Auto-Meta: Automated Gradient Based Meta Learner Search

Jaehong Kim[1] **Sangyeul Lee**[1] **Sungwan Kim**[1] **Moonsu Cha**[1] **Jung Kwon Lee**[1]

Youngduck Choi[1,2] **Yongseok Choi**[1] **Dong-Yeon Cho**[1] **Jiwon Kim**[1]

SK T-Brain[1]
Yale University[2]
{ xhark, sylee0335, sw0726.kim, ckanstnzja, jklee,
yschoi, dycho24, jk} @sktbrain.com
youngduck.choi@yale.edu

Abstract

Fully automating machine learning pipelines is one of the key challenges of current artificial intelligence research, since practical machine learning often requires costly and time-consuming human-powered processes such as model design, algorithm development, and hyperparameter tuning. In this paper, we verify that automated architecture search synergizes with the effect of gradient-based meta learning. We adopt the progressive neural architecture search [14] to find optimal architectures for meta-learners. The gradient based meta-learner whose architecture was automatically found achieved state-of-the-art results on the 5-shot 5-way Mini-ImageNet classification problem with 74.65% accuracy, which is 11.54% improvement over the result obtained by the first gradient-based meta-learner called MAML [8]. To our best knowledge, this work is the first successful neural architecture search implementation in the context of meta learning.

1 Introduction

Despite the lack of full knowledge of human learning mechanisms, many researchers tried to develop learning systems to mimic our ability to quickly adapt to new environments based on previous experiences. In [7, 17, 24, 28], for example, recurrent neural networks (RNNs) were adopted to recognize the input and output mappings represented by training data and rapidly predict outputs for the test data using the internal states of learned RNN models. In contrast to these methods that rely on expert hand-crafted architectures, model-agnostic meta-learning (MAML) [8, 19] estimated a good initialization of model parameters for the fast adaptation to new tasks purely by a gradient-based search. Although we can apply model-agnostic meta-learning techniques to a variety of learning tasks without deeply contemplating the model architectures due to its model-agnostic properties, it is natural to expect that well-designed models for given tasks have better performances than conventional architectures.

In recent years, automation of machine learning has rapidly progressed. For instance, neural network architecture search for image classification tasks has been successful. The selected architectures, when followed up with appropriate fine tuning, outperform models that are manually selected and trained by deep learning experts, a slow process that requires a large amount of trial and error guided by intuition. Progressive neural architecture search (PNAS) [14] is a particular form of the automated search that progressively expands the search candidate neural network architectures, supported by an RNN model

2nd Workshop on Meta-Learning at NeurIPS 2018, Montréal, Canada.

for predicting candidates' performances without fully training the candidate architectures. Progressive neural architecture search achieves outstanding results without the significant computational expense required by reinforcement learning or evolutionary algorithm based searches.

In this paper, we propose a new approach to automatic design of neural network architectures for gradient-based meta learners. As the gradient-based meta-learning algorithm for the task, we considered Reptile [19] which is an first-order approximation of MAML [8]. To our best knowledge, this combination is the first instance of successful scalable neural architecture search work within the meta learning literature. For 5-shot 5-way Mini-ImageNet classification problem, we obtained 74.65% accuracy, which is 11.54% improvement over the result from MAML [8].

2 Related Works

Meta-learning Meta learning as a theme is quite general, and extends well beyond the gradient based meta learners for few shot classification tasks. In fact, much of the meta learning literature focuses on the general reinforcement learning tasks [7, 28]. One of the most common approaches to meta-learning is to build a recurrent neural network as a meta-learner. In particular, RNN based methods, augmented with memory-augment network [24] or simple attention mechanism [17] have been applied to few-shot image classification tasks.

Metric learning is another popular approach to address meta-learning problems. The meta learner attempts to learn a metric which can be used to compare two different examples effectively and performs tasks in the learned metric space [27]. Some studies train a Siamese network to achieve the same objective [12]. The metric-based meta-learning has been known to perform well for few-shot image classification tasks [26, 25].

