MetAdapt: Meta-Learned Task-Adaptive Architecture for Few-Shot Classification

Sivan Doveh*[1,2], Eli Schwartz*[1,2],
Chao Xue[1], Rogerio Feris[1], Alex Bronstein[3], Raja Giryes[2], Leonid Karlinsky[1]
[1] IBM Research AI, [2] Tel Aviv University, Israel, [3] Technion, Israel

Abstract

Few-Shot Learning (FSL) is a topic of rapidly growing interest. Typically, in FSL a model is trained on a dataset consisting of many small tasks (meta-tasks) and learns to adapt to novel tasks that it will encounter during test time. This is also referred to as meta-learning. Another topic closely related to meta-learning with a lot of interest in the community is Neural Architecture Search (NAS), automatically finding optimal architecture instead of engineering it manually. In this work we combine these two aspects of meta-learning. So far, meta-learning FSL methods have focused on optimizing parameters of pre-defined network architectures, in order to make them easily adaptable to novel tasks. Moreover, it was observed that, in general, larger architectures perform better than smaller ones up to a certain saturation point (where they start to degrade due to over-fitting). However, little attention has been given to explicitly optimizing the architectures for FSL, nor to an adaptation of the architecture at test time to particular novel tasks. In this work, we propose to employ tools inspired by the Differentiable Neural Architecture Search (D-NAS) literature in order to optimize the architecture for FSL without over-fitting. Additionally, to make the architecture task adaptive, we propose the concept of 'MetAdapt Controller' modules. These modules are added to the model and are meta-trained to predict the optimal network connections for a given novel task. Using the proposed approach we observe state-of-the-art results on two popular few-shot benchmarks: miniImageNet and FC100.

1. Introduction

Recently, there has been a lot of exciting progress in the field of few-shot learning in general, and in few-shot classification (FSC) in particular. A popular method for approaching FSC is meta-learning, or learning-to-learn. In meta-learning, the inputs to both train and test phases are not images, but instead a set of few-shot tasks, $\{T_i\}$, each

* Equal contributors

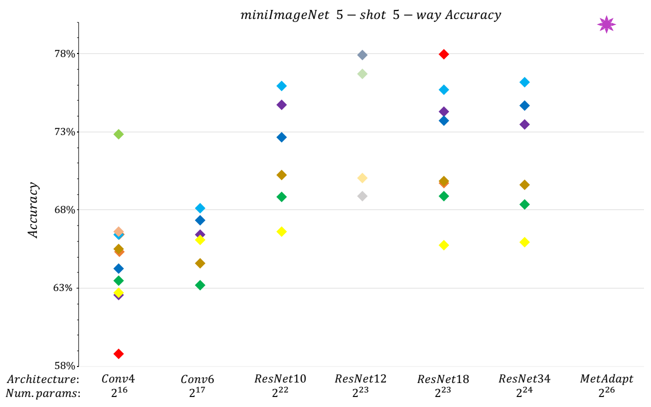

Figure 1. **Accuracy vs. Architecture** as can be seen, the general trend is that larger architectures improve performance. However, when reaching a certain size (ResNet18/34) performance saturates or even degrades due to over-fitting. We propose MetAdapt as a method for finding and training larger models that still improve performance. Markers represent results of different methods with various backbone architectures for the 5-shot / 5-way miniImageNet few-shot benchmark. Same-color markers indicate same method with different architectures.

K-shot / N-way task containing a small amount K (usually 1-5) of labeled support images and some amount of unlabeled query images for each of the N categories of the task. The goal of meta-learning is to find a base model that is easily adapted to the specific task at hand, so that it will generalize well to tasks built from novel unseen categories and fulfill the goal of FSC (see Section 2 for further review).

Many successful meta-learning based approaches have been developed for FSC [12, 9, 5] advancing its state-of-the-art. Besides continuous improvements offered by the FSC methods, some general trends affecting the performance of FSC have become apparent. One of such major factors is the CNN backbone architecture at the basis of all the modern FSC methods. Carefully reviewing and placing on a single chart the test accuracies of top-performing FSC approaches w.r.t. the backbone architecture employed reveals an interesting trend (Figure 1). It is apparent that larger architectures increase FSC performance, up to a certain size, where performance seems to saturate or even degrade. This happens since bigger backbones carry higher risk of over-fitting. It seems the overall performance of the FSC techniques cannot continue to grow by simply expanding the backbone size.

