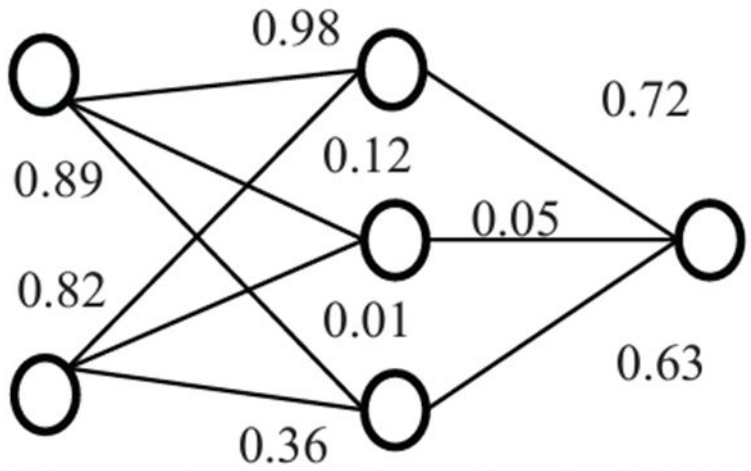
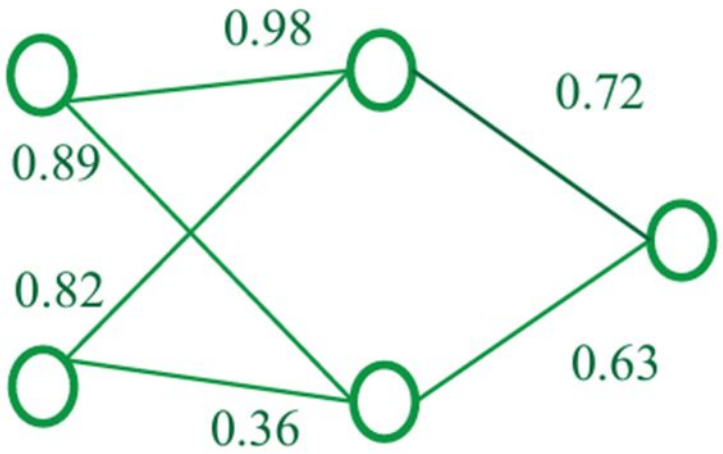
Network Compression

Network Compression

- Network Compression: 保持或最小影響模型性能的情況下減少模型的計算需求和參數量，從而適應硬體資源有限的環境，並提升模型的實用性。



(a) Non-Compression Classical NN



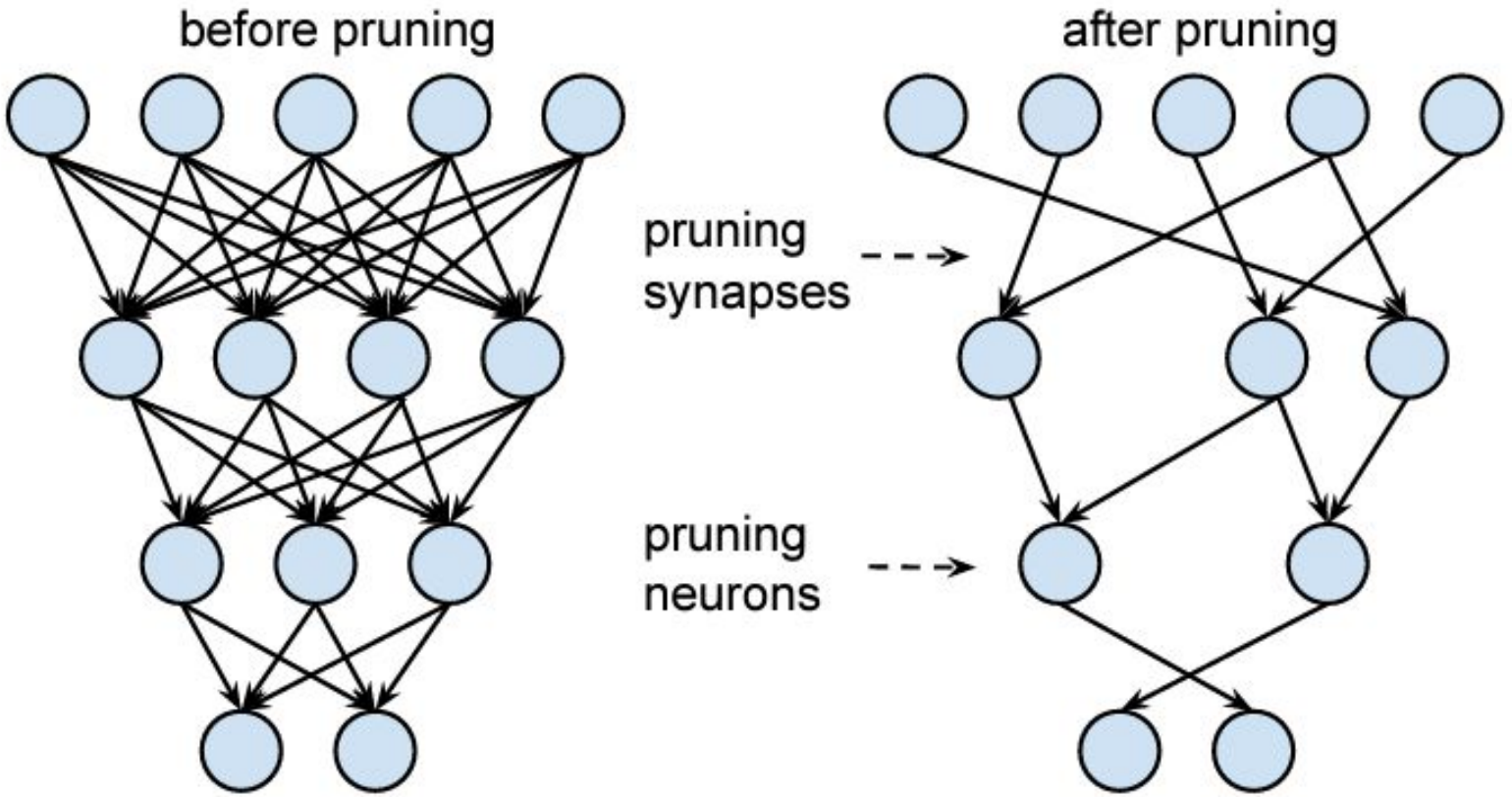
(b) Classical NN with Pruning

Why do we need network compression?

- 減少模型大小
  - 更小的體積除了減少儲存成本，也讓硬體需求降低了
- 邊緣運算裝置部署
  - 模型當然可以直接移植，但是行動裝置(手機、手錶…)的可用運算資源都不高，運算效率通常不怎麼樣
- 提高計算效率，降低延遲
  - 雲端運算再發達，要怎麼保證網路訊號及延遲始終穩定?想像一下輔助駕駛系統響應時間超過1秒...
  - 解法: 邊緣運算 – 將壓縮的大模型部署在車輛上

Network pruning

What can you prune?



Visualization of pruning weights vs pruning neurons

But, why?

問題來了，為什麼不直接訓練一個比較小的神經網路?

