Chapter 16 Meta Learning

Task 1: speech recognition

Task 2: image recognition

⋮

Task 100: text classification

Introduction

- Meta learning = Learn to learn

Learning task 1

Learning task 2

⋮

Learning task 100

I can learn task 101 better because I learn some **learning skills**

Be a better learner

Life-long: one model for all the tasks

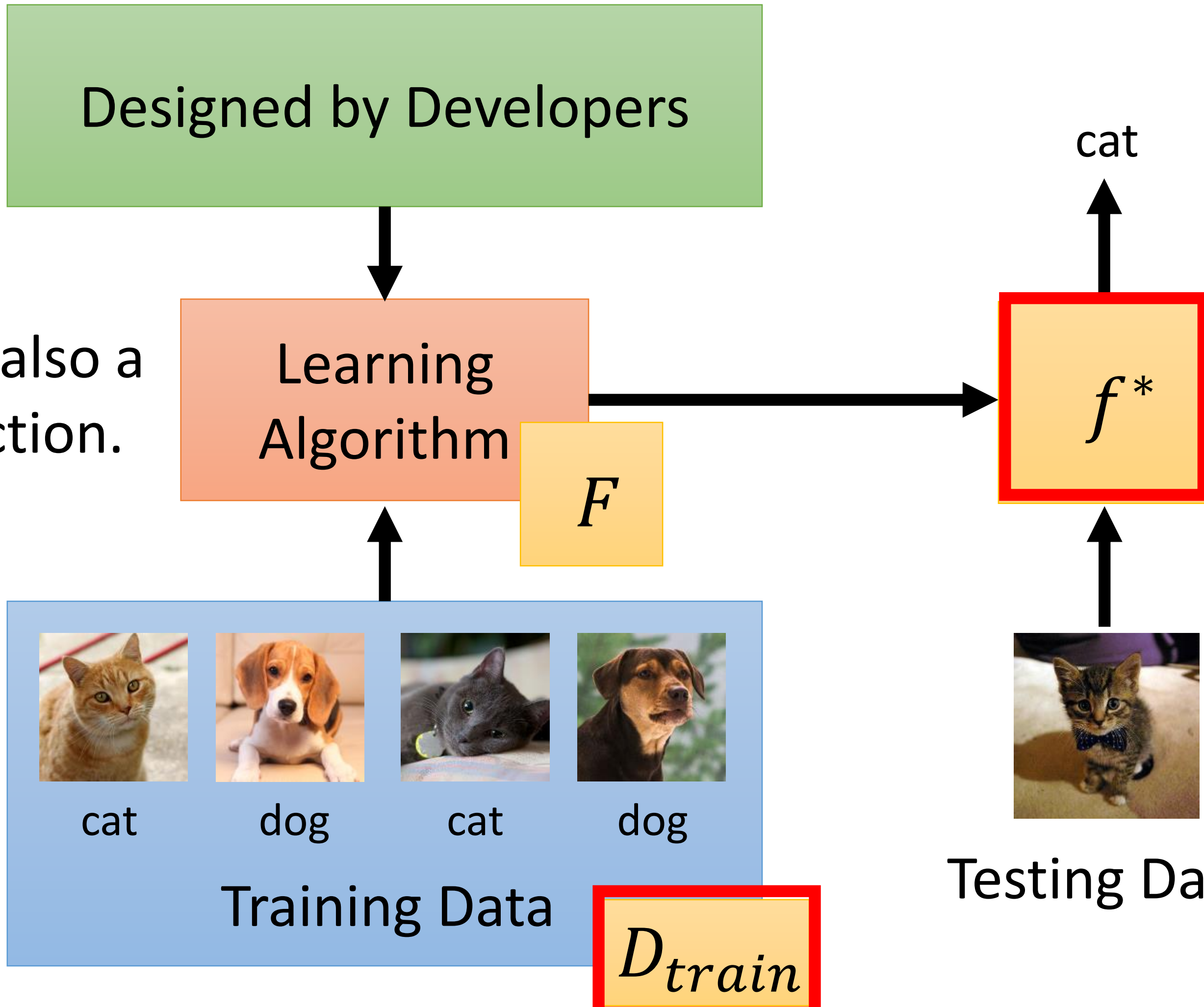
Meta: How to learn a new model

Meta Learning

$$f^* = F(D_{train})$$

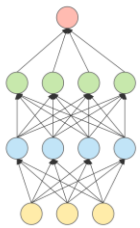
Can machine find F from data?

It is also a function.



Meta Learning

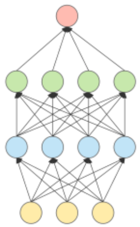
<u>Machine Learning</u> ≈ 根据资料找一個函数 f 的能力

 $f($  $)$ = “Cat”

