Meta Learning - Learn to learn

Deep learning is to adjust hyper parameters.

Can machine learning learns the hyper parameters?

Machine learning:

Industry

Using 1000 GPUs to try 1000 sets of hyperparameters

Academia

"Telepathize" (通靈) a set of good hyperparameters

$Function\ with\ unknown(f_\theta) \rightarrow Define\ loss\ function(L) \rightarrow Optimization$

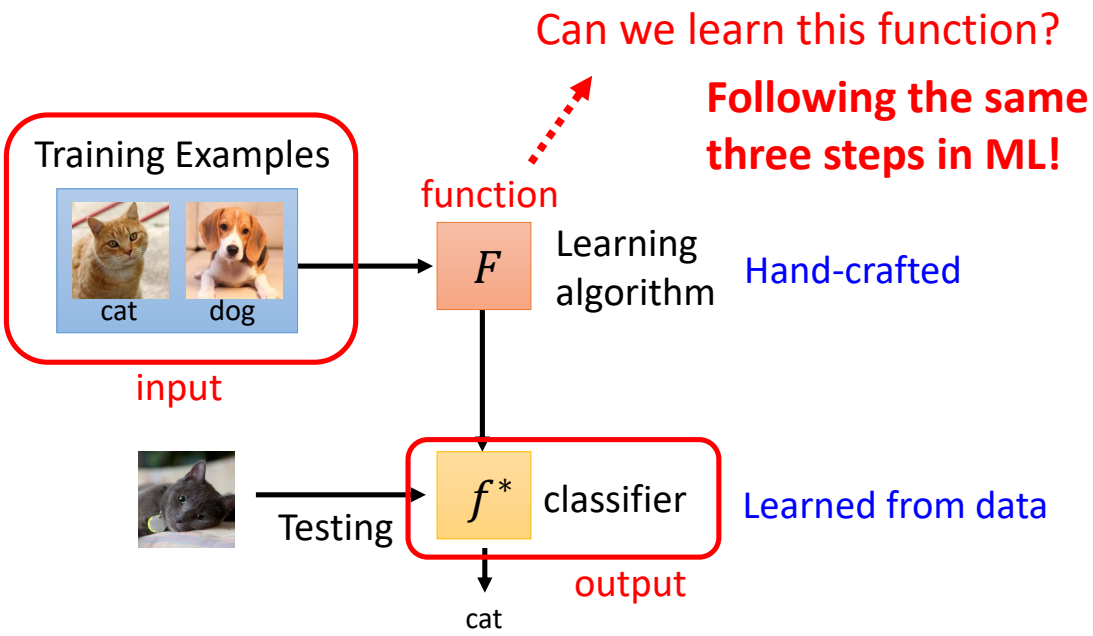
Step 1: $loss\ L(\theta) = \sum_{K=1}^{K} e_K$ sum over training examples

Step 2: $\theta^* = \underset{\theta}{argmin} L(\theta)$, done by gradient descent (vanilla)

