Reinforced Attention for Few-Shot Learning and Beyond

Jie Hong*[1,2], Pengfei Fang[1,2], Weihao Li[2], Tong Zhang*[3], Christian Simon[1,2],
Mehrtash Harandi[4], Lars Petersson[2]
[1]Australian National University, [2]Data61-CSIRO, [3]EPFL, [4]Monash University
jie.hong@anu.edu.au, pengfei.fang@anu.edu.au, weihao.li@data61.csiro.au, tong.zhang@epfl.ch,
christian.simon@anu.edu.au, mehrtash.harandi@monash.edu, lars.petersson@data61.csiro.au

Abstract

Few-shot learning aims to correctly recognize query samples from unseen classes given a limited number of support samples, often by relying on global embeddings of images. In this paper, we propose to equip the backbone network with an attention agent, which is trained by reinforcement learning. The policy gradient algorithm is employed to train the agent towards adaptively localizing the representative regions on feature maps over time. We further design a reward function based on the prediction of the held-out data, thus helping the attention mechanism to generalize better across the unseen classes. The extensive experiments show, with the help of the reinforced attention, that our embedding network has the capability to progressively generate a more discriminative representation in few-shot learning. Moreover, experiments on the task of image classification also show the effectiveness of the proposed design.

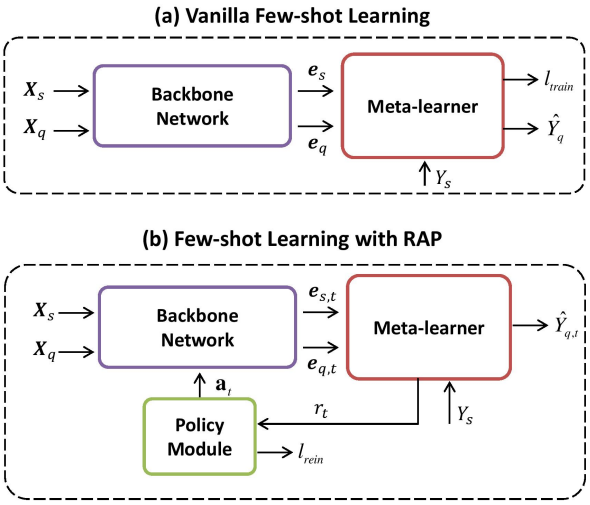

Figure 1. Few-shot learning processes. (a) Vanilla few-shot learning. (b) Few-shot learning with RAP. The learning process has been formulated as a Markov Decision Process (MDP).

1. Introduction

The success of deep learning models rely heavily on a significant amount of labeled data, but the availability of large datasets is still limited due to the labor-intensive data preparation, which motivates the significant interest in few-shot learning [22, 40, 49, 11, 51, 47, 45]. Few-shot learning aims to enable the model to classify unlabeled query examples of unseen classes, utilizing a very small number of labeled support examples. One prominent category of methods is the model-initialization based approach [40, 11, 47]. It temporarily updates the model parameter using support examples via gradient descent steps for the training tasks, and seeks a representation that generalizes well in the testing phase. Another line of work, the metric-learning based methods [22, 49, 51], is based on the complex manipulation of global embeddings learned by the backbone network.

Even though the traditional approaches work well for the task of few-shot learning, they are likely to ignore the spatial information encoded within feature maps, which make the model very sensitive to the background clutter on image examples [62]. To fully make use of the available spatial information, attention-oriented designs are recently developed for few-shot learning [53, 55, 7, 62, 42, 16, 9]. Using word embeddings as auxiliary data, semantics-guided attention modules are proposed to capture the relevant visual features among query samples [55, 7, 62]. In addition to the semantics-guided attention, the sample-guided attention designs are able to further explore the feature relevance between support samples and query samples [53, 42, 16]. While these attention models effectively make class features more representative, they tend to focus too much on designing a complex meta-learner.

In order to address the aforementioned weakness, in this work, we propose a reinforced-attention policy (RAP)

*corresponding author

model for few-shot learning, an attention mechanism trained by reinforcement learning. Specifically, an auxiliary agent is designed to equip the backbone network for computing a series of attention maps which recurrently decide where to enforce or ignore over the feature maps.

Our designed RAP enables the backbone network to identify informative parts of the feature maps of an example, and thus make generated embeddings more discriminative for the few-shot meta-learner. We formulate the feature extraction of examples as a Markov Decision Process (MDP) and optimize RAP in a reinforcement learning setting. Given example images, the agent which progressively refines the attention upon feature maps over time is optimized according to the online feedback, *i.e.*, the computed reward. The specific reward function which incorporates the performance of the meta-learner on the held-out data is designed to guide the agent towards being more generic.

More details are specified in Fig. 1. As illustrated in Fig. 1 (a), the vanilla few-shot learning process can be viewed as a two-component paradigm: the feature extractor, *i.e.* the backbone network and the meta-learner. Given several query examples $\mathbf{X}_q = \{\mathbf{X}_{q,1}, \mathbf{X}_{q,2}, ..., \mathbf{X}_{q,i}, ...\}$ and support examples $\mathbf{X}_s = \{\mathbf{X}_{s,1}, \mathbf{X}_{s,2}, ..., \mathbf{X}_{s,j}, ...\}$ with labels $Y_s = \{Y_{s,1}, Y_{s,2}, ..., Y_{s,j}, ...\}$, the meta-learner works as a classifier to identify the category $\hat{Y}_q = \{\hat{Y}_{q,1}, \hat{Y}_{q,2}, ..., \hat{Y}_{q,i}, ...\}$ of query examples based on embedding features, $\mathbf{e}_q = \{\mathbf{e}_{q,1}, \mathbf{e}_{q,2}, ..., \mathbf{e}_{q,i}, ...\}$ and $\mathbf{e}_s = \{\mathbf{e}_{s,1}, \mathbf{e}_{s,2}, ..., \mathbf{e}_{s,j}, ...\}$, computed by the backbone network. How RAP works on a few-shot learning task is further shown in Fig. 1 (b). Using RAP to equip the backbone network, we convert the few-shot learning into a MDP. The policy module continuously receives the reward r_t as feedback from the meta-learner and gives the action $\mathbf{a}_t$ towards the larger total reward. The embedding $\mathbf{e}_T$ from the last time step T is viewed as the resulting embedding. Hence, instead of $\hat{Y}_q$, we take $\hat{Y}_{q,T}$ as the final prediction. The modification to only the backbone network makes RAP skip the further design of the meta-learner, such that RAP is able to be embedded in most existing few-shot learning baselines.