- 知識轉移與壓縮效果
  - 使用大型網路進行壓縮（如知識蒸餾）能夠將大網路中的隱藏知識有效轉移到小型網路中，這些隱藏知識可能包括資料的更深層次模式或泛化能力。

- 表現能力的差異
  - 大型網路擁有更多的參數，而小型網路缺乏足夠的參數來捕捉資料中的資訊，泛化能力較差。(Overparameterization & Double Descent)

補充: Overparameterization

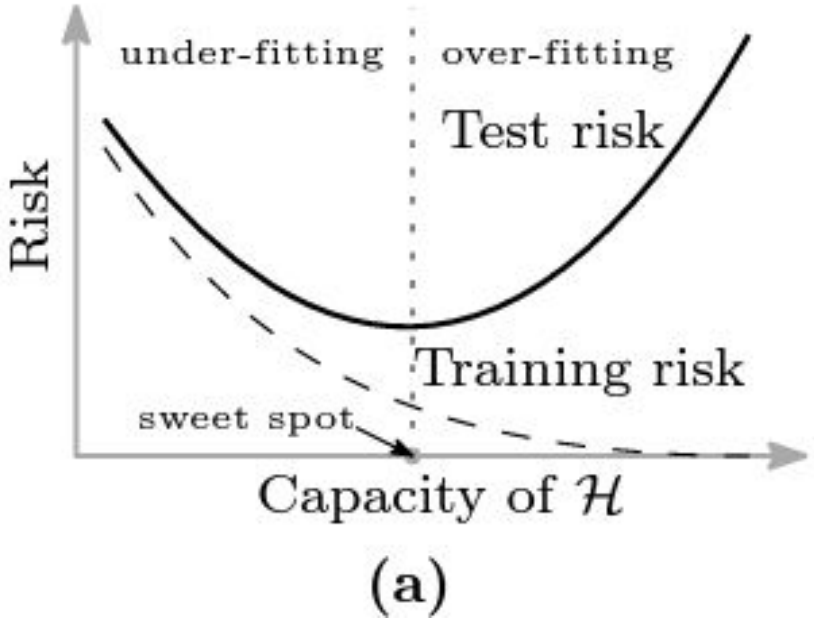
- 過參數化(Overparameterization)，是指模型的參數量超過訓練資料所需的程度

- 當模型的參數超過一定數量後，過參數化反而可以提高模型的泛化能力，使得模型在未見過的資料上也能有良好表現。
  - 圖見下頁

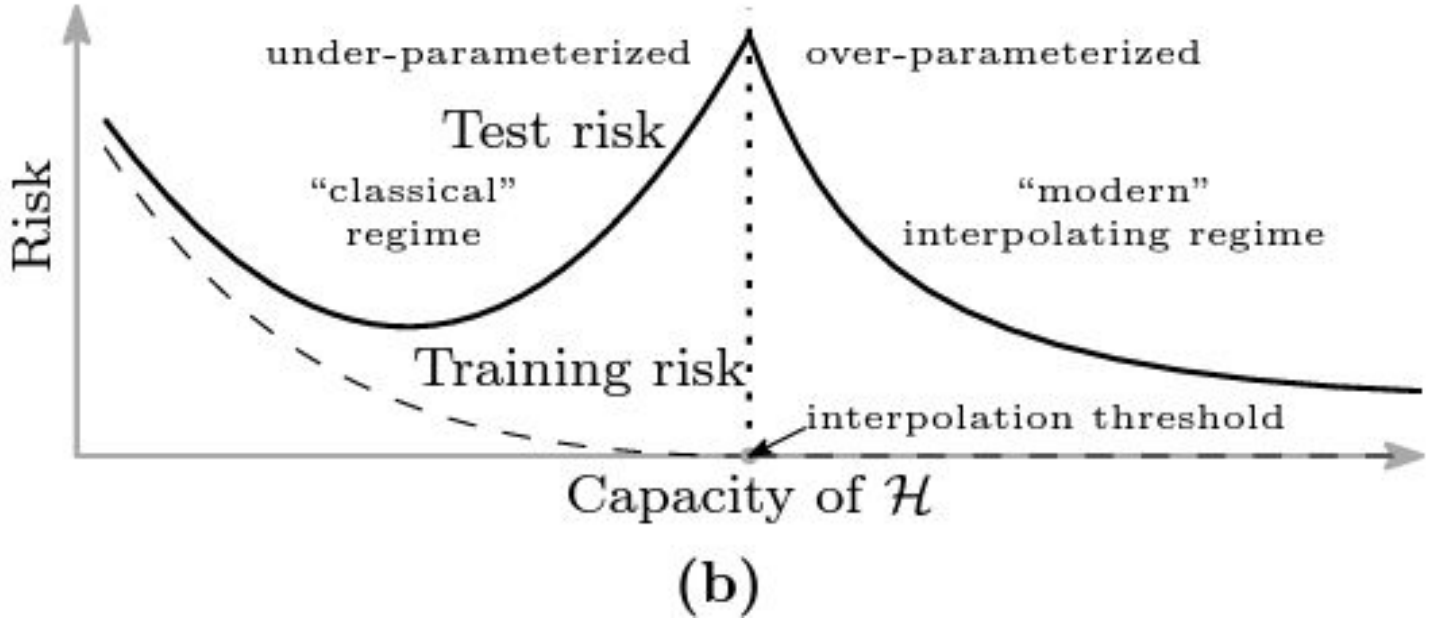
- 過參數化的神經網路能夠探索到更為平滑、穩定的解，有助於防止過擬合並提升模型性能。

補充: Double Descent

- 雙重下降(Double Descent)是指模型的test error 在參數數量增多時出現二次下降的現象
- 在高度過參數化的情況下，模型可以自動找到一個較為穩健的解釋模式，使測試誤差降低。



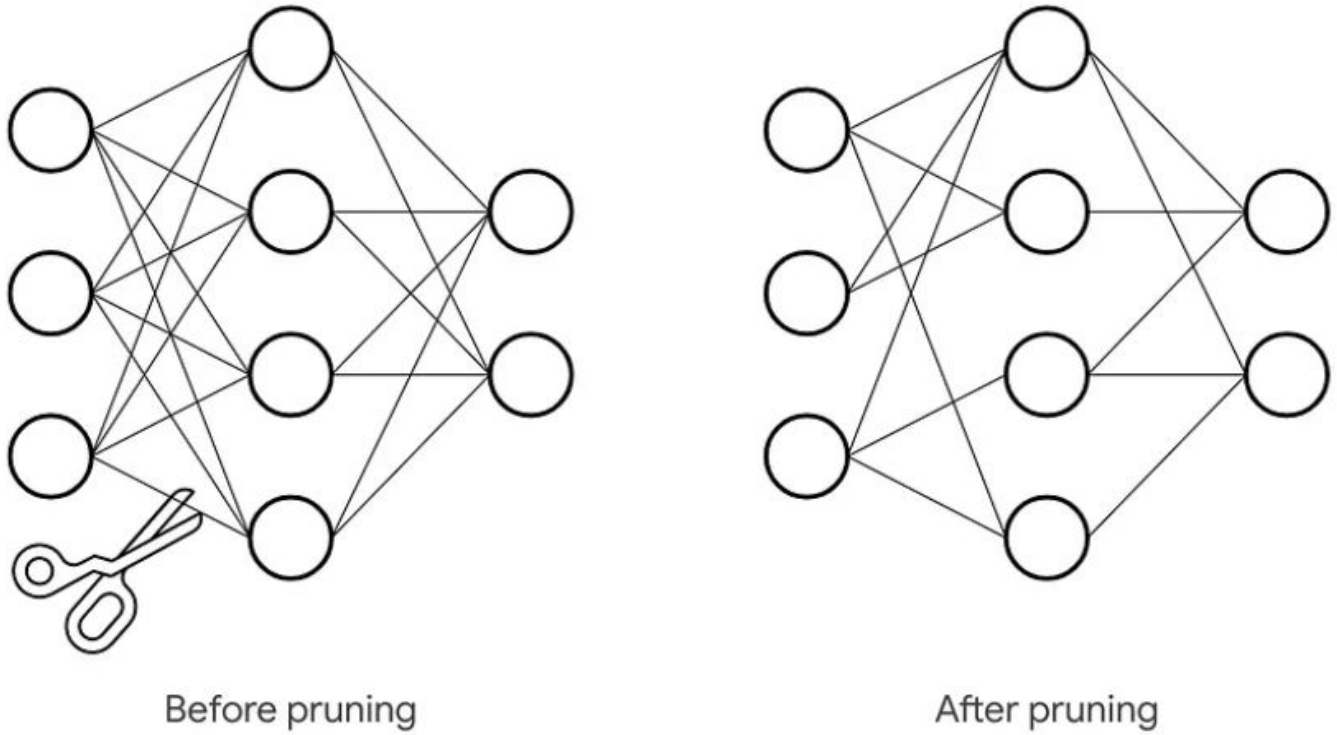
(a)



(b)