In light of the above, in this paper we set to explore methods for architecture search, their meta-adaptation and optimization for FSC. Neural Architecture Search (NAS) is a very active research field that has contributed significantly to overall improvement of the state of the art in supervised classification. Some of the recent NAS techniques, and in particular Differentiable-NAS (D-NAS), such as DARTS [7], are capable of finding optimal (and transferable) architectures given a particular task using a single GPU in the course of 1-2 days. This is due to incorporating the architecture as an additional set of neural network parameters to be optimized, and solving this optimization using SGD. Due to this use of additional architectural parameters, the training tends to over-fit. D-NAS optimization techniques are especially designed to mitigate over-fitting, making them attractive to extreme situations with the greatest risk of over-fitting, such as in the case of FSC.

So far, D-NAS techniques have been explored mainly in the context of large scale tasks, involving thousands of labeled examples for each class. Very little work has been done on NAS for few-shot. D-NAS in particular, to the best of our knowledge, has not been applied to few-shot problems yet. Meta-adaption of the architecture in task dependent manner to accommodate for novel tasks also has not been explored.

In this work, we build our few-shot task-adaptive architecture search upon a technique from D-NAS (DARTS [7]). Our goal is to learn a neural network where connections are controllable and adapt to the few-shot task with novel categories. Similarly to DARTS, we have a neural network in the form of a Directed Acyclic Graph (DAG), where the nodes are the intermediate feature maps tensors, and edges are operations. Each edge is a weighted sum of operations (with weights summing to 1), each operation is a different preset sequence of layers (convolution, pooling, BatchNorm and non-linearity). The operations set includes the identity-operation and the zero-operation to either keep the representation untouched or cut the connection. To avoid over-fitting, a bi-level (two-fold) optimization is performed where first the operation layers' weights are trained on one fold of the data and then the connections' weights are trained on the other fold.

However, unlike DARTS, our goal is not to learn a one time architecture to be used for all tasks. To be successful at FSC, we need to make our architecture task adaptive so it would be able to quickly rewire for each new target task. To this end, we employ a set of small neural networks, MetAdapt Controllers, responsible for controlling the connections in the DAG given the current task. The MetAdapt Controllers adjust the weights of the different operations, such that if some operations are better for the current task they will get higher weights, thus, effectively modifying the architecture and adapting it to the task.

To summarize, our contributions in this work are as follows: (1) We show that DARTS-like bi-level iterative optimization of layer weights and network connections performs well for few-shot classification without suffering from overfitting due to over-parameterization; (2) We show that adding small neural networks, MetAdapt Controllers, that adapt the connections in the main network according to the given task further (and significantly) improves performance; (3) using the proposed method, we obtain improvements over FSC state-of-the-art on two popular FSC benchmarks: *mini*ImageNet [12] and FC100 [9].

2. Related Work

Few-shot meta-learning These methods are trained on a set of few-shot tasks (also known as 'episodes') instead of a set of object instances, with the motivation to learn a learning strategy that will allow effective adaptation to new such (few-shot) tasks using one or few examples. In Matching Networks [12], a non-parametric k-NN classifier is meta-learned such that for each few-shot task the learned model generates an adaptive embedding space for which the task can be better solved. In [11] the metric (embedding) space is optimized such that in the resulting space different categories form compact and well separated uni-modal distributions around the category 'prototypes'. Another family of meta-learning approaches is the so-called 'gradient based approaches', that try to maximize the 'adaptability', or speed of convergence, of the networks they train to new (few-shot) tasks. In other words, the meta-learned classifiers are optimized to be easily fine-tuned on new few-shot tasks using small training data. The first of these approaches is MAML [3] that due to its universality was later extended through many works such as, Meta-SGD [6] and Meta-Networks [8]. In LEO [10], a MAML like loss is applied not directly on the model parameters, but rather on a latent representation encoding them.

Neural Architecture Search. Over the last few years Neural Architecture Search (NAS) have enabled automatic design of novel architectures that outperformed previous hand-designed architectures in terms of accuracy. Most relevant for this work are the differentiable methods. Notable among them is differentiable architecture search (DARTS) [7]. DARTS relaxes the search space into a continuous one, allowing a differentiable search and managed to search for architecture in just a few GPU days.