The other major category of meta learning is to learn an optimizer as the meta-learner which enables the learner to learn a new task more effectively [10, 1]. This approach has been applied to few-shot learning successfully [20]. Rather than using the learned optimizer, a new meta-learning scheme applicable to all gradient-based learning algorithms, model-agnostic meta-learning (MAML), has recently been proposed [8]. MAML attempts to find a set of parameters which initializes a learner for any specific task to be trained quickly only with small amount of data. Although this technique has shown the effectiveness for various few-shot learning problems including few-shot image classification and reinforcement learning, Hessian vector product calculation during training requires a large amount of computation. The first-order approximation algorithm has been proposed to avoid the Hessian computations in [19]. Some advantages of gradient based meta learners have also been discussed in [9].

Neural network architecture search Neural network architecture search (NAS) is a methodology to automatically find optimal neural network architectures for a given task. There are various types of NAS, such as reinforcement learning based NAS and evolutionary algorithm based NAS. Reinforcement learning based NAS includes REINFORCE [31], Q-learning [30, 2], and PPO-type algorithms [32]. Evolutionary algorithm based NAS has extensively been explored in [22, 16, 29, 15, 21]. For example, AmoebaNet [22] applies an evolutionary algorithm to the same search space of NASNet and achieves state-of-the-art results on image classification tasks. Other methods deploy various types of reasonable heuristics that attempt to reduce computational cost. This line of thinking is present in hypernetworks [4], co-evolving NEAT [16], boosting [6, 11], MCTS [18], early stopping of unpromising models [3], and progressive neural architecture search (PNAS) [14]. In particular, PNAS expands the search space incrementally from simple to complex so that it can search architectures in an efficient way without limiting the search to the space of fully-specified architectures.

3 Auto-Meta

Our goal is to automatically find the optimal network architecture for gradient-based meta-learners. Considering the loss function $\mathcal{L}$ for given tasks j represented by training and test data sets (D_j^{Tr}, D_j^{Te}), this can be formulated as follows:

$$\min_{A,\theta} \sum_j \mathcal{L}\big(D_j^{Te}, U(D_j^{Tr}, \theta; A)\big), \tag{1}$$

where A and θ are the neural network architecture and its parameters, respectively. U denotes the computation of parameter updates using one or more gradient descent steps. A natural and simply way to solve this problem is to minimize the loss $\mathcal{L}$ over parameters keeping the candidate architectures fixed. Then, based on some promising architectures, more complicated architectures are searched progressively. By repeating these two steps, we can obtain a good approximate solution to Equation (1).

As the gradient-based meta-learning algorithm, we adopt Reptile [19] which showed comparable performance with MAML [8] on few-shot classification problems using only first-order gradient information. This allows us to avoid the time-consuming calculation of second-order derivatives. As the network architecture search method, we consider the PNAS algorithm [14] where three layers (i.e., block, cell, and network) of abstraction for representing a neural network topology were defined. At most B blocks which represent a combination operators applied to two inputs are included in a cell. This cell is then stacked a certain number of times to create a full CNN. During the architecture search, the cells "progressively" get more complicated by adding a block to themselves. Without expensive training procedure, the performance of each cell is evaluated with a surrogate predictor, such as LSTM to rank all expanded candidate cells. Then, CNNs with the top K cells are trained and evaluated. We continue in this way until each cell has the maximum number of blocks.

4 Experiments and Results

To implement our gradient based meta-learner search, a distributed system that deploys and trains many neural networks in a stable manner is required. We apply a Kubernetes system [5] that supports utilities to deploy over clusters of GPUs, in order to achieve the parallelization of training and testing top models in each iteration of block progression. We used the system with 112 P40 GPUs for experiments described in this section. Our architecture search procedure on Mini-ImageNet dataset takes 24 hours on average after parallelization.

