Rapid Learning or Feature Reuse? Towards Understanding the Effectiveness of MAML

Aniruddh Raghu, Maithra Raghu, Samy Bengio, Oriol Vinyals

https://arxiv.org/abs/1909.09157

Alex Fedorov

June 26, 2020

Agenda

Rewind

- Few-Shot Learning, MAML

Question

- Rapid Learning vs Feature Reuse

Solution

- ANIL

Evaluation

Beyond

Rewind

Few-Shot Learning

- **Few-Shot** classification is an instantiation of **meta-learning** in the field of supervised learning. The dataset D is often split into two parts, a support set for learning S and a prediction set B for training or testing, $D = \langle S, B \rangle$.
- K-shot N-class classification task: the support set contains K labelled examples for each of N classes.

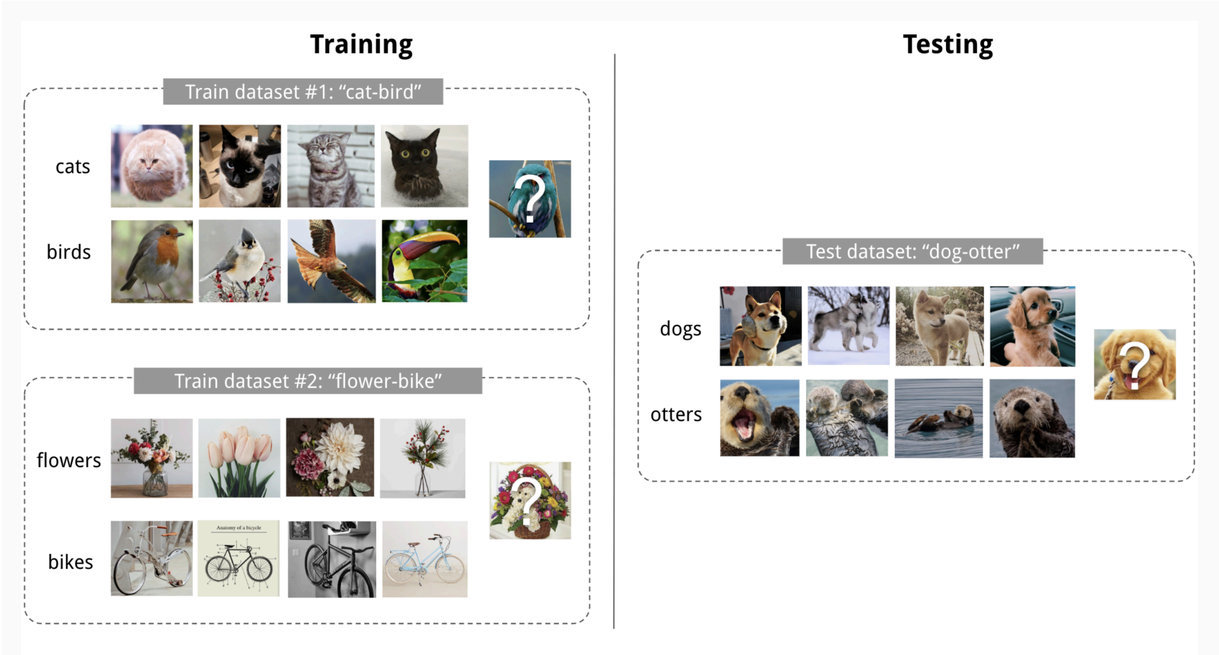
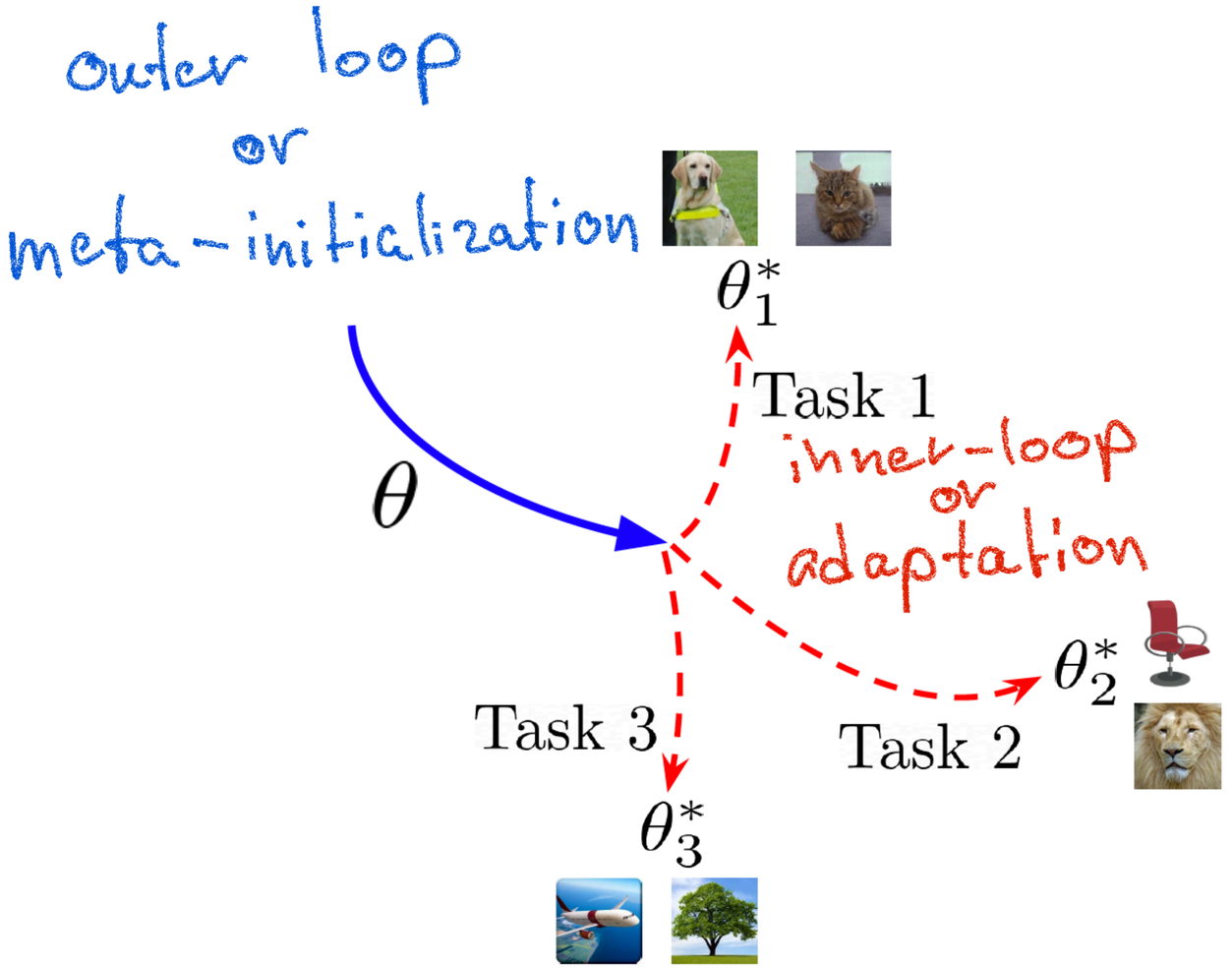