<u>Meta Learning</u>

≈ 根据资料找一個找一個函數 f 的函数 F 的能力

$F($ 

$) = f^*$ 

~~Machine~~ Learning is Simple

Meta

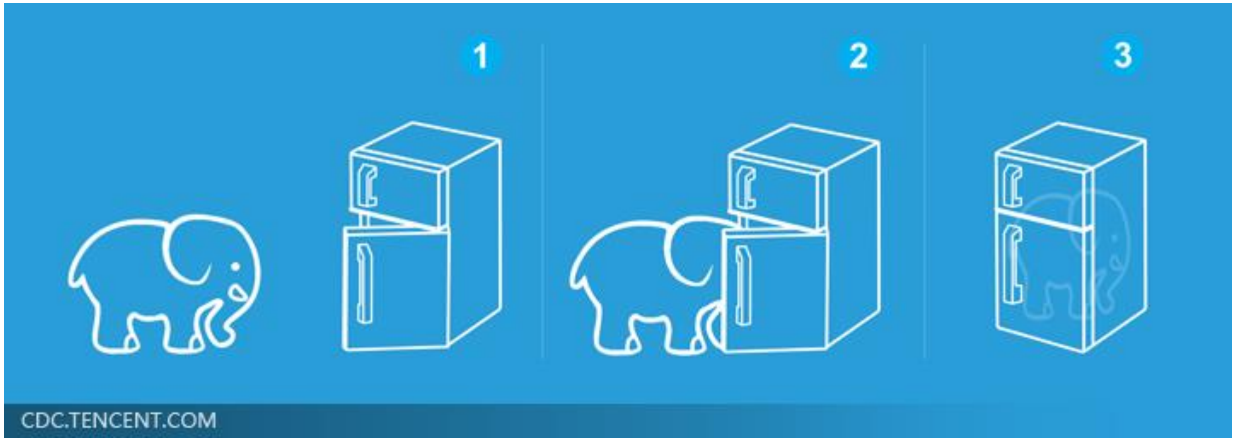
Step 1: define a set of ~~function~~

Step 2: goodness of ~~function~~

Step 3: pick the best ~~function~~

Function f → Learning algorithm F

就好像把大象放进冰箱 ……

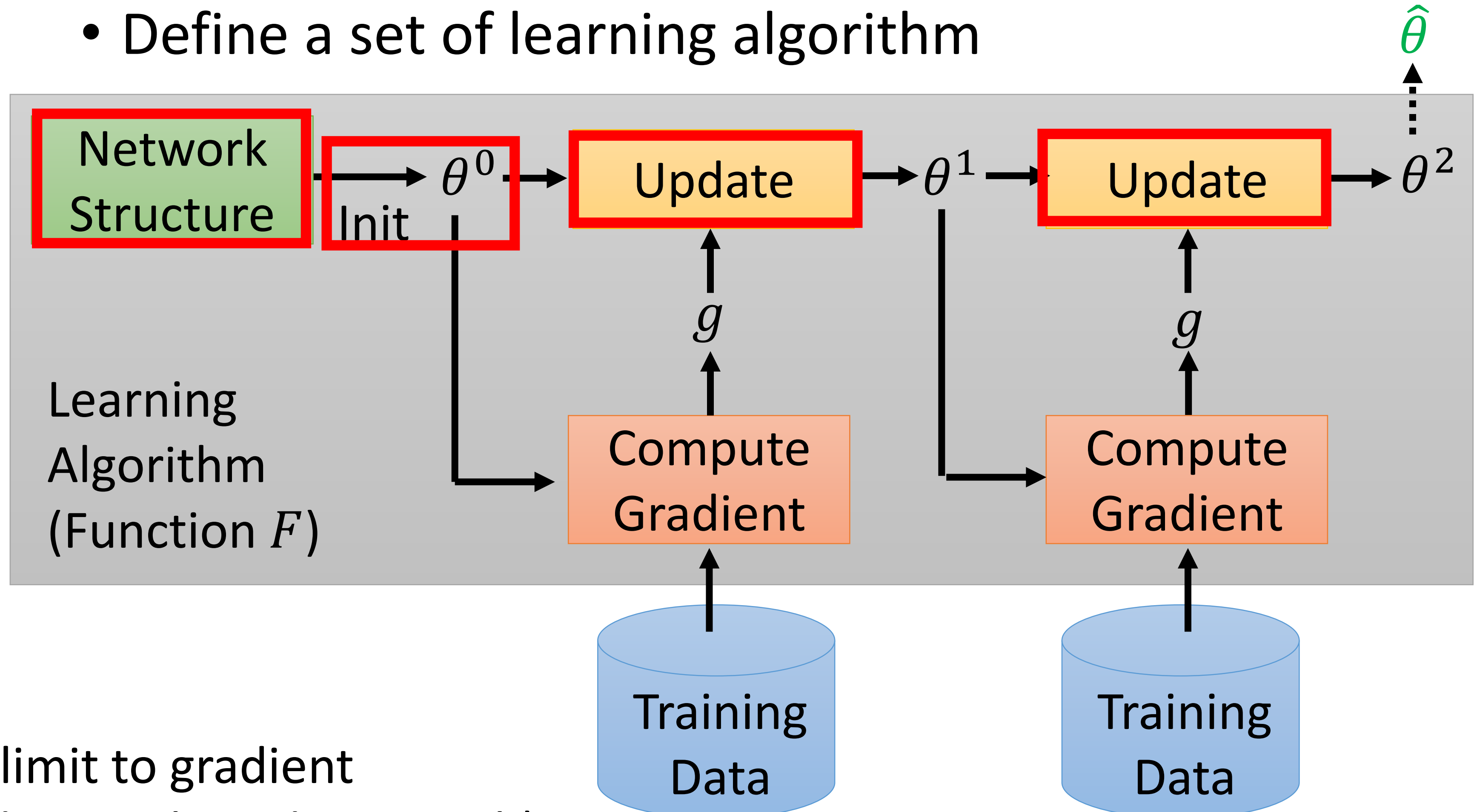


Meta Learning

Different decisions in the red boxes lead to different algorithms.

What happens in the red boxes is decided by humans until now.

- Define a set of learning algorithm



(limit to gradient descent based approach)