Weight Pruning

- Weight Pruning 的基本概念是將權重移除，從而減少模型中的連接數量。



Weight Pruning method

1. 學習連接性
   - 先進行常規的網路訓練，建立神經網路中的所有連接。

1. 剪去低權重連接
   - 將所有權重小於閾值的連接進行剪枝，型成一個稀疏的網路結構。

1. 微調稀疏網路
   - 對剩下的稀疏連接進行重新訓練，以學習這些剪枝後的連接的最終權重

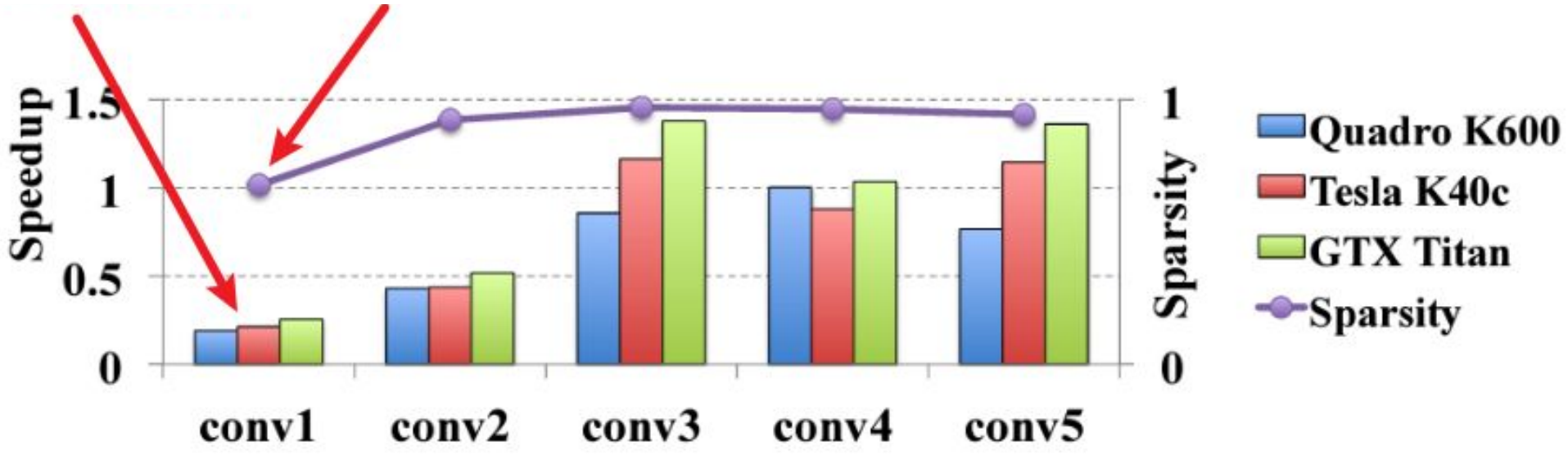
Weight Pruning的特點

- 非結構化剪枝
  - 移除單一參數時，不考慮模型的整體架構，容易使模型產生一種不規則的稀疏結構
- 硬體限制
  - 權重剪枝會使原本的矩陣變得稀疏。不是所有的硬體都支援稀疏矩陣的運算
- 雖然不實際減少參數，但是依然有壓縮效果
  - 剪枝動作會將權重設為 0，但是儲存時採用CSR 或 CSC 格式可以忽略0值減少大小

Weight Pruning

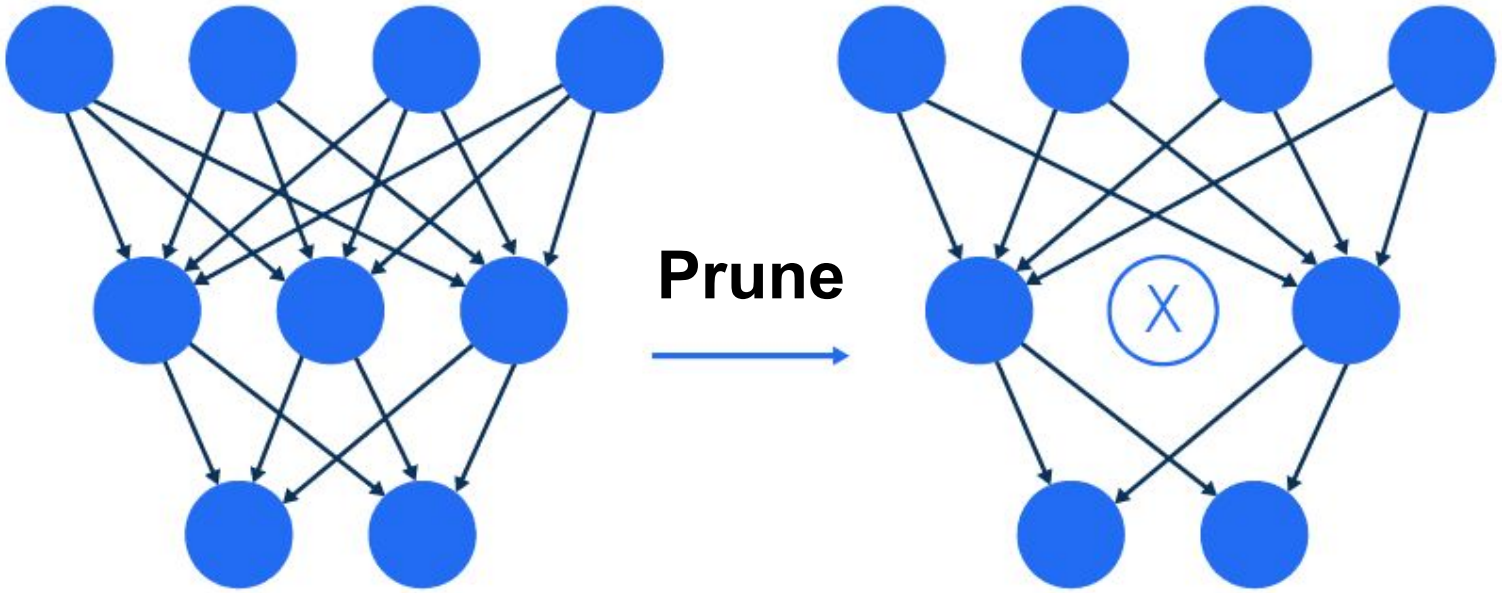
- 透過下圖的實驗結果可以觀察到在大部分情況下運算沒有加速
  - Sparsity 代表節省的參數比例 (0的比例)
  - Speedup 代表加速倍率
  - conv<num> 代表神經網路的某一層

加速小於1 (開倒車)　**去掉的參數比例大約 75%**



Neuron Pruning

- Neuron Pruning 的基本概念是將神經網絡中的整個神經元刪除，而不是只移除單獨的權重



Neuron Pruning 流程

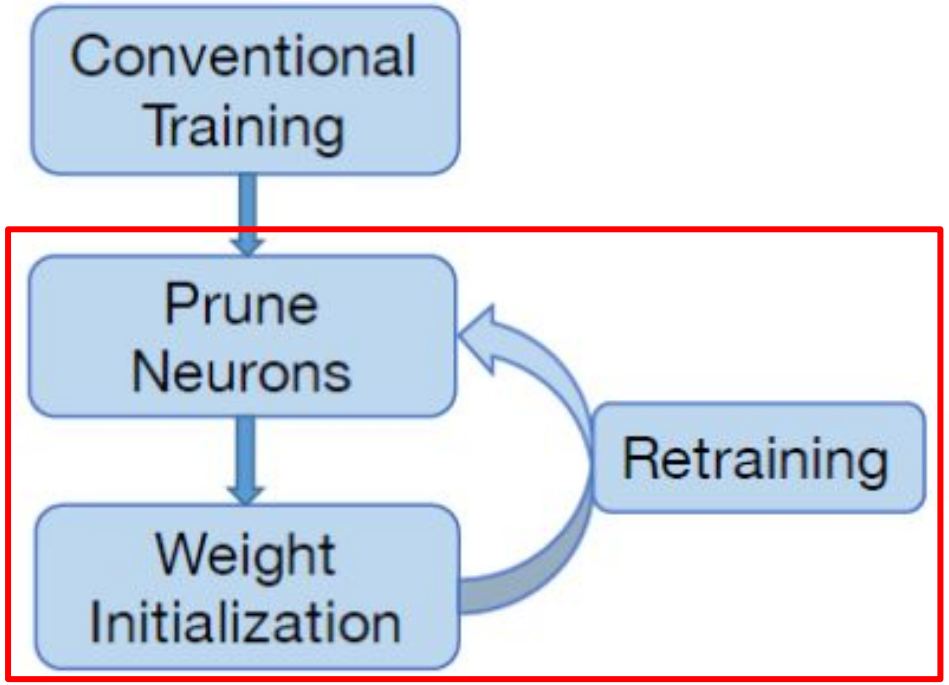
- 訓練
  - 訓練一個完整的神經網絡模型，達到較好的性能。
- 剪枝
  - 根據預設的剪枝規則與閾值，選擇性地將對輸出影響較小的神經元整體移除。
- 微調 (Fine-tuning)
  - 剪枝後的模型可能會出現一定的性能損失，因此需要對模型進行微調，以恢復或提升其準確度。