Fig. 1. An example of 4-shot 2-class image classification. (Image thumbnails are from Pinterest)

https://lilianweng.github.io/lil-log/2018/11/30/meta-learning.html

MAML
Rewinded

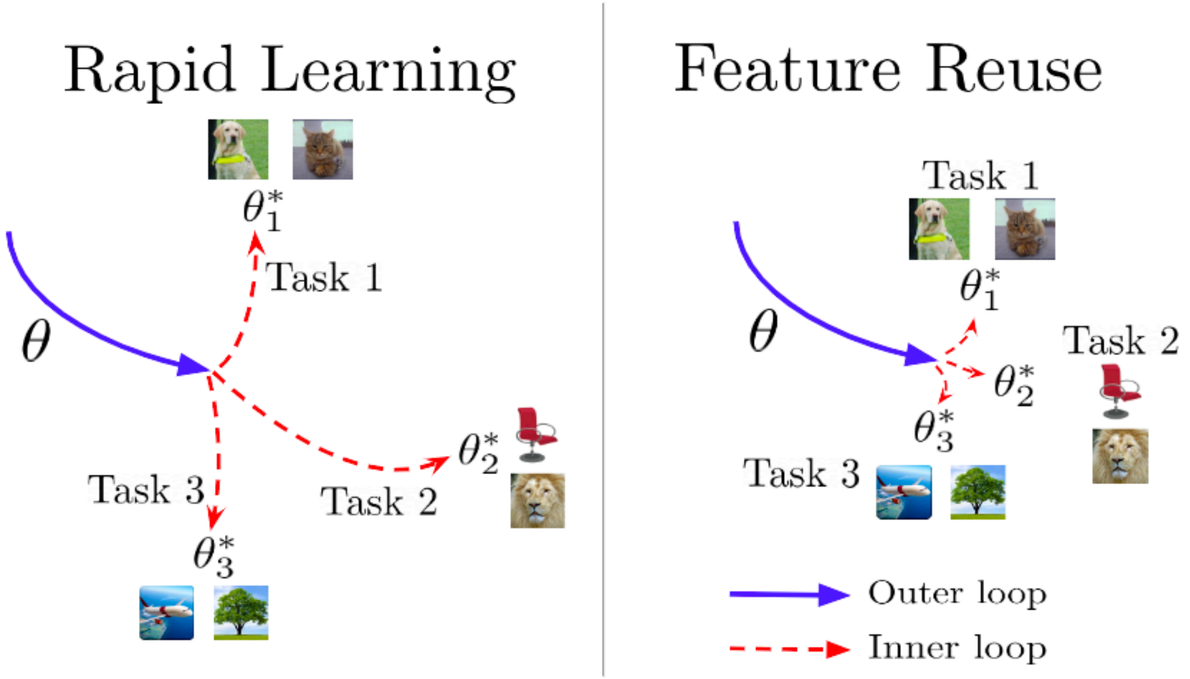


http://learn2learn.net/tutorials/anil_tutorial/ANIL_tutorial/

Question

Rapid Learning vs. Feature Reuse

- In **rapid learning**, the meta-initialization in the outer loop results in a parameter setting that is favorable for fast learning
  - Significant adaptation to new tasks can rapidly take place in the inner loop.
- In **feature reuse**, the meta-initialization already contains useful features that can be reused
  - Little adaptation on the parameters is required in the inner loop.



http://learn2learn.net/tutorials/anil_tutorial/ANIL_tutorial/

Matching Networks by Vinyals et al. (2016)

https://arxiv.org/abs/1606.04080

$$S = \{(x_i, y_i)\}_{i=1}^{k}$$

Encode each member

$$\hat{y} = \sum_{i=1}^{k} a(\hat{x}, x_i)\, y_i$$

$$a(\hat{x}, x_i) = \frac{e^{s(f(\hat{x}), g(x_i))}}{\sum_{j=1}^{k} e^{s(f(\hat{x}), g(x_j))}}$$

