# Auto-Meta: Automated Gradient Based Meta Learner Search

**Jaehong Kim**[1] **Youngduck Choi**[1,2]

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

**Jiwon Kim**[1]

SK T-Brain[1]
Yale University[2]
{ xhark, jklee, ckanstnzja, ylee0335, sw0726.kim,
yschoi, jk} @sktbrain.com
youngduck.choi@yale.edu

## Abstract

Fully automating machine learning pipeline is one of the outstanding challenges of general artificial intelligence, as practical machine learning often requires costly human driven process, such as hyper-parameter tuning, algorithmic selection, and model selection. In this work, we consider the problem of executing automated, yet scalable search for finding optimal gradient based meta-learners in practice. As a solution, we apply progressive neural architecture search to proto-architectures by appealing to the model agnostic nature of general gradient based meta learners. In the presence of recent universality result of Finn *et al.*[9], our search is a priori motivated in that neural network architecture search dynamics—automated or not—may be quite different from that of the classical setting with the same target tasks, due to the presence of the gradient update operator. A posteriori, our search algorithm, given appropriately designed search spaces, finds gradient based meta learners with non-intuitive proto-architectures that are narrowly deep, unlike the inception-like structures previously observed in the resulting architectures of traditional NAS algorithms. Along with these notable findings, the searched gradient based meta-learner achieves state-of-the-art results on the few shot classification problem on Mini-ImageNet with $76.29\%$ accuracy, which is an $13.18\%$ improvement over results reported in the original MAML paper. To our best knowledge, this work is the first successful AutoML implementation in the context of meta learning.

## 1 Introduction

In recent years, automating machine learning practice has been an active area of research with the rise of availability of computational resources and demand for off-the-shelf models for non-expert uses. Automated neural network architecture search for the image classification tasks, for instance, has been quite successful in that the searched architectures, when followed up with appropriate fine tuning, achieve comparable results with models that are manually selected and trained by computer vision experts, whose process naturally requires a large amount of trial and error, merely guided by intuition. Progressive neural architecture search [13] is a particular form of the automated search that

Preprint. Work in progress.

progressively expands the search candidate neural network architectures, combined with an explicit neural network model for predicting the performance of candidate architectures, if they were to be fully trained and fine tuned. Progressive neural architecture search achieves state-of-the-art results without the burdens of computational expenses, that are required by either reinforcement learning or evolutionary algorithm based searches.

Within Machine Learning literature, meta learners refer to various mechanism for learning to learn arbitrary tasks. While sacrificing meta learning's great generality, we focus primarily on gradient based meta learners, such as MAML [8], for few shot classification tasks. Though gradient based meta learners are model agnostic, in a sense that they are compatible with any neural network architecture, to employ these models in practice, a particular choice of proto-architecture(neural network embedded in the meta learner) is not only necessary, but essential for achieving good performance on targeted tasks. The recent finding in regards to universality of gradient based meta learners[9] suggests that the neural network architecture search dynamic might be quite different from that of ordinary search dynamic in the absence of the gradient update operator. The above context in combination with the relative unfamiliarity with gradient based meta learners for most ML practitioners naturally invites us to consider an automated search for finding the optimal gradient based meta learners for given tasks.

In this work, we propose an algorithm for first-order gradient based meta learner search by applying progressive neural architecture search to the space of proto-architectures of MAML. Guided by the universality result, we design the combinatorial search space of PNAS to contain network topologies that are much narrowly deeper, compared to the original proto-architecture in the original MAML work. The search space, however, naturally introduces computational complexity that is an order of magnitude higher than the previous gradient based meta learner computation. We resolve this issue by leveraging Reptile[18], which is an first-order approximation to MAML. Through this method, one can avoid the computation of the Hessian vector product that MAML loss requires, as a consequence of having nested loss functions in the objective. Interestingly enough, the resulting architectures of the search are quite non-intuitive and are in fact narrowly deep, unlike the InceptionNet-like structures that appear in the classical NAS algorithms when applied to image classification tasks.

