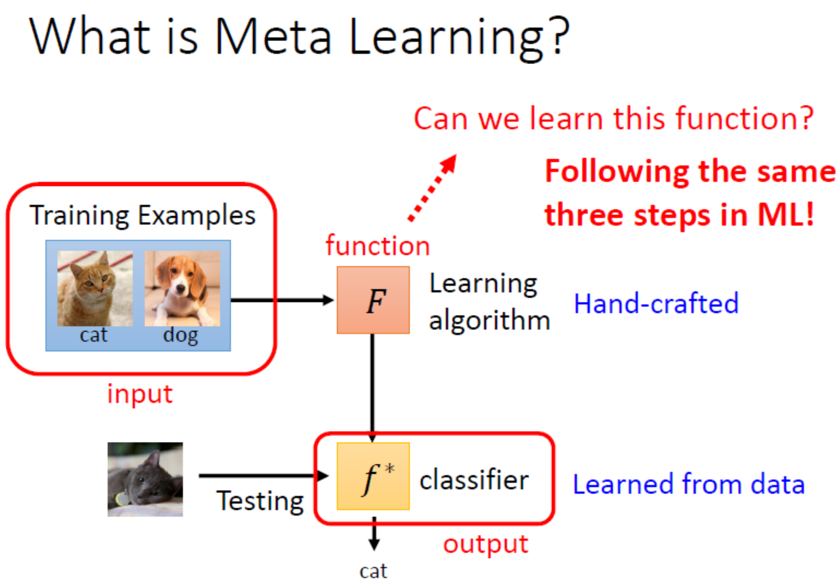
15-Meta Learning（元學習）

1. 什麼是 Meta Learning？

將訓練資料輸入進 F，F 直接輸出一個模型 f^* 可以直接進行測試



meta learning 就是要找一個 **learning algorithm** F

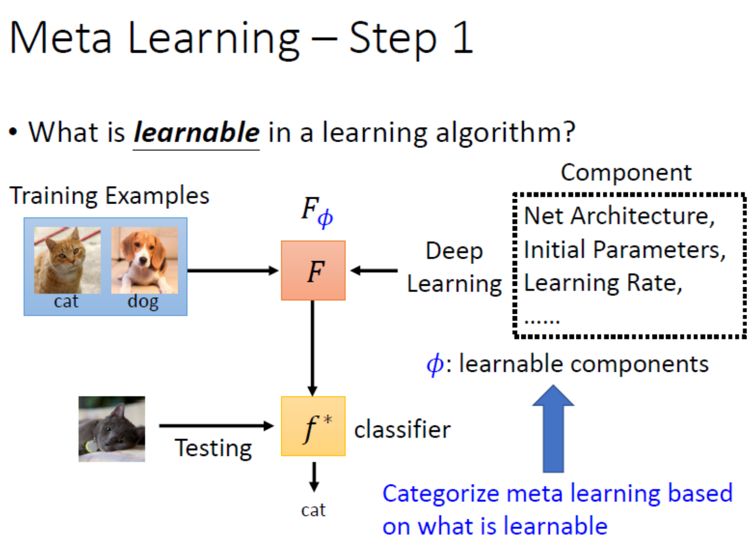
2. 尋找 Learning Algorithm 三步驟

注意：

任務有**訓練任務**與**測試任務**之別

Step 1：What is learnable？

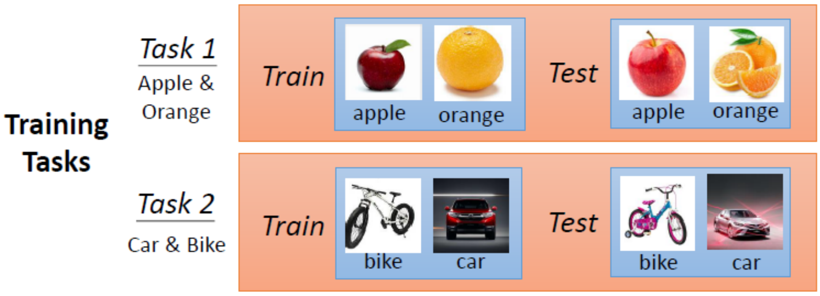
決定 learning algorithm 中要被學的 components（網路架構、初始參數、學習率等等），以 ϕ 表示