4.1 Few Shot Image Classification

To evaluate the automated architecture search algorithm for the gradient-based meta-learning, we applied our method to few-shot image classification problems. We used two benchmark datasets Omniglot [13] and Mini-ImageNet [27][20]. In the Omniglot dataset, there are 1623 handwritten characters from 50 different alphabets. The Mini-ImageNet dataset consists of $60,000$ images (84×84 pixels) with 100 classes. In particular, we conducted the 5-way, 1-shot and 5-shot classification tasks through the proposed gradient based meta learner search. Meta-test accuracy of meta learner was used as score for the LSTM predictor. Based on predicted scores from surrogate the LSTM, we trained 100 most promising cell candidates. After search completed, we chose the cell structure with best score for final training. We use 3×3 convolution, 5×5 factorized convolution, identity, 3×3 average pooling, and 3×3 max pooling as operations for blocks. For training parameters of the network and surrogate LSTM predictor, we used Adam optimizer with learning rate 0.01. Other hyperparameters used in our experiments are given in Table A1.

Algorithm	#Params	Accuracy
Ours ($F = 10$)	25.9k	**97.44 ± 0.07%**
Ours^{Transduction} ($F = 10$)	25.9k	**98.94 ± 0.05%**
MAML^{Transduction} [8]	112.0k	98.7 ± 0.4%
1st-order MAML^{Transduction} [19]	112.0k	98.3 ± 0.5%
Reptile [19]	113.2k	95.39 ± 0.09%
Reptile^{Transduction} [19]	113.2k	97.68 ± 0.04%
Matching-Nets [27]	-	98.1%
Prototypical-Nets [26]	111.9k	98.8%

Table 1: Results on Omniglot dataset for the 1-shot 5-way classification
(*F*: The number of filters in the first convolution layer, *Transduction*: Prediction at test time using batch normalization with batch mean and variance [19])

As shown in Table 1, our architecture search algorithm successfully found an excellent cell in 1-shot 5-way image classification for the Omniglot dataset so that the full convolution neural network (CNN)

constructed with the cell outperformed the human-designed CNN trained with the Reptile [19]. It is also worth pointing out that our CNN model has much smaller number of parameters than other ones.

		1-shot 5-way		5-shot 5-way	
	Algorithm	#Params	Accuracy	#Params	Accuracy
small setting	Ours ($F = 10, 10$)	28k	**49.58 ± 0.20%**	28k	**65.09 ± 0.24%**
	Ours^{Transduction}($F = 10, 10$)	28k	**54.02 ± 0.13%**	28k	**69.77 ± 0.31%**
	MAML^{Transduction} [8]	32.8k	$48.70 \pm 1.84\%$	32.8k	$63.11 \pm 0.92\%$
	Reptile [19]	34.7k	$47.07 \pm 0.26\%$	34.7k	$62.74 \pm 0.37\%$
	Reptile^{Transduction} [19]	34.7k	$49.96 \pm 0.32\%$	34.7k	$65.99 \pm 0.58\%$
large setting	Ours ($F = 12, 12$)	98.7k	**51.16 ± 0.17%**	94.0k	**69.18 ± 0.14%**
	Ours^{Transduction}($F = 12, 12$)	98.7k	**57.58 ± 0.20%**	94.0k	**74.65 ± 0.19%**
	Matching-Nets [27]	-	43.60%	-	55.30%
	Ravi and Laroche [20]	-	$43.40 \pm 0.77\%$	-	$60.20 \pm 0.71\%$
	Prototypical-Nets [26]	113.1k	$49.42 \pm 0.78\%$	113.1k	$68.20 \pm 0.66\%$

Table 2: Number of parameters and accuracy on Mini-ImageNet for 5-way, 1-shot and 5-shot classification tasks

Results for Mini-ImageNet dataset are shown in Table 2. For fair comparison, we tried to make full CNN models with the searched cell have similar capacity to models adopted by previous work in *small* setting. For the 1-shot and 5-way task, our method showed the comparable performance with others. However, the results clearly showed the superiority of the automated search for the 5-shot and 5-way task. Without restricting the model size in terms of the number of parameters, we constructed the best performing CNN with the best cell by appropriately choosing the value of *F* in *large* setting. For both 1-shot and 5-shot tasks, our model had the best accuracy. More specifically, our gradient based meta learner which have the automatically searched cell achieved 74.65% accuracy on the 5-shot 5-way task. This implies that our approach not only drastically improved performance of other meta-learners but also can compete with state-of-the-art techniques such as [23] for few shot classification tasks. The best cell architectures found in these tasks were shown in Figure 1.