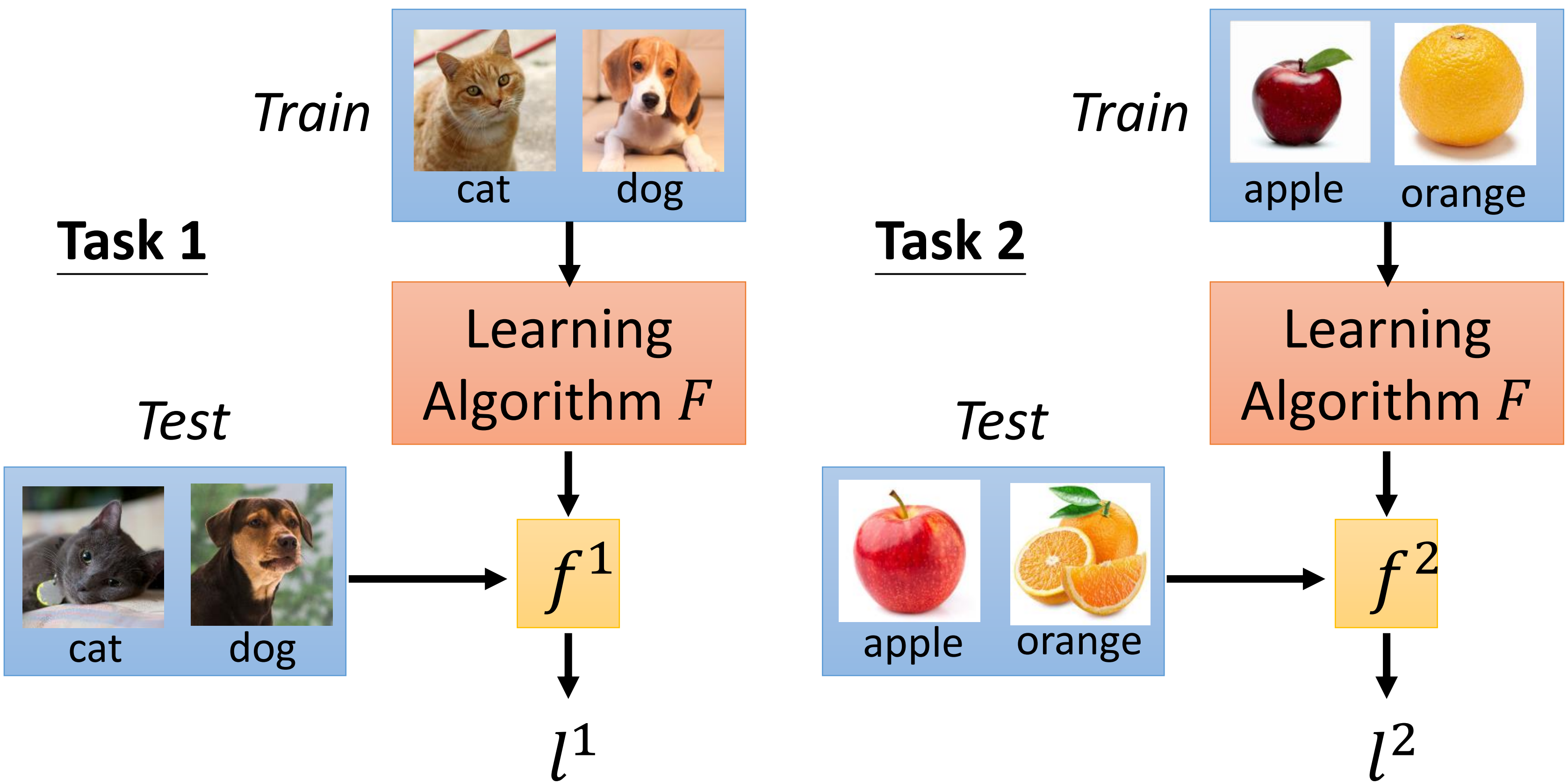
Meta Learning

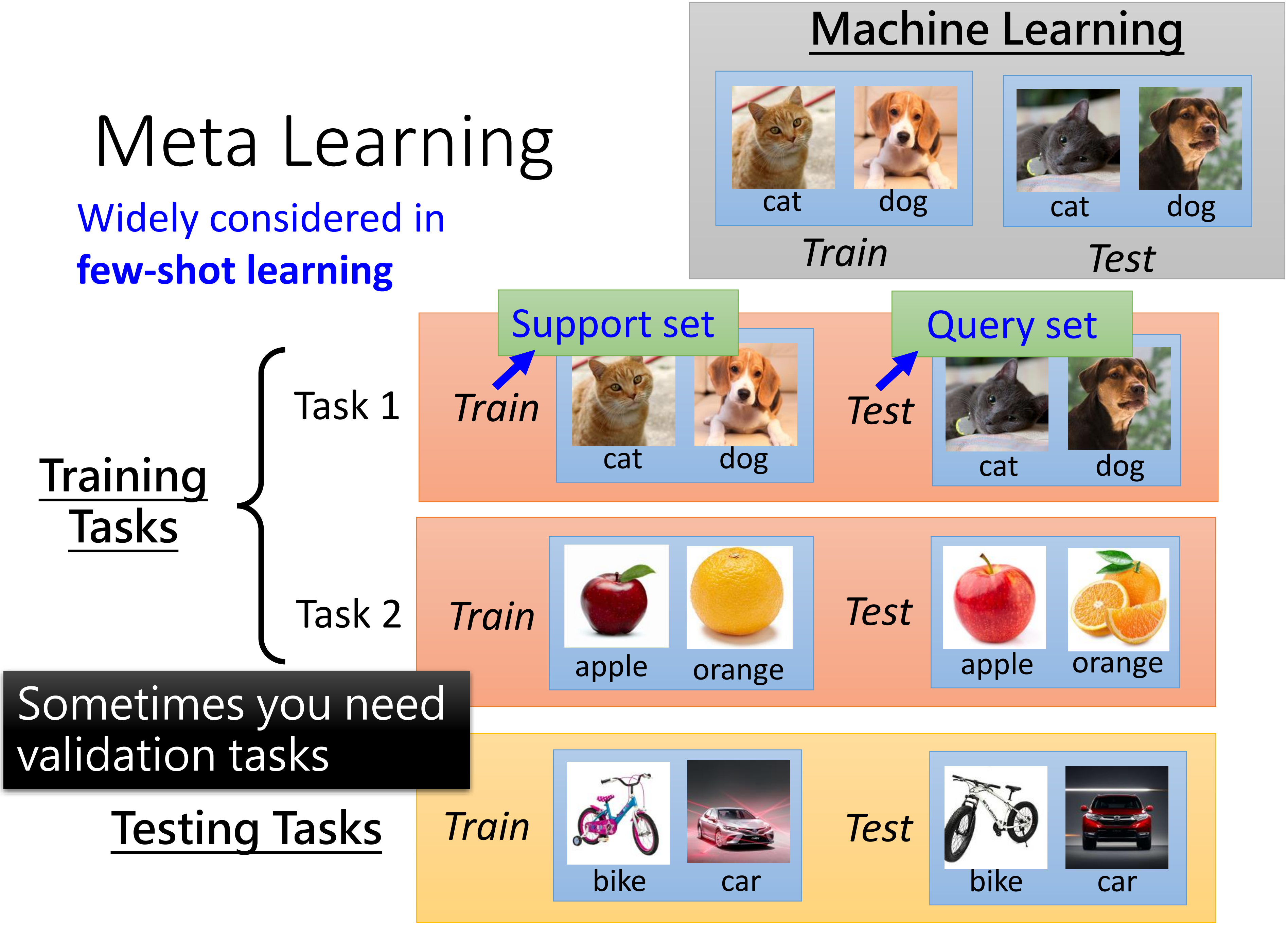
$$L(F) = \sum_{n=1}^{N} l^n$$

N tasks

Testing loss for task n after training

- Defining the goodness of a function F





Meta Learning
Widely considered in
few-shot learning
Machine Learning
cat
dog
Train
cat
dog
Test
Training Tasks
Task 1
Train
Support set
cat
dog
Test
Query set
cat
dog
Task 2
Train
apple
orange
Test
apple
orange
Sometimes you need validation tasks
Testing Tasks
Train
bike
car
Test
bike
car

Meta Learning

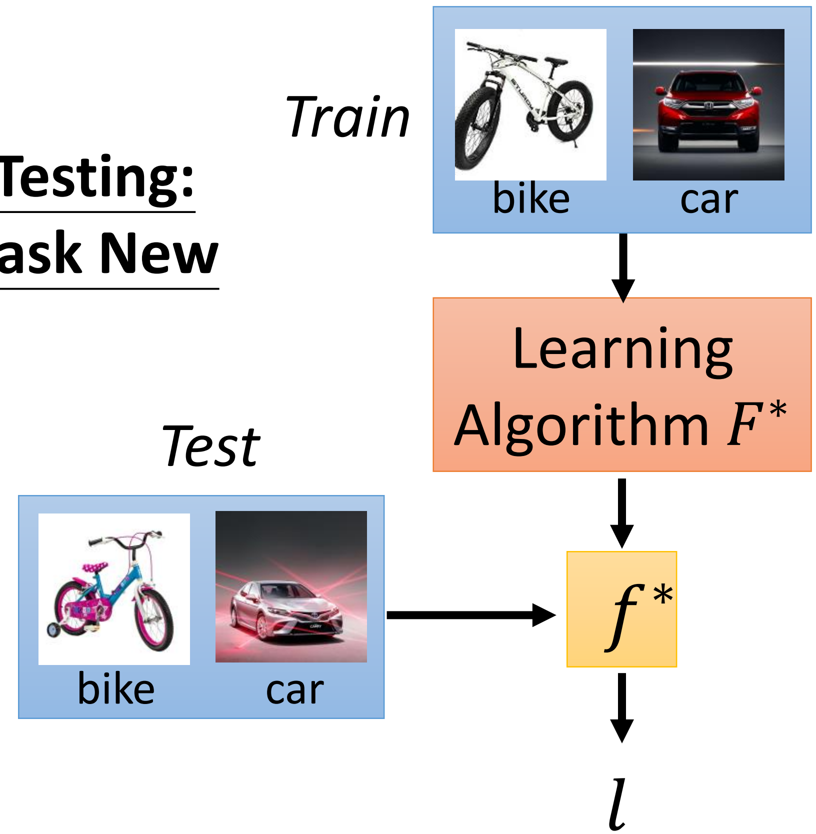
- Defining the goodness of a function F

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

- Find the best function F^*

$$F^* = arg\min_{F} L(F)$$

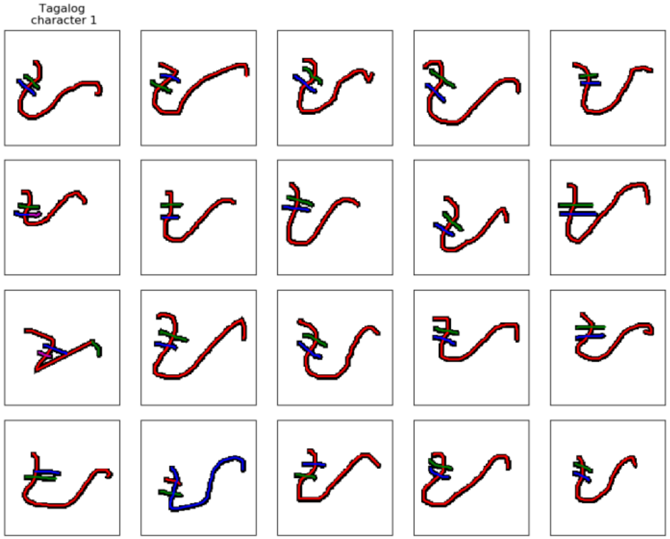
Testing:
Task New



Omniglot

https://github.com/brendenlake/omniglot

- 1623 characters
- Each has 20 examples



Demo of Reptile:
https://openai.com/blog/reptile/

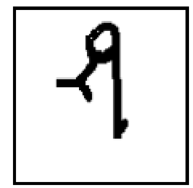
Omniglot – Few-shot Classification

- N-ways K-shot classification: In each training and test tasks, there are N classes, each has K examples.

20 ways
1 shot

Each character represents a class

Training set (Support set)



Testing set (Query set)

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

Techniques Today

- **MAML**
  - Chelsea Finn, Pieter Abbeel, and Sergey Levine, “Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks”, ICML, 2017

- **Reptile**
  - Alex Nichol, Joshua Achiam, John Schulman, On First-Order Meta-Learning Algorithms, arXiv, 2018