Step 3: f_{θ^*} is the function learned by learning algorithm from data

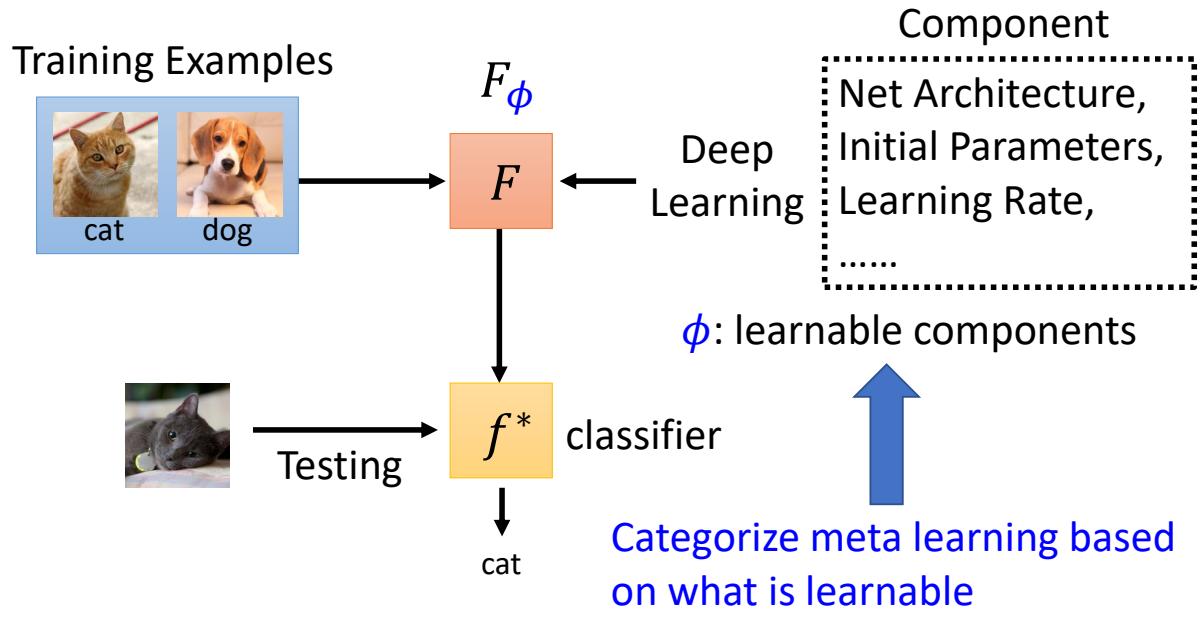
Introduction of Meta Learning

Learning is a function F



$Training\ Examples(dataset) \rightarrow Learning\ algorithm(F) \rightarrow classifier(f^*) \rightarrow result$
Learning algorithm is hand-crafted.
Question: Can we learn this function by machine learning steps?

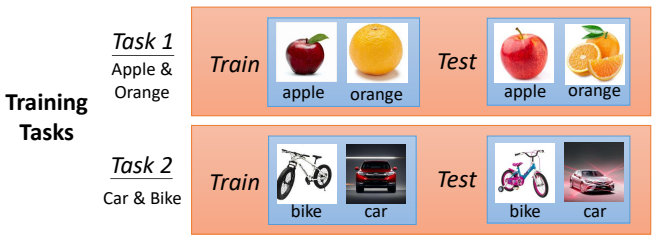
Step 1



What is learnable in a learning algorithm?
All the learnable components are defined as ϕ

Step 2

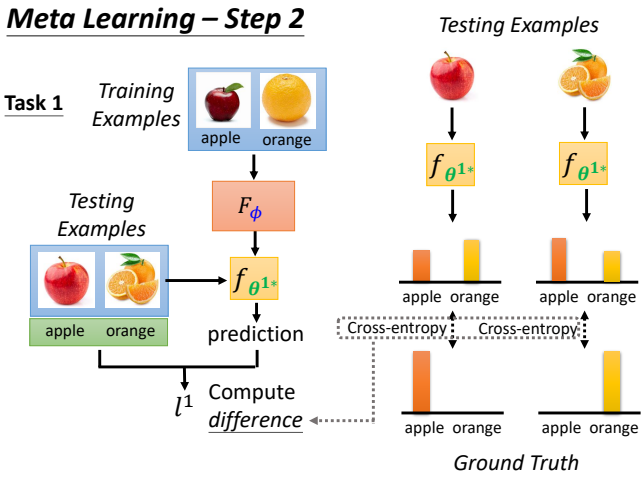
Define loss function $L(\phi)$for learning algorithm F_ϕ.
Prepare lots of training tasks



How to define $L(\phi)$

θ^{1*}:parameters of the classifier learned by F_ϕ using the training examples of task 1
Then we evaluate the classifier on testing set to know how good the classifier is.

$l^1 = \sum$ cross entropy



Total loss $L(\phi) = \sum_{n=1}^{N} l^n$ (N is the number of the training tasks)

We use test data to calculate the loss but in usual machine learning task we use training data.

