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

Aniruddh Raghu*, Maithra Raghu*,
Samy Bengio, Oriol Vinyals

Few Shot Learning

Many tasks, **little data** for each task

Task 1

Dog/Cat

Task 2

Chair/Lion

Task 3

Plane/Tree

Few Shot Learning

(**Optimization**-based) **Meta Learning Algorithms**

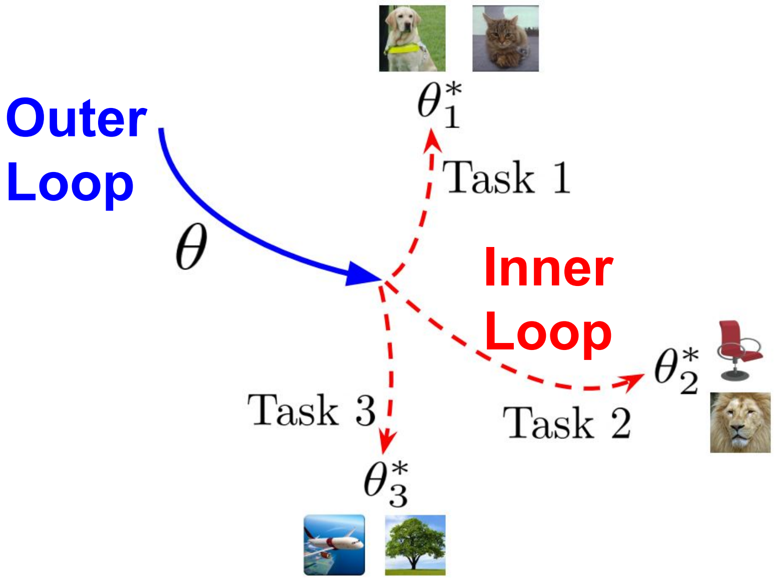
- *Model Agnostic Meta Learning, (Finn et al), ICML 2017*

Few Shot Learning

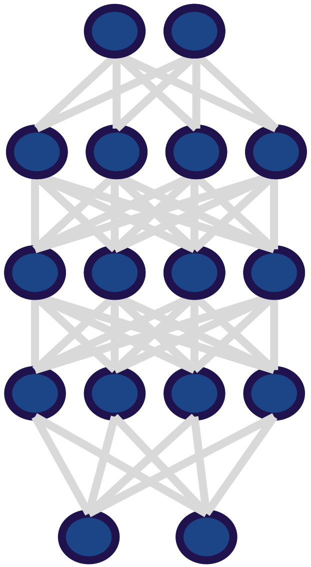
(Optimization-based) Meta Learning Algorithms