softmax

cosine distance

Encode Support Set S

$$f(\hat{x}, S) = \text{attn LSTM}(f'(\hat{x}), g(S), k)$$

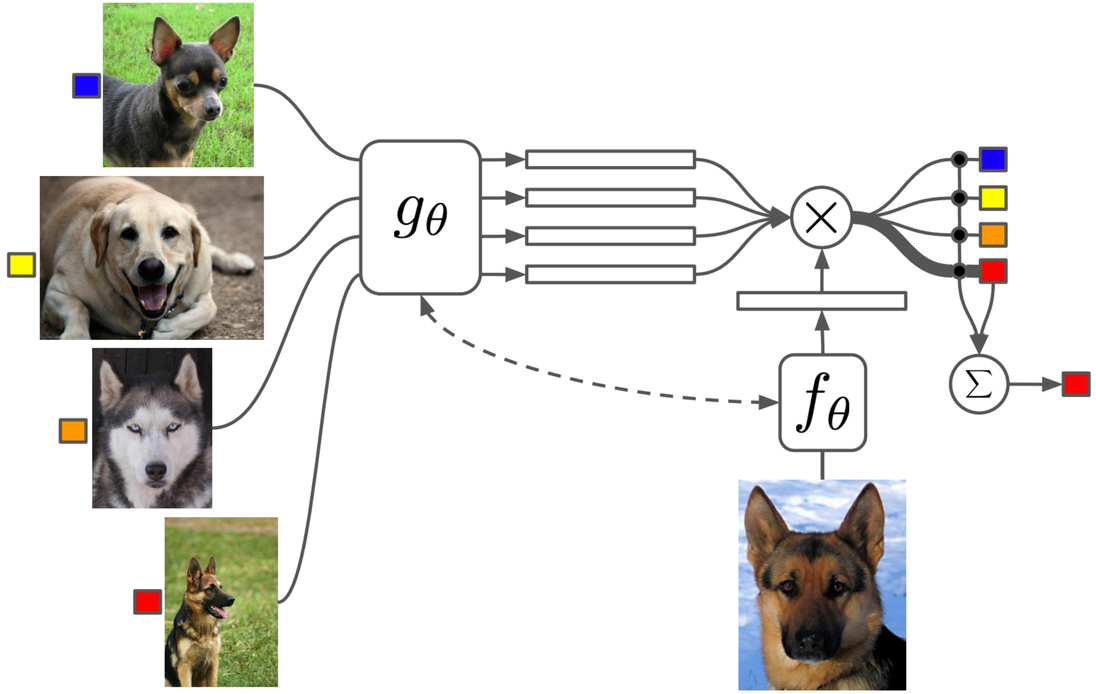


Figure 1: Matching Networks architecture

$$\theta = \arg\max_{\theta} \mathbb{E}_{L \sim T}\Big[\mathbb{E}_{S \sim L, B \sim L}\Big[\sum_{(x,y) \in B} \log P_\theta(y \mid x, S) \Big] \Big]$$

Is MAML's efficacy mostly due Rapid Learning or Feature Reuse?

- We can answer this question by accessing layers of the NN
- Earlier layers vs head layer
- Head layer have to change more
  - Since in each task there are different classes
- To study this question
  - Access Few-Shot learning performance by freezing layers during adaptation
  - Representational similarity tools: **CCA** (https://arxiv.org/abs/1706.05806, https://arxiv.org/abs/1806.05759, tutorial (https://github.com/google/svcca)) and **CKA** (https://arxiv.org/abs/1905.00414)

$$\max_{w,s} \frac{\langle w^T A, s^T B \rangle}{||w^T A|| \cdot ||s^T B||} = \max_{w,s} \frac{w^T \Sigma_{AB} s}{\sqrt{w^T \Sigma_{AA} w} \cdot \sqrt{s^T \Sigma_{BB} s}}$$

Freezing experiment

Freeze layers	MiniImageNet-5way-1shot	MiniImageNet-5way-5shot
None	46.9 ± 0.2	63.1 ± 0.4
1	46.5 ± 0.3	63.0 ± 0.6
1,2	46.4 ± 0.4	62.6 ± 0.6
1,2,3	46.3 ± 0.4	61.2 ± 0.5
1,2,3,4	46.3 ± 0.4	61.0 ± 0.6

Table 1: **Freezing successive layers (preventing inner loop adaptation) does not affect accuracy, supporting feature reuse.** To test the amount of feature reuse happening in the inner loop adaptation, we test the accuracy of the model when we freeze (prevent inner loop adaptation) a contiguous block of layers at test time. We find that freezing even all four convolutional layers of the network (all layers except the network head) hardly affects accuracy. This strongly supports the feature reuse hypothesis: layers don't have to change rapidly at adaptation time; they already contain good features from the meta-initialization.