3. MetAdapt

In this section we describe the architecture and training procedure for MetAdapt. We introduce the task-adaptable block, it has a graph structure with adaptable connections that can modulate the architecture, adapting it to the

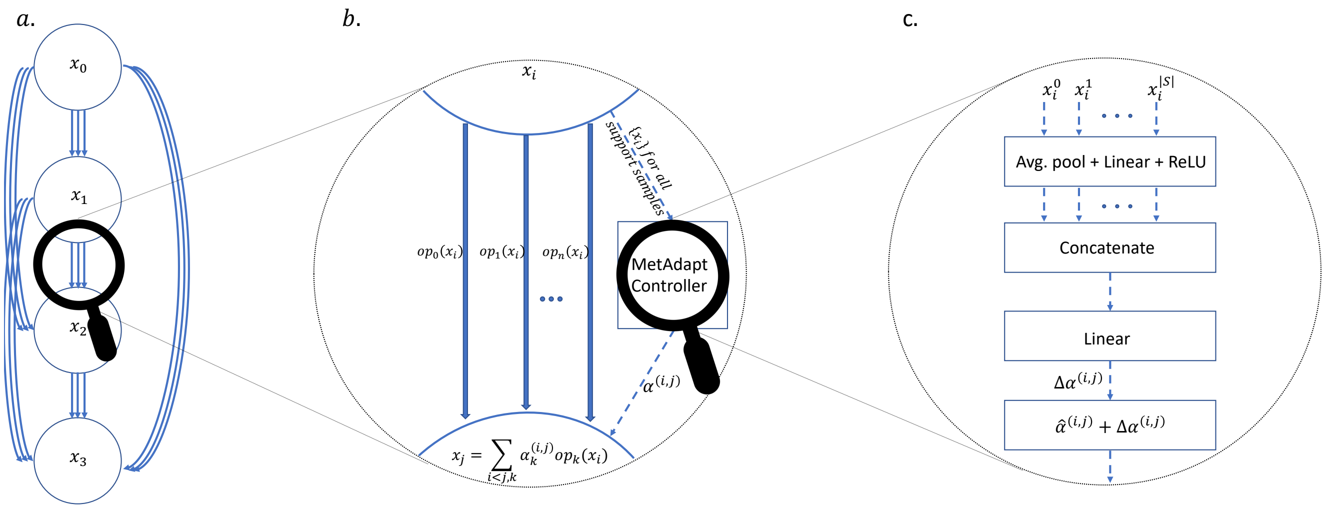
a.
b.
c.
x_0
x_1
x_2
x_3
x_i
$op_0(x_i)$
$op_1(x_i)$
$op_n(x_i)$
$\{x_i\}$ for all support samples
MetAdapt Controller
$\alpha^{(i,j)}$
$x_j = \sum_{i<j,k} \alpha_k^{(i,j)} op_k(x_i)$
x_i^0 x_i^1 $x_i^{|S|}$
Avg. pool + Linear + ReLU
Concatenate
Linear
$\Delta\alpha^{(i,j)}$
$\hat{\alpha}^{(i,j)} + \Delta\alpha^{(i,j)}$

Figure 2. **MetAdapt Block Architecture Illustration.** MetAdapt block has a DAG structure for the network with edges being operations (network layers) and nodes being the feature maps calculated as a convex combination of all edges arriving at a node. The weighting of each edge is optimized in a bi-level manner in parallel with the layer parameters. Additionally, MetAdapt controllers receive as input the concatenated average pooled feature grids for all the support-samples of the current episode and adapts the edges' convex combination weights according to the current task.

few-shot task at hand. We then describe the sub-models, MetAdapt Controllers, that predict the change in connectivity that is needed in the learned graph as a function of the current task. Finally, we describe the training procedure.