- *Model Agnostic Meta Learning, (Finn et al), ICML 2017*

Outer Loop: meta-initialization; **Inner Loop**: adaptation

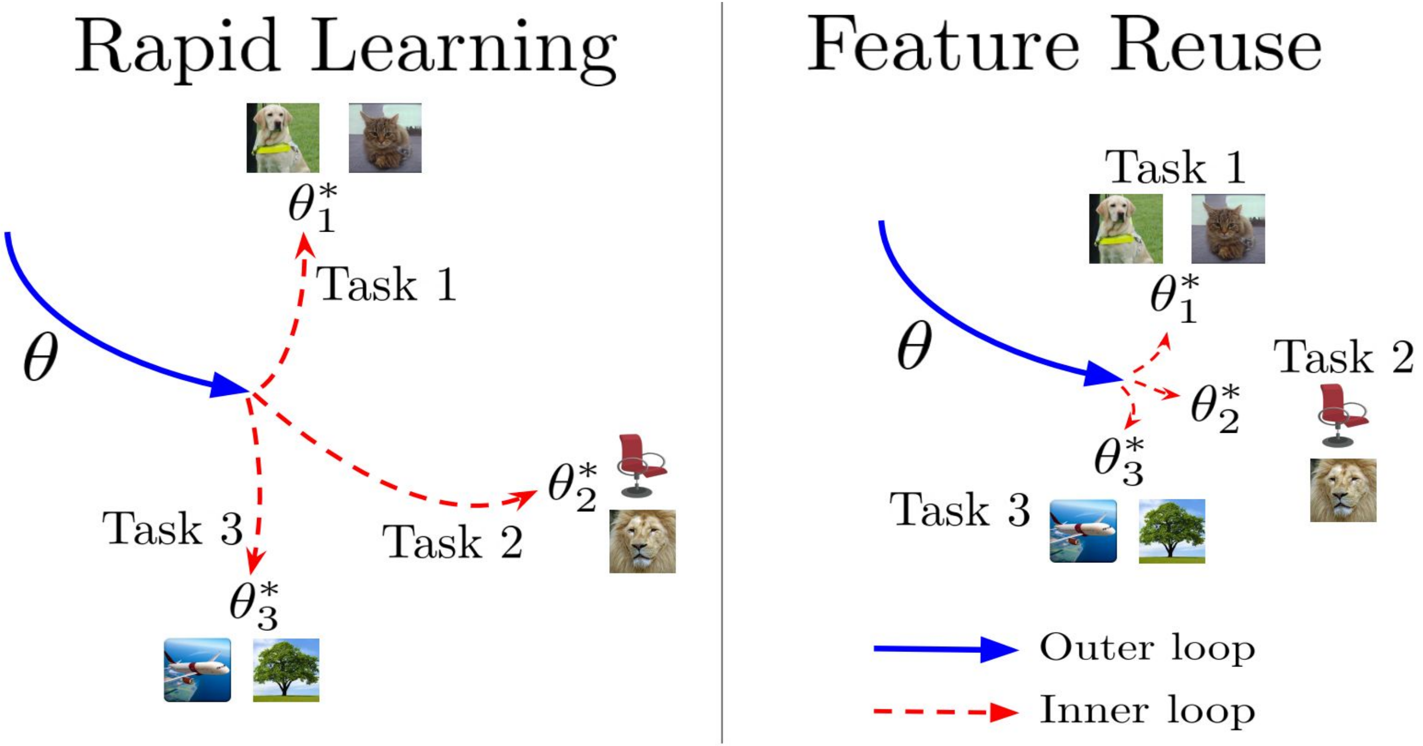

Outer Loop

Inner Loop



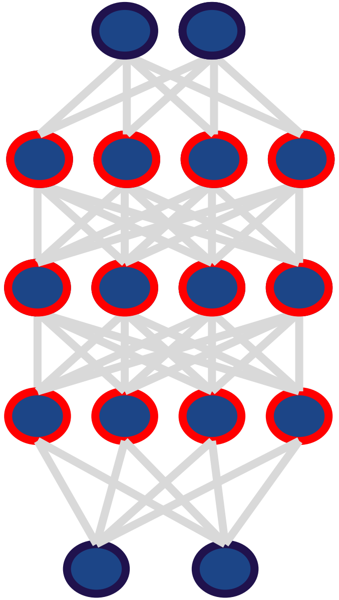
Rapid Learning or Feature Reuse?

Outer Loop: meta-initialization; **Inner Loop**: adaptation



Rapid Learning or Feature Reuse?

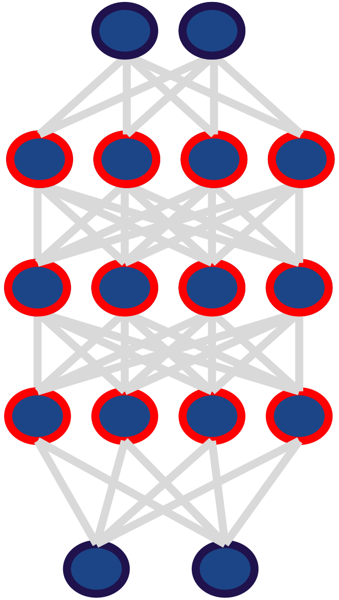
How do **hidden representations** behave (during inner loop)?



Rapid Learning or Feature Reuse?

How do **hidden representations** behave (during inner loop)?