MAML

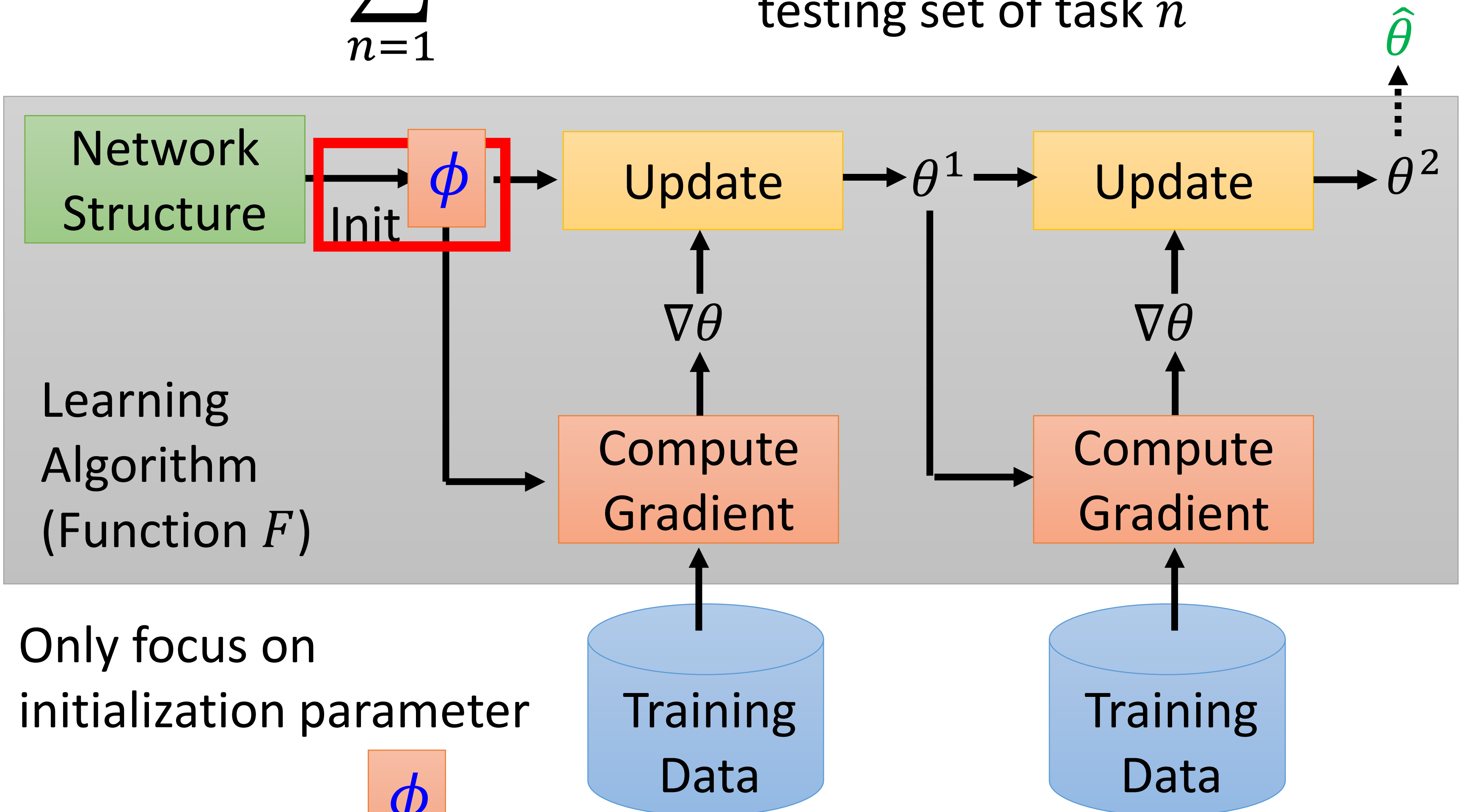
$\hat{\theta}^n$: model learned from task n

$\hat{\theta}^n$ depends on ϕ

Loss Function:

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

$l^n(\hat{\theta}^n)$: loss of task n on the testing set of task n



Only focus on initialization parameter

ϕ

MAML

$\hat{\theta}^n$: model learned from task n

$\hat{\theta}^n$ depends on ϕ

Loss Function:

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

$l^n(\hat{\theta}^n)$: loss of task n on the testing set of task n

How to minimize $L(\phi)$? Gradient Descent

$$\phi \leftarrow \phi - \eta \nabla_{\phi} L(\phi)$$

Model Pre-training

Widely used in transfer learning

Loss Function:

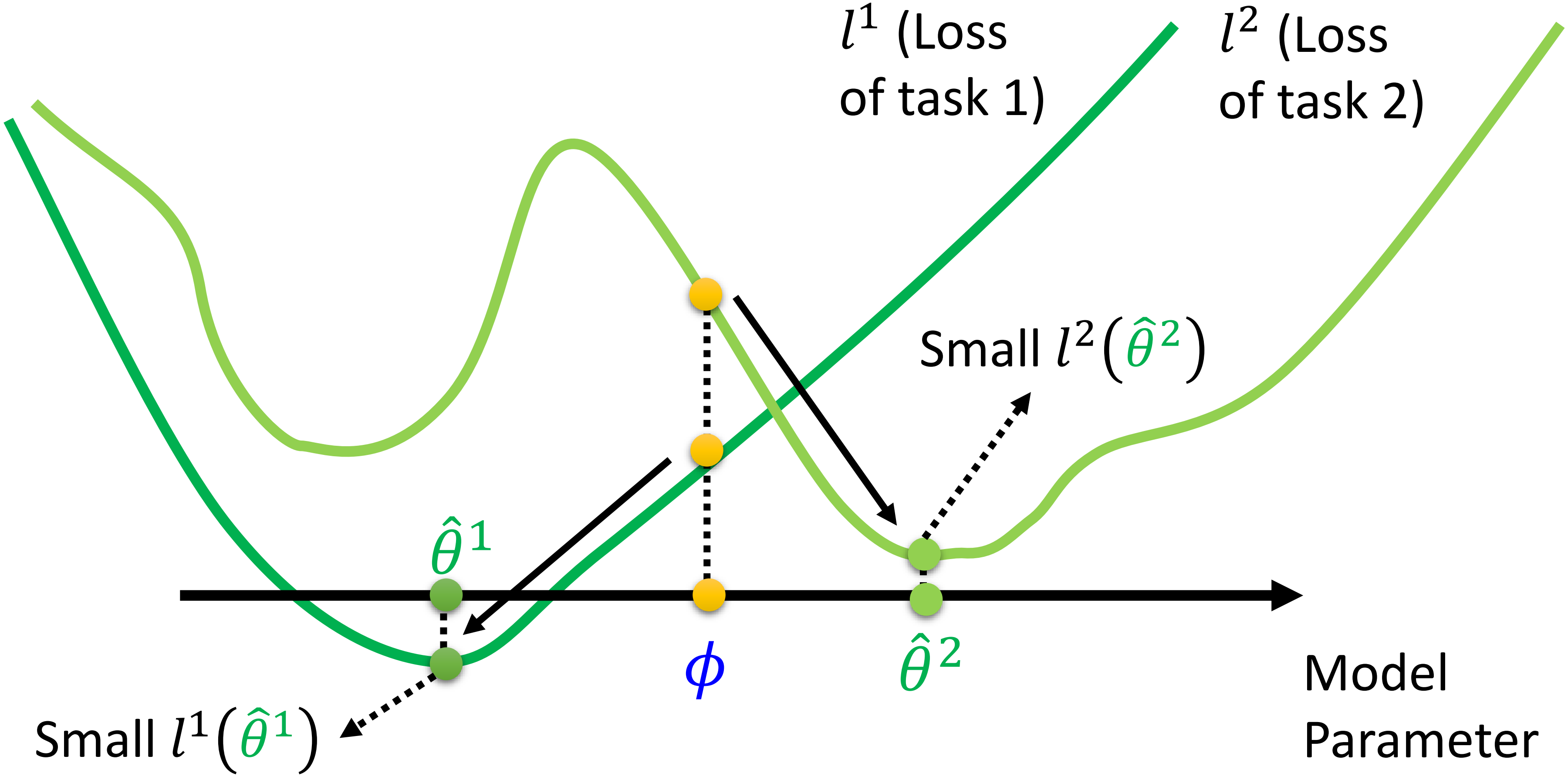
$$L(\phi) = \sum_{n=1}^{N} l^n(\phi)$$

MAML

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

我们不在意 ϕ 在 training task 上表现如何

我们在意用 ϕ 训练出來的 $\hat{\theta}^n$ 表现如何

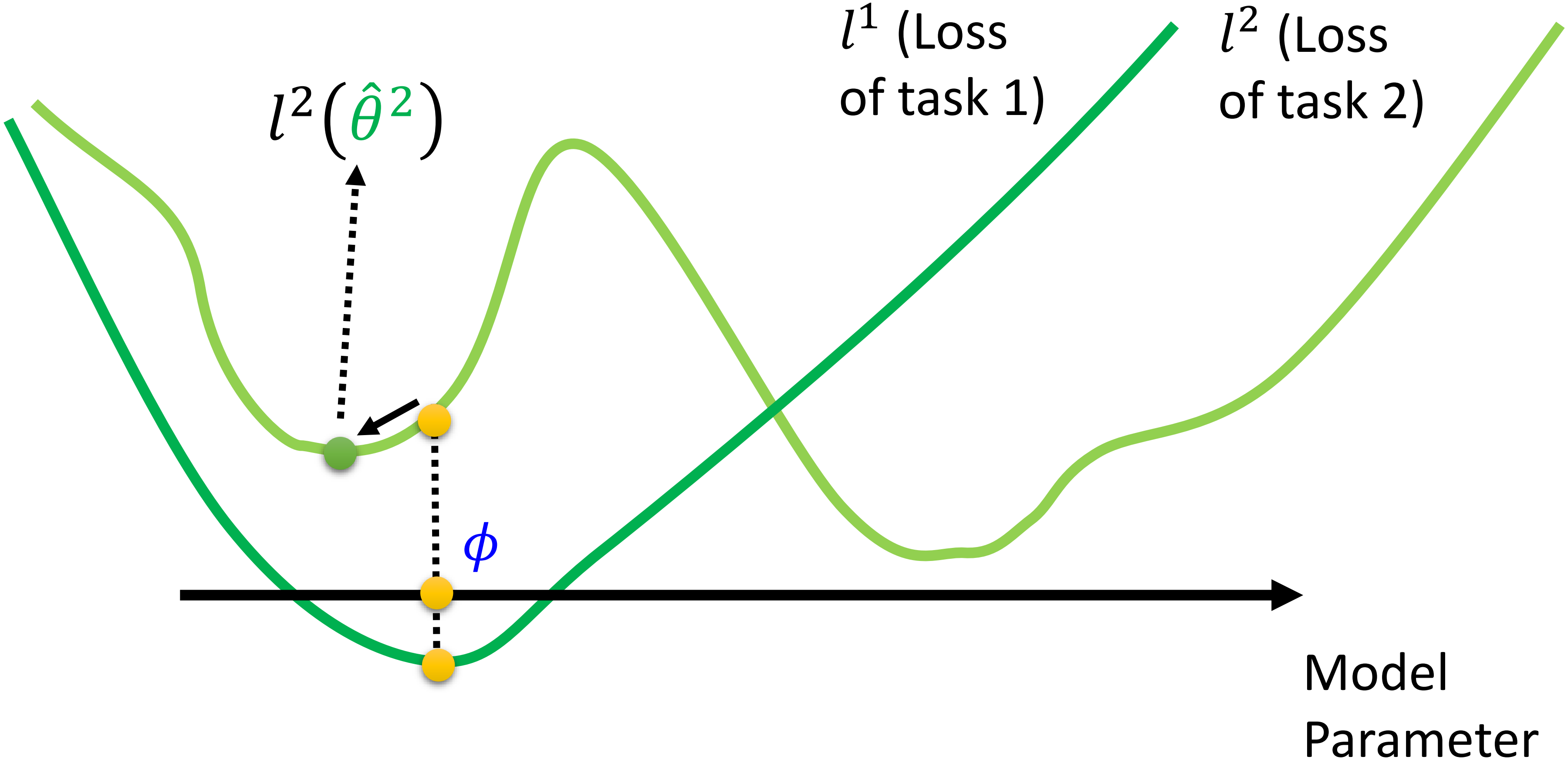


Model Pre-training

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

找寻在所有 task 都最好的 ϕ

并不保证拿 ϕ 去训练以后会得到好的 $\hat{\theta}^n$



MAML

$\hat{\theta}^n$: model learned from task n

Loss Function:

$\hat{\theta}^n$ depends on ϕ

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

$l^n(\hat{\theta}^n)$: loss of task n on the testing set of task n

How to minimize $L(\phi)$? Gradient Descent

$$\phi \leftarrow \phi - \eta \nabla_{\phi} L(\phi)$$

Find ϕ achieving good performance **after training** | 潜力

Model Pre-training

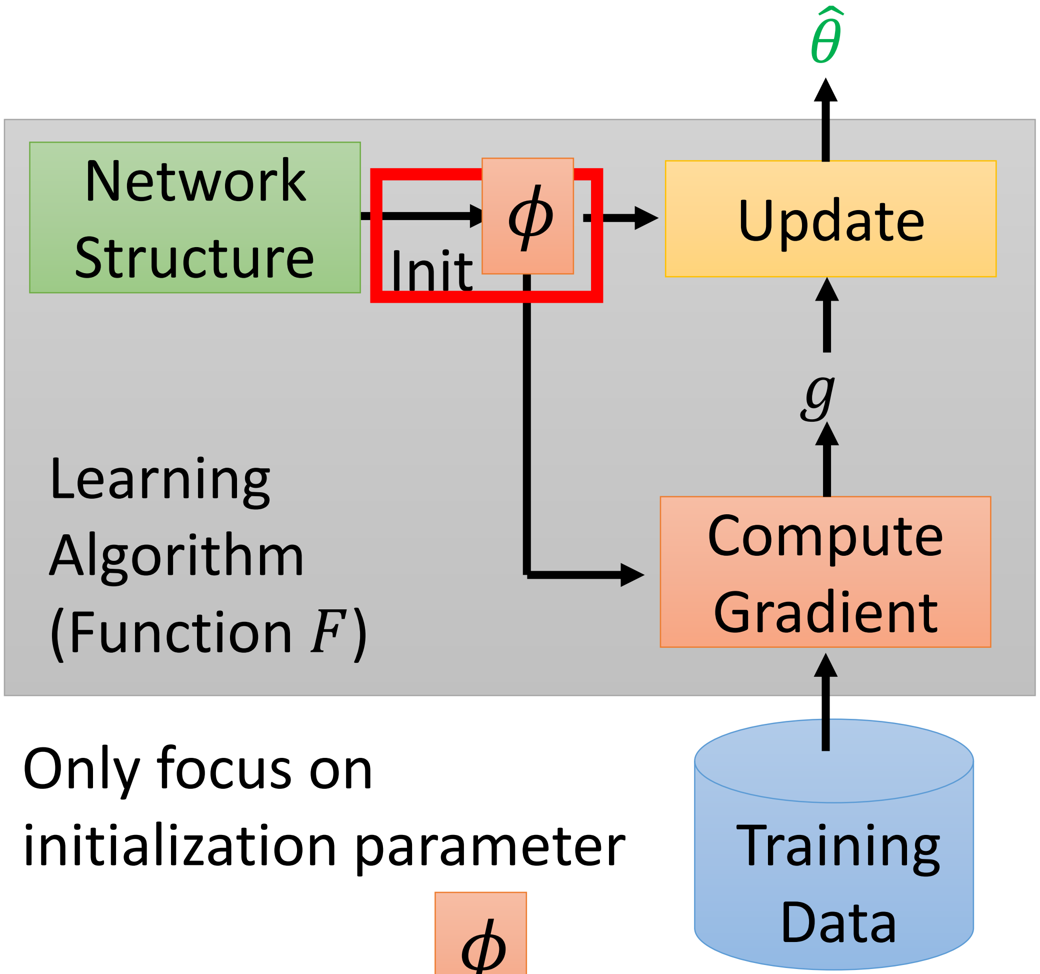
Loss Function:

Widely used in transfer learning

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

Find ϕ achieving good performance | 现在表现如何

MAML



Only focus on initialization parameter ϕ

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

$$\phi \leftarrow \phi - \eta \nabla_{\phi} L(\phi)$$

Considering one-step training:

$$\hat{\theta} = \phi - \varepsilon \nabla_{\phi} l(\phi)$$

Warning of Math

$$\phi \leftarrow \phi - \eta \nabla_{\phi} L(\phi)$$

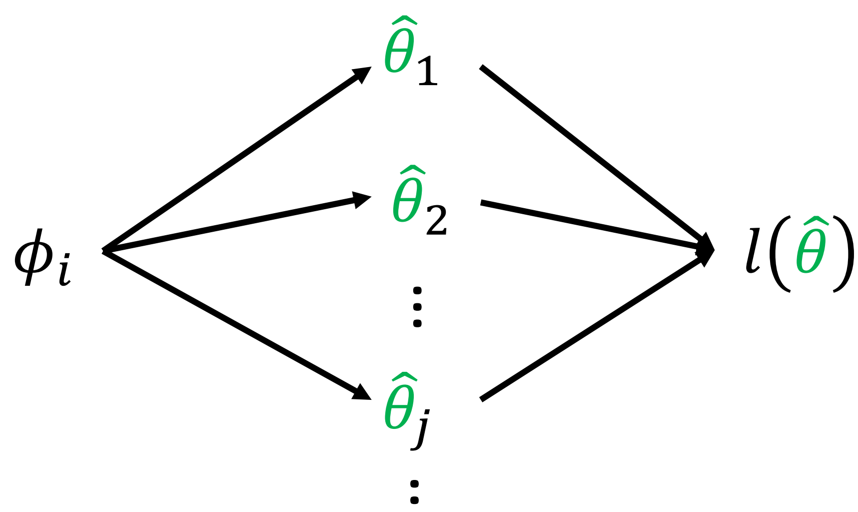
$$L(\phi) = \sum_{n=1}^{N} l^{n}\left(\hat{\theta}^{n}\right)$$