不同的 meta learning 方法的差異在於 components 的選擇

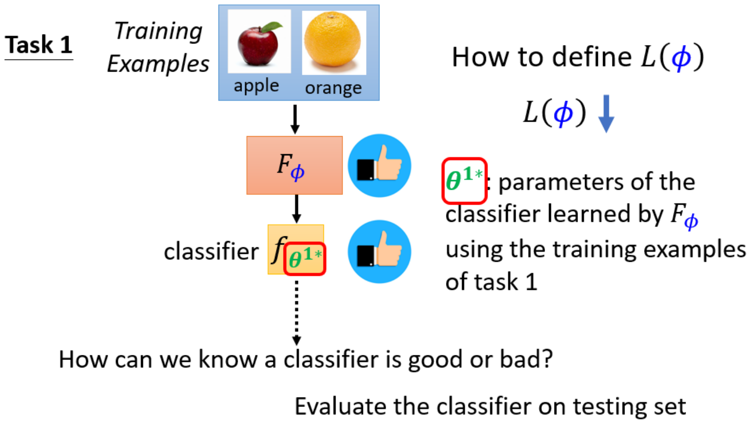
Step 2：Define loss function $L(\phi)$

訓練資料來自很多訓練任務，每個任務中有訓練集和測試集

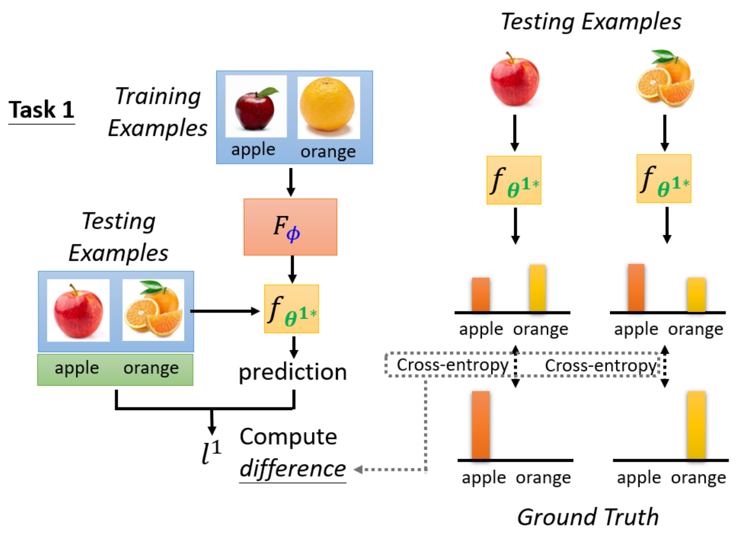


定義 loss function $L(\phi)$：

1. 將某一任務的訓練資料輸入進 learning algorithm F_ϕ，得到模型 $f_{\theta^{1*}}$

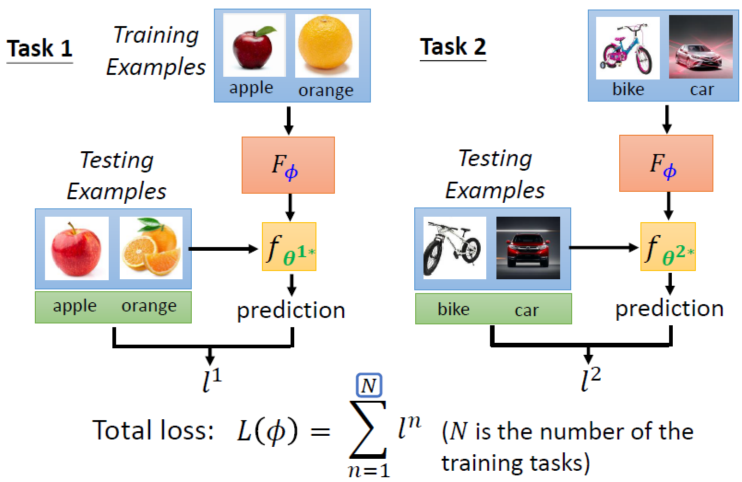


2. 使用對應任務的測試資料對模型 $f_{\theta^{1*}}$ 進行測試，計算每個預測資料的結果與 ground truth 之間的 cross entropy，並將全部的 cross entropy 加總得到 l^1



- l^1 若越小，表示模型 $f_{\theta^{1*}}$ 越好，代表是好 learning algorithm F_ϕ
- l^1 若越大，表示模型 $f_{\theta^{1*}}$ 越不好，代表是差 learning algorithm F_ϕ

3. 將下一任務的訓練資料輸入進 learning algorithm F_ϕ，得到模型 $f_{\theta^{2*}}$，並計算每個預測資料的結果與 ground truth 之間的 cross entropy，並將全部的 cross entropy 加總得到 l^2
4. 以此類推得到全部訓練任務的 l，並加總得到 learning algorithm 的 loss $L(\phi)$



注意：

在一般機器學習中，loss 是根據訓練資料得來的；而在 **meta learning 中，loss 是根據訓練任務中的測試資料得來的**

Step 3：Optimazation

- Loss function for learning algorithm $L(\phi) = \sum_{n=1}^{N} l^n$
- Find ϕ that can minimize $L(\phi)$ $\phi^* = arg \min_{\phi} L(\phi)$
- Using the optimization approach you know

  If you know how to compute $\partial L(\phi)/\partial \phi$

  Gradient descent is your friend.

  What if $L(\phi)$ is not differentiable?

  Reinforcement Learning / Evolutionary Algorithm

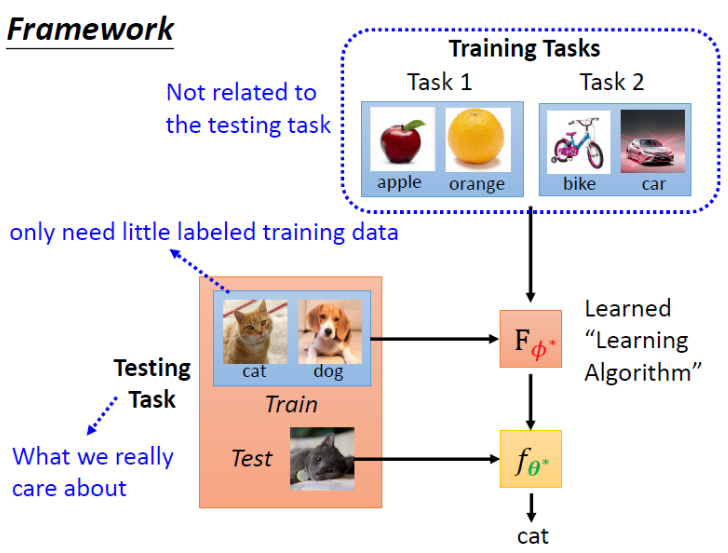
  Now we have a learned "learning algorithm" F_{ϕ^*}