The contributions of this work can be summarized as follows: i) Our proposed RAP is capable of attending to informative regions of feature maps while avoiding the extra cumbersome meta-learner design. Additionally, most of the few-shot learning baselines can be equipped with RAP, since RAP is essentially a flexible extension specific to the backbone network. ii) We provide a novel solution to train the attention mechanism by using reinforcement learning. Intuitively, the recurrent formulation in a reinforcement learning manner can help the attention mechanism to incrementally locate useful parts of the features due to the characteristic that reinforcement learning is able to substantially learn from experience.

In the experimental part of our few-shot learning, we select several baselines for which RAP agents are trained. In effect, our embedded design pushes the backbone network to produce embeddings which become more discriminative. Aside from few-shot learning, our design is applicable to image classification. The effectiveness is demonstrated via experiments on multiple benchmark datasets.

2. Related Work

Attention Design. The attention mechanism, aimed at attending to the discriminative areas adaptively, have been studied extensively for 2D and 3D visual tasks [17, 60, 56, 36, 69, 10, 39]. Those attention blocks either focus on the channel encoding [17, 39] or spatial context connection [56]. Hu *et al*. first propose the Squeeze-and-Excitation Network (SENet), to weight each slice of the feature map [17]. Later work, termed Convolutional Block Attention Module (CBAM) [60], further employs hybrid spatial and channel features for attention design. In [56], the spatial context connection is established by the visual similarity between the features of the query and the key. Different from existing works, our attention design selects the informative areas by recurrently attending to the feature maps in a reinforcement learning manner. The work in [36, 69] also uses the recurrent model to learn an attention mask. In our work, RAP is expected to boost its generalization power by the learning experience on the held-out data.

Few-shot Learning. Few-shot learning originates from the task in imitation of the human learning ability. Human beings are able to recognize a class given a few instances. Model-Agnostic Meta-Learning (MAML) [11] and Prototypical Network (ProtoNet) [49] are viewed as representatives for model-initialization based methods and metric-learning based methods, respectively. The former quickly adapts the classifier to the target task by learning a sensitive initialization. The latter accurately calculates the prototype of each class, and minimizes the distance in a embedding space between query samples and their corresponding prototype per class. Negative margin loss (Neg-Margin) is introduced to metric-learning based methods in [31]. Some methods, which we call graph-construction based methods, are recently studied by exploring the structural information among examples. Liu *et al.* [32] develop Transductive Propagation Network (TPN) where a graph to propagate labels from labeled samples to unlabeled samples is learnt. Similar to TPN, the framework named Transfer Simplified Graph Convolutional Network (Transfer+SGC) proposed in [18] comes with the use of the graph convolutional network to share the information between the unlabeled examples and labeled examples. Ziko *et al.* [68] propose a graph clustering method (LaplacianShot) which encourages the neighboring query samples to have the same label assignment. Keeping the original meta-learner design, we extend

four baseline models with RAP: MAML, ProtoNet, LaplacianShot and Neg-Margin.

Attention in Few-shot Learning. There are two main categories of attention mechanism applied in few-shot learning. The first one is semantics-guided attention [55, 7, 62, 6]. Guided by word embeddings, Multi-attention Network [55] generates attention maps over visual features of examples, hence the representative parts of an example can be precisely captured. Like Multi-attention Network, an attention generator is developed in [7] to localize the relevant regions in an image with the help of word embeddings. Yan *et al.* [62] develop a dual attention network (STANet) trained with a semantic-aware loss. The other category is sample-guided attention [53, 42, 16]. Matching Network (MatchingNet) [53] uses an attention based on the softmax function to fully specify the prediction of the meta-learner classifier. Similar to the Matching Network, Cross Attention Network (Cross-Attention) proposed by Hou *et al.* [16] aims at modeling the semantic dependency between support samples and query samples. The relevant regions on the query samples are adaptively localized such that the discrimination of embedding features benefit. MatchingNet provides attention on embeddings while Cross-Attention manipulates feature maps. Under the setting of incremental few-shot learning, an attention attractor model [42] is presented to regulate the meta-learning of unseen classes by attending to seen classes. In contrast to these works, our developed RAP attempts to sequentially capture the important information within the feature maps of the backbone network. Its simple extension to the backbone network avoids the complicated meta-learner structure design. In addition, the external semantic data resource is not needed.

Reinforcement Learning on Visual Tasks. Deep learning is widely investigated in the domain of reinforcement learning. By introducing the deep neural network to building the value function or the policy function, the reinforcement learning agent has a powerful capacity to learn the dynamics of the environment with which it interacts. Deep Q Network (DQN) [37] and Deep Deterministic Policy Gradient (DDPG) [30] are the very first attempts to apply the deep neural network in a reinforcement learning setting. Recently, a number of works introduce deep reinforcement learning to the computer vision community. Reinforcement learning is easily utilized in sequential visual tasks, like tracking [64, 3, 41], action recognition [63, 52, 4], visual navigation [57, 67], video summarization [29] and person re-id [25]. The applications are also studied on more classic tasks, such as classification [61], object detection [2, 35], segmentation [1, 66] and pose estimation [24, 44]. However, few works employing reinforcement learning have been done so far in the domain of few-shot learning.

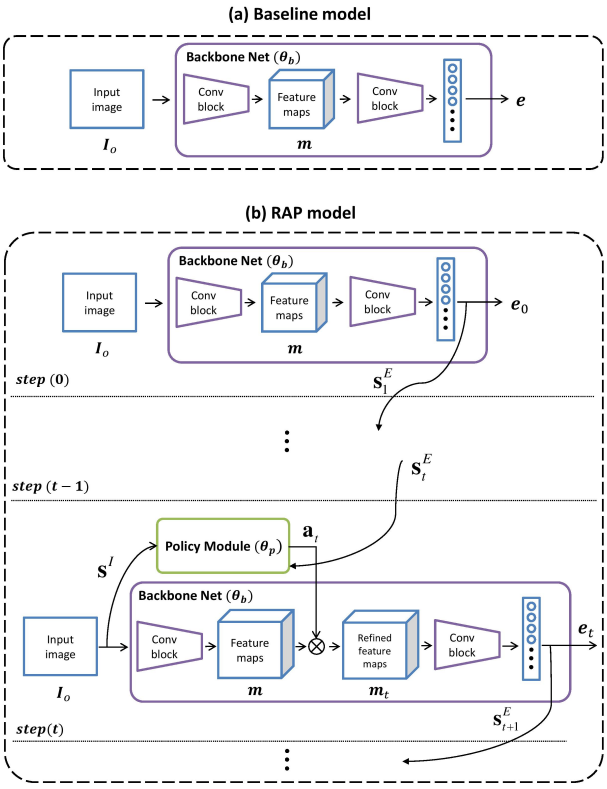

Figure 2. The frameworks of baseline model and RAP model. (a) Baseline model. Given the input image $\mathbf{I}_o$, the backbone network generates the embedding $\mathbf{e}$. (b) RAP model. The recurrent formulation is built where each input image is processed over multiple time-step. The designed policy module is combined with the backbone network to compute the action $\mathbf{a}_t$. The embedding computed after the final time step $\mathbf{e}_T$ is the resulting embedding.