To our best knowledge, this paper presents the first instance of successful scalable AutoML work within the meta learning literature. We obtain state of the art result on 5 way 5 shot classification problem on MiniImageNet dataset with $76.29\%$ accuracy, which is an $13.18\%$ improvement over the result from the original MAML paper.

## 2 Related Works

**Meta-learning** As mentioned before, 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, 26]. 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[22] or simple attention mechanism[16] 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 [25]. 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 [24, 23].

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 [19]. 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]. Technically, 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 the training requires a large amount of computation. The first-order approximation algorithm has been proposed to reduce the Hessian computations in [18]. The universality of gradient based meta learners has 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[29], Q-learning [28, 2], and PPO [30] type algorithms. In particular, Zoph et al.[29] uses the REINFORCE algorithm to train a controller that samples candidate architectures. Evolutionary algorithm based NAS has extensively been explored in [21, 15, 27, 14, 20]. For example, AmoebaNet[21] applies 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 [15], boosting [6, 11], MCTS [17], early stopping of unpromising models [3], and progressive neural architecture search(PNAS) [13]. In particular, PNAS is quite computational cost efficient. PNAS expands the search space incrementally from simple to complex and does not limit the search to the space of fully-specified architectures.

## 3 Auto-Meta

### 3.1 Gradient Based Meta-Learners and the Search Problem

MAML by Finn et al. considered the following problem: "find an initial set of parameters, $\phi$, such that for a randomly sampled task $\tau$ with corresponding loss $L_\tau$ , the learner will have low loss after $k$ updates" [8]. This can be formulated as an optimization problem via

$$\min_{\phi} \mathbb{E}_\tau \left[L_{\tau,B}(U_{\tau,A}(\phi))\right]$$

where $U_\tau^k$ is the "operator" that updates $\phi$ k times using data sampled from $\tau$, such as k-step gradient update, and $(A, B)$ represents training and test samples coming from two different sources, which corresponds to directly optimizing for cross-validation scores [18]. A $\phi$ that achieves the minimum is usually called a set of meta-parameters. The idea is that once we have a set of meta-parameters, we can simply initialize our neural network model with the set, so that the model will do a fast adaptation through few iterations of back-propagation to a particular task that the model encounters in test time. It is often emphasized that the above formulation is model agnostic in that it is compatible with any neural network model. In practice, however, a particular choice of neural network topology is not only required, but also quite essential to achieving good performance. In fact, the recent theoretical results in universality of gradient based meta learner reveals that, due to the presence of the $U_\tau$ term, the neural network search dynamic—automated or not—is quite different from the ordinary neural network search dynamic, where the neural network in the search space, corresponds to the entire inference step, even when the target tasks are the same in test time. This naturally motivates the following minimization problem, adapted from the MAML objective:

$$\min_{T\in\mathfrak{T},\ \phi\in\mathbb{R}^{|T|}} \mathbb{E}_\tau[L_{\tau,B,T}(U_{\tau,A,T}(\phi))], \tag{1}$$

where $\mathfrak{T}$ represents all possible neural network topologies. Clearly, the above minimization problem is intractable in a practical sense, as there are uncountably many many architectures one would need to try. Even in a practical sense, where $\mathfrak{T}$ is reduced to some efficient combinatorial search space, combined with the potential need for including models with higher expressibility with a large number of parameters, (1), when compared to the original MAML objective, incurs much larger additional computational complexity. Hence, to solve (1), we resort to an approximation to the gradient of MAML objective through the following approximation:

$$\begin{aligned} g_{\text{MAML}} &= \frac{\partial}{\partial\phi_1} L_k(\phi_k) = \frac{\partial}{\partial\phi_1} L_k(U_{k-1}(...U_1(\phi_1)...)) \\ &= \left(\prod_{j=1}^{k-1}(I-\alpha L_j^{''}(\phi_j))\right) g_k \approx \sum_{j=1}^{k} g_j, \end{aligned} \tag{2}$$