To investigate how our progressive network architecture search algorithm can find the best cells for few-shot image classification tasks, we observed the distribution of depths of the promising cells as the search progresses. As shown in Figure 2, the distributions are moving toward deeper architectures in both settings. Surely, further empirical studies would be required to have a more conclusive remark on this phenomenon.

5 Conclusion and Future Work

Our gradient based meta learner search was motivated by the current trend in machine learning communities. There have been many attempts to automate the machine learning pipelines in different manner. We tried a simple, but natural combination of two automation techniques: neural architecture search and gradient-based meta-learning. To our best knowledge, this implementation is the first successful AutoML execution in the context of meta learning.

Our results indeed show that automated architecture search is beneficial for improvement of the performance of meta-learning algorithms which tried to find good initialization for different tasks. Our gradient based meta learners with automatically search architectures have much better results than other meta-learners with human-crafted models on some few shot image classification tasks and compatible results with state-of-the-art techniques which employed more sophisticated auxiliary components such as encoder and decoder networks for the tasks.

Future work includes exploring the effect of operators in our search method, and further empirical studies on the shape of cells in terms of depth and width for good performance. Also, various kinds of neural architecture search methods should be explored in the context of meta learning. For example, it would be quite interesting to see if the architecture and parameters can be jointly optimized by formulating the model search in a differentiable manner. The applicability of our method can be tested to other domains which require the fast-adaptation property.

References

[1] M. Andrychowicz, M. Denil, S. G. Colmenarejo, M. W. Hoffman, D. Pfau, T. Schaul, and N. de Freitas. Learning to learn by gradient descent by gradient descent. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 3981–3989, 2016.

[2] B. Baker, O. Gupta, N. Naik, and R. Raskar. Designing neural network architectures using reinforcement learning. *CoRR*, abs/1611.02167, 2016.

[3] B. Baker, O. Gupta, R. Raskar, and N. Naik. Accelerating neural architecture search using performance prediction. *CoRR*, abs/1705.10823, 2017.

[4] A. Brock, T. Lim, J. M. Ritchie, and N. Weston. SMASH: one-shot model architecture search through hypernetworks. *CoRR*, abs/1708.05344, 2017.

[5] B. Burns, B. Grant, D. Oppenheimer, E. Brewer, and J. Wilkes. Borg, omega, and kubernetes. *ACM Queue*, 14:70–93, 2016.

[6] C. Cortes, X. Gonzalvo, V. Kuznetsov, M. Mohri, and S. Yang. Adanet: Adaptive structural learning of artificial neural networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pages 874–883, 2017.

[7] Y. Duan, J. Schulman, X. Chen, P. L. Bartlett, I. Sutskever, and P. Abbeel. Rl^2: Fast reinforcement learning via slow reinforcement learning. *CoRR*, abs/1611.02779, 2016.

[8] C. Finn, P. Abbeel, and S. Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pages 1126–1135, 2017.

[9] C. Finn and S. Levine. Meta-learning and universality: Deep representations and gradient descent can approximate any learning algorithm. *CoRR*, abs/1710.11622, 2017.

[10] S. Hochreiter, A. S. Younger, and P. R. Conwell. Learning to learn using gradient descent. In *Artificial Neural Networks - ICANN 2001, International Conference Vienna, Austria, August 21-25, 2001 Proceedings*, pages 87–94, 2001.

[11] F. Huang, J. T. Ash, J. Langford, and R. E. Schapire. Learning deep resnet blocks sequentially using boosting theory. *CoRR*, abs/1706.04964, 2017.

[12] G. Koch. Siamese neural networks for one-shot image recognition. In *ICML workshop*, 2015.