- 若 $\frac{\partial L(\phi)}{\partial \phi}$ 可微，則可以使用 gradient descent 找出 ϕ^* 最小化 $L(\phi)$
- 若 $\frac{\partial L(\phi)}{\partial \phi}$ 不可微（ϕ 有可能是一個 network 架構），使用 RL 硬 train，或其他方法

最終得到一 learning algorithm F_{ϕ^*} 使 $L(\phi)$ 最小化

Framework

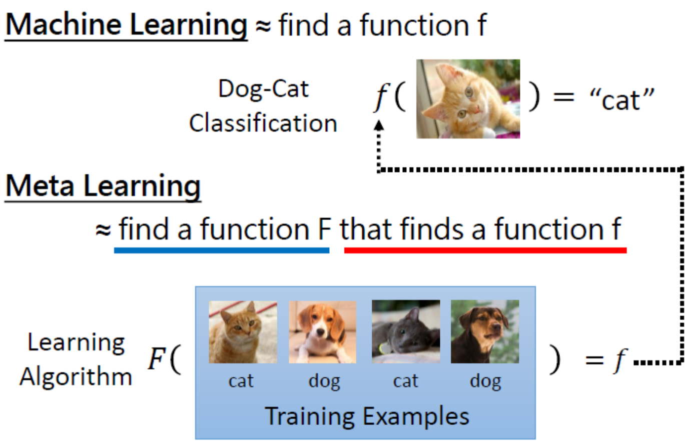
我們真正關心的是在**測試任務**上，learning algorithm F_{ϕ^*} 的性能



將**測試任務**中的訓練資料輸入進 learning algorithm F_{ϕ^*} 進行訓練得到模型 f_{θ^*}，f_{θ^*} 就是我們最終想要的模型

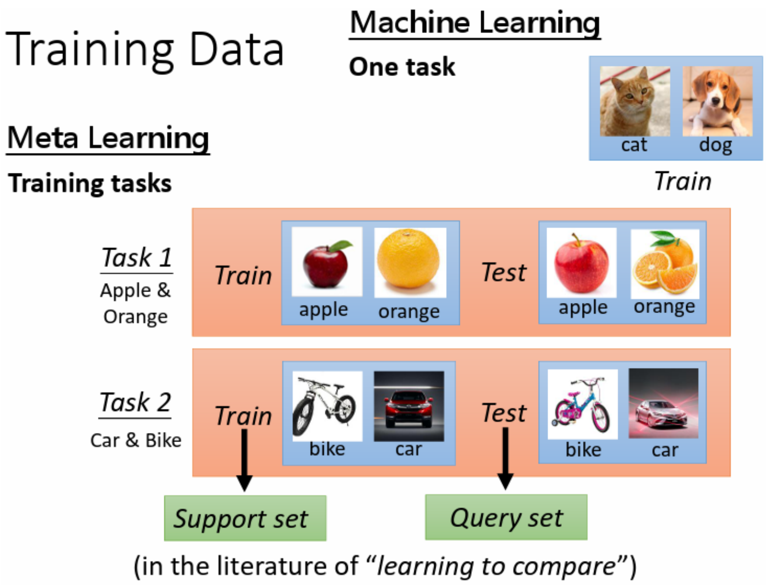
3. ML vs Meta

3.1 Goal



- ML：找到一個能完成任務的函數 f
- Meta：找到一個 learning algorithm F，能夠找到能完成任務的函數 f

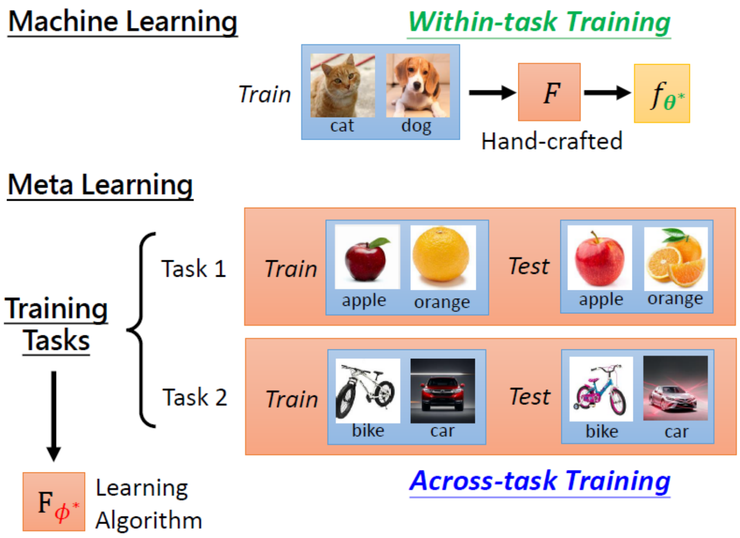
3.2 Training Data



- ML：使用一個任務中的訓練資料進行訓練
- Meta：使用若干個**訓練任務**進行訓練，每個**訓練任務中都有訓練資料及測試資料**
  - **Support set：訓練任務**中的**訓練資料**
  - **Query set：訓練任務**中的**測試資料**