where $g_i = L_i^{'}(\phi_i)$, $\phi_{i+1} = \phi_i - \alpha g_i$ for $1 \leq i \leq n$, representing mini-batches, and the last approximation arises as a second-order taylor series analysis with respect to $\alpha$. The intuition behind this approximation is that moving a point via linear interpolation with computed gradients then computing the gradient at the moved point ends up gifting us the approximate Hessian vector product, computed—not on the moved point—on the initial point that one would need to compute to deal with the MAML loss. In a nutshell, the above approximation has an algorithmic implication that we simply do SGD updates, while updating the parameters accordingly.

### 3.2 Progressive Neural Architecture Search

We now discuss the issue of defining an efficient combinatorial search space $\mathfrak{T}$, which appears in (1). Progressive neural architecture search introduces three layers of abstraction for representing a neural network topology: blocks, cells and full neural networks. Blocks, which typically ranges from 1 to 5, are combined to form cells, and cells are joined to create a full CNN by repeating itself as a layer, where the number of layers is fixed for a given task. The cells "progressively" get more complicated by adding a block to itself. The algorithm essentially adds a block structure with varying connections to existing cells and predict how well these newly introduced cells will perform with a surrogate predictor, such as LSTM, if they were to be fitted after being rolled-out by a fixed number of layers by a particular optimization technique, such as ADAM. When predictions are out, the good cells, judged by the surrogate predictor, are kept, and the algorithm actually proceed to train the rolled-out CNNs of the cells with (train,validate) pairs. When the rolled-out CNNs are trained and evaluated, we get back the performance of each cell and with those we update the surrogate predictor(LSTM). In the end, after b=5, we return top K(typically 100 in our experiments) CNNs that we estimate to perform the best on the given task.

### 3.3 Gradient based Meta-learning and Universality

The classical universal function approximation theorem states the following: let $\varphi(\cdot)$ be a non-constant, bounded, and monotonically-increasing continuous function, $I_m$ be the m-dimensional unit cube. If

$$\mathcal{F} \quad = \quad \left\{ F(x) = \sum_{i=1}^{N} v_i \varphi(w_i^T x + b_i) \ : N \in \mathbb{N}, \forall, 1 \le i \le N \ \ v_i, b_i \in \mathbb{R} \right\},$$

then $\mathcal{F}$ is dense in $(C(I_m), || \cdot ||_{\text{sup}})$. In the context of machine learning, one can view $\mathcal{F}$ as a feed-forward network with a single hidden layer containing a finite number of neurons, and each $f \in C(I_m)$ as given data, where the unit-cube assumption arises with the usual setting of finite sample. Though algorithmic feasibility of obtaining such approximation may not be granted, universality is a desirable property for an neural network architecture.

The universality property of gradient based meta learners behaves quite differently from the above classical case, due to the presence of the gradient update operators. Recall that for one-shot classification tasks, one can write:

$$\begin{aligned} y^* &= f_{\text{MAML}}(D_T, x^*; \theta) \\ &= f\left( x^*; \theta - \alpha \nabla_\theta \frac{1}{K} \sum_{k=1}^{K} l(y_k, f(x_k; \theta)) \right), \end{aligned}$$

where $f$ is the proto-architecture of the meta learner, $(D_T) = \{(x_k, y_k)\}$ is the few-shot data. Observe that the above form of $f_{\text{MAML}}$ does not go through the assumptions of the classical universal function approximation theorem. Finn *et al.* in [9] proves a necessary condition on $f$ for $f_{\text{MAML}}$ to be a universal function approximator: if $f$ is a form of NN, *it needs to be sufficiently deep*, which marks a clear distinction with the classical case. The key insight comes from the fact that a single gradient step can only represent a rank$-1$ update to the weight matrix. This effect, however, disappears when weight matrices are composed together. For instance, with $W$ being a product of $N$ matrices, a single gradient step can represent a rank$-n$ update to $W$. This indicates that the wider the weight matrices are in gradient based meat learners, due to the presence of gradient step operator, the deeper the proto-architecture needs to be to compensate for the restriction on the expressibility of a single gradient update. This theory also generalizes to the case of multi-shot case, and one recovers an UFA type theorem for functions that are invariant to the ordering of training data-points.