Representational Similarity

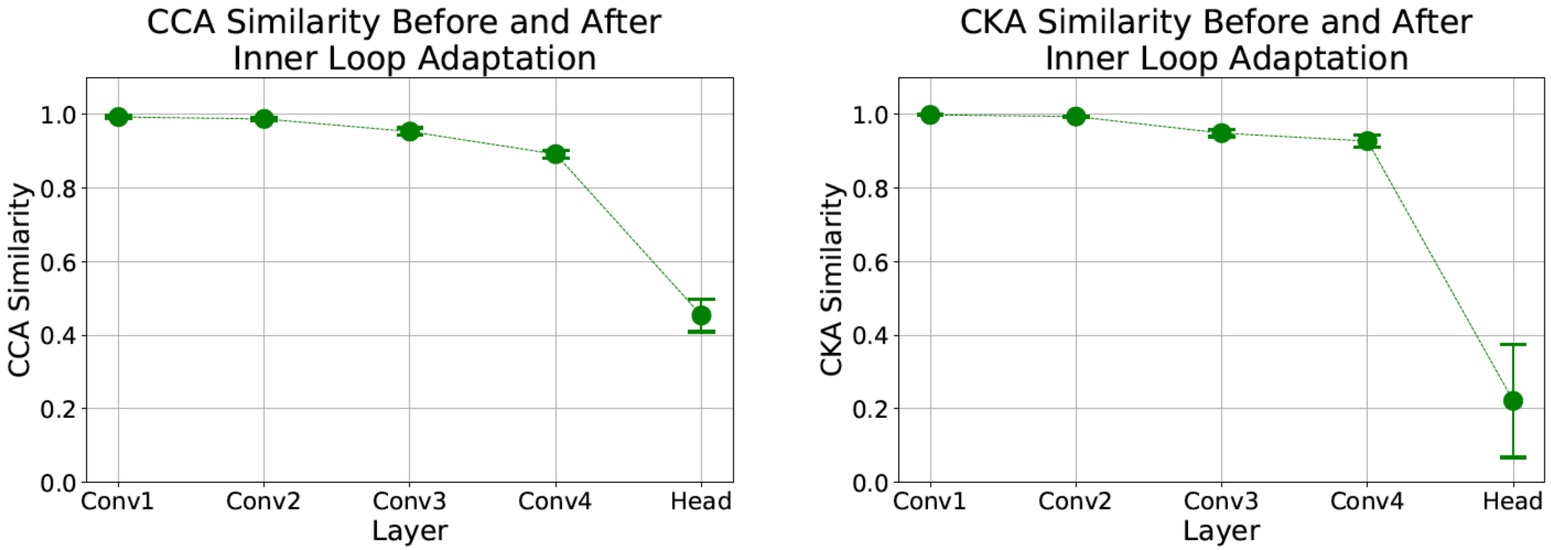


Figure 2: **High CCA/CKA similarity between representations before and after adaptation for all layers except the head.** We compute CCA/CKA similarity between the representation of a layer before the inner loop adaptation and after adaptation. We observe that for all layers except the head, the CCA/CKA similarity is almost 1, indicating perfect similarity. This suggests that these layers do not change much during adaptation, but mostly perform feature reuse. Note that there is a slight dip in similarity in the higher conv layers (e.g. conv3, conv4); this is likely because the slight representational differences in conv1, conv2 have a compounding effect on the representations of conv3, conv4. The head of the network must change significantly during adaptation, and this is reflected in the much lower CCA/CKA similarity.