[13] B. M. Lake, R. Salakhutdinov, and J. B. Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.

[14] C. Liu, B. Zoph, J. Shlens, W. Hua, L. Li, L. Fei-Fei, A. L. Yuille, J. Huang, and K. Murphy. Progressive neural architecture search. *CoRR*, abs/1712.00559, 2017.

[15] H. Liu, K. Simonyan, O. Vinyals, C. Fernando, and K. Kavukcuoglu. Hierarchical representations for efficient architecture search. *CoRR*, abs/1711.00436, 2017.

[16] R. Miikkulainen, J. Z. Liang, E. Meyerson, A. Rawal, D. Fink, O. Francon, B. Raju, H. Shahrzad, A. Navruzyan, N. Duffy, and B. Hodjat. Evolving deep neural networks. *CoRR*, abs/1703.00548, 2017.

[17] N. Mishra, M. Rohaninejad, X. Chen, and P. Abbeel. A simple neural attentive meta-learner. *CoRR*, abs/1707.03141, 2017.

[18] R. Negrinho and G. J. Gordon. Deeparchitect: Automatically designing and training deep architectures. *CoRR*, abs/1704.08792, 2017.

[19] A. Nichol, J. Achiam, and J. Schulman. On first-order meta-learning algorithms. *CoRR*, abs/1803.02999, 2018.

[20] S. Ravi and H. Larochelle. Optimization as a model for few-shot learning. In *ICLR 2017*, 2017.

[21] E. Real, A. Aggarwal, Y. Huang, and Q. V. Le. Regularized evolution for image classifier architecture search. *CoRR*, abs/1802.01548, 2018.

[22] E. Real, S. Moore, A. Selle, S. Saxena, Y. L. Suematsu, J. Tan, Q. V. Le, and A. Kurakin. Large-scale evolution of image classifiers. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pages 2902–2911, 2017.

[23] A. A. Rusu, D. Rao, J. Sygnowski, O. Vinyals, R. Pascanu, S. Osindero, and R. Hadsell. Meta-learning with latent embedding optimization. *CoRR*, abs/1807.05960, 2018.

[24] A. Santoro, S. Bartunov, M. Botvinick, D. Wierstra, and T. P. Lillicrap. Meta-learning with memory-augmented neural networks. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, pages 1842–1850, 2016.

[25] P. Shyam, S. Gupta, and A. Dukkipati. Attentive recurrent comparators. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pages 3173–3181, 2017.

[26] J. Snell, K. Swersky, and R. S. Zemel. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 4080–4090, 2017.

[27] O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, and D. Wierstra. Matching networks for one shot learning. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 3630–3638, 2016.

[28] J. X. Wang, Z. Kurth-Nelson, D. Tirumala, H. Soyer, J. Z. Leibo, R. Munos, C. Blundell, D. Kumaran, and M. Botvinick. Learning to reinforcement learn. *CoRR*, abs/1611.05763, 2016.

[29] L. Xie and A. L. Yuille. Genetic CNN. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 1388–1397, 2017.

[30] Z. Zhong, J. Yan, and C. Liu. Practical network blocks design with q-learning. *CoRR*, abs/1708.05552, 2017.

[31] B. Zoph and Q. V. Le. Neural architecture search with reinforcement learning. *CoRR*, abs/1611.01578, 2016.

[32] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le. Learning transferable architectures for scalable image recognition. *CoRR*, abs/1707.07012, 2017.

Appendix

Architecture Search Hyperparameters

Progressive neural architecture search	
Max num blocks (B)	5
Num filters in first layer (F)	4, 10, 12, 32
Beam size (K)	100
Num times to unroll cell (N)	0, 1
Feature Scale Rate	1, 2
Surrogate predictor [14]	Cell size $= 100$, Number of layers $= 1$
Gradient-based meta-learning	
Inner iterations	8
Inner-batch size	10
Inner Adam learning rate	0.001, 0.01
Meta-batch size	5
Outer step size	0.3, 1.0
Outer iterations	1000

Table A1: Hyperparameters

Further Empirical Analysis

Figure 1 shows the cell architectures that have the highest classification accuracies for 1-shot 5-way Mini-Imagenet classification tasks in the small and large settings respectively. The depths of both are equivalent to three vertically stacked blocks while their constituent operations and connections are different from each other.