APoZ(Average Percentage of Zeros)

- 一個神經元經過Activation function後為輸出為0的比例
- APoZ值越高代表該神經元對最終結果影響越低
- 該指標原本用於CNN，但只要修改一下公式也可以適用於其他神經網路結構

Neuron Pruning method

- 迭代式修剪
  - 每次修剪後，保留其餘神經元的權重進行初始化(非歸零)，並對模型進行微調。
- 逐層或逐步修剪
  - 若一次修剪過多神經元，會對模型性能造成較大損失



Three main steps for trimming

神經元剪枝後的初始化

- 修剪後的網路如果不進行初始化，在重新訓練時會有更多神經元的激活值為零
  - 代表網路的有效神經元數量下降，計算資源得不到充分利用

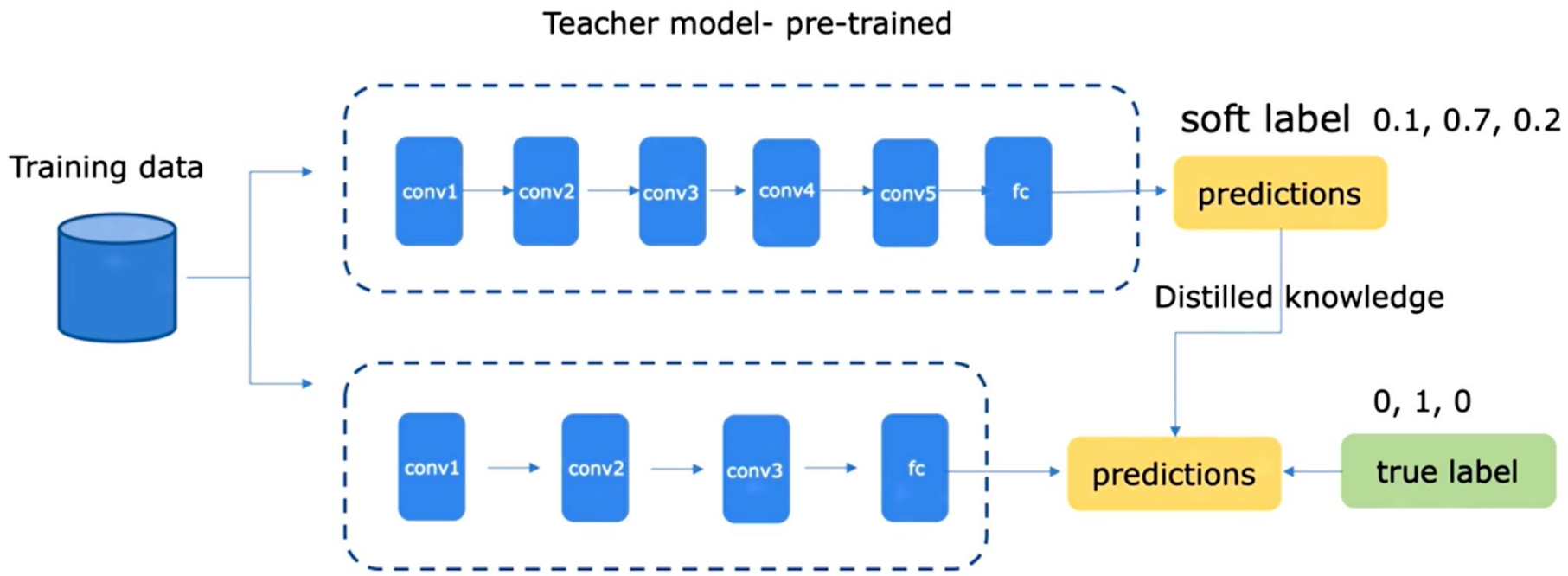
	With Weight Init			Without Weight Init		
Number of Neurons in FC1	500	426	349	500	420	303
Accuracy (%)	99.31	99.29	99.30	99.31	99.23	99.18
Mean APoZ (%)	52.30	45.85	42.70	52.30	55.08	55.08
#{APoZ>0.6}	154	77	17	154	160	110
#{APoZ>0.7}	87	10	0	87	102	78
#{APoZ>0.8}	54	0	0	54	61	49
#{APoZ>0.9}	33	0	0	33	40	32

Knowledge Distillation

Knowledge Distillation

讓小模型(學生模型)學習大模型(教師模型)的輸出

- 學生模型的訓練目標有兩個:
  - 跟True label 之間的loss
  - 跟教師模型輸出之間的loss



Knowledge Distillation 基本流程

1. 訓練教師模型
   - 訓練一個大型且精度高的教師模型，通常是DNN

1. 準備軟標籤（Soft Labels）
   - 教師模型會對訓練數據進行預測，產生軟標籤

1. 定義溫度係數
   - 用來平滑模型的概率分佈，會在損失函數中使用

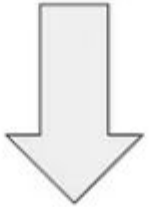
1. 訓練學生模型

1. 微調

Softmax temperature

- Softmax temperature引入T在Softmax函數中，通過控制數值的放大或縮小，使得概率分佈更加平滑或更加尖銳。

$$p_i = \mathrm{Softmax}(z_i) = \frac{\exp(z_i)}{\sum_{j=1}^{n} \exp(z_j)}$$



$$p_i(T) = \mathrm{Softmax}(z_i/T) = \frac{\exp(z_i/T)}{\sum_{j=1}^{n} \exp(z_j/T)}$$

Soft label

- 軟標籤是指教師模型對每個類別輸出的概率分佈（即Softmax輸出），而不是單一的硬標籤（Hard Label）。
  - 在標準分類問題中，模型最終會輸出一個「硬標籤」，即只選擇機率最高的那個類別。

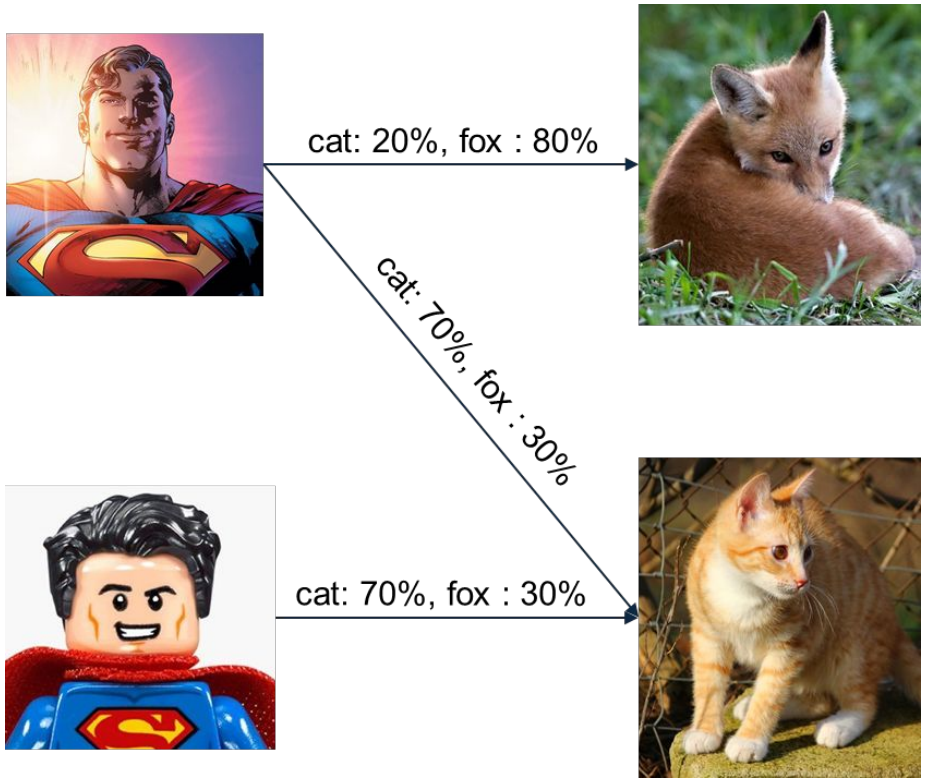
- 軟標籤會保留對所有類別的概率值，即保留了「預測強度」。

- 軟標籤包含了更豐富的信息
  - 如果一張圖像被教師模型分類為[cat, dog, fox] = [0.8, 0.15, 0.05]，那麼學生模型可以學到「貓」和「狗」在某些特徵上比較相似，而不是單純只知道「貓」的正確標籤。