### 3.4 Gradient based Meta-learner Search through Proto-architecture search

Building upon the above sections, we finally discuss how to do a gradient based meta learner search concretely in practice. Rather than executing a general search for gradient based meta-learner, we execute progressive neural architecture search for proto-architectures with the loss presented (2) for computational efficiency. With the few shot vision classification tasks, we use $3 \times 3$ convolution, $5 \times 5$ factorized convolution, identity, $3 \times 3$ average pooling, and $3 \times 3$ max pooling as operations

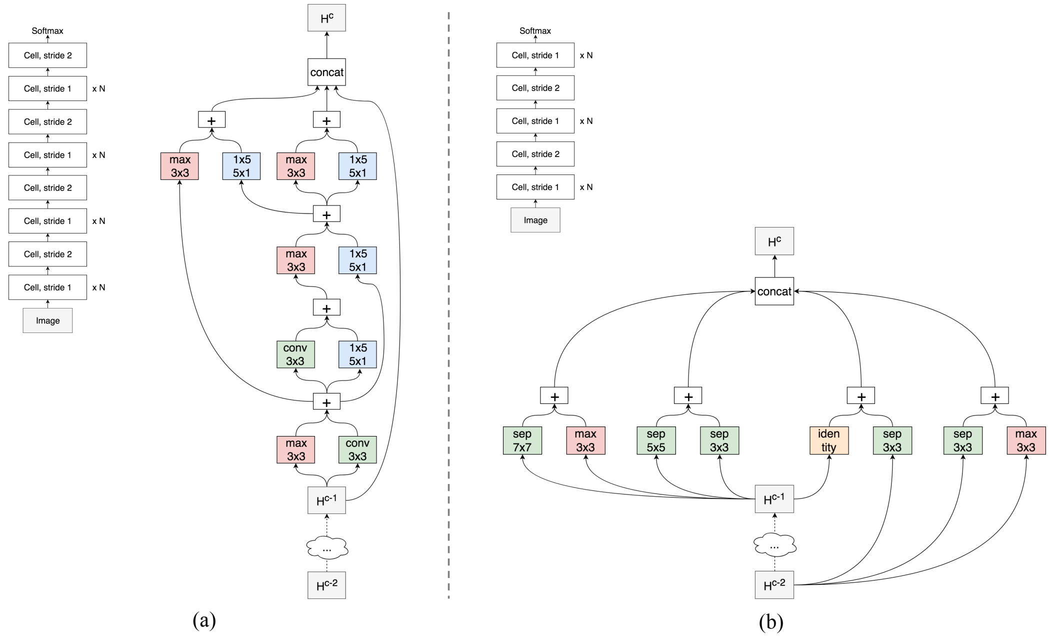


Figure 1: Searched cell structures and cell2CNN unrolling for image classification tasks. (a) Gradient based meta learning(ours) (b) Classical neural architecture search [13]. One can see that our searched proto-architecture is quite non-intuitive, and narrowly deep, when compared to the classical NAS case.

for possible block combinations in progressive neural architecture search. Importantly, the style of search that we propose naturally introduces an interaction with the universality theory. Hence, in our experiments, we design our search spaces to typically include architectures that are narrowly deeper than the original MAML proto-architecture.

## 4 Experiments and Results

In this section, we first explain implementation details. We then describe experimental setup and results for the few shot image classification tasks on the Mini-ImageNet dataset. We also show cell progression of top performing cells and explore various depth related statistics of conducted searches, motivated by the universality property of gradient based meta learners. This extends the empirical analysis in [9] to a much larger scale.

### 4.1 Implementation Details

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 have applied Kubernetes system [5] that supports utilities to deploy, maintain, and extend 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.