$$\hat{\theta} = \phi - \varepsilon \nabla_{\phi} l(\phi)$$

$$\nabla_{\phi} L(\phi) = \nabla_{\phi} \sum_{n=1}^{N} l^{n}\left(\hat{\theta}^{n}\right) = \sum_{n=1}^{N} \underline{\nabla_{\phi} l^{n}\left(\hat{\theta}^{n}\right)}$$

$$\frac{\partial l(\hat{\theta})}{\partial \phi_i} = \sum_{j} \frac{\partial l(\hat{\theta})}{\partial \hat{\theta}_j} \frac{\partial \hat{\theta}_j}{\partial \phi_i}$$

$$\nabla_{\phi} l(\hat{\theta}) = \begin{bmatrix} \partial l(\hat{\theta}) / \partial \phi_1 \\ \partial l(\hat{\theta}) / \partial \phi_2 \\ \vdots \\ \boxed{\partial l(\hat{\theta}) / \partial \phi_i} \\ \vdots \end{bmatrix}$$



$$\phi \leftarrow \phi - \eta \nabla_{\phi} L(\phi)$$

$$L(\phi) = \sum_{n=1}^{N} l^n\left(\hat{\theta}^n\right)$$

$$\boxed{\hat{\theta} = \phi - \varepsilon \nabla_{\phi} l(\phi)}$$

$$\nabla_{\phi} L(\phi) = \nabla_{\phi} \sum_{n=1}^{N} l^n\left(\hat{\theta}^n\right) = \sum_{n=1}^{N} \underline{\nabla_{\phi} l^n\left(\hat{\theta}^n\right)}$$

$$\frac{\partial l\left(\hat{\theta}\right)}{\partial \phi_i} = \sum_j \frac{\partial l\left(\hat{\theta}\right)}{\partial \hat{\theta}_j} \boxed{\frac{\partial \hat{\theta}_j}{\partial \phi_i}} \approx \frac{\partial l\left(\hat{\theta}\right)}{\partial \hat{\theta}_i}$$

$$\hat{\theta}_j = \phi_j - \varepsilon \frac{\partial l(\phi)}{\partial \phi_j}$$

$i \neq j$:

$$\frac{\partial \hat{\theta}_j}{\partial \phi_i} = -\varepsilon \cancel{\frac{\partial l(\phi)}{\partial \phi_i \partial \phi_j}} \approx 0$$

$i = j$:

$$\frac{\partial \hat{\theta}_j}{\partial \phi_i} = 1 - \varepsilon \cancel{\frac{\partial l(\phi)}{\partial \phi_i \partial \phi_j}} \approx 1$$

$$\nabla_{\phi} l\left(\hat{\theta}\right) = \begin{bmatrix} \partial l\left(\hat{\theta}\right)/\partial \phi_1 \\ \partial l\left(\hat{\theta}\right)/\partial \phi_2 \\ \vdots \\ \boxed{\partial l\left(\hat{\theta}\right)/\partial \phi_i} \\ \vdots \end{bmatrix}$$

$$\phi \leftarrow \phi - \eta \nabla_{\phi} L(\phi)$$

$$L(\phi) = \sum_{n=1}^{N} l^n\left(\hat{\theta}^n\right)$$

$$\hat{\theta} = \phi - \varepsilon \nabla_{\phi} l(\phi)$$

$$\nabla_{\phi} L(\phi) = \nabla_{\phi} \sum_{n=1}^{N} l^n\left(\hat{\theta}^n\right) = \sum_{n=1}^{N} \nabla_{\phi} l^n\left(\hat{\theta}^n\right)$$

$$\hat{\theta}^n$$

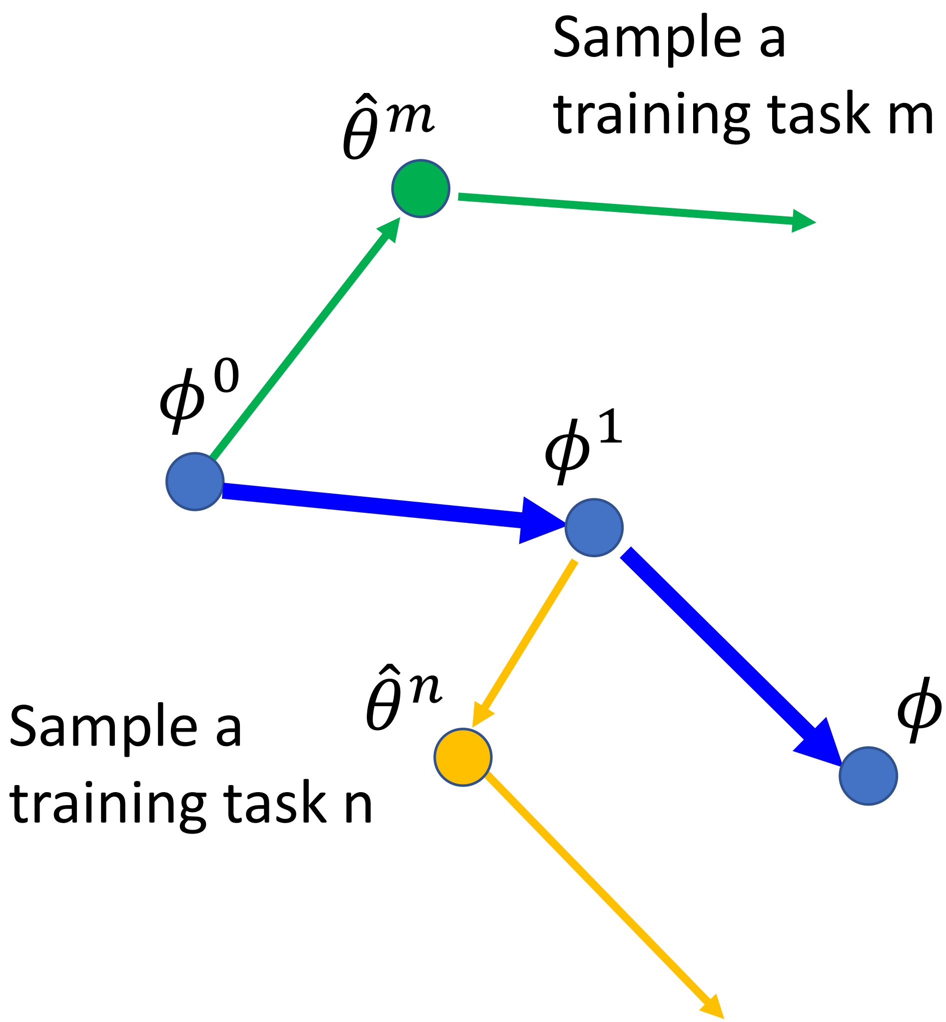
$$\frac{\partial l(\hat{\theta})}{\partial \phi_i} = \sum_j \frac{\partial l(\hat{\theta})}{\partial \hat{\theta}_j} \frac{\partial \hat{\theta}_j}{\partial \phi_i} \approx \frac{\partial l(\hat{\theta})}{\partial \hat{\theta}_i}$$

products, which is supported by standard deep learning libraries such as TensorFlow (Abadi et al., 2016). In our experiments, we also include a comparison to dropping this backward pass and using a first-order approximation, which we discuss in Section 5.2.