3. Methodology

The overviews of relevant frameworks are depicted in Fig. 2. As shown in Fig. 2 (b), the baseline model (see Fig. 2 (a)) is equipped with a policy module which receives recurrent signals and reinforces the backbone network to attend to the discriminative areas within the feature maps. Conditioned on an example image, the RAP model sequentially seeks the scoring tensor for the corresponding feature maps generated by the backbone network. During this process, we leverage reinforcement learning to train a reward-directed agent, *i.e.*, the policy module (see Fig. 3). Note that how RAP model is applied in few-shot learning is illustrated in Fig. 1.

3.1. Problem Statement

As described in Fig. 2 (b), one execution of RAP can be understood as a MDP. RAP agent, *i.e.*, the policy mod-

ule interacts with the backbone network via multiple time steps. Its behavior can be represented as state-action pairs. At time step t, the agent observes the state $\mathbf{s}_t$, executes the action $\mathbf{a}_t$ and receives the reward r_t towards optimizing the policy. The action at current time step $\mathbf{a}_t$ only depends on the current state $\mathbf{s}_t$ and the next state $\mathbf{s}_{t+1}$ is conditioned on both $\mathbf{s}_t$ and $\mathbf{a}_t$.

To be specific, given feature maps from the backbone network, our RAP module will recurrently attend to informative areas within feature maps, as the RAP agent will determine attention maps until more informative areas in feature maps are found. In our setting, the action $\mathbf{a}_t$ equals the attention maps. The varying factors, such as the input image $\mathbf{I}_o$ and the computed embeddings $\mathbf{e}_t$, are modeled as states $\mathbf{s}_t$ in RL. In the following, we will introduce more details about state, action and reward.

State: Under reinforcement learning settings, we normally encode the observed environment dynamics as states. As depicted in Fig. 2 (b), the dataset and backbone network are modeled as the environment to the agent. The image itself $\mathbf{I}_o \in \mathbb{R}^{H\times W\times 3}$ and the embedding feature vector $\mathbf{e}_t \in \mathbb{R}^{N_{SE}\times 1}$ generated from the last convolutional block are aggregated as the state $\mathbf{s}_t = \{\mathbf{s}^I = \mathbf{I}_o, \mathbf{s}_t^E = \mathbf{e}_t\}$ at time step $t \in \{0, 1, 2, ..., T\}$. T is the total execution times to each image. It is noted that $\mathbf{s}^I$ is fixed in one sequence of processing. We need to address the issue that the high variance exists among tremendous images [36, 34]. For this purpose, we treat the image $\mathbf{I}_o$ as one of states $\mathbf{s}^I$. Hence, RAP is capable of learning the variance among input images. In addition, the setting of the state enables the agent to fuse both the low-level information and the high-level information. The high-level information from $\mathbf{s}_t^E$ indicates the how action affects the backbone network. The low-level information which is encoded within $\mathbf{s}^I$ represents the image variance.

Action: Attention computed at time t for weighting each pixel on feature map are defined as the action $\mathbf{a}_t \in \mathbb{R}^{h\times w\times c}$ where h, w and c are its height, width and length. As illustrated in Fig. 3. the action before reshaping operation $\mathbf{a}_t^v \in \mathbb{R}^{hw\times 1}$ is drawn from the distribution π: $\mathbf{a}_t^v \sim \pi(.|g(\mathbf{s}_t|\boldsymbol{\theta}_p))$ where π is the distribution function parameterized by the computation from policy function g. Then, $\mathbf{a}_t^v \in \mathbb{R}^{hw\times 1}$ is reshaped and duplicated to $\mathbf{a}_t \in \mathbb{R}^{h\times w\times c}$ which has the same size as the operated feature map. The element-wise multiplication is performed to give the refined feature map as follows:

$$\mathbf{m}_t = \mathbf{a}_t \otimes \mathbf{m}. \tag{1}$$

where $\mathbf{m} \in \mathbb{R}^{h\times w\times c}$ ($\mathbf{m}_t \in \mathbb{R}^{h\times w\times c}$) is the feature map before (after) the action being taken at time t. Given the state $\mathbf{s}_t$, the policy function g can decide which pixel along the channel in the feature map should be heightened or weakened at time t.

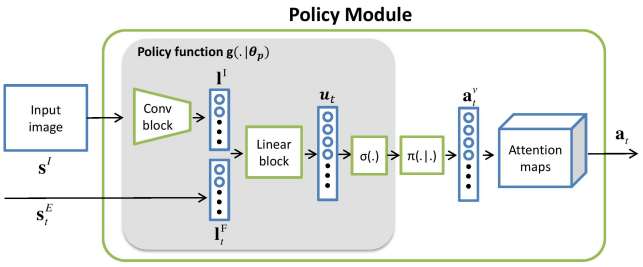

Figure 3. The framework of the policy module. The module essentially maps the state $\mathbf{s}_t$ to the action $\mathbf{a}_t$. $\sigma(.)$ is Sigmoid function.

Reward: After the execution of the action $\mathbf{a}_t$ over feature maps at step t, the agent receives the reward r_t which evaluates $\mathbf{a}_t$. The reward function matters in the sense that it criticizes the direction that the policy is optimized towards. One goal of introducing RAP is to adaptively allocate the informative areas over feature maps via a sequential process. In doing so, one option is to closely monitor and react to the model performance on the held-out data. This can be achieved with the help of the validation set. Hence, the reward function is proposed as

$$\mathbf{r}_t = -\alpha \ell_{val,t} \tag{2}$$

where $\ell_{val,t}$ is the loss built on the validation set at time t and α is the coefficient. In our case, the reward design aims at directing the policy to achieve the higher prediction on the validation data. By explicitly pursuing the better performance on the validation data, RAP is able to have the better generalization ability and therefore pay attention to more useful information on feature maps.

3.2. Policy Design

In this case, we exploit reinforcement learning to learn the policy. The architecture of the policy module is illustrated in Fig. 3. Let $g(\mathbf{s}_t|\boldsymbol{\theta}_p)$ be the policy function which outputs the vector $\mathbf{u}_t \in \mathbb{R}^{hw\times 1}$, representing the parameter value of the distribution function $\pi(.|g(.|\boldsymbol{\theta}_p))$. Under this distribution, $\mathbf{a}_t^v \in \mathbb{R}^{hw\times 1}$ can be chosen.