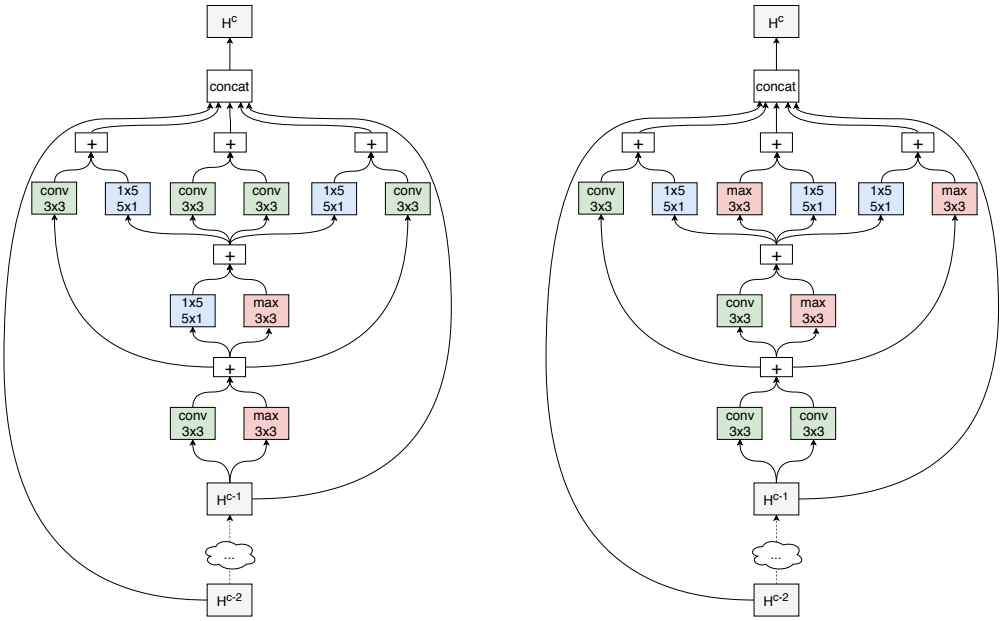

Figure 1: The best cell architectures for 1-shot 5-way Mini-Imagenet tasks in the small (left) and large (right) settings (some pre-/post-processing operations are omitted for the sake of simplicity)

Figure 2 shows the distribution of depths of the best cells at each stage in terms of the number of blocks as the search progresses from $b = 3$ to $b = 5$. We can see the distributions are moving toward deeper architectures in both settings while the degree of the change seems much larger in the large setting.

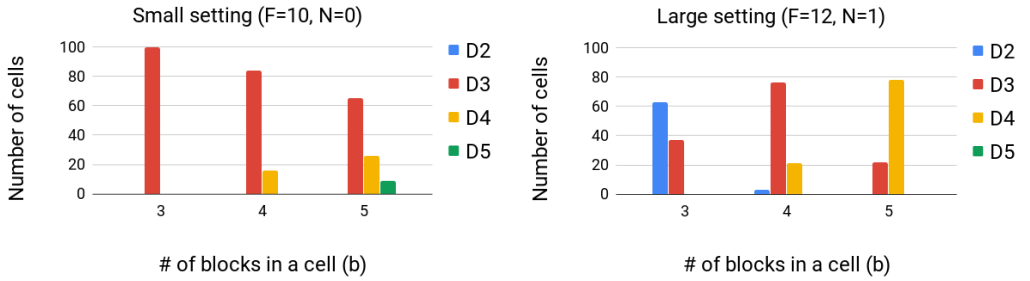

Figure 2: Evolution of the cell depth distribution during the progressive network architecture search (Dx refers to cells whose depth is x in terms of blocks.)