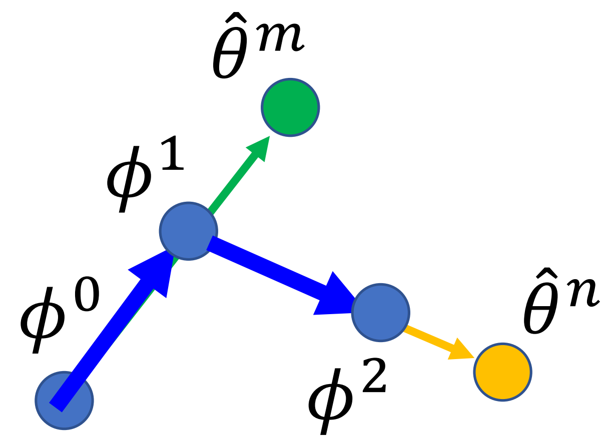
$$\nabla_{\phi} l(\hat{\theta}) = \begin{bmatrix} \partial l(\hat{\theta})/\partial \phi_1 \\ \partial l(\hat{\theta})/\partial \phi_2 \\ \vdots \\ \partial l(\hat{\theta})/\partial \phi_i \\ \vdots \end{bmatrix} = \begin{bmatrix} \partial l(\hat{\theta})/\partial \hat{\theta}_1 \\ \partial l(\hat{\theta})/\partial \hat{\theta}_2 \\ \vdots \\ \partial l(\hat{\theta})/\partial \hat{\theta}_i \\ \vdots \end{bmatrix} = \nabla_{\hat{\theta}} l(\hat{\theta})$$

End of Warning

MAML – Real Implementation



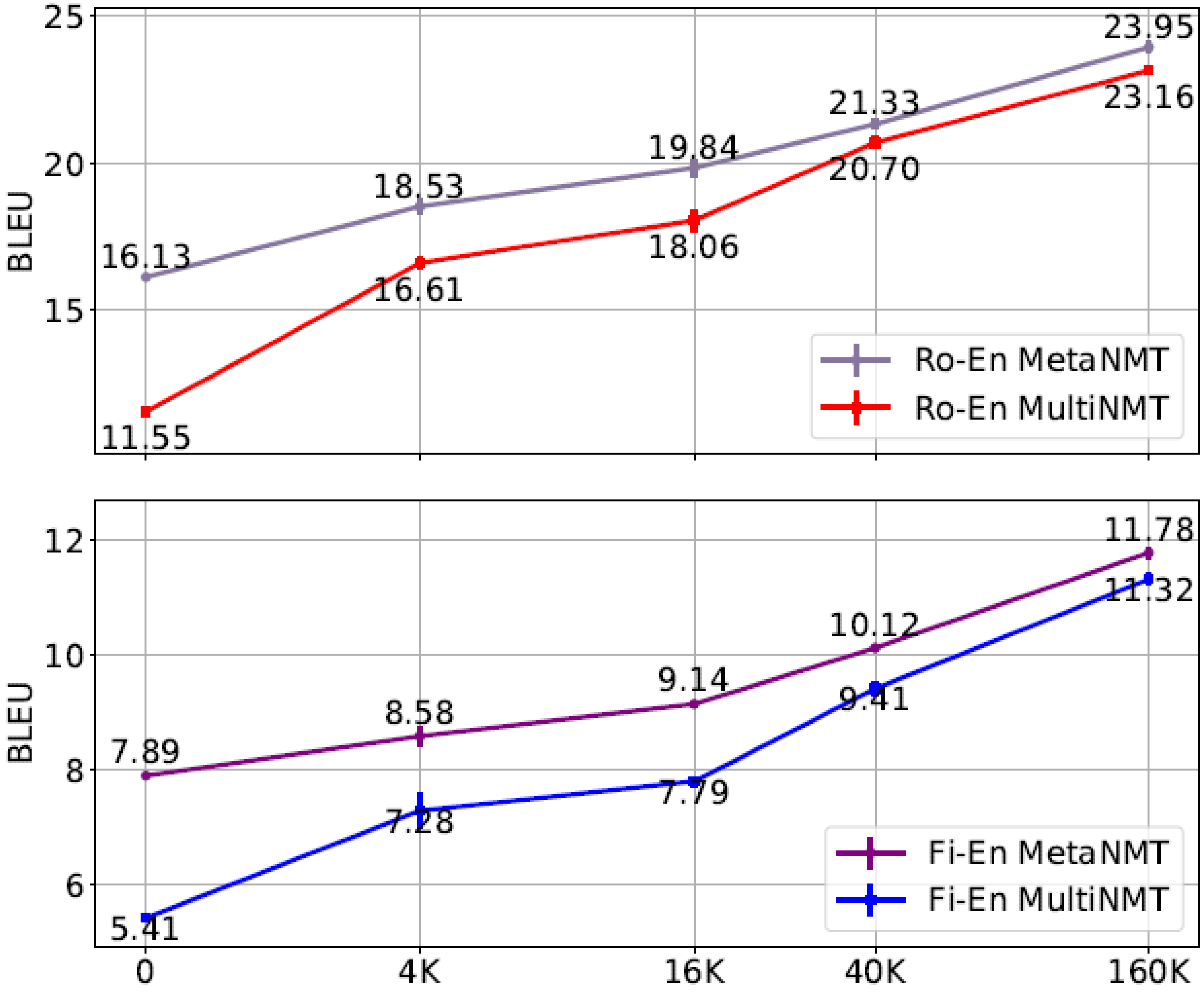
Model Pre-training



Translation

18 training tasks: 18 different languages translating to English

2 validation tasks: 2 different languages translating to English



Ro = Romanian

Fi = Finnish

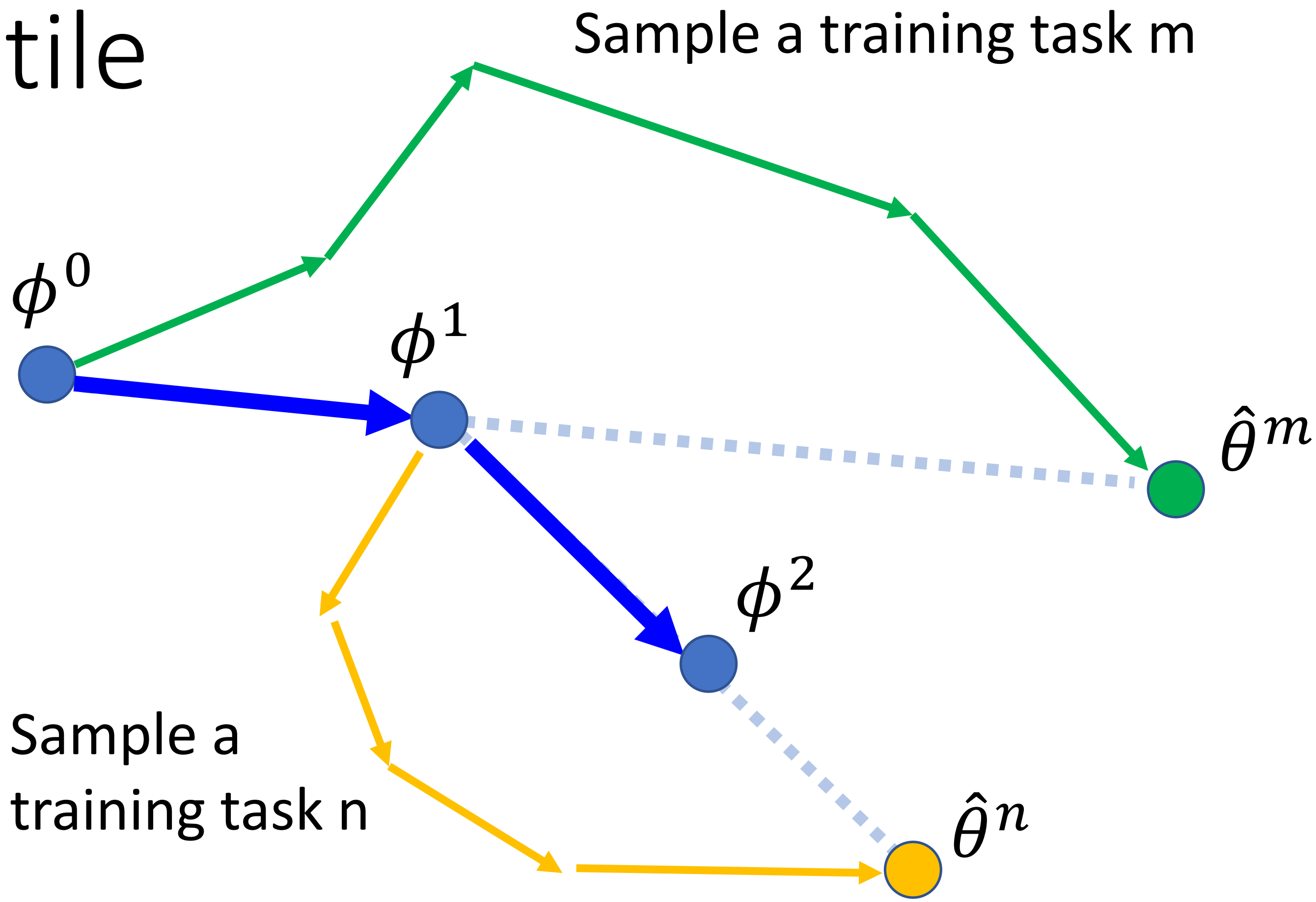
https://arxiv.org/abs/1808.08437

Techniques Today

- **MAML**
  - Chelsea Finn, Pieter Abbeel, and Sergey Levine, “Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks”, ICML, 2017
- **Reptile**
  - Alex Nichol, Joshua Achiam, John Schulman, On First-Order Meta-Learning Algorithms, arXiv, 2018