Measure Representation Similarity

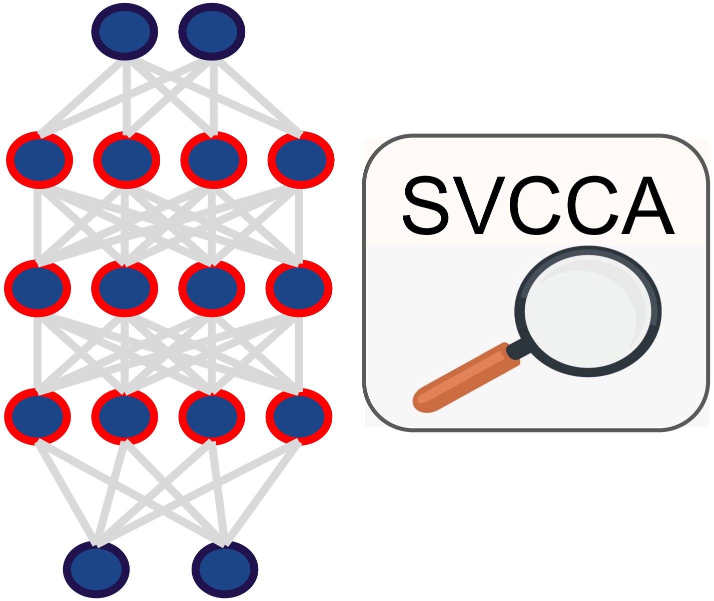


SVCCA

https://github.com/google/svcca

Rapid Learning or Feature Reuse?

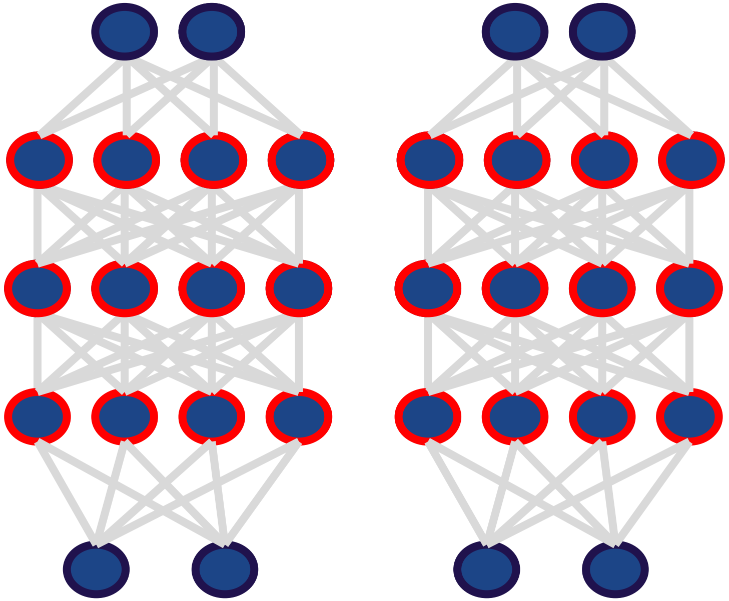
How do **hidden representations** behave (during inner loop)?