### 4.2 Mini-ImageNet Few Shot Image Classification

We use Mini-ImageNet dataset composed of $84 \times 84$ $128,779$ images with $100$ classes. Among them, we use $82,099$ images with $64$ classes for training, $25,964$ images with $20$ classes for testing, and $20,716$ images with $16$ classes as a validation set. In particular, we use the $5-$shot, $5-$way classification task to conduct the gradient based meta learner search. Test task accuracy of meta learner was used as score for LSTM predictor. When searching, $N = 0, 1, F = 4, 32, B = 5$ are used for the progressive neural architecture search part. Based on predicted scores from surrogate

LSTM, we use $K = 100$ cell candidates to train. After finished search stage $B = 5$, we choose the cell structure with best score as promising cell structure for final training. As mentioned in Section 3.4, we use $3 \times 3$ convolution, $5 \times 5$ factorized convolution, identity, $3 \times 3$ average pooling, and $3 \times 3$ max pooling as operations for blocks. For network training and surrogate LSTM predictor, we use Adam optimizer with learning rate $0.01$. Meta batch size is 5, and meta iterations is 1000. Meta step is primarily set to be 2 and then decays linearly to 0 during training.

To compare the performance of found cell structures, we first set $F = 10, N = 0$ to build a final CNN with nearly same amount of parameters of original meta learners and train the model for both $5-$shot, $5-$way classification and $1-$shot, $5-$way classification (*small* settings). Also, we build the best performing model by appropriately setting F and N (*large* settings).

| Algorithm | Params | 5-shot 5-way |
|---|---|---|
| MAML + Transduction [8] | 35k | $63.11 \pm 0.92\%$ |
| 1st-order MAML + Transduction [18] | 35k | $63.15 \pm 0.91\%$ |
| Reptile + Transduction [18] | 35k | $65.99 \pm 0.58\%$ |
| Ours + Transduction ($F = 10$) | 28k | $\mathbf{69.77 \pm 0.31}\%$ |
| Reptile [18] | 35k | $62.74 \pm 0.37\%$ |
| Ours ($F = 10$) | 28k | $\mathbf{65.09 \pm 0.24}\%$ |

Table 1: Results on Mini-ImageNet in *small* setting. For *small* setting, we use $F = 10$, feature scale rate $= 1.0$, $N = 0$, $B = 5$, inner learning rate $= 0.01$, meta iteration $= 35.2k$, and meta learning rate $= 1.0$.

The above results clearly show the superiority of the search method in that even when the number of parameters are smaller than that of the original work, the corresponding gradient based meta learner achieves a higher performance score. The resulting proto-architecture in this setting is indeed narrowly deep.

| Algorithm | Params | 5-shot 5-way |
|---|---|---|
| MAML + Transduction [8] | 35k | $63.11 \pm 0.92\%$ |
| Reptile + Transduction [18] | 35k | $65.99 \pm 0.58\%$ |
| Ours + Transduction ($F = 64$) | 1,094k | $\mathbf{76.29 \pm 0.38}\%$ |
| Reptile [18] | 35k | $62.74 \pm 0.37\%$ |
| Matching-Nets [25] | - | 55.30% |
| Ravi and Laroche [19] | - | $60.20 \pm 0.71\%$ |
| Prototypical-Nets [24] | - | $68.20 \pm 0.66\%$ |
| Ours ($F = 64$) | 1,094k | $\mathbf{70.87 \pm 0.23}\%$ |

Table 2: Results on Mini-ImageNet in *large* setting. For *large* setting, we use $F = 64$, feature scale rate $= 1.0$, $N = 0$, $B = 5$, inner learning rate $= 0.01$, meta iteration $= 11.2k$, and meta learning rate $= 1.0$.

The result in Table 2 is the final performance of the searched model without the restriction on the number of parameters, which was done for comparison. The resulting gradient based meta learner achieves $76.29\%$ accuracy on the 5-shot 5-way task. This not only drastically improves the previous gradient based meta learner results, but also compares favorably to other state-of-the-art techniques for few shot classification tasks, which are less general.