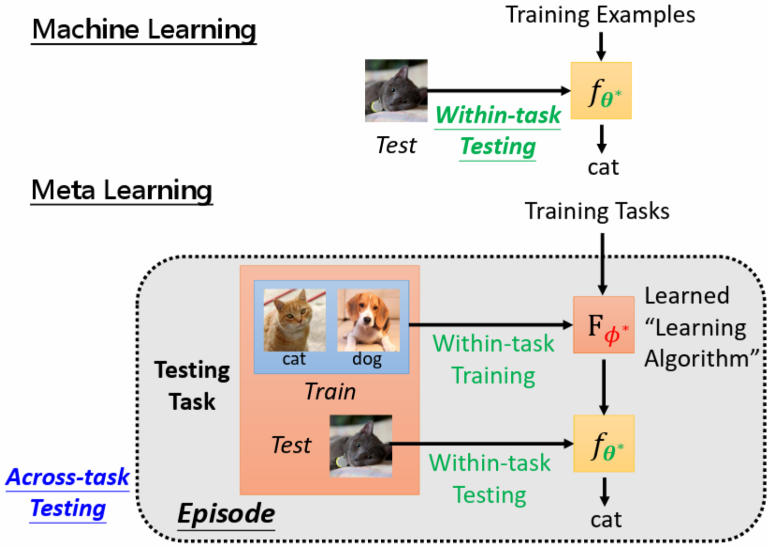
3.3 Framework

3.3.1 Training



- ML：人為設定 learning algorithm，稱作 **Within-task Training**
- Meta：多個任務上訓練得到 learning algorithm，稱作 **Across-task Training**

3.3.2 Testing

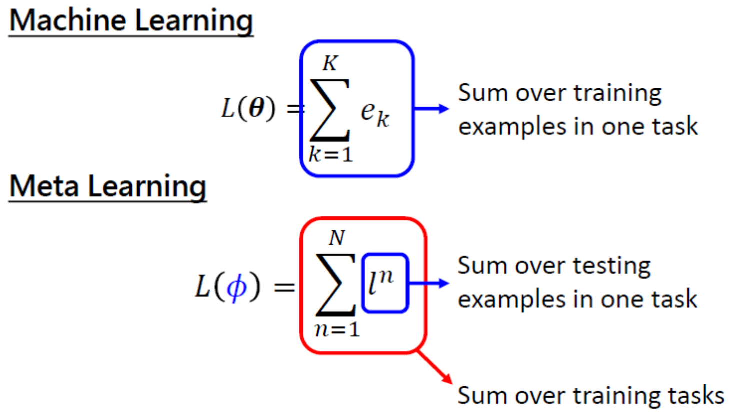


- ML：直接使用訓練得到的模型在任務中對測試資料進行測試，稱作 **Within-task Testing**
- Meta：需要測試的是 learning algorithm，稱作 **Across-task Testing**
    - learning algorithm 以測試任務的訓練資料做為訓練，稱作 **Within-task Training**

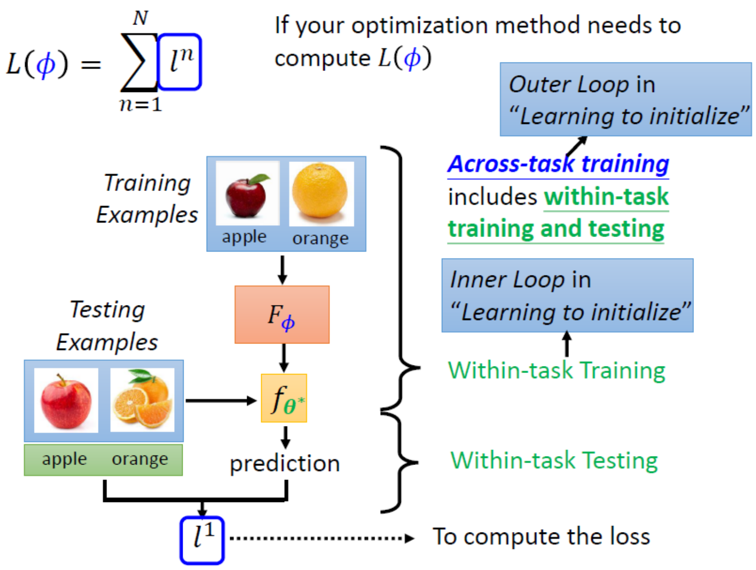
- ∘ 以測試任務的測試資料測試模型，稱作 **Within-task Testing**

Episode = Within-task Training + Within-task Testing

3.3.3 Loss



- ML：對一個任務中所有的測試數據的損失之和
- Meta：l 是一個訓練任務的 loss，L 是所有訓練任務的 loss 總和



計算一個 l，需要一次的 Within-task Training + Within-task Testing 即一個 episode。
將 Within-task Training 稱作 Inner Loop；Across-task training 稱作 Outer Loop