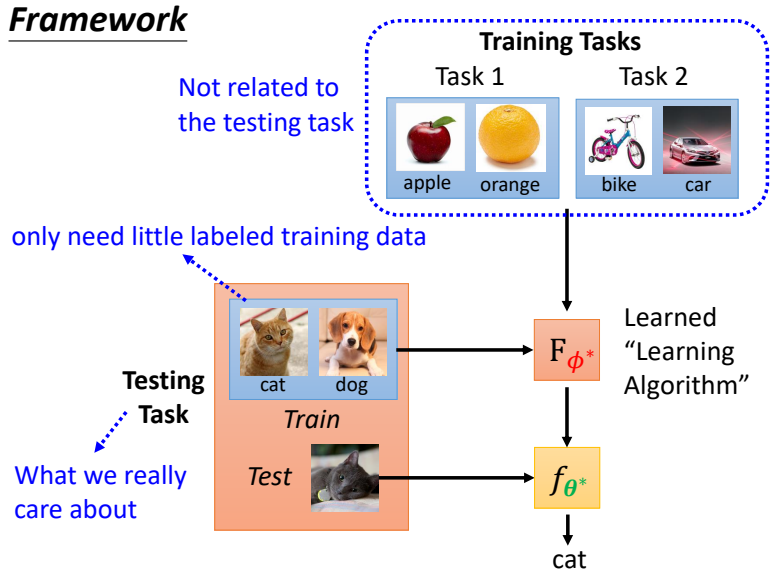
Because we are training tasks not missions.

Step 3

1. Loss function for learning algorithm $L(\phi) = \sum_{n=1}^{N} l^n$
2. Find ϕthat can minimize $L(\phi)$, $\phi^* = arg\min_{\phi} L(\phi)$
3. Using the optimization approach you know
    1. If you know how to compute $\frac{\partial L(\phi)}{\partial \phi}$ → Gradient descent!

       Once $L(\phi)$ is not differentiable?
        1. Reinforcement learning/Evolutionary Algorithm (硬作)
4. Now we have a learned "learning algorithm" F_{ϕ^*}

Framework of Meta Learning



It might be confused with few-shop learning.

Usually few-shot learning algorithms are fetched by meta learning.

Training Tasks/Testing Tasks

Testing Tasks can not be seen in training step.

Comparison of ML and Meta Learning

Goal

Machine learning ≈ find a function f

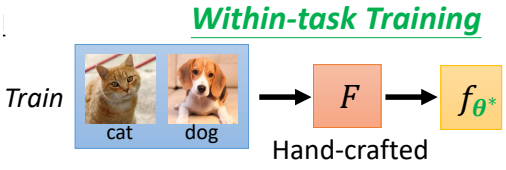
$f(input) = target$

Meta Learning≈ find a function F that finds a function f.

Learning Algorithm $F(tasks) = f \rightarrow f(input) = target$

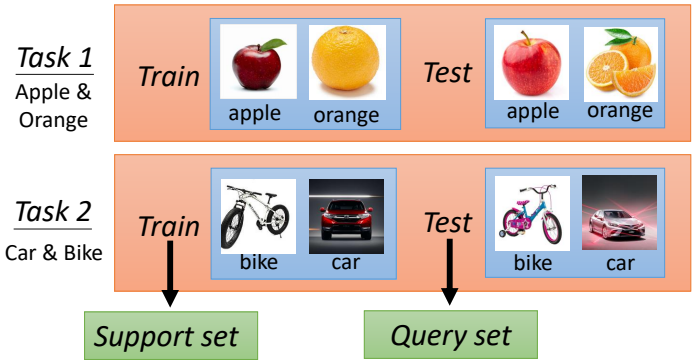
Training Data

Machine learning:



One task

Within-task Training



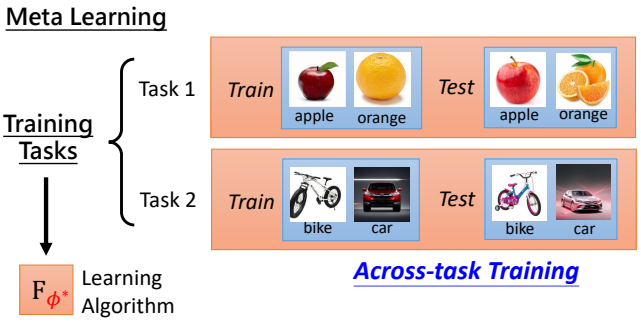
(in the literature of "*learning to compare*")