The results in Table 3 show the case, when the search is conducted through 5-shot 5-way case, but the fine tuning stage goes through the 1-shot 5-way case. We do observe that the search is transferable to some degree, but as expected, the performance is not as high as the case when the search and fine tuning are both conducted through 5-shot 5-way case.

### 4.3 Further Empirical Analysis

The below figure not only shows the top$-1$ cell at $b = 5$, but also shows the progression of how the cell has been established, starting from $b = 3$. One can see that upto $b = 4$, the progression has only chosen to build itself vertically without any horizontal expansion. At $b = 5$, we finally see the last block addition to happen vertically at the top part of the cell.

|  | Algorithm | Params | 1-shot 5-way |
|---|---|---|---|
| *small* settings | Ours + Transduction ($F = 10$) | 28k | $48.35 \pm 0.35\%$ |
|  | Ours ($F = 10$) | 28k | $45.92 \pm 0.27\%$ |
|  | MAML + Transduction [8] | 35k | $48.70 \pm 1.84\%$ |
|  | 1st-order MAML + Transduction [18] | 35k | $47.07 \pm 0.26\%$ |
|  | Reptile + Transduction [18] | 35k | $\mathbf{49.96 \pm 0.32}\%$ |
|  | Reptile [18] | 35k | $47.07 \pm 0.26\%$ |
| *large* settings | Ours + Transduction ($F = 24$) | 157k | $\mathbf{51.76 \pm 0.30}\%$ |
|  | Ours ($F = 24$) | 157k | $49.09 \pm 0.26\%$ |
|  | Matching-Nets [25] | - | 43.60% |
|  | Ravi and Laroche [19] | - | $43.40 \pm 0.77\%$ |
|  | Prototypical-Nets [24] | - | $49.42 \pm 0.78\%$ |

Table 3: Results on $1-$shot $5-$way Mini-ImageNet. For *small* settings, $F = 10$, feature scale rate $= 1.0$, $N = 0$, $B = 5$, inner learning rate$= 0.01$, meta iteration$= 32k$, meta learning rate$= 0.3$. For *large* settings, $F = 24$, feature scale rate $= 1.0$, $N = 0$, $B = 5$, inner learning rate$= 0.01$, meta iteration$= 33k$, and meta learning rate$= 0.235$.

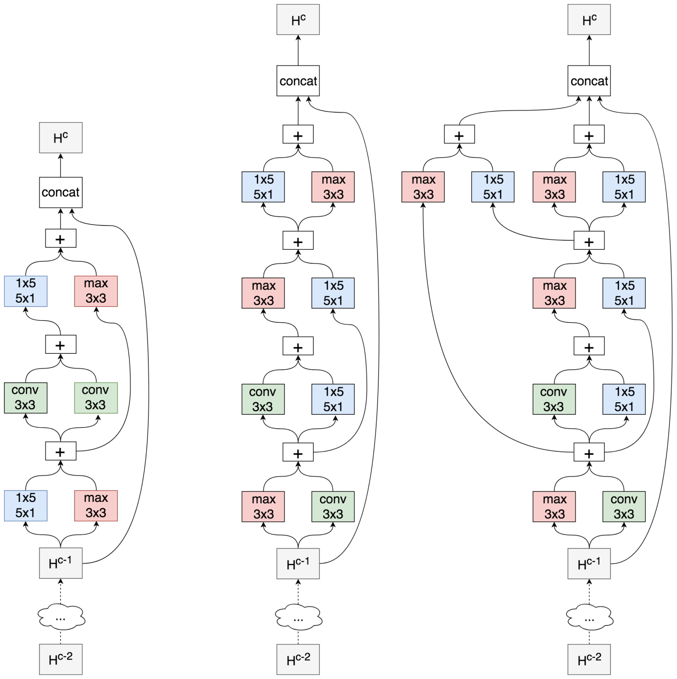
|  | PNAS parameters |
|---|---:|
| Max num blocks(B) | 5 |
| Num filters in first layer(F) | 4, 32 |
| Beam size(K) | 100 |
| Num times to unroll cell (N) | 0, 1 |
| Feature Scale Rate | 1, 2 |
| Surrogate predictor parameters[13] | Cell size $= 100$, Num layers $= 1$ |
|  | Meta-learning parameters |
| Inner iterations | 8 |
| Inner-batch size | 10 |
| Inner Adam learning rate | 0.005 |
| Meta-batch size | 5 |
| Outer step size | 1 |
| Outer iterations | 1k |