As shown in Fig. 3, given the image $\mathbf{s}^I$, the conv block, which contains standard conv operations followed by the global average pooling along each channel, is used to compute a N_{l1}-dimensional representation $\mathbf{l}^I \in \mathbb{R}^{N_{lI}\times 1}$. The resulting vector $\mathbf{l}^I$ is then concatenated with another state vector $\mathbf{l}_t^E = \mathbf{s}_t^E \in \mathbb{R}^{N_{lE}\times 1}$ passed from the last time step to form the vector $\mathbf{l}_t = \{\mathbf{l}^I, \mathbf{l}_t^E\}$ which encodes the low-level spatial information from the given example image and the high-level semantic information from the backbone network. Taking $\mathbf{l}_t$ as input, the linear block follows to cal-

culate $\mathbf{u}_t$. The configuration of the linear block is constructed with fully connected (FC) layers. The computed action vector $\mathbf{a}_t^v$ should be reshaped and extended c times to $\mathbf{a}_t \in \mathbb{R}^{h\times w\times c}$ where c indicates the channel number of the feature map to be operated on. We need to avoid the situation that the proposed auxiliary mechanism itself contributes too much to boosting the performance, for which the designed convolution block is usually much shallower than the first conv block of the backbone network before the operation on the feature map.

3.3. Policy Training

The learnable parameter $\boldsymbol{\theta} = \{\boldsymbol{\theta}_b, \boldsymbol{\theta}_p\}$ contains the parameter of the backbone network $\boldsymbol{\theta}_b$ and the policy module $\boldsymbol{\theta}_p$. The objective of policy training is to maximize the sum of the reward $J^\pi = \sum_t^T r_t$ under the distribution π given the input image $\mathbf{s}^I$. We directly write the reinforcement loss ℓ_{rein} as follows:

$$\begin{aligned}\ell_{rein} &= -\frac{1}{NT}\sum_{i=1}^{N} J_i^\pi \\ &= -\frac{1}{NT}\sum_{i=1}^{N}\sum_{t=1}^{T}\log(\pi(\mathbf{u}_{i,t}|\mathbf{s}_{i,t},\boldsymbol{\theta}_p))r_{i,t}\end{aligned} \tag{3}$$

where $\mathbf{u}_t = g(\mathbf{s}_t|\boldsymbol{\theta}_p)$ on the current state $\mathbf{s}_t$ is the predicted parameter (*e.g.*, mean) to the distribution function π in choosing $\mathbf{a}_t^v$ and $\log\pi$ is the log-probability. N is the batch size of images that the model processes in each time step. Using the REINFORCE algorithm [59], the gradient can be expressed as:

$$\nabla_\theta J^\pi = \sum_{t=1}^{T}\nabla_{\boldsymbol{\theta}}\log\big(\pi(\mathbf{u}_t|\mathbf{s}_t;\boldsymbol{\theta}_p)\big)r_t, \tag{4}$$

The gradient $\nabla_\theta J^\pi$ in (4) encourages the optimal parameter to produce a larger reward. The parameter θ_p determines the sensitivity of the policy to the reward r_t. Its motivation is to seek a suitable scale acting on the feedback from the validation. Suppose $r_{i,t}$ becomes zero, the gradient $\nabla_\theta J^\pi$ likewise becomes zero, which would not push the weight to be optimized. One possible concern is that only a reinforcement learning loss ℓ_{rein} that does not include the train data would make the model more firmly rooted in the validation data and thus negatively degrades the accuracy. Incorporating the loss based on the train set can explicitly avoid this problem. It is therefore worth considering a composite form of the total loss as:

$$\ell_{total} = \ell_{rein} + \ell_{train} \tag{5}$$

where ℓ_{train} is the loss built on the train data. When the training is under progress with ℓ_{total}, $\boldsymbol{\theta}$ is free to be updated. The train loss ℓ_{train} based on the train data guarantees the basic performance of the backbone network whereas the reinforce loss ℓ_{rein} based on the validation data enables the policy module to actively identify the informative areas on the feature map of the backbone. When it comes to the inference stage, $\boldsymbol{\theta}$ becomes frozen.

4. Experiment

The proposed design is mainly evaluated on the task of few-shot learning. We assess our RAP model over the *mini*ImageNet [40] and Caltech-UCSD Birds (CUB-200-2011) [54] datasets where it consistently outperforms the corresponding baseline model. Then, to verify that the feasibility of our design is not restricted to few-shot learning alone, experiments on image classification are performed on CIFAR10/100 and STL-10. The standard data augmentations, *i.e.*, random crop and random horizontal flips, are applied to the training samples throughout all experiments. The Adam optimizer [21] is employed to optimize the model weight. Moreover, the trained model that achieves the best validation accuracy is selected to do the inference. All deep models are implemented using PyTorch [38]. We opt for Gaussian function as distribution function π whose mean value is computed by policy function g.

4.1. Datasets

***mini*ImageNet.** For few-shot learning, *mini*ImageNet [40] is the most popular benchmark (60000 images of size 84×84 selected from ImageNet) which is comprised of 64 training classes, 16 validation classes and 20 test classes of images. Every class has 600 images. The size ratio of the train set and validation set is 4:1.

CUB-200-2011. The CUB-200-2011 dataset [54] consists of 11788 images of 84×84 from 200 classes. Following the data preparation from [15], we randomly divide it into 100 train classes, 50 validation classes and 50 test classes.

CIFAR10/100. CIFAR10/100 [23] comprises 60000 images from 10 classes/100 classes. The size of each image is 32 × 32. There are originally 50000 train images and 10000 test images. Our method needs a validation set, thus, we randomly divide the original train set into 40000 train samples and 10000 validation samples.

STL-10. STL-10 [8] contains 10 classes where each class has 500 train images and 800 test images of 96×96. Similar to CIFAR, the original train data is proportionately split into train data and validation data by a ratio of 4:1.

4.2. Few-shot Learning

Training a few-shot learning machine is characterized as an episode training. An episode, which is randomly sam-

pled from training data, is a simple N-way K-shot Q-query classification task. The training process is to classify $Q \times N$ query samples from N classes correctly given K samples per class (*e.g.*, $K = 1$ or 5).

The 1-shot and 5-shot scenarios are examined. Our entire set of experiments are performed using the setting of 5-way and 16-Query. Moreover, we arrange 1200 episodes at random for MAML and ProtoNet (10000 and 600 episodes for LaplacianShot and Neg-Margin, respectively). The final performance is provided averaging the classification accuracy over episodes. For the design details of the policy module, the conv block includes three convolutional layers with 3×3 convolutions where each layer is followed by a batch normalization, ReLU and max pooling. One-layer FC is following as a linear block. The total time step T is kept fixed as 5. The coefficient α in (2) is set to 1e-4. Different architectures of backbone network are assessed including Conv-4 [53], Conv6, ResNet-10, ResNet-12, ResNet-18 [14] and DenseNet-121 [19]. We insert our attention module after the second (or third) conv block of Conv-4 and Conv-6 (or ResNet-10, ResNet-12, ResNet-18 and DenseNet-121).