Meta Learning

Training set ➜ support set
Test set ➜ Query set

in the literature of "learning to compare"

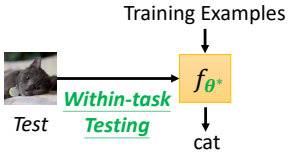
Across-task Training



Test Procedure

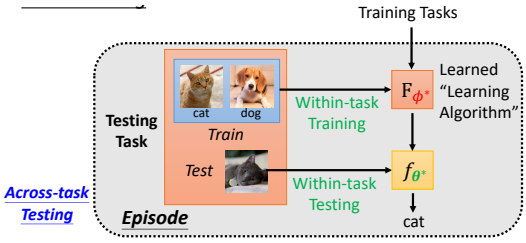
Machine learning:

Test



Meta Learning:

Within-task Training
Within-task Testing



Loss

Machine Learning

$$loss\ L(\theta) = \sum_{K=1}^{K} e_K$$

Sum over training examples in one task

Meta Learning

$$L(\phi) = \sum_{n=1}^{N} l^n$$

l: Sum over testing examples in one task

$\sum_{n=1}^{N} l^n$: Sum over training tasks

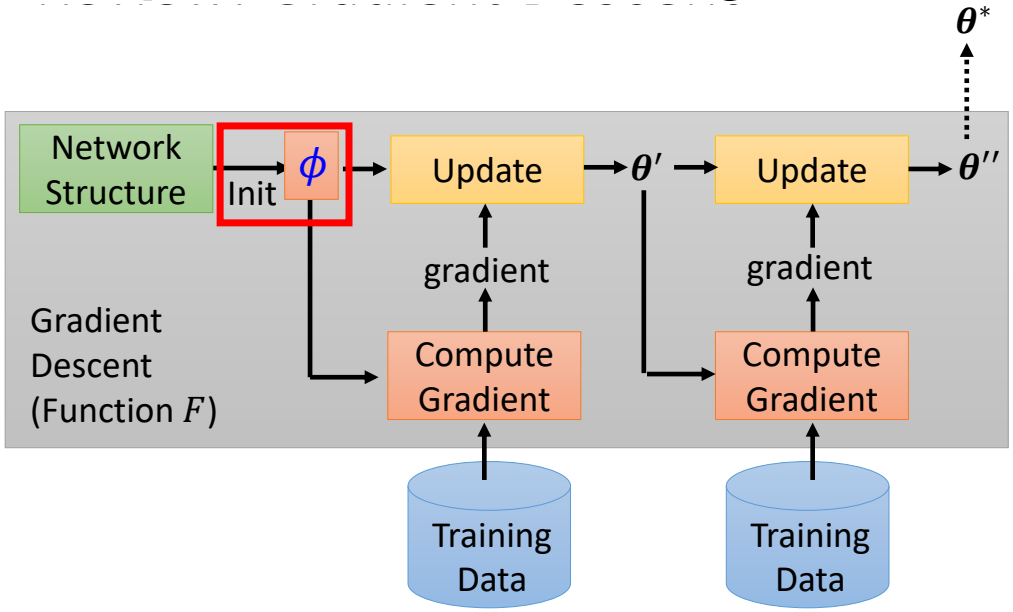
You need to go over an episode to get l^1
Across-task training includes within-task training and testing.
Across-task training: outer loop in “Learning to initialize”
Within-task training and testing: Inner loop in “Learning to initialize”

Shared tips between ML and Meta Learning

What you know about ML can usually apply to meta learning:

- Overfitting on training tasks
  - Get more training tasks to improve performance
  - Task augmentation
- There are also hyperparameters when learning a learning algorithm .....
  Development task 🙂

Examples of Meta Learning



Learning to initialize - to find the best θ^0

θ^0 is trainable in Meta learning

MAML

Pre-training (Self-supervised Learning)

Trained by proxy tasks
(fill-in the blanks, etc.)