3.1. Task-Adaptable Block

The architecture of the adaptable block used in MetAdapt is defined, similarly to DARTS [7], as a Directed Acyclic Graph (DAG). The block is built from feature maps $V = \{x_i\}$ that are linked by mixtures of operations. The input feature map to the block is x_0 and its output is $x_{|V|-1}$. A *Mixed Operation*, $\bar{o}^{(i,j)}$, is defined as

$$\bar{o}^{(i,j)}(x) = \frac{\sum_{o\in\mathcal{O}} exp(\alpha_o^{(i,j)})o(x)}{\sum_{o\in\mathcal{O}} exp(\alpha_o^{(i,j)})}, \tag{1}$$

where $\mathcal{O}$ is a set of the search space operations, $o(x)$ is an operation applied to x, and $\alpha_o^{(i,j)}$ is an optimised coefficient for operation o at edge (i,j). Later, we will describe how αs can be adapted per task (K-shot, N-way episode). The search space operations include 5×5 and 3×3 convolutions (with and without batch-norm); max and average pooling; and the zero-operation and identity-operation that can fully or partially (depending on the corresponding $\alpha_o^{(i,j)}$) cut the connection or make it a residual one (skip-connection). Each feature map x_i in the block is connected to all previous maps by setting it to be:

$$x_i = \sum_{j<i} \bar{o}^{(j,i)}(x_j). \tag{2}$$

The task-adaptive block defined above can be appended to any backbone feature extractor. Potentially, more than one block can be used. We use ResNet9 followed by a single task-adaptive block with 4 nodes ($|V| = 4$) in our experiments, resulting in about 8 times more parameters compared with the original ResNet12 (due to large set of operations on all connections combined). Note that as we use 4 nodes in our block, there exists a single path in our search space that is the regular residual block (ResNet3 block). Figure 2a schematically illustrates the block architecture.

3.2. MetAdapt Controllers

The task-adaptive block is accompanied by a set of MetAdapt Controller modules, one per edge. They are responsible for predicting, given a few-shot task, the best way of adapting the mixing coefficients ($\alpha_o^{(i,j)}$) for the corresponding edge operations. Let $\alpha^{(i,j)}$ be the vector of all $\alpha_o^{(i,j)}$. Let $\hat{\alpha}^{(i,j)}$ be the globally optimized coefficients (optimization process described below), then MetAdapt controllers predict the task-specific residuals $\Delta\alpha^{(i,j)}$, that is the modification required to make to $\hat{\alpha}^{(i,j)}$ for the current task (few-shot episode). Finally,

$$\alpha^{(i,j)} = \hat{\alpha}^{(i,j)} + \Delta\alpha^{(i,j)} \tag{3}$$

are the final task-adapted coefficients used for the *Mixed Operation* calculation as in Equation 1.

The architecture for each MetAdapt Controller, predicting $\Delta\alpha^{(i,j)}$, is as follows. It receives the input feature maps of the corresponding edge x_i, computed for all the support samples of the episode. For a support-set of size S, number of channels D and feature map spatial resolution $M\times M$, the input is a tensor of dimensions (S,D,M,M). We perform global average pooling to obtain a (S,D) tensor, followed by a bottleneck linear layer (with ReLU activation) that operates on each sample individually, to get a $(S, D_{bottleneck})$ size tensor. Then, all support samples representations are concatenated to form a single vector of size $S\cdot D_{bottleneck}$. Finally, another linear layer maps the concatenation of all support-samples to the predicted $\Delta\alpha^{(i,j)}$. The MetAdapt controller architecture and the way it is used in our adaptable block structure are schematically illustrated on Figure 2b+c.

3.3. Training

Replacing simple sequence of convolutional layers with the suggested DAG, with its many layers and parameters, in conventional gradient descent training will result in a larger over-fitting. This is even worse for FSL, where it is harder to achieve generalization due to scarcity of the data and the domain differences between the training and test sets. Researchers have faced the same problem with differentiable architecture search, where the objective is to train a large neural network with weighted connections that are then pruned to form the final chosen architecture.