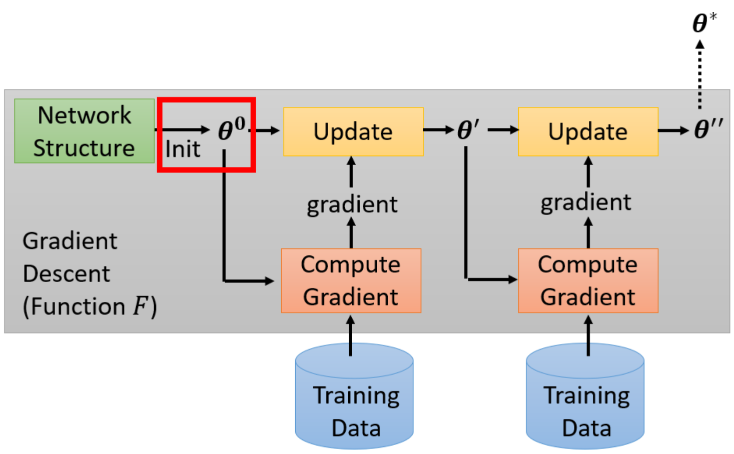
3.4 相同點

- 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 ☺

4. What is learnable in a learning algorithm？

4.1 模型初始參數 θ^0

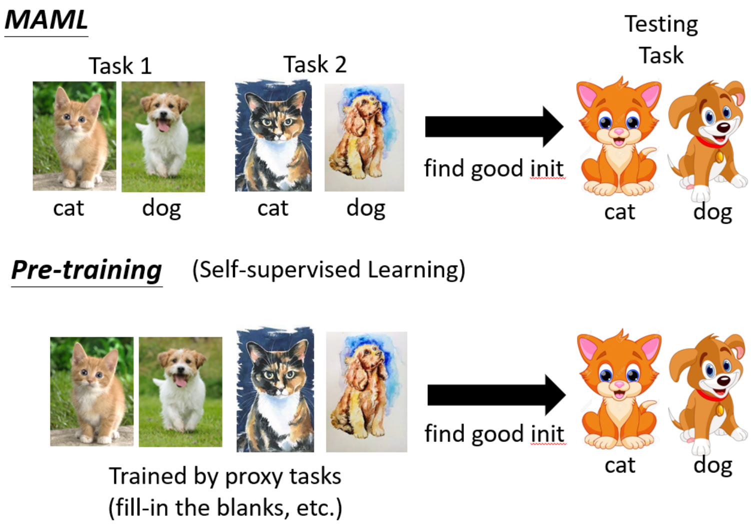
選擇 θ^0 作為 meta learning 要學習的參數 ϕ



方法：

- Model Agnostic MetaLearning（**MAML**）：https://youtu.be/mxqzGwP_Qys
- First order MAML（**FOMAML**）：https://youtu.be/3z997JhL9Oo
- Reptile：https://youtu.be/9jJe2AD35P8

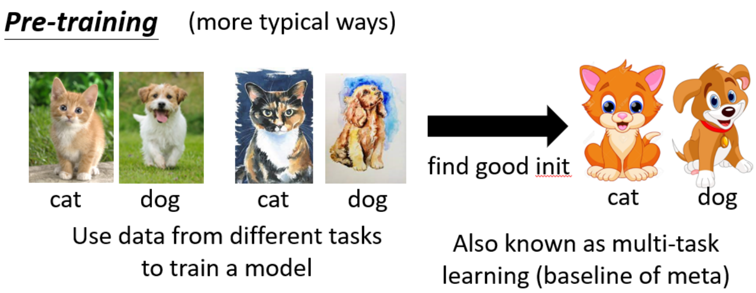
4.1.1 MAML vs Pre-training



- MAML 需要用到有標註的資料
- pre-training（self-supervised learning）使用的資料沒有標註

注意：

meta learning 中所謂不同任務的訓練，實際上就是不同的 domain，所以**也可以說 meta learning 是 domain adaptation 的一種方法**



過去 pre-training 還有其他的方法，如將來自不同任務的資料混在一起進行訓練（稱作 multi-task training）

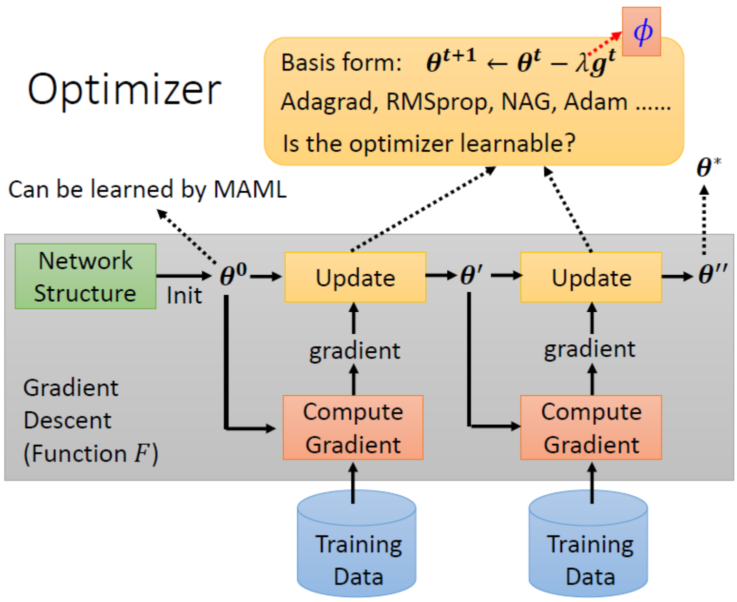
multi-task training 通常作為 meta-learning 的 baseline