Table 4: Hyperparameter combinations used in our search.

The Figure 3 shows the depth statistics($x-$axis is depth) of two executed searches with block progression from $b = 3$ to $b = 5$, which includes total of 600 cells($K = 100$), thus corresponding 600 CNN statistics. This substantially extends the depth-universality analysis of Finn to a much larger scale. One can see that the depths of top 100 CNNs in the search are consistently on the higher end of the $x$-axis. The depth phenomenon that has been suggested by the diagram of the top-1 cell's progression is also present in many other top 100 cells. Interestingly enough, the depth histograms shift to the left, as the CNN unrolling parameter increase from 0 to 1. This might suggest that one only needs sufficient amount of depth as a function of number of parameters in the proto-architecture. Of course, 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 has been motivated by the universality result of Finn. Our intention has been to explore whether or not the neural network search dynamics of the proto-architecture should retain the intuition that ML experts have gathered from the previous iterations of neural network search in the absence of gradient update operators. Our findings indeed show that the search dynamics of the proto-architecture for gradient based meta learners might be quite non-intuitive and much is left to be explored. The searched gradient based meta learners typically contain proto-architectures that are non-intuitive and narrowly deep, unlike the inception-like structures previously reported in the literature. Our searched gradient based meta learners achieve state-of-the art results on the few shot image classification tasks and improve the original MAML results by a fair



(b = 3) (b = 4) (b = 5)

Figure 2: Top-1 cell progression from b=3 to b=5.

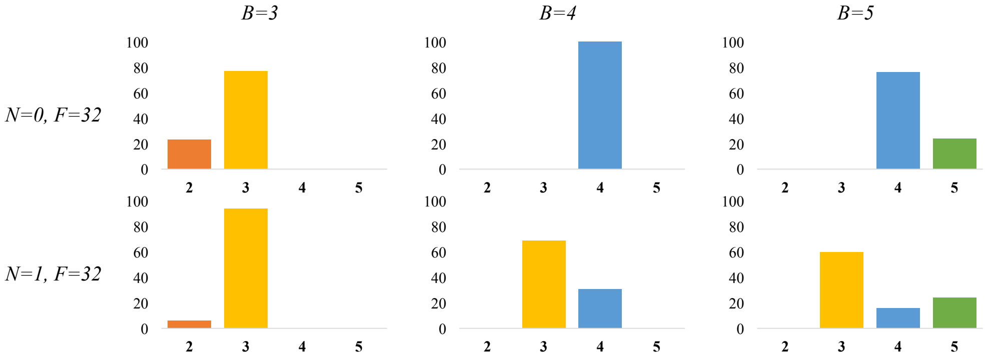


Figure 3: Depth statistics in relation to $B, N, F$

margin. To our best knowledge, this implementation is the first successful AutoML execution in the context of meta learning.

Future work includes exploring the effect of inductive bias in our search method, and further empirical studies on the fact that the universality theory only requires sufficiently deep, but not arbitrarily deep architectures. Also, various kinds of neural architecture search methods should be explored in this precise context of meta learning. It would be quite interesting to see if reinforcement learning type search methods can be formulated with rewards that consider the universality constraints introduced by the presence of the gradient update operators.

## 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$ˆ2$: 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] 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.

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

[15] 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.

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

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

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

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

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

[21] 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.

[22] 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.

[23] 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.

[24] 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.

[25] 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.

[26] 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.

[27] 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.

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

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

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