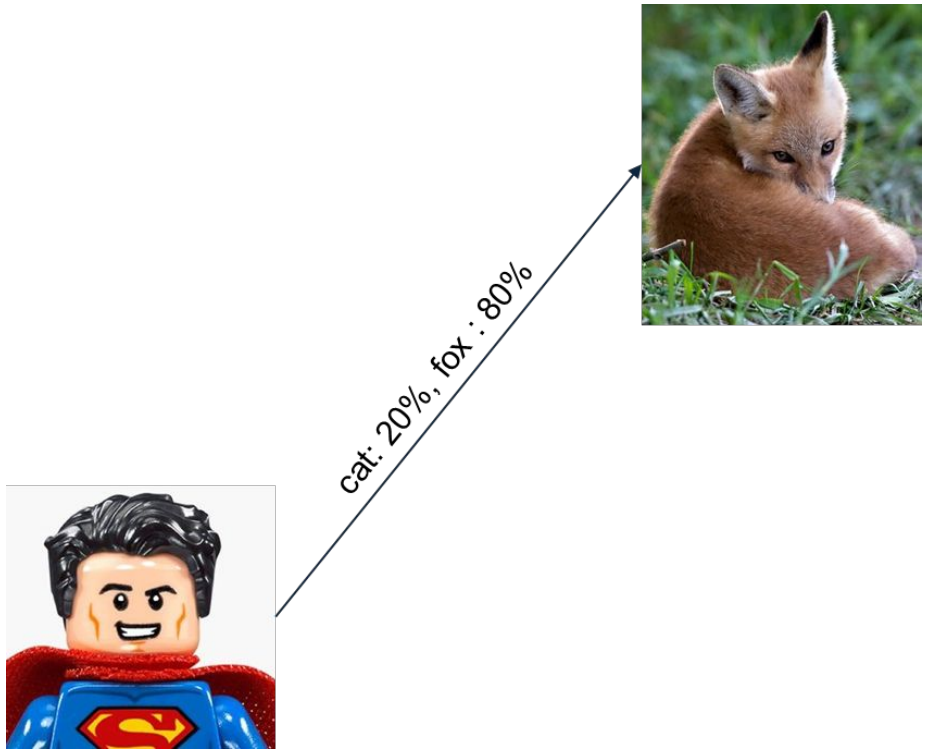
知識蒸餾的特點(1/2)

學生模型學習的不只是具體的分類資訊，還包含了類別之間的關係，即使小模型沒看過資料，依然有機會可以預測出沒看過別

Learning



Inference



知識蒸餾的特點(2/2)

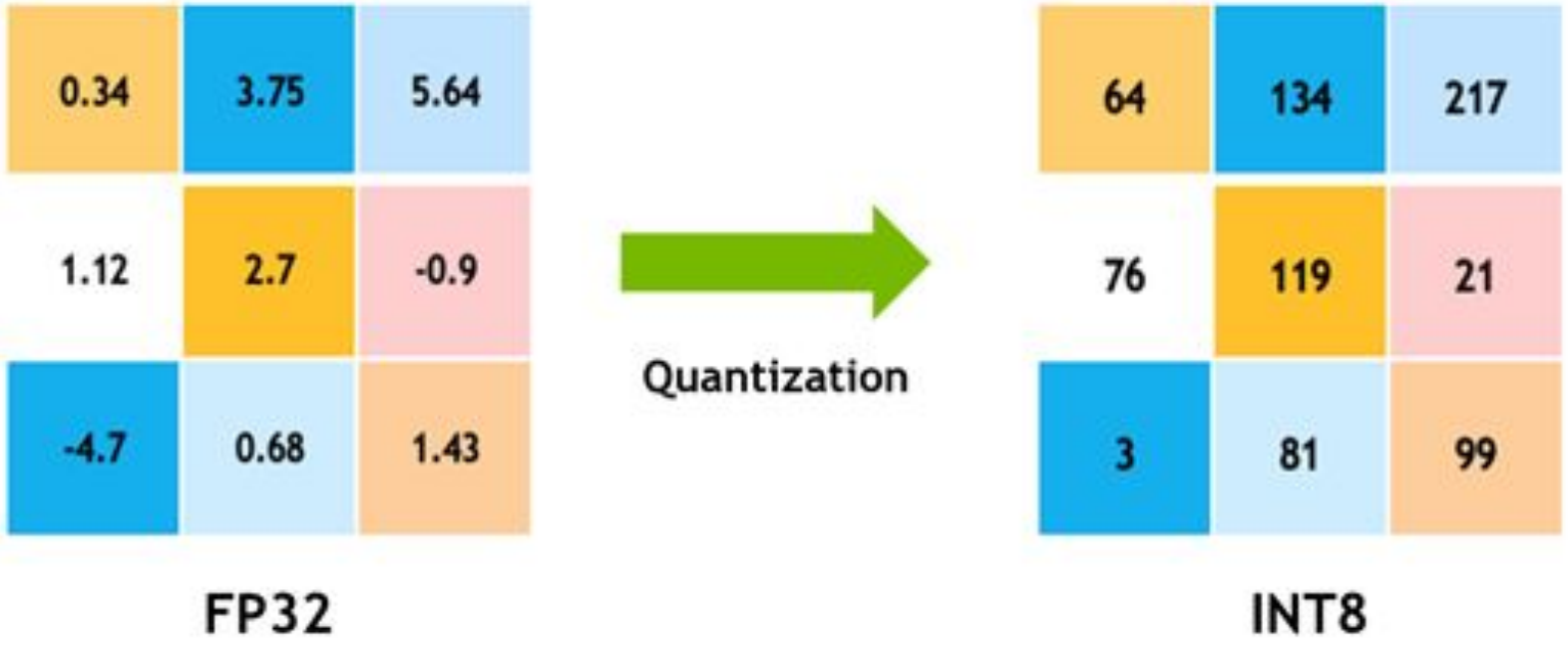
- 知識蒸餾的目的是讓學生模型學習到教師模型的泛化能力，而不是去擬合訓練資料。
- 知識蒸餾不限於特定的架構
- 知識蒸餾的過程也是訓練的過程，同樣需要足夠資料並耗費大量的時間
- 可能需要花時間額外設計學生模型
  - 視部署環境決定，有現成的，如MobileNet等輕量模型