4.2.1 *mini*ImageNet

In the first part of our few-shot learning experiments, we compare the performance on *mini*ImageNet. The results are presented in Table 1. We equip the classic baselines including MAML and ProtoNet with our proposed RAP. LaplacianShot, an excellent baseline, is also considered. In most cases, our RAP models outperform their corresponding baseline counterparts except for RAP-LaplacianShot with DenseNet-121 in the 5-way 1-shot setting. Another observation is that the choice of baseline do affect the performance improvement against baselines. RAP-MAML and RAP-ProtoNet yield a greater performance improvement than RAP-LaplacianShot. Comparisons with other few-shot learning baselines which exploit attention are provided in Table 2. As we can see from the table, our performance is better than others given the same backbone.

4.2.2 CUB-200-2011

Below, experiments are performed on CUB-200-2011. In this case, besides MAML and ProtoNet and LaplacianShot, we test another strong baseline, Neg-Margin. The obtained accuracy are reported in Table 3. It is obvious that using MAML, RAP models have an accuracy increase of around 2% to 6% against the baseline models whatever the choice of backbone. Similarly, it is also the case for ProtoNet, LaplacianShot and Neg-Margin, that the performance of RAP models are significantly superior to the corresponding baselines. The competitive outcomes prove that the learned policy by reinforcement learning helps the backbone network to localize the informative areas of the feature map and thus generate discriminative embeddings. It should also be emphasized, again, that our RAP is compatible with a number of various few-shot learning baselines.

Model	Backbone	5-way Acc. 1-shot	5-way Acc. 5-shot
MAML [11]	Conv-4	48.70±1.84	63.15±0.92
	Conv-6	50.96±0.92	66.09±0.71
	ResNet-10	54.69±0.89	66.62±0.83
RAP-MAML	Conv-6	52.57±0.61	66.96±0.52
	ResNet-10	**56.13±0.62**	**68.74±0.54**
ProtoNet [49]	Conv-4	49.42±0.78	68.20±0.66
	Conv-6	50.37±0.83	67.33±0.67
	ResNet-10	51.98±0.84	72.64±0.64
RAP-ProtoNet	Conv-6	51.72±0.72	69.18±0.45
	ResNet-10	**53.64±0.60**	**74.54±0.45**
TPN [32]	ResNet-12	59.46	75.65
LTS [28]	ResNet-12	70.1±1.9	78.7±0.8
MetaOptNet-SVM [27]	ResNet-12	64.09±0.62	80.00±0.45
Neg-Margin [31]	ResNet-12	63.85±0.81	81.57±0.56
DSN-MR [46]	ResNet-12	67.09±0.68	81.65±0.69
Su *et al*. [50]	ResNet-18	-	76.60±0.70
Hyperbolic ProtoNet [20]	ResNet-18	59.47±0.20	76.84±0.14
LwoF [12]	WRN	60.06±0.14	76.39±0.11
wDAE-GNN [13]	WRN	61.07±0.15	76.75±0.11
EPNet [43]	WRN	70.74±0.85	84.34±0.53
LaplacianShot [68]	ResNet-10*	69.47±0.20	79.78±0.15
	ResNet-12*	72.29±0.20	82.85±0.14
	DenseNet-121	75.57±0.19	84.72±0.13
RAP-LaplacianShot	ResNet-10	71.34±0.19	81.98±0.14
	ResNet-12	74.29±0.20	84.51±0.13
	DenseNet-121	**75.58±0.20**	**85.63±0.13**

Table 1. Few-shot classification accuracy on *mini*ImageNet. Published results of MAML and ProtoNet with Conv4 are provided in [11] and [49], respectively while their outcomes under Conv-6 and ResNet-10 are reported in [5]. "*" indicates the result is obtained using networks implemented by us.

Model	Backbone	5-way Acc. 1-shot	5-way Acc. 5-shot
Attention Attractor [42]	ResNet-10	54.95±0.30	63.04±0.30
RAP-MAML	ResNet-10	56.13±0.62	68.74±0.54
RAP-ProtoNet	ResNet-10	53.64±0.60	74.54±0.45
RAP-LaplacianShot	ResNet-10	**71.34±0.19**	**81.98±0.14**
STANet [62]	ResNet-12	58.35±0.57	71.07±0.39
Cross-Attention [16]	ResNet-12	67.19±0.55	80.64±0.35
RAP-LaplacianShot	ResNet-12	**74.29±0.20**	**84.51±0.13**

Table 2. Comparison with other attention designs applied to few-shot learning on *mini*ImageNet. Different attention models are compared under the same backbone.

4.3. Image Classification

For the task of image classification, all models are trained with a mini-batch size of 128 for 4000 epochs (2000 epochs on STL10). Pre-trained backbones are used. The learning rate is set to 1e-6. The RAP models execute actions for T=5 times. The results of the classification accuracy on three datasets are listed in Table 4 where RAP models surpass the corresponding baseline models with a healthy mar-

Model	Backbone	5-way Acc. 1-shot	5-shot
MAML [11]	Conv-4	54.73±0.97	75.75±0.76
	Conv-6	66.26±1.05	78.82±0.70
	ResNet-10	70.32±0.99	80.93±0.71
	ResNet-18	68.42±1.07	83.47±0.62
RAP-MAML	Conv-4	61.49±0.70	77.15±0.50
	Conv-6	69.95±0.68	81.48±0.44
	ResNet-10	74.33±0.65	83.29±0.42
	ResNet-18	**75.04±0.70**	**86.07±0.43**
ProtoNet [49]	Conv-4	50.46±0.88	76.39±0.64
	Conv-6	66.36±1.00	82.03±0.59
	ResNet-10	73.22±0.92	85.01±0.52
	ResNet-18	72.99±0.88	86.64±0.51
RAP-ProtoNet	Conv-4	56.71±0.66	78.70±0.44
	Conv-6	67.79±0.66	83.78±0.41
	ResNet-10	**75.17±0.63**	88.29±0.34
	ResNet-18	74.09±0.60	**89.23±0.31**
DeepEMD [65]	ResNet-12	76.65±0.83	88.69±0.50
MatchingNet [53]	ResNet-18	72.36±0.90	83.64±0.60
RelationNet [51]	ResNet-18	67.59±1.02	82.75±0.58
Chen *et al*. [5]	ResNet-18	67.02	83.58
SimpleShot [58]	ResNet-18	70.28	86.37
Manifold [33]	WRN	80.68±0.81	90.85±0.44
EPNet [43]	WRN	87.75±0.70	94.03±0.33
Neg-Margin [31]	ResNet-18	72.66±0.85	89.40±0.43
RAP-Neg-Margin	ResNet-18	**75.37±0.81**	**90.61±0.39**
LaplacianShot [68]	ResNet-18	80.96	88.68
RAP-LaplacianShot	ResNet-18	**83.59±0.18**	**90.77±0.10**

Table 3. Few-shot classification accuracy on CUB-200-2011. The results of MAML and ProtoNet are provided from [5].

gin. We can observe that on each dataset, RAP models have a similar accuracy improvement given different backbone architectures.