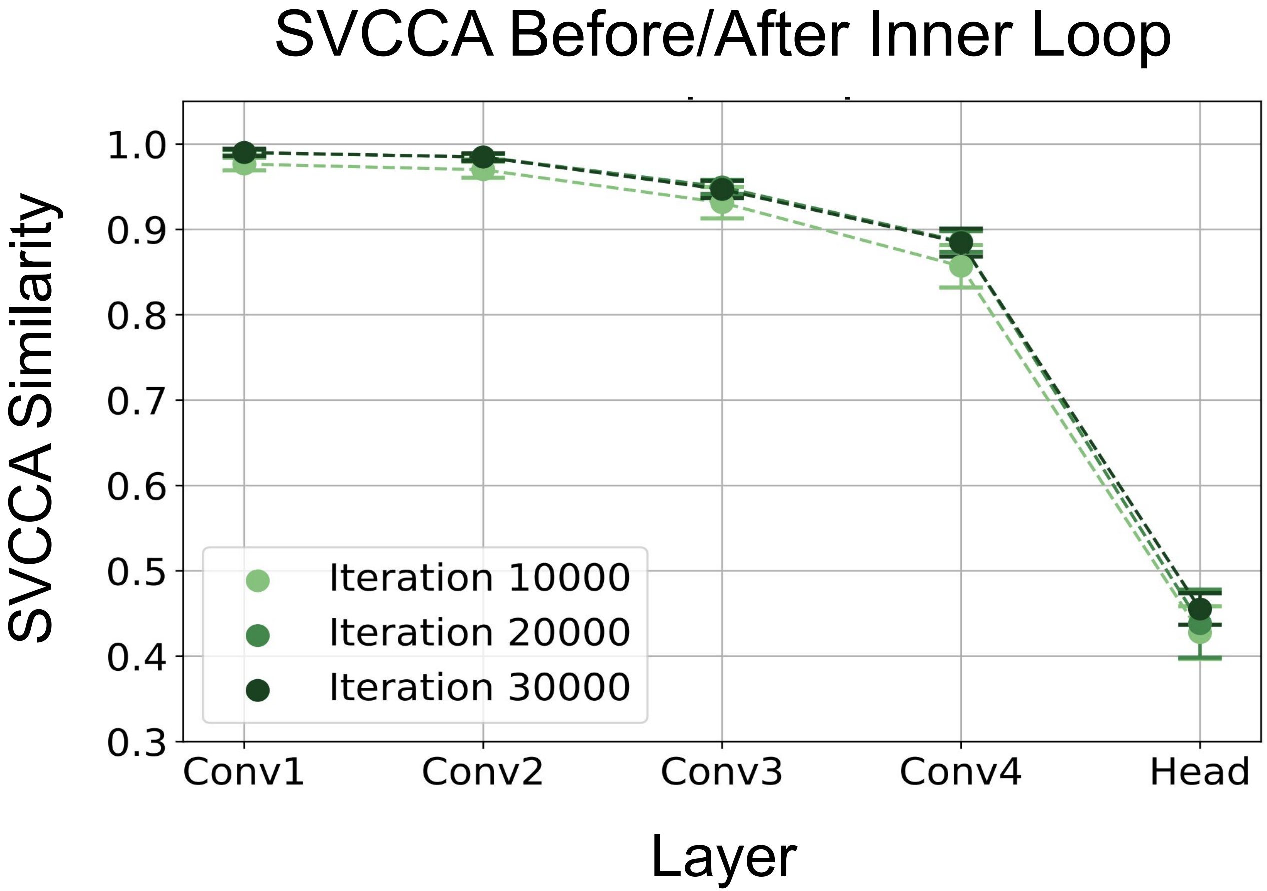
Measure Representation Similarity

Before inner loop

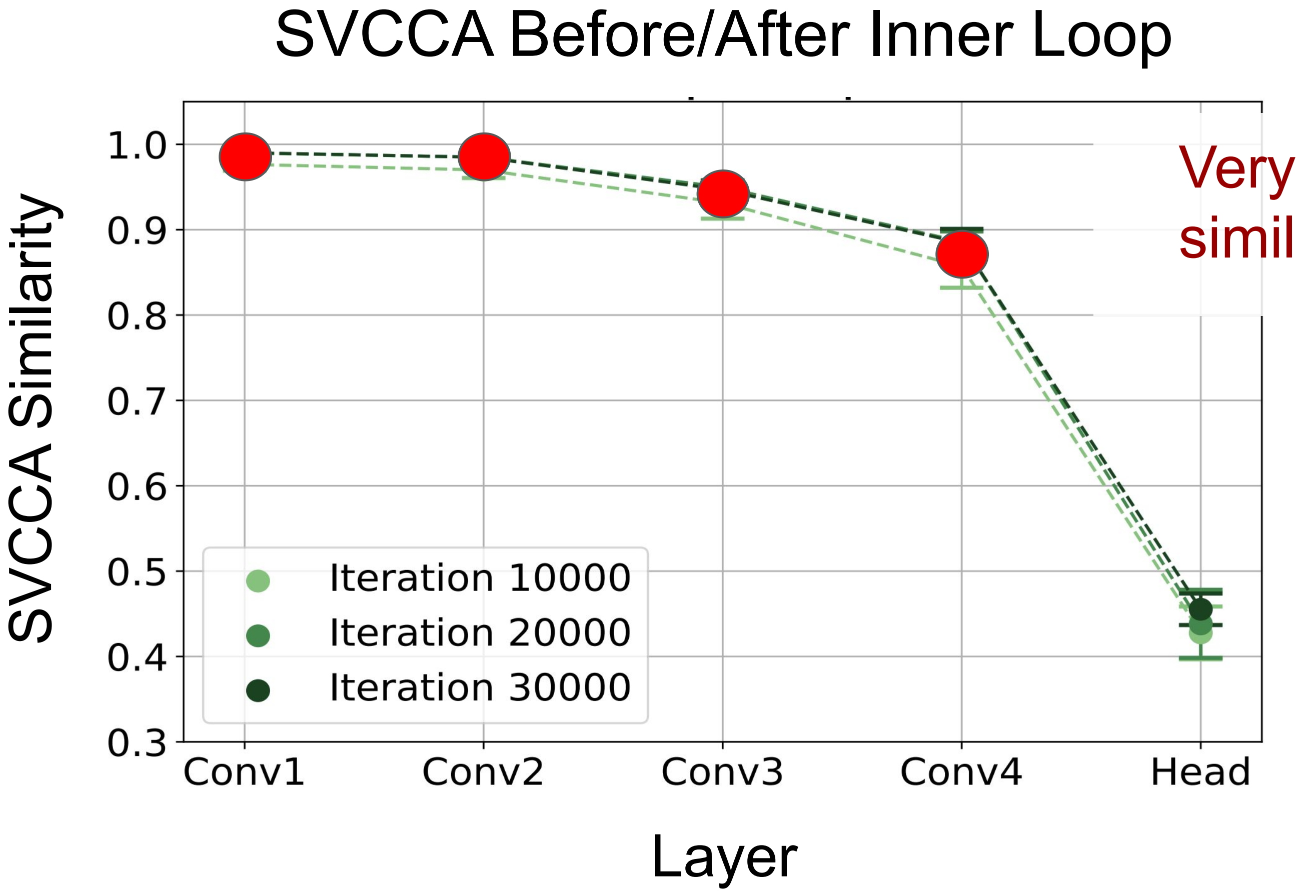