Feature Reuse happens early in Learning

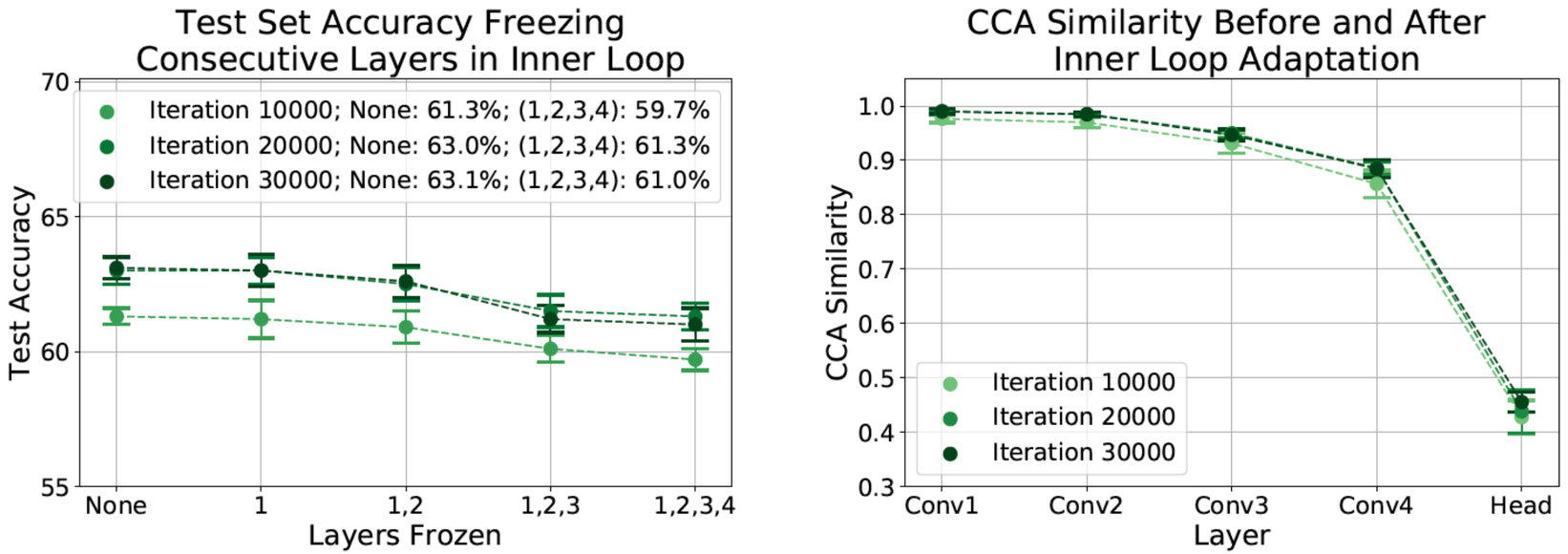
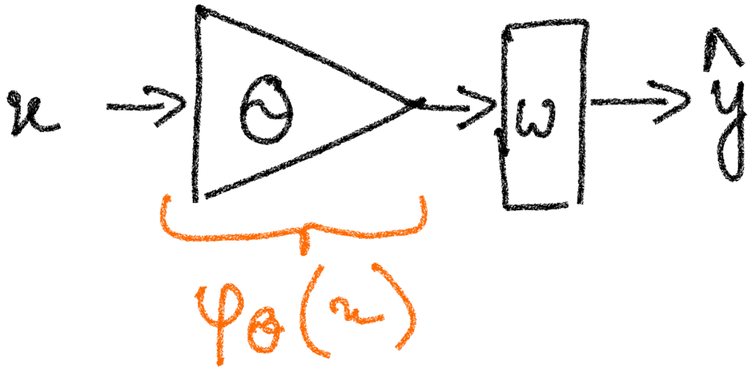


Figure 3: **Inner loop updates have little effect on learned representations from early on in learning.** Left pane: we freeze contiguous blocks of layers (no adaptation at test time), on MiniImageNet-5way-5shot and see almost identical performance. Right pane: representations of all layers except the head are highly similar pre/post adaptation – i.e. features are being reused. This is true from early (iteration 10000) in training.

Solution

MAML vs ANIL

ANIL (Almost No Inner Loop)



$\hat{y} = w^T \phi_\theta(x)$ - prediction

T - task

$\mathcal{L}_T$ - loss of a task T

Outer Loop

w.r.t. θ_i and w_i

$$\theta_{i+1} = \theta_i - \alpha \nabla_{\theta_i} \mathcal{L}_T(w_i^T \phi_{\theta_i}(x), y)$$

$$w_{i+1} = w_i - \alpha \nabla_{w_i} \mathcal{L}_T(w_i^T \phi_{\theta_i}(x), y)$$

$\mathcal{L}$ is computed with θ_i and w_i

Inner Loop

MAML

$$\theta_i = \theta_i - \beta \nabla_{\theta_i} \mathcal{L}_T(w_i^T \phi_{\theta_i}(x), y)$$

ANIL

$$\theta_i = \theta_i$$

$$w_i = w_i - \beta \nabla_{w_i} \mathcal{L}_T(w_i^T \varphi_{\theta_i}(x), y)$$

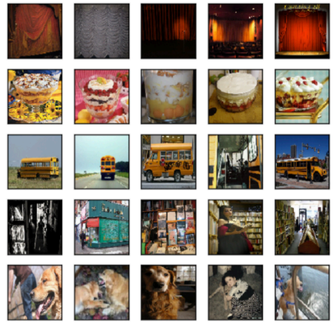
http://learn2learn.net/tutorials/anil_tutorial/ANIL_tutorial/