4.4. Analysis

Validation Set. RAP models trained by $\ell_{total} = \ell_{train} + \ell_{rein}$ work under the specific setting that ℓ_{train} and ℓ_{rein} are built on train data and validation data, respectively. Any batch is formed by images from either the training set or the validation set. The reward r_t of RAP incorporates the feedback from the validation loss ℓ_{val}, hence, to justify the comparison between RAP models and baseline models, it is necessary to train the baseline models by using both the train set and the validation set. Thus, the loss $\ell_{total} = \ell_{train}$ to the baseline model is replaced by $\ell_{total} = \ell_{train} + \ell_{val}$. As reported in Table 5, even under the same data settings, RAP models still beat baseline models with different backbones. Another fact is that RAP models are sensitive to α. ℓ_{rein} does contribute even if α is set to 1e-4. We thus have to choose α carefully since α decides what extend validation data should be leveraged (see Table 6).

Attention Design. In order to further verify the advantage of our method, we compare RAP against two conventional attention designs, SENet and CBAM. The policy module of RAP is substituted by different attention mechanisms, but the remaining settings are kept the same. As can be seen in Table 7, models equipped with SENet and CBAM are shown to outperform the corresponding baseline models. However, their performance do not exceed our RAP models except that they have a nearly identical accuracy in the 5-way 1-shot classification using the Conv-4 backbone.

Dataset	Model	Backbone	Accuracy (%)
CIFAR10	Baseline	LeNet*	75.94
		VGG-16*	91.48
		ResNet-18*	92.68
		ResNet-50*	93.02
	RAP	LeNet	77.15 (↑ 1.21)
		VGG-16	92.05 (↑ 0.57)
		ResNet-18	93.25 (↑ 0.57)
		ResNet-50	**93.56 (↑ 0.54)**
CIFAR100	Baseline	LeNet*	41.63
		VGG-16*	67.23
		ResNet-18*	72.56
		ResNet-50*	74.22
	RAP	LeNet	43.02 (↑ 1.39)
		VGG16	68.45 (↑ 1.22)
		ResNet-18	73.76 (↑ 1.20)
		ResNet-50	**75.36 (↑ 1.14)**
STL10	Baseline	VGG-16*	74.43
		ResNet-18*	78.21
		ResNet-34*	76.31
	RAP	VGG-16	76.16 (↑ 1.73)
		ResNet-18	**80.15 (↑ 1.94)**
		ResNet-34	77.74 (↑ 1.43)

Table 4. Image classification accuracy of all models on CIFAR10/CIFAR100 and STL10. LeNet [26], VGG-16 [48], ResNet-18, ResNet34 and ResNet-50 are chosen as backbones. Baseline models are trained on both train set and validation set. "*" indicates the result is obtained using networks implemented by us.

Model	Backbone	5-way Acc. 1-shot	5-shot
MAML [11]	Conv-4*	57.55±0.67	75.61±0.49
	Conv-6*	65.07±0.72	79.89±0.48
RAP-MAML	Conv-4	61.49±0.70	77.15±0.50
	Conv-6	69.95±0.68	81.48±0.44
ProtoNet [49]	Conv-4*	51.61±0.65	75.29±0.48
	Conv-6*	64.72±0.72	81.35±0.43
RAP-ProtoNet	Conv-4	56.71±0.66	78.70±0.44
	Conv-6	67.79±0.66	83.78±0.41

Table 5. Fair comparisons on CUB-200-2011 where baseline models are trained on both the train set and validation set like RAP models. "*" indicates the result is obtained using networks implemented by us.

Recurrent Process. A natural question is whether the recurrent modeling of reinforcement learning contributes to the increased performance. To provide evidence that the

Model	α	Backbone	5-way Acc. 1-shot	5-shot
RAP-ProtoNet	0	Conv-4	53.83±0.66	77.10±0.46
	1e-4	Conv-4	56.71±0.66	78.70±0.44
	1e-2	Conv-4	44.68±0.61	71.02±0.49
	1	Conv-4	44.90±0.63	65.07±0.50
RAP-ProtoNet	0	ResNet-10	72.71±0.61	86.68±0.33
	1e-4	ResNet-10	75.17±0.63	88.29±0.34
	1e-2	ResNet-10	72.61±0.64	85.95±0.37
	1	ResNet-10	66.33±0.64	80.47±0.42

Table 6. The results of 5-way classification of RAP-ProtoNet on CUB-200-2011 using different α.

	Backbone	5-way Acc. 1-shot	5-shot
SENet [17]	Conv-4*	55.18±0.61	77.07±0.46
	ResNet-10*	73.89±0.57	87.32±0.38
CBAM [60]	Conv-4*	55.52±0.55	76.80±0.34
	ResNet-10*	72.43±0.65	86.70±0.48
RAP	Conv-4	56.71±0.66	78.70±0.44
	ResNet-10	75.17±0.63	88.29±0.34

Table 7. Few-shot classification accuracy using ProtoNet as a baseline with different regular attention designs on CUB-200-2011. "*" indicates the result is obtained using networks implemented by us.

behavior of the sequential process has a positive effect, we perform the experiments by varying the total time step T. Under the same few-shot learning scenario, a 2-step model and a 5-step model are compared. We can see from Table 8 that the performance has increased nearly 1% to 2% with the greater number of time steps. The results suggest that our reinforced attention continuously adapt to important information of the feature maps over time. The fact that classification accuracy gains as more time steps of actions are executed reinforces the contribution and improvement attained by the learned policy. For image classification, we take LeNet on CIFAR10 and VGG-16 on STL10 as instantiations. The curves of the test accuracy are plotted in Fig 4. Interestingly, the accuracy of the 2-step RAP model increases as quickly as the 5-step model in the initial epochs, but it is gradually surpassed by the 5-step model. In our setting, the reward is closely linked to the classification accuracy. Fig. 4 also suggests that the learned policy maximizes the reward. The experiments strongly indicate that reinforcement learning can effectively learn from experience in this context, and that it benefits from a greater number of iterations.

5. Conclusion

In this paper, we propose RAP, a novel attention design which is able to fit nicely into most few-shot learning baselines to refine the feature map towards a more discriminative

Model	T	Backbone	5-way Acc. 1-shot	5-shot
RAP-MAML	2	Conv-4	60.15±0.71	75.76±0.50
	5	Conv-4	61.49±0.70	77.15±0.50
	2	Conv-6	68.33±0.70	80.43±0.45
	5	Conv-6	69.95±0.68	81.48±0.44
RAP-ProtoNet	2	Conv-4	54.01±0.65	78.12±0.45
	5	Conv-4	56.71±0.66	78.70±0.44
	2	Conv-6	66.27±0.68	82.58±0.42
	5	Conv-6	67.79±0.66	83.78±0.41

Table 8. Comparisons among RAP models with different time step T on CUB-200-2011.