After inner loop



Rapid Learning or Feature Reuse?



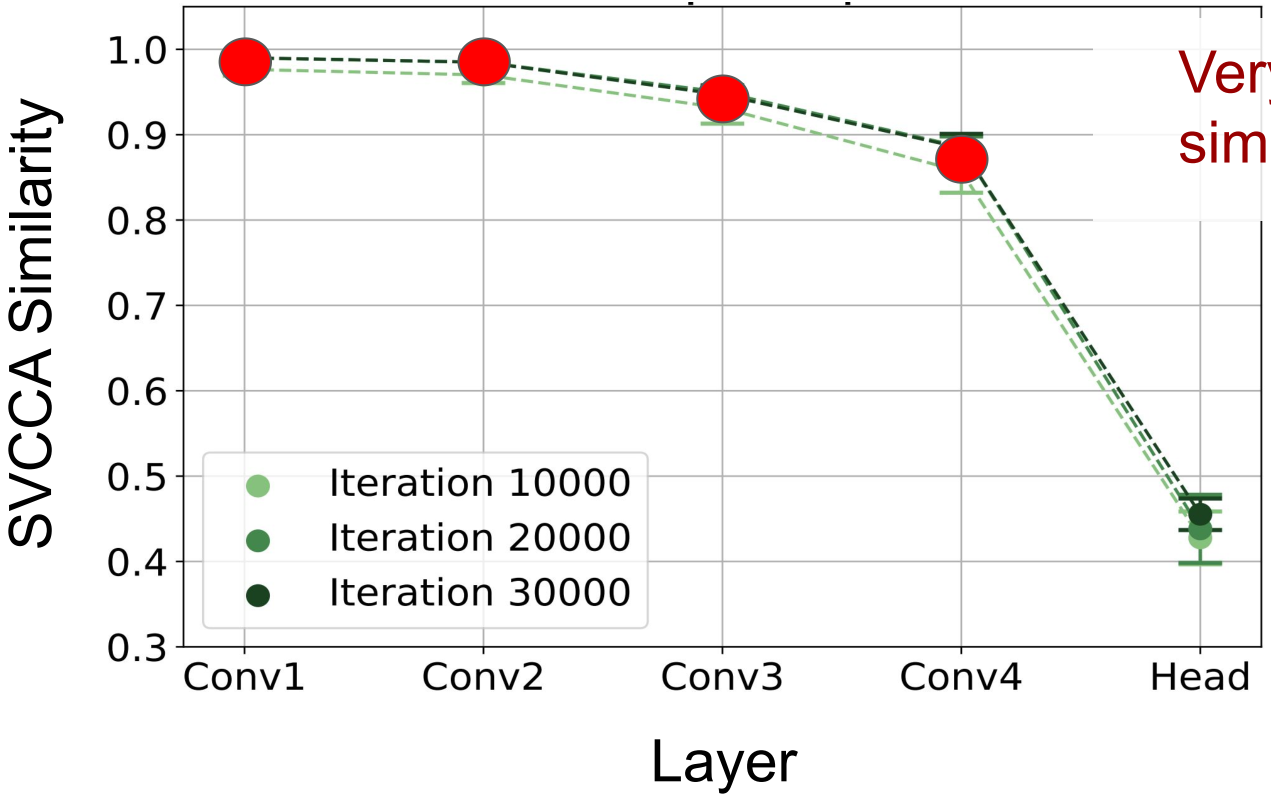
Rapid Learning or Feature Reuse?