Model-AgnosticMeta-Learning(MAML) (mammal)

- Reptile

How to train your MAML?

Pre-training (self-supervised learning)

Trained by proxy tasks (fill-in the blanks, etc.

More typical ways: Use data from different tasks to train a model which also known as **multi-task learning (baseline of meta)**

Meta learning is similar as domain adaptation/ transfer learning.

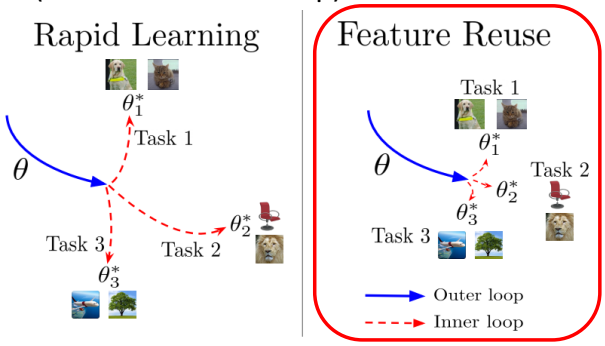
The reason MAML is good:

ANIL (Almost No Inner Loop)

Rapid Learning is the major cause to make MAML good.
Feature Reuse

First order MAML (FOMAML)
Reptile

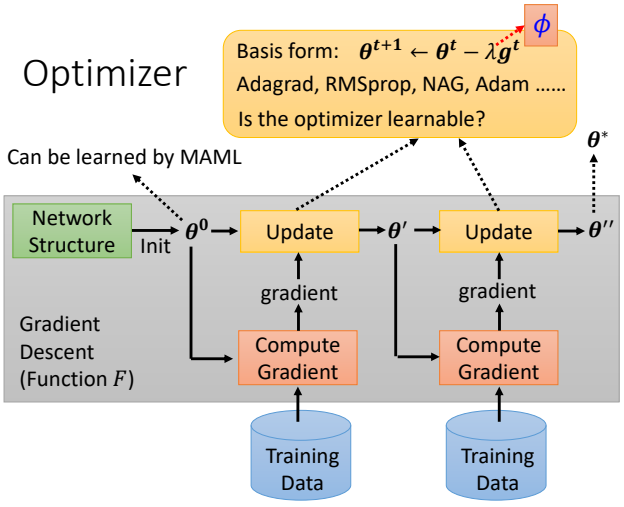
- ANIL (Almost No Inner Loop)



Aniruddh Raghu, Maithra Raghu, Samy Bengio, Oriol Vinyals, Rapid Learning or Feature Reuse? Towards Understanding the Effectiveness of MAML, ICLR, 2020

Learning Optimizer

Basis form:$\theta^{t+1} \rightarrow \theta^t - \lambda g^t$, λ here related to ϕ.



Learning Network Architecture Search (NAS)

$\hat{\phi} = \underset{\phi}{argmin} L(\phi),\ \nabla_{\phi} L(\phi) = ?$

Here ϕ means **network architecture.**

If you can not find a differentiation, we can use reinforcement learning to reach it.

Apply Reinforcement learning

An agent uses a set of actions to determine the network architecture.

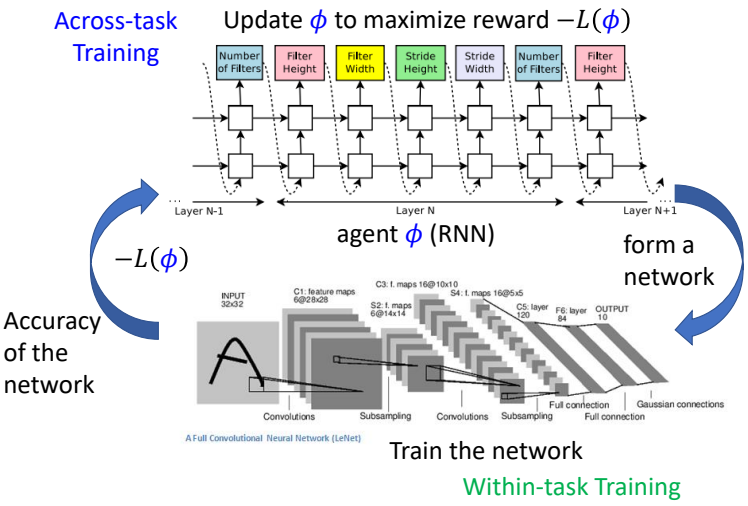
In reinforcement learning, ϕ represents the agent's (A) parameters.