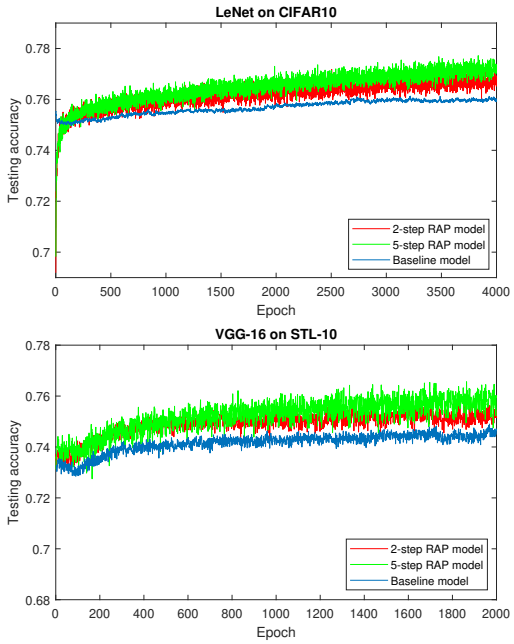

Figure 4. The recorded accuracy of image classification of LeNet on CIFAR10 and VGG-16 on STL10.

representation. To the best of our knowledge, our approach is the first attempt to address few-shot learning via applying an attention mechanism trained by reinforcement learning. We show that the reward design based on validation data enables the model to learn a more diverse distribution. More importantly, the key idea underlying our method is that the recurrent formulation of reinforcement learning has the nature of locating the available information from experience over time and thus enables the agent to make more accurate decisions.

Acknowledgements

Tong Zhang is partly supported by the Swiss National Science Foundation via the Sinergia grant CRSII5-180359. We would like to thank the anonymous reviewers for their useful feedbacks.

References

[1] Nikita Araslanov, Constantin A Rothkopf, and Stefan Roth. Actor-critic instance segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8237–8246, 2019. 3

[2] Juan C Caicedo and Svetlana Lazebnik. Active object localization with deep reinforcement learning. In *Proceedings of the IEEE international conference on computer vision*, pages 2488–2496, 2015. 3

[3] Boyu Chen, Dong Wang, Peixia Li, Shuang Wang, and Huchuan Lu. Real-time'actor-critic'tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 318–334, 2018. 3

[4] Lei Chen, Jiwen Lu, Zhanjie Song, and Jie Zhou. Part-activated deep reinforcement learning for action prediction. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 421–436, 2018. 3

[5] Wei-Yu Chen, Yen-Cheng Liu, Zsolt Kira, Yu-Chiang Wang, and Jia-Bin Huang. A closer look at few-shot classification. In *International Conference on Learning Representations*, 2019. 6, 7

[6] Ali Cheraghian, Shafin Rahman, Pengfei Fang, Soumava Kumar Roy, Lars Petersson, and Mehrtash Harandi. Semantic-aware knowledge distillation for few-shot class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021. 3

[7] Wen-Hsuan Chu and Yu-Chiang Frank Wang. Learning semantics-guided visual attention for few-shot image classification. In *2018 25th IEEE International Conference on Image Processing (ICIP)*, pages 2979–2983. IEEE, 2018. 1, 3

[8] Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 215–223, 2011. 5

[9] Qi Fan, Wei Zhuo, Chi-Keung Tang, and Yu-Wing Tai. Few-shot object detection with attention-rpn and multi-relation detector. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4013–4022, 2020. 1

[10] Pengfei Fang, Jieming Zhou, Soumava Kumar Roy, Lars Petersson, and Mehrtash Harandi. Bilinear attention networks for person retrieval. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019. 2

[11] 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-Volume 70*, pages 1126–1135. JMLR. org, 2017. 1, 2, 6, 7

[12] Spyros Gidaris and Nikos Komodakis. Dynamic few-shot visual learning without forgetting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4367–4375, 2018. 6

[13] Spyros Gidaris and Nikos Komodakis. Generating classification weights with gnn denoising autoencoders for few-shot learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 21–30, 2019. 6

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 6

[15] Nathan Hilliard, Lawrence Phillips, Scott Howland, Artëm Yankov, Courtney D Corley, and Nathan O Hodas. Few-shot learning with metric-agnostic conditional embeddings. *arXiv preprint arXiv:1802.04376*, 2018. 5

[16] Ruibing Hou, Hong Chang, MA Bingpeng, Shiguang Shan, and Xilin Chen. Cross attention network for few-shot classification. In *Advances in Neural Information Processing Systems*, pages 4005–4016, 2019. 1, 3, 6

[17] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018. 2, 8

[18] Yuqing Hu, Vincent Gripon, and Stéphane Pateux. Exploiting unsupervised inputs for accurate few-shot classification. *arXiv preprint arXiv:2001.09849*, 2020. 2

[19] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017. 6

[20] Valentin Khrulkov, Leyla Mirvakhabova, Evgeniya Ustinova, Ivan Oseledets, and Victor Lempitsky. Hyperbolic image embeddings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6418–6428, 2020. 6

[21] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. iclr (2015). *arXiv preprint arXiv:1412.6980*, 9, 2015. 5

[22] Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. Siamese neural networks for one-shot image recognition. In *ICML deep learning workshop*, volume 2. Lille, 2015. 1

[23] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 5

[24] Alexander Krull, Eric Brachmann, Sebastian Nowozin, Frank Michel, Jamie Shotton, and Carsten Rother. Poseagent: Budget-constrained 6d object pose estimation via reinforcement learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6702–6710, 2017. 3

[25] Xu Lan, Hanxiao Wang, Shaogang Gong, and Xiatian Zhu. Deep reinforcement learning attention selection for person re-identification. *arXiv preprint arXiv:1707.02785*, 2017. 3

[26] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 7

[27] Kwonjoon Lee, Subhransu Maji, Avinash Ravichandran, and Stefano Soatto. Meta-learning with differentiable convex optimization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 10657–10665, 2019. 6

[28] Xinzhe Li, Qianru Sun, Yaoyao Liu, Qin Zhou, Shibao Zheng, Tat-Seng Chua, and Bernt Schiele. Learning to self-

train for semi-supervised few-shot classification. In *Advances in Neural Information Processing Systems*, pages 10276–10286, 2019. 6

[29] Yandong Li, Liqiang Wang, Tianbao Yang, and Boqing Gong. How local is the local diversity? reinforcing sequential determinantal point processes with dynamic ground sets for supervised video summarization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 151–167, 2018. 3