Quantization

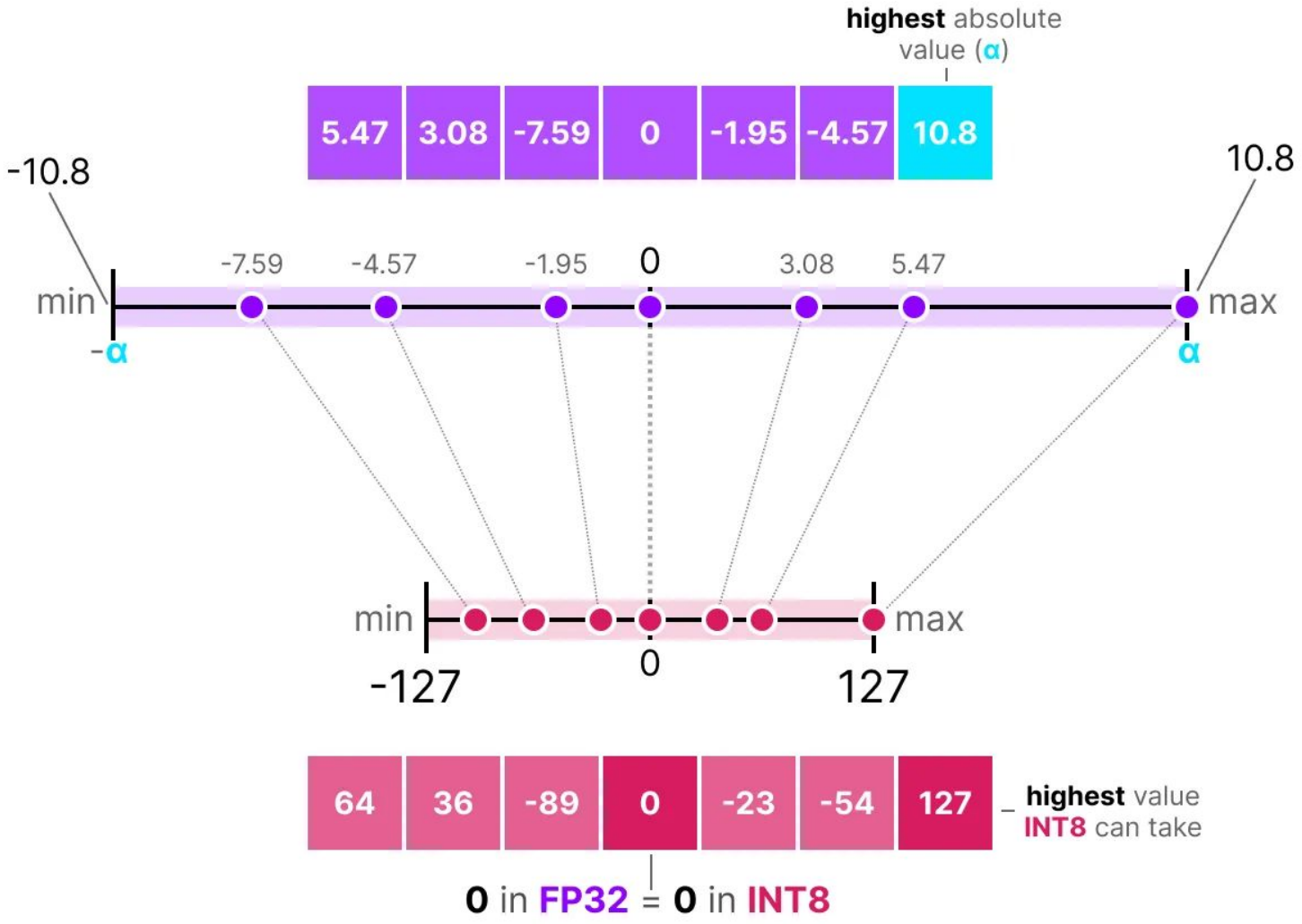
Post-Training Quantization

- 訓練後使用更少的位元儲存參數
  - 例如把參數32-bit(4 Byte)浮點數轉換為8-bit(1 Byte)整數

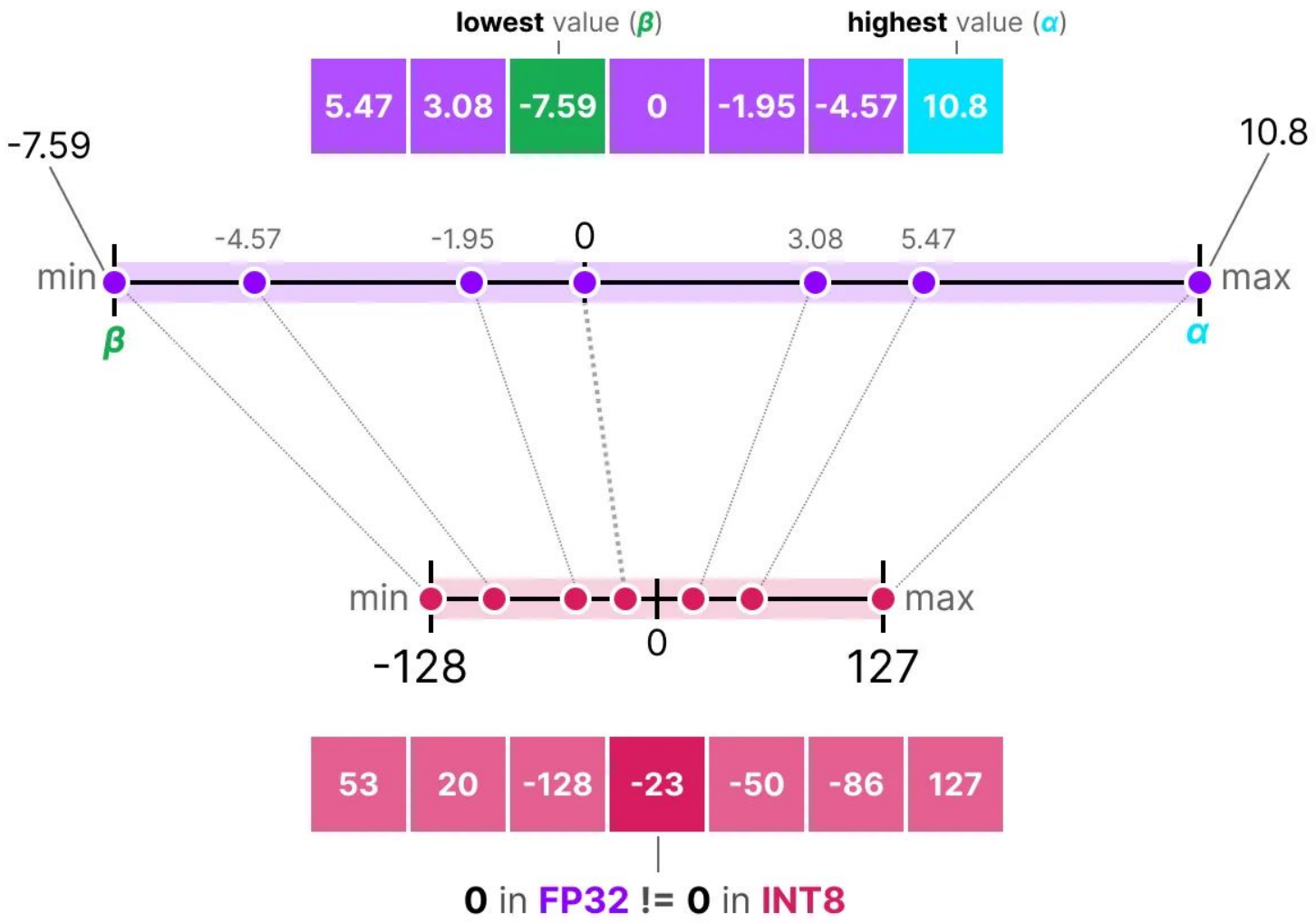
    假設整個神經網路的參數量約有一千萬左右，壓縮前需要大約381.47MB，壓縮後只需要95.37 MB



對稱量化


highest absolute value (α)
5.47
3.08
-7.59
0
-1.95
-4.57
10.8
-10.8
10.8
-7.59
-4.57
-1.95
0
3.08
5.47
min
max
-α
α
min
max
-127
0
127
64
36
-89
0
-23
-54
127
highest value INT8 can take
0 in FP32 = 0 in INT8

非對稱量化


lowest value (β)
highest value (α)
5.47
3.08
-7.59
0
-1.95
-4.57
10.8
-7.59
10.8
-4.57
-1.95
0
3.08
5.47
min
max
β
α
min
max
-128
0
127
53
20
-128
-23
-50
-86
127
0 in FP32 != 0 in INT8