https://openai.com/blog/reptile/

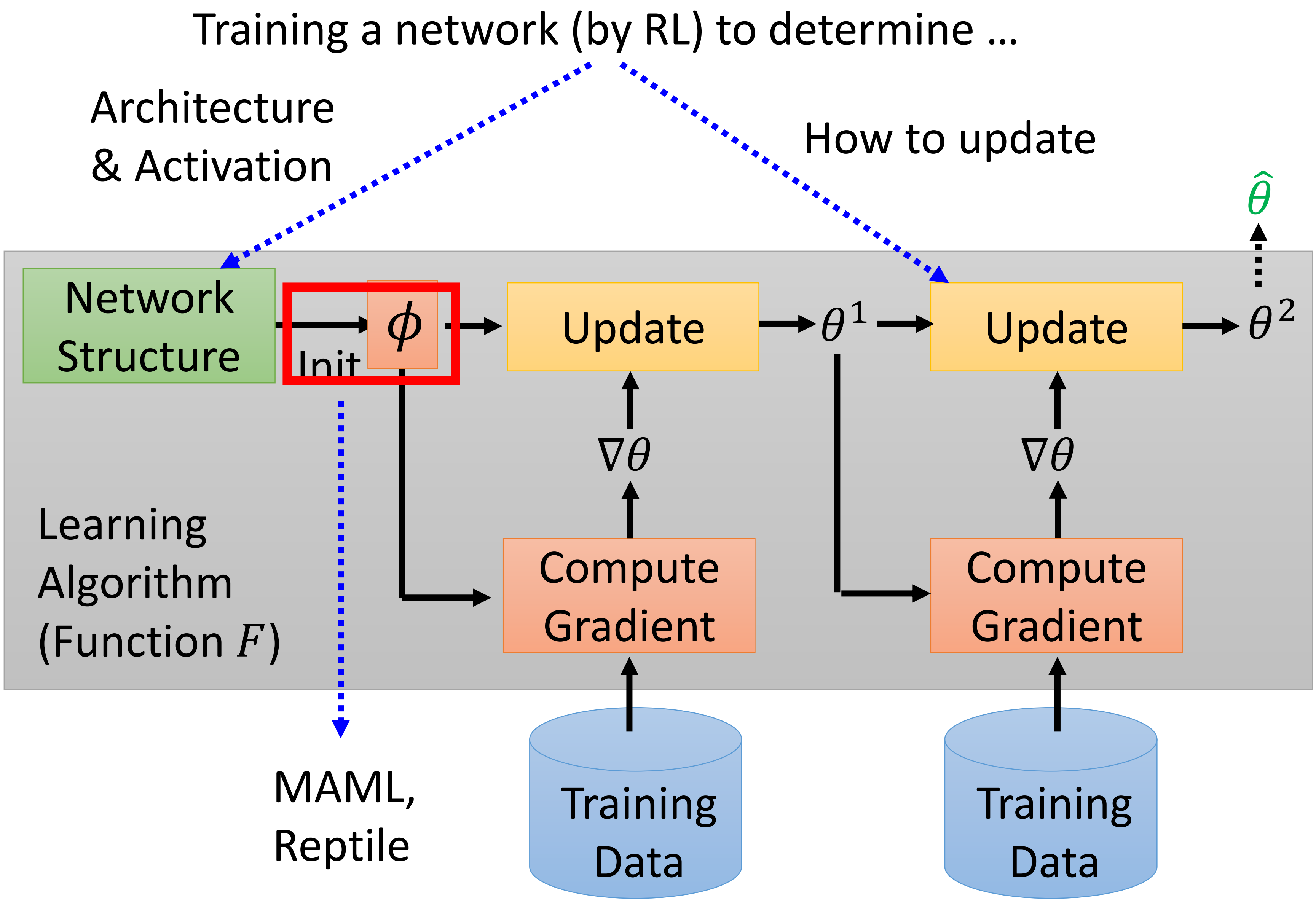
Reptile



You might be thinking "isn't this the same as training on the expected loss $\mathbb{E}_\tau[L_\tau]$?" and then checking if the date is April 1st. Indeed, if the partial minimization consists of a single gradient step, then this algorithm corresponds to minimizing the expected loss:

(this sentence is removed in the updated version)

More … Video: https://www.youtube.com/watch?v=c10nxBcSH14



Turtles all the way down ...... ?

- We learn the initialization parameter ϕ by gradient descent
- What is the initialization parameter ϕ^0 for initialization parameter ϕ?

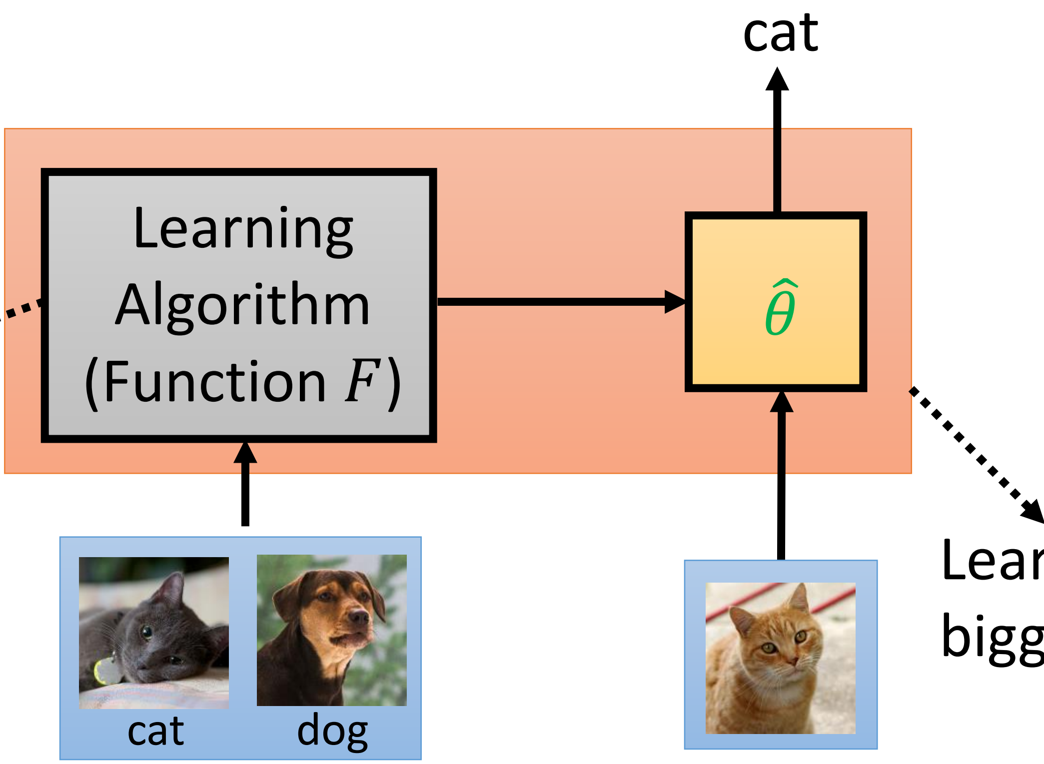
Learn

Learn to learn

Learn to learn to learn

Crazy Idea?

- How about learning algorithm beyond gradient descent?



Just a network

Design network architecture
= Design training algorithm?

Learn an even bigger function

Training Data

Testing Data