學習更多：

https://youtu.be/vUwOA3SNb_E

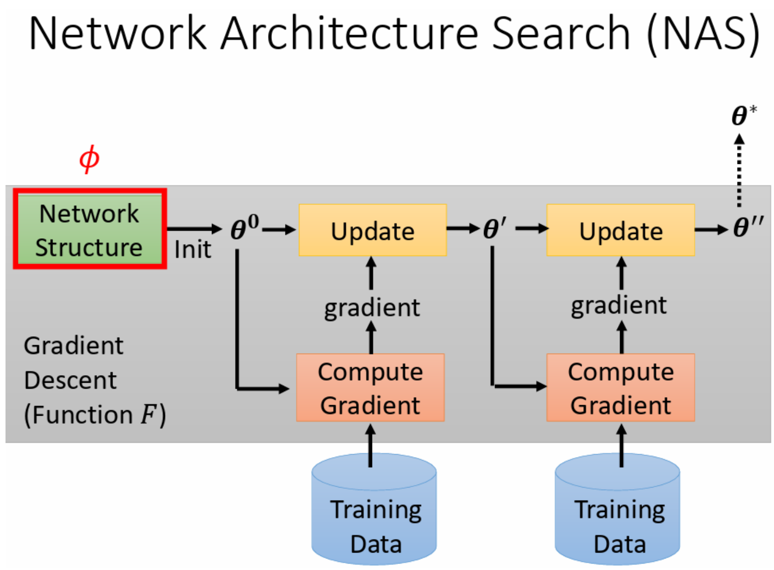
4.2 Optimizer（learning rate, momentum）

選擇 learning rate, momentum 等 optimizer 中的參數作為 meta learning 要學習的參數 ϕ



4.3 Network Architecture Search（NAS）

選擇 network 架構作為 meta learning 要學習的參數 ϕ

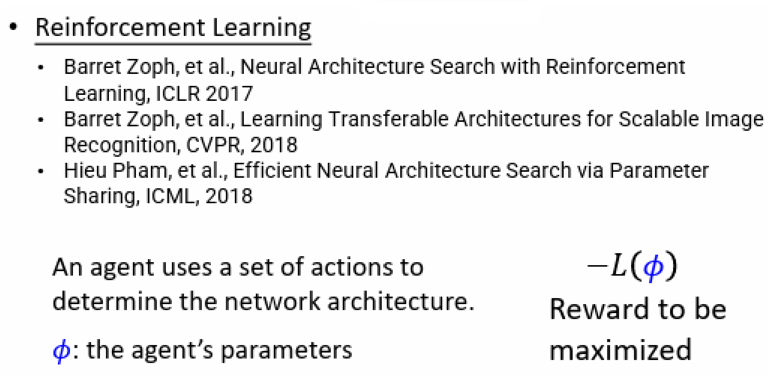


問題：

ϕ 是一個 network 架構，$L(\phi)$ 對 ϕ 不可微

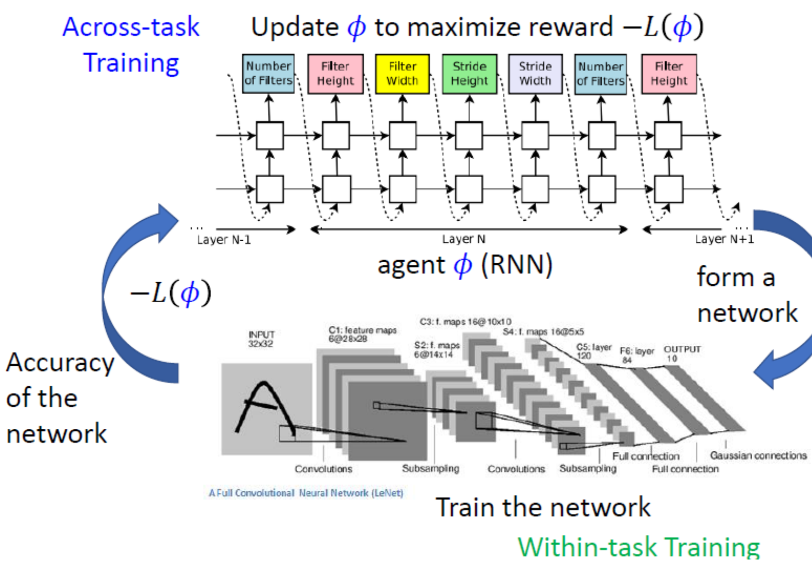
4.3.1 解法 1：Reinforcement Learning

用 RL 硬 train



- ϕ：the agent's parameters
- actor 的輸出：network 寬度、深度等等
- Reward to be maximized：$-L(\phi)$

舉例：



actor 是 RNN 架構；environment 為 network

1. RNN 輸出網路架構（action）
2. 搭建網路架構
3. 測試網路的精確度（observation）
4. 更新 RNN

4.3.2 解法 2：Evolution Algorithm

- Evolution Algorithm
  - Esteban Real, et al., Large-Scale Evolution of Image Classifiers, ICML 2017
  - Esteban Real, et al., Regularized Evolution for Image Classifier Architecture Search, AAAI, 2019
  - Hanxiao Liu, et al., Hierarchical Representations for Efficient Architecture Search, ICLR, 2018