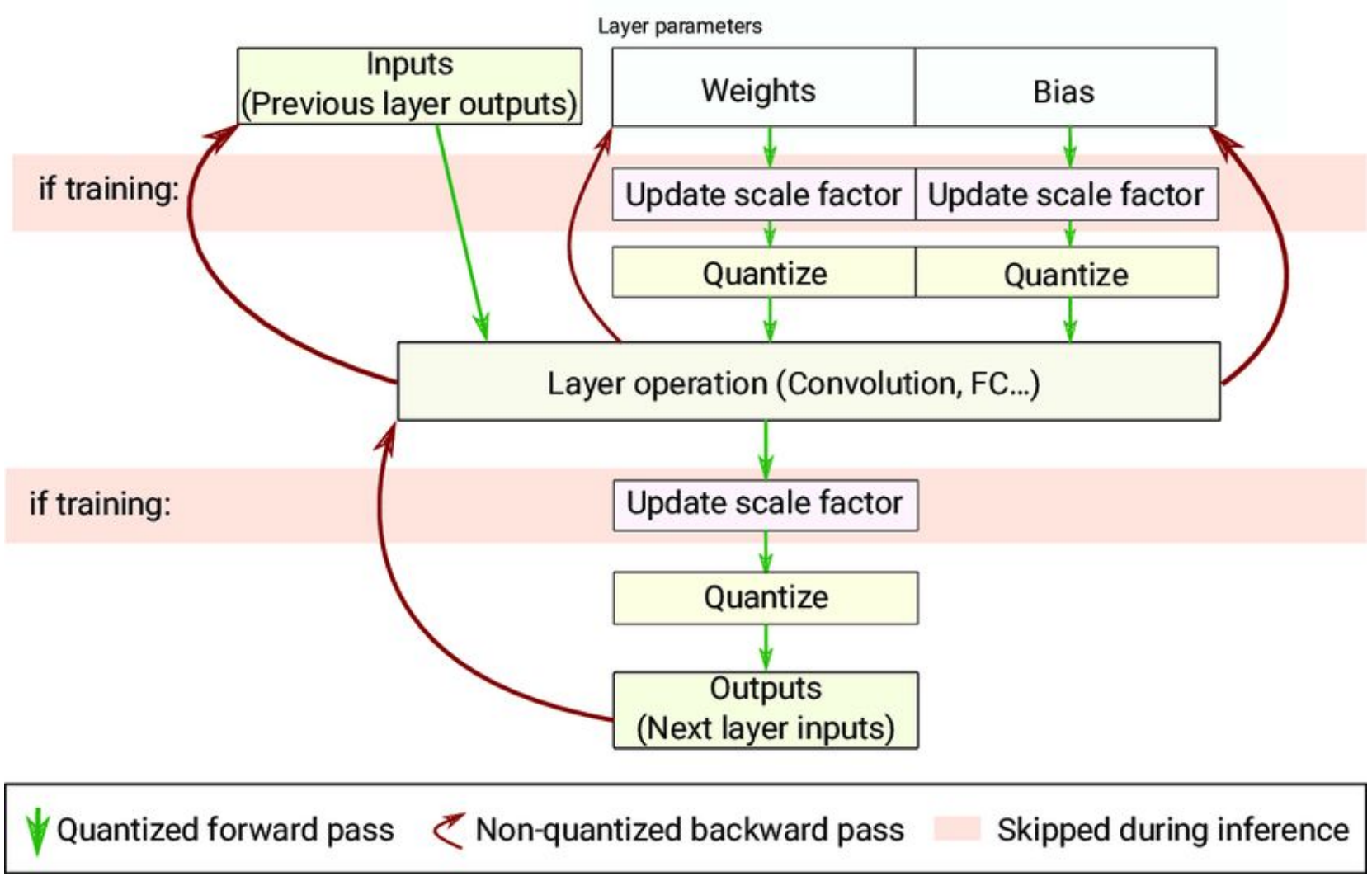
Quantization Aware Training
量化感知訓練

- 大部分模型都是在fp32的精度下訓練的，做PTQ後性能多少會有損失

- 量化感知訓練是指在訓練時進行假量化，來讓模型在訓練階段針對量化後的誤差做調整

- QAT可以在PTQ之後作為微調權重恢復精度的手段或是直接加入在訓練過程中
  - 差別在訓練後做QAT要凍結縮放

Quantization Aware Training



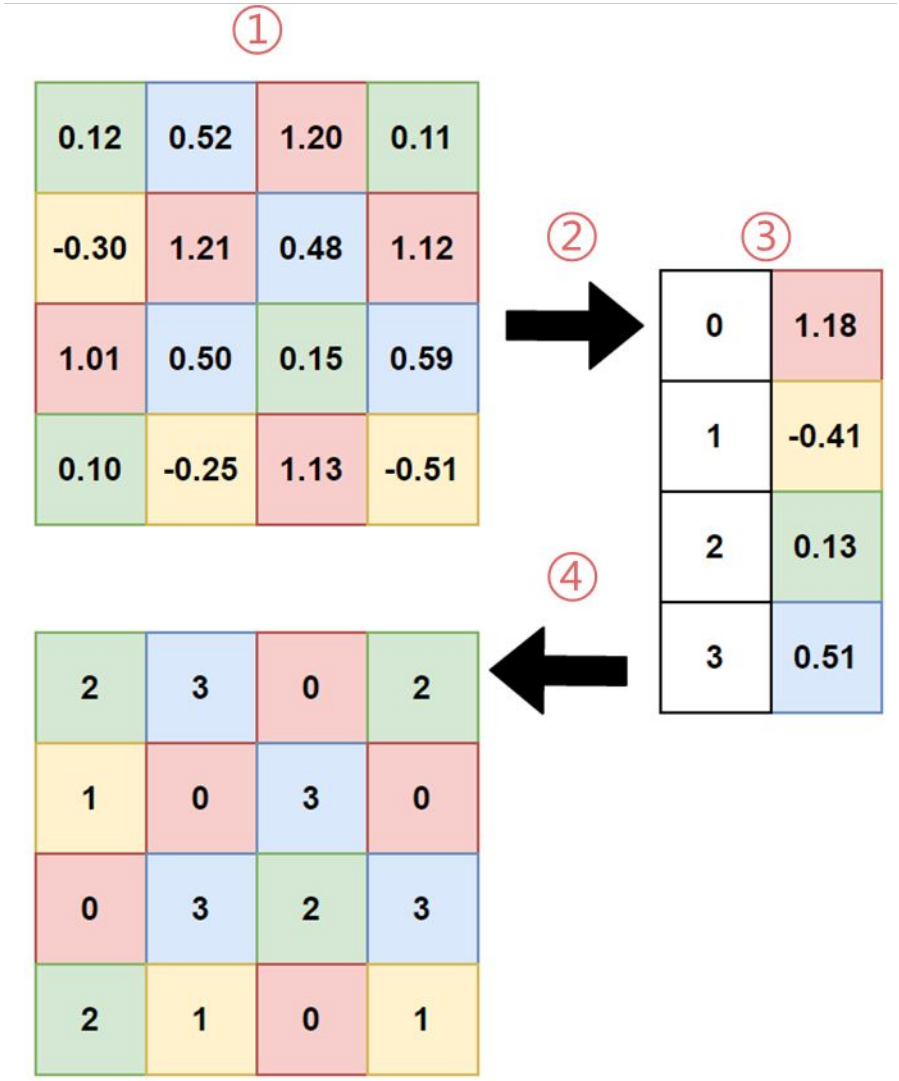
權重聚類分析

1. 將權重分群，數值相近的分為一群

1. 依照策略將同一群的參數共享權重

1. 建立權重映射表

1. 參數從權重數值改成權重映射表的標記

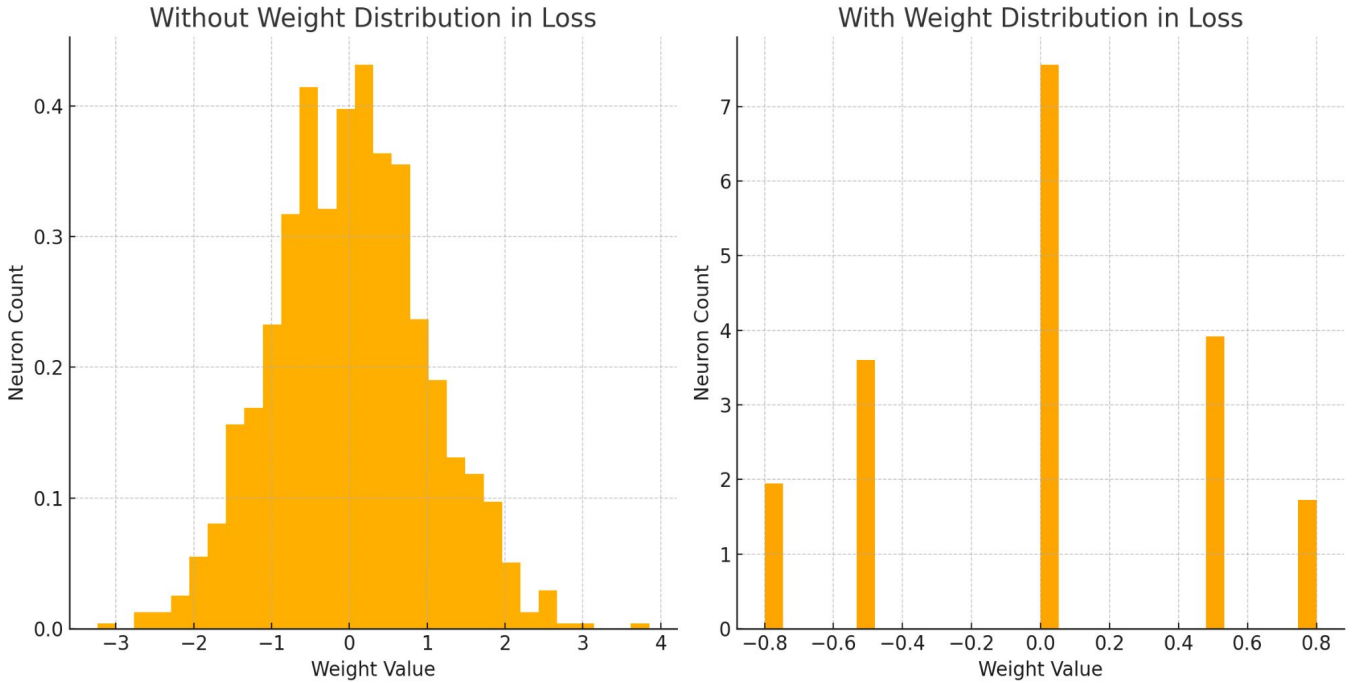


權重聚類分析特點

- 可以透過調整聚類的數量實現高壓縮率
- 在訓練時透過將權重的分佈作為loss計算的一部分可以降低聚類後對精度的影響
- 聚類後可以結合 PTQ 再進一步壓縮

將權重的分佈作為loss

- 在loss function 中加入一個機制，鼓勵權重分佈集中在特定值附近
- 例如: $\mathcal{L}_{cluster} = \sum_{w \in W} \min_{c \in C} (w - c)^2$ (W:權重集合 | C: 聚類中心)