Reward to be maximized is $-L(\phi)$.

Apply Evolution Algorithm

DARTS

Differentiable Architecture Search



Learning Data Processing

Data Augmentation

Sample Reweighting

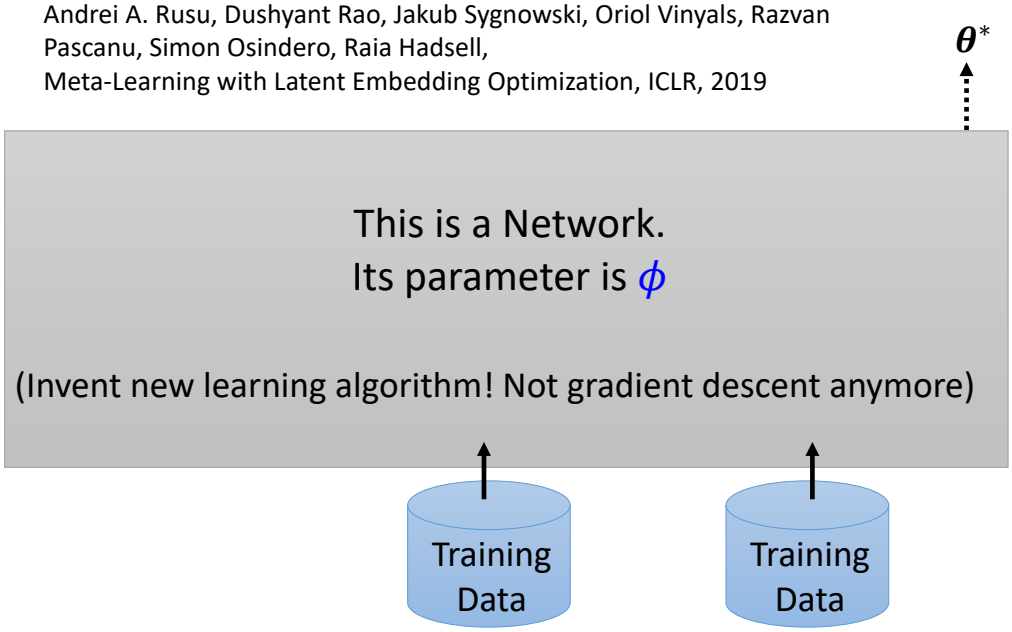
Give different samples different weights

Sample Weighting Strategies

Develop Novel Learning Algorithms by Learning - Beyond Gradient Descent

Beyond Gradient Descent

Andrei A. Rusu, Dushyant Rao, Jakub Sygnowski, Oriol Vinyals, Razvan Pascanu, Simon Osindero, Raia Hadsell,
Meta-Learning with Latent Embedding Optimization, ICLR, 2019



Learning to compare

Metric-based approach

There is no boundary between training and testing.

Applications

Few-shot image classification.

Each class only have a few images.

3-ways 2-shot

N-ways K-shot classification: In each task, there are
3-ways 2-shot, N classes, each has K examples.
In meta learning, you need to **prepare many N-ways K-shot tasks** as training and testing tasks.

Benchmark dataset: Omniglot
To create N way K shot:

20 ways
1 shot

Each character represents a class

Testing set
(Query set)

Training set
(Support set)

- Split your characters into training and testing characters
  - Sample N training characters, sample K examples from each sampled characters → one training task
  - Sample N testing characters, sample K examples from each sampled characters → one testing task