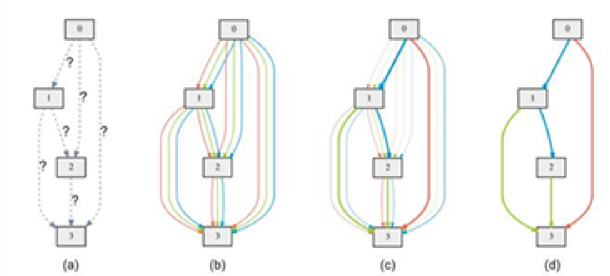
4.3.3 解法 3：DARTS

Differentiable Architecture Search（DARTS）方法修改 network architecture，使之可以微分

$$\hat{\phi} = \arg\min_{\phi} L(\phi) \qquad \nabla_{\phi} L(\phi) = ?$$

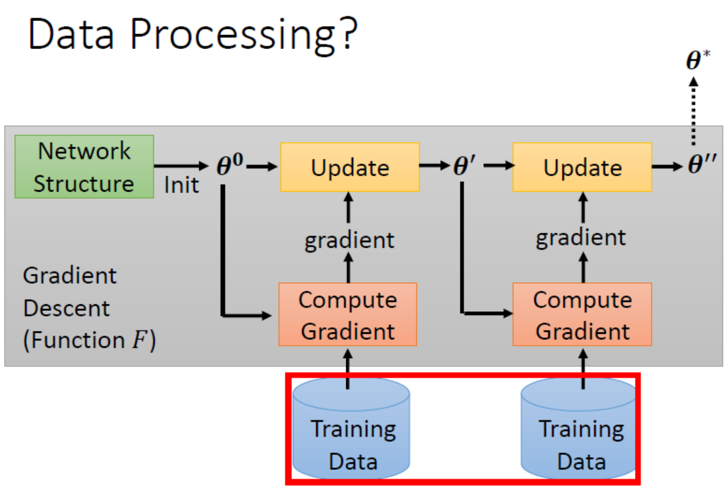
Network Architecture

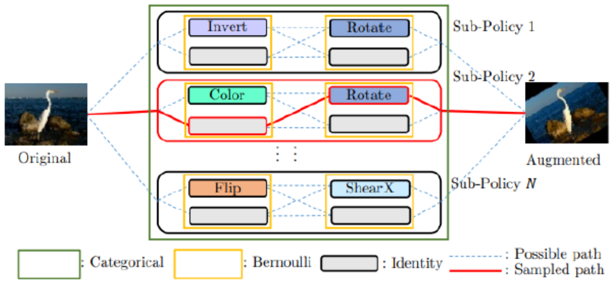
- DARTS Hanxiao Liu, et al., DARTS: Differentiable Architecture Search, ICLR, 2019



4.4 Data Augmentation

選擇 data（augmentation）作為 meta learning 要學習的參數 ϕ

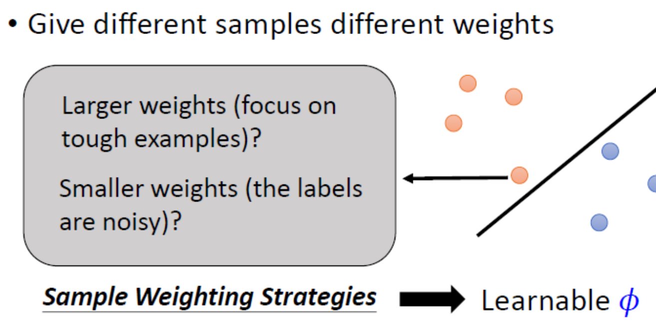




Yonggang Li, Guosheng Hu, Yongtao Wang, Timothy Hospedales, Neil M. Robertson, Yongxin Yang, DADA: Differentiable Automatic Data Augmentation, ECCV, 2020
Daniel Ho, Eric Liang, Ion Stoica, Pieter Abbeel, Xi Chen, Population Based Augmentation: Efficient Learning of Augmentation Policy Schedules, ICML, 2019
Ekin D. Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, Quoc V. Le, AutoAugment: Learning Augmentation Policies from Data, CVPR, 2019

4.5 Sample Reweightnig

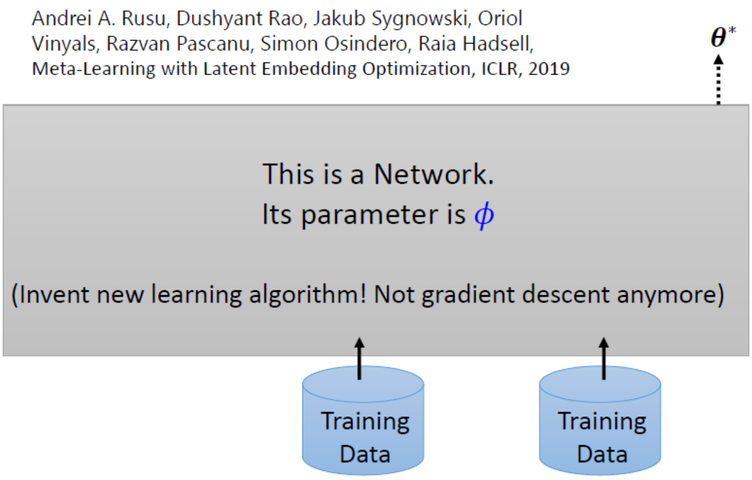
選擇 sample 的 weight 作為 meta learning 要學習的參數 ϕ



Jun Shu, Qi Xie, Lixuan Yi, Qian Zhao, Sanping Zhou, Zongben Xu, Deyu Meng, Meta-Weight-Net: Learning an Explicit Mapping For Sample Weighting, NeurIPS, 2019
Mengye Ren, Wenyuan Zeng, Bin Yang, Raquel Urtasun, Learning to Reweight Examples for Robust Deep Learning, ICML, 2018