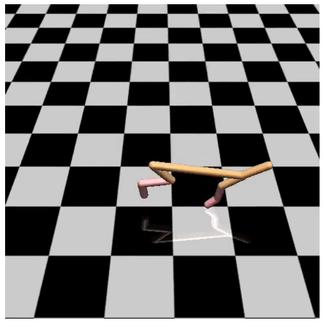
Evaluation

Computational Benefit of ANIL

- Almost no inner loop
  - Training and inference is faster
  - 1.7x per training iteration over MAML

Benchmarks





Method	Omniglot-20way-1shot	Omniglot-20way-5shot	MiniImageNet-5way-1shot	MiniImageNet-5way-5shot
MAML	93.7 ± 0.7	96.4 ± 0.1	46.9 ± 0.2	63.1 ± 0.4
ANIL	96.2 ± 0.5	98.0 ± 0.3	46.7 ± 0.4	61.5 ± 0.5

Method	HalfCheetah-Direction	HalfCheetah-Velocity	2D-Navigation
MAML	170.4 ± 21.0	-139.0 ± 18.9	-20.3 ± 3.2
ANIL	363.2 ± 14.8	-120.9 ± 6.3	-20.1 ± 2.3

Table 2: **ANIL matches the performance of MAML on few-shot image classification and RL.** On benchmark few-shot classification tasks MAML and ANIL have comparable accuracy, and also comparable average return (the higher the better) on standard RL tasks (Finn et al., 2017).