[30] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. 2016. 3

[31] Bin Liu, Yue Cao, Yutong Lin, Qi Li, Zheng Zhang, Mingsheng Long, and Han Hu. Negative margin matters: Understanding margin in few-shot classification. *arXiv preprint arXiv:2003.12060*, 2020. 2, 6, 7

[32] Yanbin Liu, Juho Lee, Minseop Park, Saehoon Kim, Eunho Yang, Sung Ju Hwang, and Yi Yang. Learning to propagate labels: Transductive propagation network for few-shot learning. *arXiv preprint arXiv:1805.10002*, 2018. 2, 6

[33] Puneet Mangla, Nupur Kumari, Abhishek Sinha, Mayank Singh, Balaji Krishnamurthy, and Vineeth N Balasubramanian. Charting the right manifold: Manifold mixup for few-shot learning. In *The IEEE Winter Conference on Applications of Computer Vision*, pages 2218–2227, 2020. 7

[34] Hongzi Mao, Shaileshh Bojja Venkatakrishnan, Malte Schwarzkopf, and Mohammad Alizadeh. Variance reduction for reinforcement learning in input-driven environments. *arXiv preprint arXiv:1807.02264*, 2018. 4

[35] Stefan Mathe, Aleksis Pirinen, and Cristian Sminchisescu. Reinforcement learning for visual object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2894–2902, 2016. 3

[36] Volodymyr Mnih, Nicolas Heess, Alex Graves, et al. Recurrent models of visual attention. In *Advances in neural information processing systems*, pages 2204–2212, 2014. 2, 4

[37] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015. 3

[38] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems*, pages 8026–8037, 2019. 5

[39] Shi Qiu, Saeed Anwar, and Nick Barnes. Geometric back-projection network for point cloud classification. *arXiv preprint arXiv:1911.12885*, 2019. 2

[40] Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017. 1, 5

[41] Liangliang Ren, Xin Yuan, Jiwen Lu, Ming Yang, and Jie Zhou. Deep reinforcement learning with iterative shift for visual tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 684–700, 2018. 3

[42] Mengye Ren, Renjie Liao, Ethan Fetaya, and Richard Zemel. Incremental few-shot learning with attention attractor networks. In *Advances in Neural Information Processing Systems*, pages 5276–5286, 2019. 1, 3, 6

[43] Pau Rodríguez, Issam Laradji, Alexandre Drouin, and Alexandre Lacoste. Embedding propagation: Smoother manifold for few-shot classification. *arXiv preprint arXiv:2003.04151*, 2020. 6, 7

[44] Jianzhun Shao, Yuhang Jiang, Gu Wang, Zhigang Li, and Xiangyang Ji. Pfrl: Pose-free reinforcement learning for 6d pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11454–11463, 2020. 3

[45] Christian Simon, Piotr Koniusz, and Mehrtash Harandi. On learning the geodesic path for incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021. 1

[46] Christian Simon, Piotr Koniusz, Richard Nock, and Mehrtash Harandi. Adaptive subspaces for few-shot learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4136–4145, 2020. 6

[47] Christian Simon, Piotr Koniusz, Richard Nock, and Mehrtash Harandi. On modulating the gradient for meta-learning. In *European Conference on Computer Vision*, pages 556–572. Springer, 2020. 1

[48] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 7

[49] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *Advances in neural information processing systems*, pages 4077–4087, 2017. 1, 2, 6, 7

[50] Jong-Chyi Su, Subhransu Maji, and Bharath Hariharan. When does self-supervision improve few-shot learning? *arXiv preprint arXiv:1910.03560*, 2019. 6

[51] Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy M Hospedales. Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1199–1208, 2018. 1, 7

[52] Yansong Tang, Yi Tian, Jiwen Lu, Peiyang Li, and Jie Zhou. Deep progressive reinforcement learning for skeleton-based action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5323–5332, 2018. 3

[53] 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. 1, 3, 6, 7

[54] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011. 5

[55] Peng Wang, Lingqiao Liu, Chunhua Shen, Zi Huang, Anton van den Hengel, and Heng Tao Shen. Multi-attention network for one shot learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2721–2729, 2017. 1, 3

[56] Xiaolong Wang, Girshick, Abhinav Gupta, and Kaiming He. Non-local Neural Networks. In *The IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 2

[57] Xin Wang, Qiuyuan Huang, Asli Celikyilmaz, Jianfeng Gao, Dinghan Shen, Yuan-Fang Wang, William Yang Wang, and Lei Zhang. Reinforced cross-modal matching and self-supervised imitation learning for vision-language navigation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6629–6638, 2019. 3

[58] Yan Wang, Wei-Lun Chao, Kilian Q Weinberger, and Laurens van der Maaten. Simpleshot: Revisiting nearest-neighbor classification for few-shot learning. *arXiv preprint arXiv:1911.04623*, 2019. 7

[59] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992. 5

[60] Sanghyun Woo, Jongchan Park, Joon-Young Lee, and In So Kweon. Cbam: Convolutional block attention module. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 3–19, 2018. 2, 8

[61] Zhen Xu, Andrew M Dai, Jonas Kemp, and Luke Metz. Learning an adaptive learning rate schedule. *arXiv preprint arXiv:1909.09712*, 2019. 3

[62] Shipeng Yan, Songyang Zhang, Xuming He, et al. A dual attention network with semantic embedding for few-shot learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 9079–9086, 2019. 1, 3, 6

[63] Serena Yeung, Olga Russakovsky, Greg Mori, and Li Fei-Fei. End-to-end learning of action detection from frame glimpses in videos. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2678–2687, 2016. 3

[64] Sangdoo Yun, Jongwon Choi, Youngjoon Yoo, Kimin Yun, and Jin Young Choi. Action-decision networks for visual tracking with deep reinforcement learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2711–2720, 2017. 3

[65] Chi Zhang, Yujun Cai, Guosheng Lin, and Chunhua Shen. Deepemd: Few-shot image classification with differentiable earth mover's distance and structured classifiers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12203–12213, 2020. 7

[66] Yizhou Zhou, Xiaoyan Sun, Zheng-Jun Zha, and Wenjun Zeng. Context-reinforced semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4046–4055, 2019. 3

[67] Fengda Zhu, Linchao Zhu, and Yi Yang. Sim-real joint reinforcement transfer for 3d indoor navigation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11388–11397, 2019. 3

[68] Imtiaz Masud Ziko, Jose Dolz, Eric Granger, and Ismail Ben Ayed. Laplacian regularized few-shot learning. *arXiv preprint arXiv:2006.15486*, 2020. 2, 6, 7

[69] Daniel Zoran, Mike Chrzanowski, Po-Sen Huang, Sven Gowal, Alex Mott, and Pushmeet Kohli. Towards robust image classification using sequential attention models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9483–9492, 2020. 2