Very high similarity!

Rapid Learning or Feature Reuse?

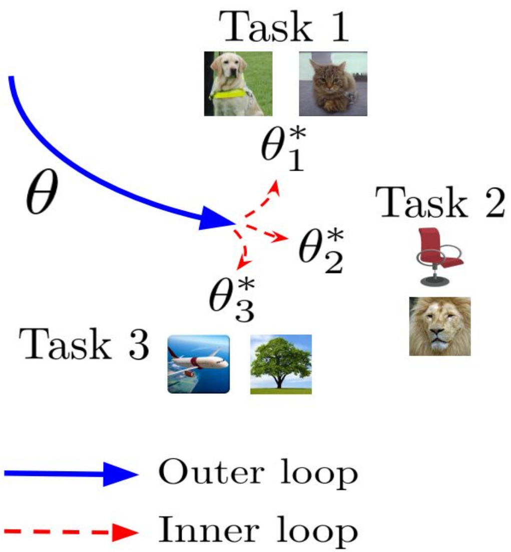
SVCCA Before/After Inner Loop



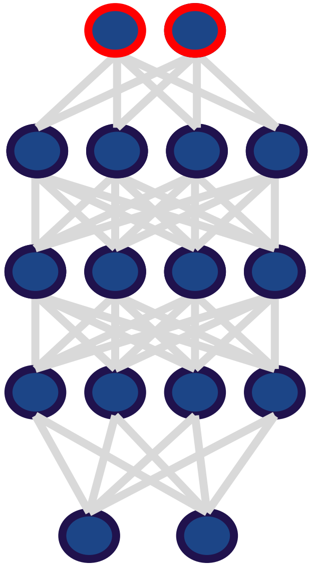
Very high similarity!

ANIL: Almost No Inner Loop Algorithm

ANIL: Almost No Inner Loop Algorithm

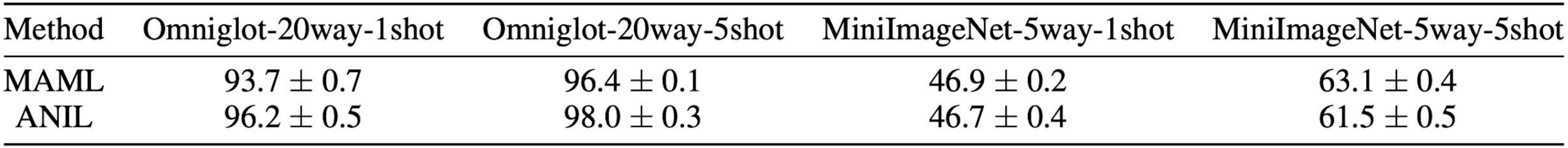


- **Removes inner loop** for all but head of network
- Much more **computationally efficient**, **same performance**
- **Insights** into meta learning and few shot learning