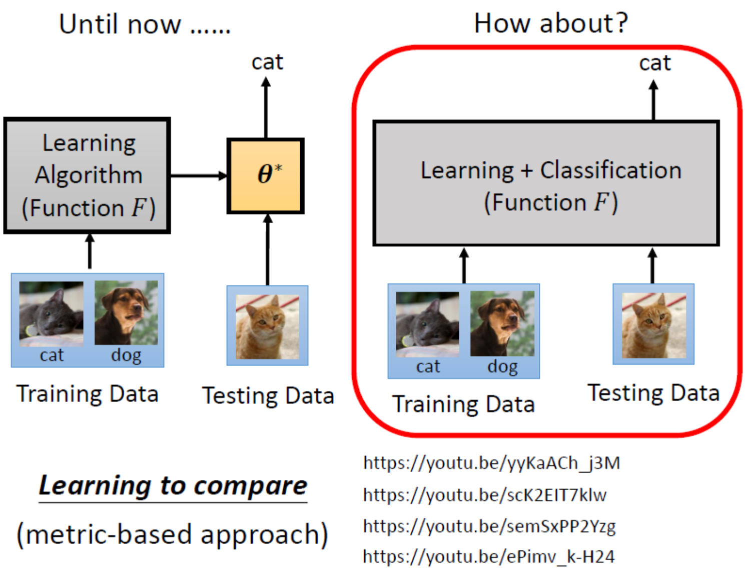
4.6 Beyond Gradient Descent

輸入數據，直接輸出模型



5. Learning to compare

learning to compare 直接輸入訓練資料和測試資料，學出 learning + classification，就直接輸出測試的結果



學習更多：

https://youtu.be/yyKaACh_j3M

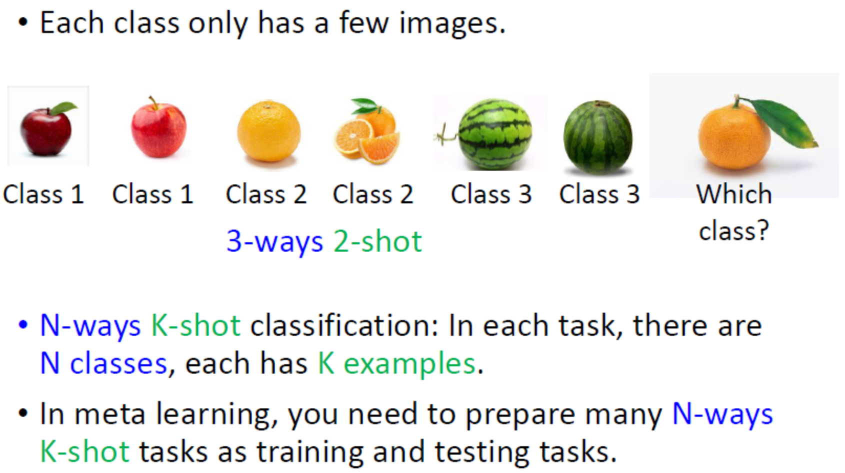
https://youtu.be/scK2EIT7klw

https://youtu.be/semSxPP2Yzg

https://youtu.be/ePimv_k-H24

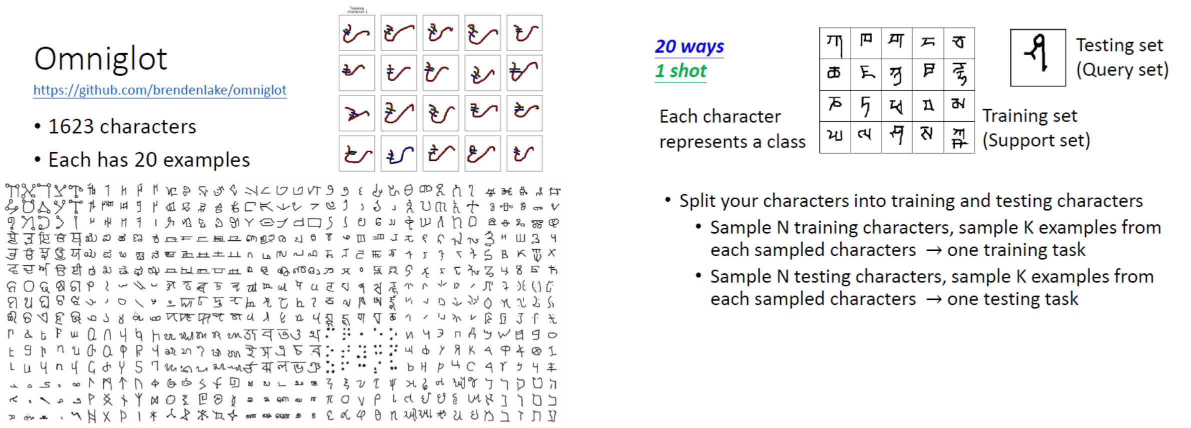
6. Application

6.1 Few-shot Image Classification



- N-ways K-shot：N 個類別、每個類別 K 個樣本
- 在 meta learning 中，需準備多個 N-ways K-shot 的任務作為訓練任務與測試任務

一般做 meta learning 的實驗通常會使用 Omniglot 資料集



6.2 More application

更多 meta learning 應用可參考：

http://speech.ee.ntu.edu.tw/~tlkagk/meta_learning_table.pdf