Benchmarks

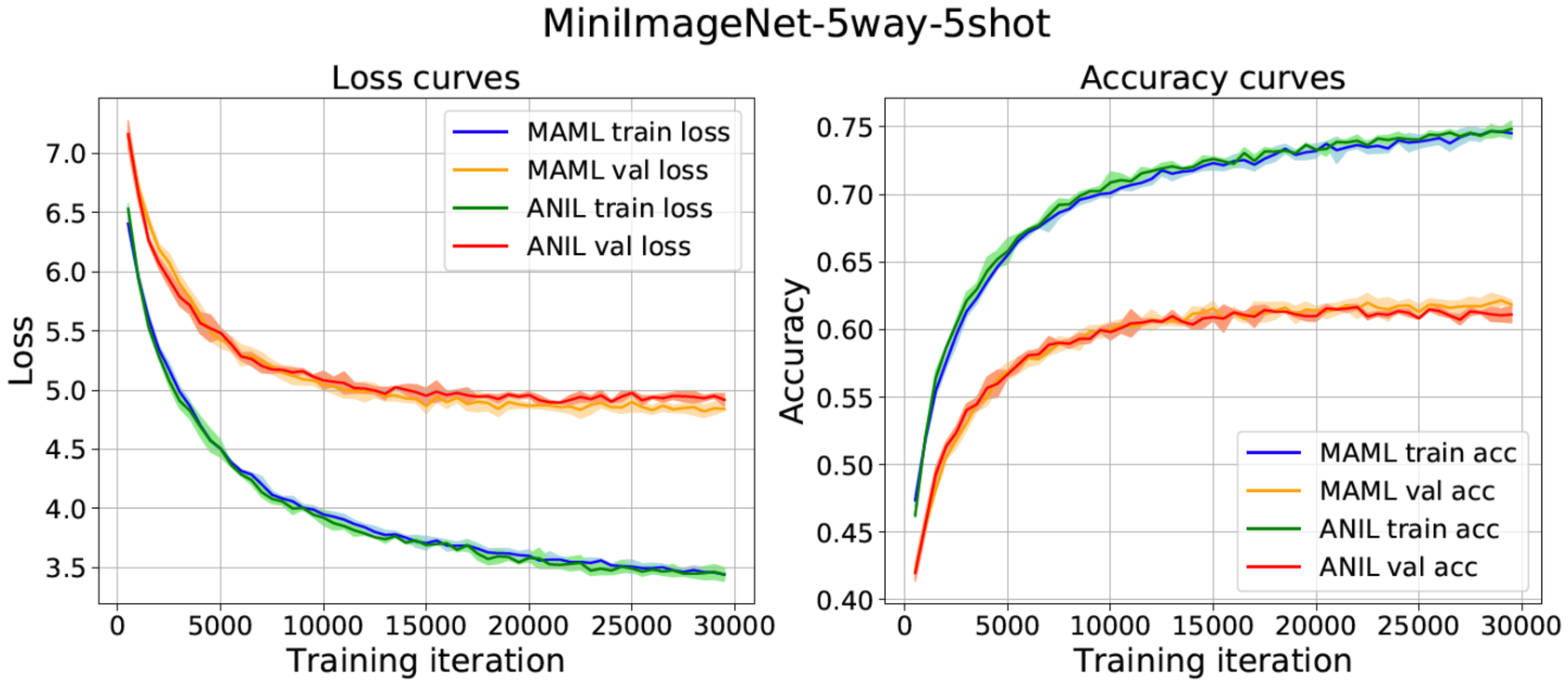


Figure 5: **MAML and ANIL learn very similarly**. Loss and accuracy curves for MAML and ANIL on MiniImageNet-5way-5shot, illustrating how MAML and ANIL behave similarly through the training process.

Comparison of the NN body

Using CCA and CKA

$$\max_{w,s} \frac{\langle w^T A, s^T B \rangle}{||w^T A|| \cdot ||s^T B||} = \max_{w,s} \frac{w^T \Sigma_{AB} s}{\sqrt{w^T \Sigma_{AA} w} \cdot \sqrt{s^T \Sigma_{BB} s}}$$

Model Pair	CCA Similarity	CKA Similarity
MAML-MAML	0.51	0.83
ANIL-ANIL	0.51	0.86
ANIL-MAML	0.50	0.83

Table 3: **MAML and ANIL models learn comparable representations.** Comparing CCA/CKA similarity scores of the of MAML-ANIL representations (averaged over network body), and MAML-MAML and ANIL-ANIL similarity scores (across different random seeds) shows algorithmic differences between MAML/ANIL does not result in vastly different types of features learned.

Remove Head

NIL (No Inner Loop)

- Train ANIL or MAML
- Remove head at test time
- Use Matching Networks at test time by Vinyals et al. (2016)

Method	Omniglot-20way-1shot	Omniglot-20way-5shot	MiniImageNet-5way-1shot	MiniImageNet-5way-5shot
MAML	93.7 ± 0.7	96.4 ± 0.1	46.9 ± 0.2	63.1 ± 0.4
ANIL	96.2 ± 0.5	98.0 ± 0.3	46.7 ± 0.4	61.5 ± 0.5
NIL	96.7 ± 0.3	98.0 ± 0.04	48.0 ± 0.7	62.2 ± 0.5

Table 4: **NIL algorithm performs as well as MAML and ANIL on few-shot image classification.** Performance of MAML, ANIL, and NIL on few-shot image classification benchmarks. We see that with no test-time inner loop, and just learned features, NIL performs comparably to MAML and ANIL, indicating the strength of the learned features, and the relative lack of importance of the head at test time.

Was Head useful?

Method	MiniImageNet-5way-1shot	MiniImageNet-5way-5shot
MAML training-NIL head	48.4 ± 0.3	61.5 ± 0.8
ANIL training-NIL head	48.0 ± 0.7	62.2 ± 0.5
Multiclass training-NIL head	39.7 ± 0.3	54.4 ± 0.5
Multitask training-NIL head	26.5 ± 1.1	34.2 ± 3.5
Random features-NIL head	32.9 ± 0.6	43.2 ± 0.5
NIL training-NIL head	38.3 ± 0.6	43.0 ± 0.2

Table 5: **MAML/ANIL training leads to superior features learned, supporting importance of head at training.** Training with MAML/ANIL leads to superior performance over other methods which do not have task specific heads, supporting the importance of the head at training.

Beyond

Beyond optimization based Meta-Learning

Model Based Meta-Learning

- Models are not optimized for specific task on support set
- Instead,
  - Condition model's output on representation of the task definition
    - Jointly encode the support set (Vinyals et al., 2016; Sung et al., 2018)
      - Rapid Learning
    - Encode each member from support set
      - Feature Reuse
- However, Feature Reuse is dominant (*Conclusion of the authors based on analyzing performance of related work*)

Thank you