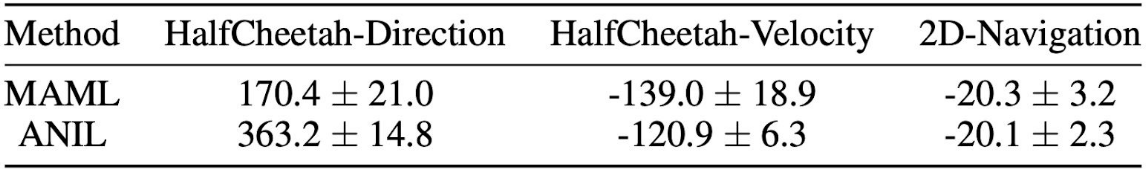


ANIL: Performance Results

Matches performance of MAML in few-shot classification and RL



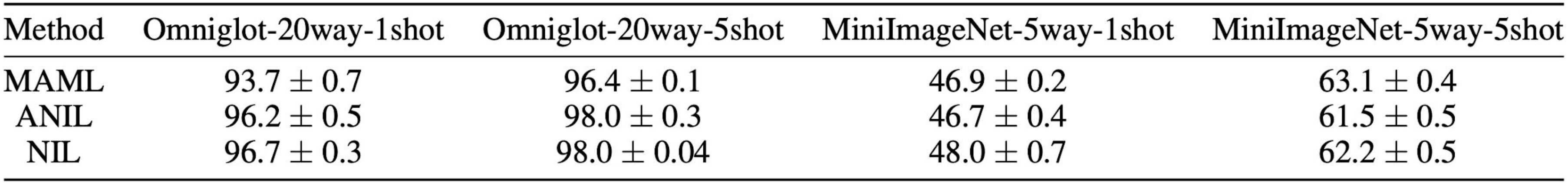
Method	Omniglot-20way-1shot	Omniglot-20way-5shot	MiniImageNet-5way-1shot	MiniImageNet-5way-5shot
MAML	93.7 ± 0.7	96.4 ± 0.1	46.9 ± 0.2	63.1 ± 0.4
ANIL	96.2 ± 0.5	98.0 ± 0.3	46.7 ± 0.4	61.5 ± 0.5



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

ANIL and NIL (No Inner Loop)

NIL: No Inner Loop (at test time), performs equally well



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

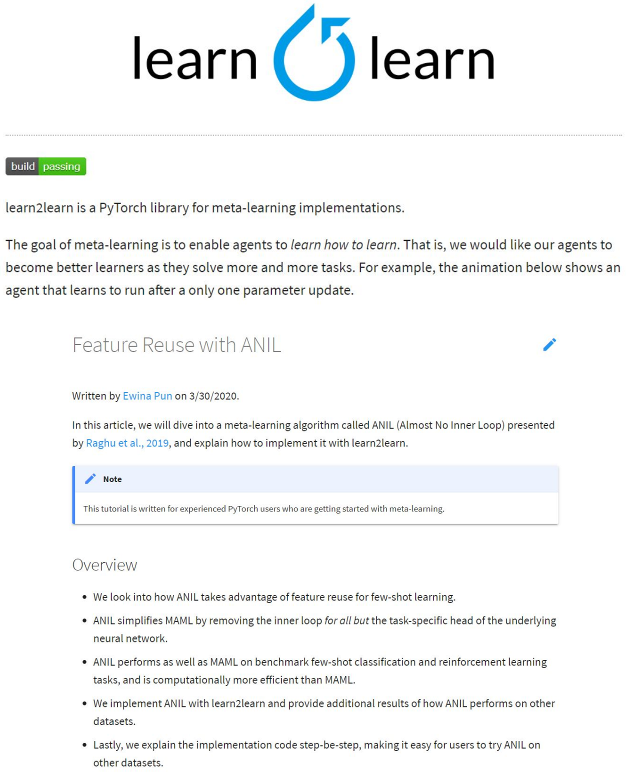
Thanks and Future Directions

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

- **Exploring Medium Shot Learning?**
- **Meta-Learning as Pretraining?**
- **Learning Regimes for Interpolating between Rapid Learning and Feature Reuse?**

Theoretical Analysis: *Few-Shot Learning via Learning the Representation, Provably, (Du, Hu, Kakade, Lee, Lei)*

Analyzing Feature Reuse: *Rethinking Few-Shot Image Classification: a Good Embedding Is All You Need?, (Tian, Wang, Krishnan, Tenenbaum, Isola)*



learn learn

build passing

learn2learn is a PyTorch library for meta-learning implementations.

The goal of meta-learning is to enable agents to *learn how to learn*. That is, we would like our agents to become better learners as they solve more and more tasks. For example, the animation below shows an agent that learns to run after a only one parameter update.

Feature Reuse with ANIL

Written by Ewina Pun on 3/30/2020.

In this article, we will dive into a meta-learning algorithm called ANIL (Almost No Inner Loop) presented by Raghu et al., 2019, and explain how to implement it with learn2learn.

> **Note**
>
> This tutorial is written for experienced PyTorch users who are getting started with meta-learning.

Overview

- We look into how ANIL takes advantage of feature reuse for few-shot learning.
- ANIL simplifies MAML by removing the inner loop *for all but* the task-specific head of the underlying neural network.
- ANIL performs as well as MAML on benchmark few-shot classification and reinforcement learning tasks, and is computationally more efficient than MAML.
- We implement ANIL with learn2learn and provide additional results of how ANIL performs on other datasets.
- Lastly, we explain the implementation code step-be-step, making it easy for users to try ANIL on other datasets.