonyan, and Yiming Yang. DARTS: Differentiable architecture search. In *International Conference on Learning Representations*, 2019. 1, 4, 5

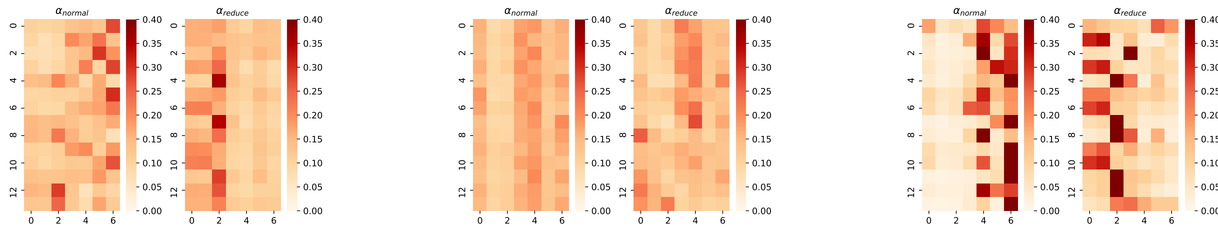
[12] Boris Oreshkin, Pau Rodríguez López, and Alexandre Lacoste. Tadam: Task dependent adaptive metric for improved few-shot learning. In *Advances in Neural Information Processing Systems*, pages 721–731, 2018. 1, 4

[13] Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V Le. Regularized evolution for image classifier architecture search. In *Proceedings of the aaai conference on artificial intelligence*, volume 33, pages 4780–4789, 2019. 4, 5

[14] Qianru Sun, Yaoyao Liu, Tat-Seng Chua, and Bernt Schiele. Meta-transfer learning for few-shot learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 403–412, 2019. 1, 4

[15] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. In *Advances in neural information processing systems*, pages 3630–3638, 2016. 4

[16] Han-Jia Ye, Hexiang Hu, De-Chuan Zhan, and Fei Sha. Few-shot learning via embedding adaptation with set-to-set func-



(a) Our method (b) Without CAML (c) Without PCC

Figure 3. Heatmap while we do pruning. (a). We use standard CAML with progressive connection consolidation (PCC). (b). We treat connection parameters and network weights equally. (c). We only prune the supernet at the end of searching.

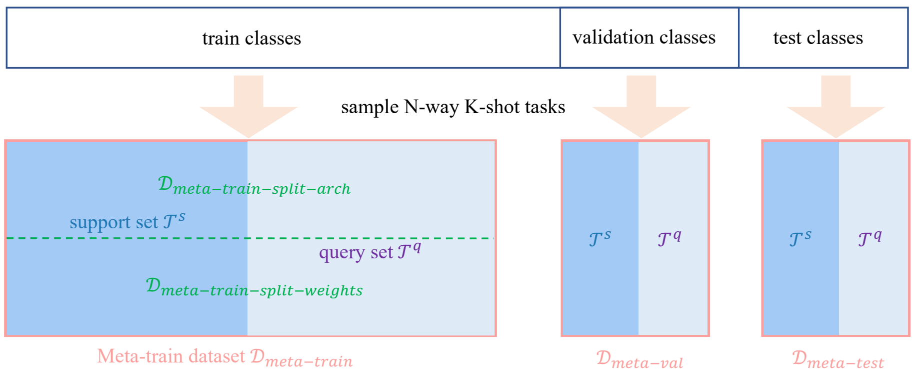


Figure 4. Dataset split

tions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8808–8817, 2020. 1, 4

[17] Min Zhang, Donglin Wang, and Sibo Gai. Knowledge distillation for model-agnostic meta-learning. In *ECAI 2020*, pages 1355–1362. IOS Press, 2020. 1, 4

[18] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. Learning transferable architectures for scalable image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8697–8710, 2018. 4, 5

Method	Arch.	Params (K)	Accuracy (%)		Pretrained
			1-shot	5-shot	
TADAM [12]	ResNet12	2039.2	58.5 $\pm$ 0.3	76.7 $\pm$ 0.3	Y
MTL [14]	ResNet12	2039.2	61.2 $\pm$ 1.8	75.5 $\pm$ 0.8	Y
FEAT [16]	ResNet18	11415.5	66.8	82.1	Y
P-MAML [17]	4CONV	32.9	49.0 $\pm$ 0.7	-	N
MAML (first order) [3]	4CONV	32.9	48.7 $\pm$ 1.8	63.1 $\pm$ 0.9	N
MAML [3]	4CONV	32.9	48.1 $\pm$ 1.8	63.2 $\pm$ 0.9	N
Auto-Meta [5]	Cell	28.0	49.6 $\pm$ 0.2	65.1 $\pm$ 0.2	N
Auto-MAML [9]	Cell	26.1	51.2 $\pm$ 1.8	64.1 $\pm$ 1.1	N
Ours	Cell	**24.2**	52.2 $\pm$ 0.4	68.1 $\pm$ 0.3	N

Table 1. Complete average 5-way classification accuracy on Mini-Imagenet with methods utilizing the pretrained model and other NAS-based methods.

Method	Accuracy (%)	
	1-shot	5-shot
Siamese nets [7]	97.3	98.4
Matching nets [15]	98.1	98.9
Neural statistician [2]	98.1	99.5
Memory mod. [4]	98.4	99.6
Meta-SGD [8]	99.53 $\pm$ 0.26	99.93 $\pm$ 0.09
MAML ([3])	98.7 $\pm$ 0.4	99.9 $\pm$ 0.1
MAML++ ([1])	99.47	99.93
Auto-Meta ([5])	97.44 $\pm$ 0.07	-
Auto-MAML ([9])	98.95 $\pm$ 0.38	99.91 $\pm$ 0.09
Ours	99.31 $\pm$ 0.07	99.93 $\pm$ 0.03

Table 2. Average 5-way classification accuracy in percent with 95% confidence interval on Omniglot.

Method	Test Error (%)	Params (M)	Search Cost (GPU days)
Random search baseline + cutout	3.29 $\pm$ 0.15	3.2	-
NASNet-A + cutout [18]	2.65	3.3	180
AmoebaNet-A + cutout [13]	3.34	3.2	3150
PNAS [10]	3.41 $\pm$ 0.09	3.2	225
DARTS (first order) [11]	3.00 $\pm$ 0.14	3.3	1.5
DARTS (second order) [11]	2.76 $\pm$ 0.09	3.37	4
Ours + cutout	3.05 $\pm$ 0.14	2.83	0.5

Table 3. Comparison with state-of-the-art NAS methods on CIFAR-10.

Method	Test Error(%)		Params	Search Cost
	top-1	top-5	(M)	(GPU days)
NASNet-A [18]	26.0	8.4	5.3	1800
NASNet-B [18]	27.2	8.7	5.3	1800
NASNet-C [18]	27.5	9.0	4.9	1800
AmoebaNet-A [13]	25.5	8.0	5.1	3150
AmoebaNet-B [13]	27.2	8.7	5.3	3150
AmoebaNet-C [13]	27.5	9.0	4.9	3150
PNAS [10]	25.8	8.1	5.1	~ 255
DARTS [11]	26.9	9.0	4.9	4
Ours	27.3	9.0	4.1	0.5

Table 4. Comparison with state-of-the-art NAS methods on ImageNet.