We follow the solution proposed in DARTS [7], solving a bi-level iterative optimization of the layers' weights w and the coefficients of operations α between the nodes. The training set is split to $train_w$ for weights training and $train_\alpha$ for training the α's. Iteratively optimizing w and α to convergence is prohibitively slow. So, like in DARTS, w is optimized with a standard SGD:

$$w = w - \mu \nabla_w Loss_{train_w}(w, \alpha), \tag{4}$$

where μ is the learning rate. The α's are optimized using SGD with a second-order approximation of the model after convergence of w, by applying:

$$\alpha = \alpha - \eta \nabla_\alpha Loss_{train_\alpha}(w - \mu Loss_{train_w}(w, \alpha), \alpha) \tag{5}$$

where η is the learning rate for α. The 'MetAdapt Controllers' parameters are trained as a final step, with all other parameters freezed, using SGD on the entire training set for a single epoch.

	miniImageNet		FC100	
Method	1-shot	5-shot	1-shot	5-shot
TADAM [9]	58.50	76.70	40.10	56.10
MetaOptNet [5]	62.64	78.63	41.37	55.30
BF3S [4]	62.93	79.87	-	-
Robust-dist [2]	63.06	**80.63**	-	-
MetAdapt (Ours)	**64.80**	**80.64**	**44.83**	**58.47**

Table 1. **Few-shot 5-way accuracy**

4. Experiments

Implementation Details We use the SVM classifier head as suggested in MetaOptNet [5]. We begin with training a ResNet12 backbone on the training set of the relevant dataset for 60 epochs. We then replace the last residual block of the ResNet12 backbone with our DAG task-adaptive block, keeping the first 3 ResNet blocks (ResNet9) fixed and perform the architecture search for 10 epochs. Finally, we train the 'MetAdapt controllers' for a single epoch. Each epoch consists of 8000 episodes with mini-batch of 4 episodes. Following previous works, e.g. [3, 1], we perform test time augmentations and fine-tuning.

Results Tables 1 compares the MetAdapt performance with the state-of-the-art few-shot classification methods that use plain ResNet backbones. We observe improved results for FC100 1-shot (+3.46%) and 5-shot (+3.17%) and also for miniImageNet 1-shot (+1.74%) and similar results for 5-shot. We see that despite having a larger model we do not suffer from severe over-fitting and perform comparably or better than top performing methods.

5. Conclusions

In this work we have proposed MetAdapt, a few-shot learning approach that enables meta-learned network architecture that is adaptive to novel few-shot tasks. The proposed approach effectively applies tools from the Neural Architecture Search (NAS) literature, extended with the concept of 'MetAdapt Controllers', in order to learn adaptive architectures. These tools help mitigate over-fitting to the extremely small data of the few-shot tasks and domain shift between the training set and the test set. We demonstrate that the proposed approach successfully improves state-of-the-art results on two popular few-shot benchmarks, *mini*ImageNet and FC100.

References

[1] Wei-Yu Chen. A Closer Look At Few-Shot Classification. In *ICLR*, pages 1–16, 2018. 4

[2] Nikita Dvornik, Cordelia Schmid, and Julien Mairal. Diversity with cooperation: Ensemble methods for few-shot classification. In *ICCV*, October 2019. 4

[3] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. *arXiv:1703.03400*, 2017. 2, 4

[4] Spyros Gidaris, Andrei Bursuc, Nikos Komodakis, Patrick Pérez, and Matthieu Cord. Boosting few-shot visual learning with self-supervision. In *ICCV*, 2019. 4

[5] Kwonjoon Lee, Subhransu Maji, Avinash Ravichandran, and Stefano Soatto. Meta-learning with differentiable convex optimization. In *CVPR*, 2019. 1, 4

[6] Zhenguo Li, Fengwei Zhou, Fei Chen, and Hang Li. Meta-SGD: Learning to Learn Quickly for Few-Shot Learning. *arXiv:1707.09835*, 2017. 2

[7] Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. In *ICLR*, 2019. 2, 3, 4

[8] Tsendsuren Munkhdalai and Hong Yu. Meta Networks. *arXiv:1703.00837*, 2017. 2

[9] Boris N. Oreshkin, Pau Rodriguez, and Alexandre Lacoste. TADAM: Task dependent adaptive metric for improved few-shot learning. *NeurIPS*, 5 2018. 1, 2, 4

[10] Andrei A. Rusu, Dushyant Rao, Jakub Sygnowski, Oriol Vinyals, Razvan Pascanu, Simon Osindero, and Raia Hadsell. Meta-Learning with Latent Embedding Optimization. In *ICLR*, 7 2018. 2

[11] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical Networks for Few-shot Learning. *NIPS*, 2017. 2

[12] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. Matching Networks for One Shot Learning. *NIPS*, 2016. 1, 2