Dynamic Computation

Dynamic Computation

- 神經網路自動調整運算量
  - 根據裝置的計算能力或是可調用的剩餘資源量(ex: 電量)調整

- 為什麼不直接準備多個不同尺寸的神經網路做切換就好
  - 裝置儲存空間有限

- **實際上模型尺寸沒有縮小**

Dynamic Depth(1/2)

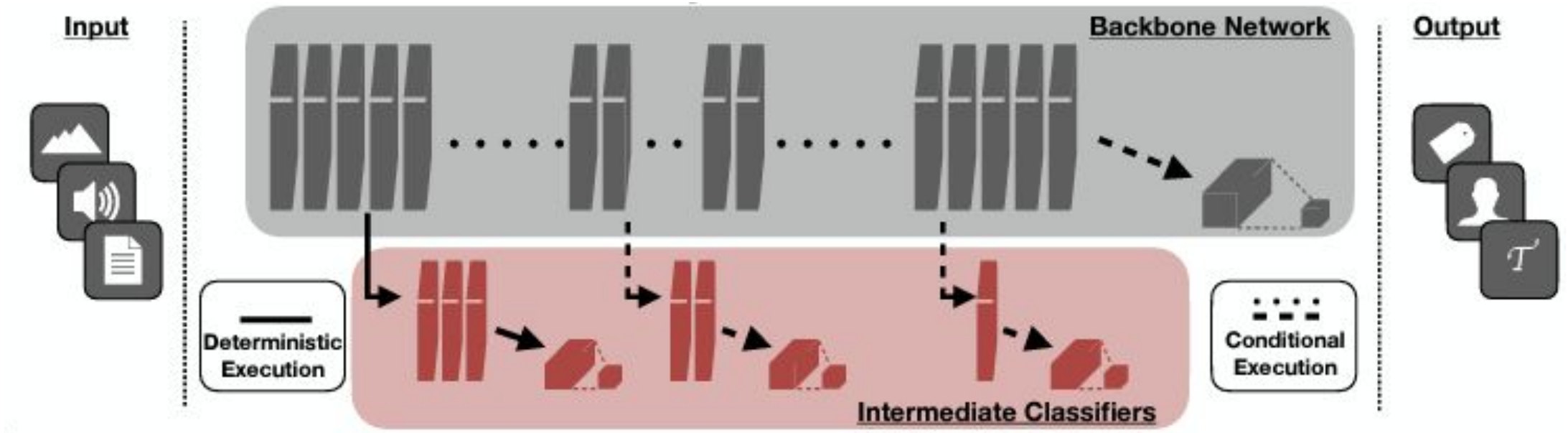
- 傳統的神經網路通常具有固定的深度，所有層皆會執行
- 允許神經網路根據輸入複雜度決定調用多少層來進行處理
- 早退(Early Exiting)
  - 在處理較為簡單的資料時，模型可以「提前退出」，以此降低運算成本。
  - *請注意這是早退不是早停(Early Stopping)

Early Exiting

- 信心度評估
  - 在每個早退點，模型評估該層的預測信心度。
  - 通常通過Softmax機率的熵值或最高機率（Top-1）判斷樣本的分類信心度。

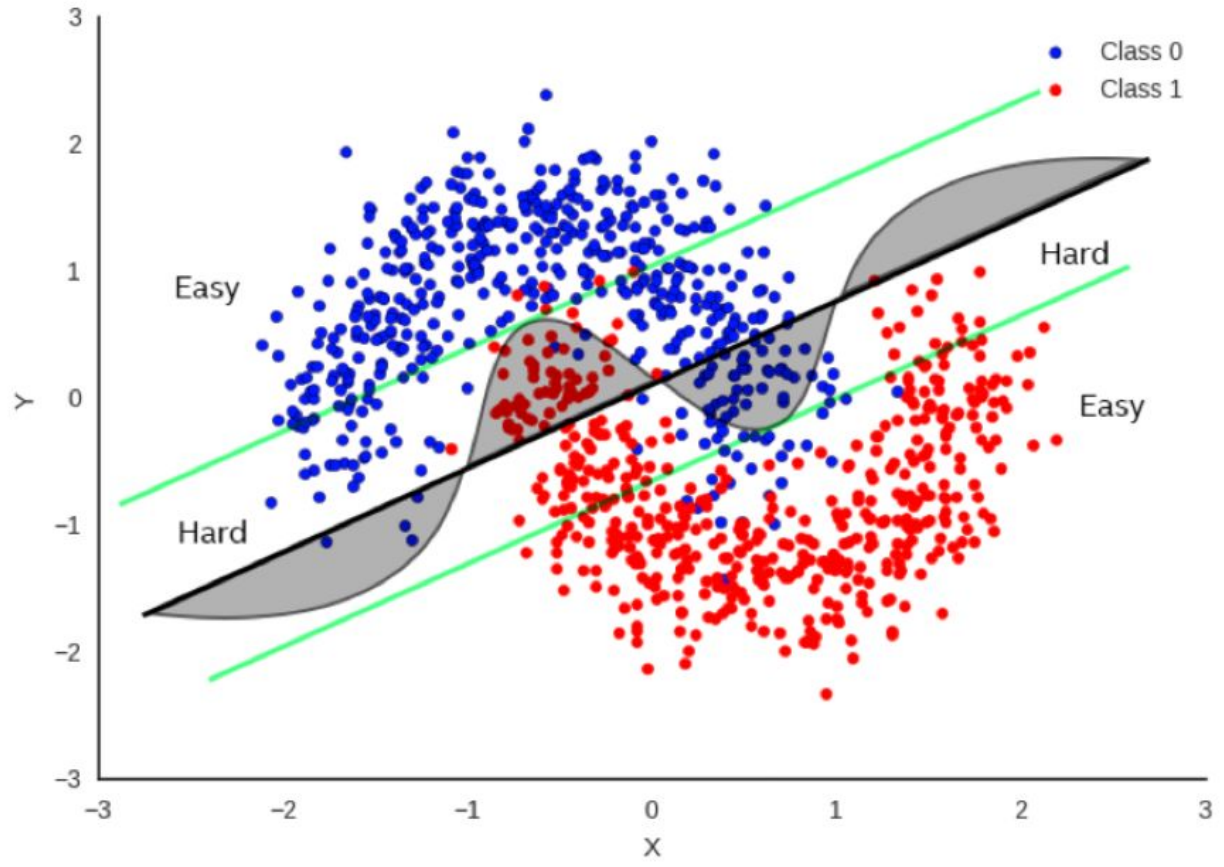
- 模型結構和早退點位置
  - 早退點的數量和位置會影響模型的性能和計算開銷。
  - 密集放置的早退點可以減少平均計算量，但也會增加網絡結構的復雜性和訓練難度。

Early-exit network architecture



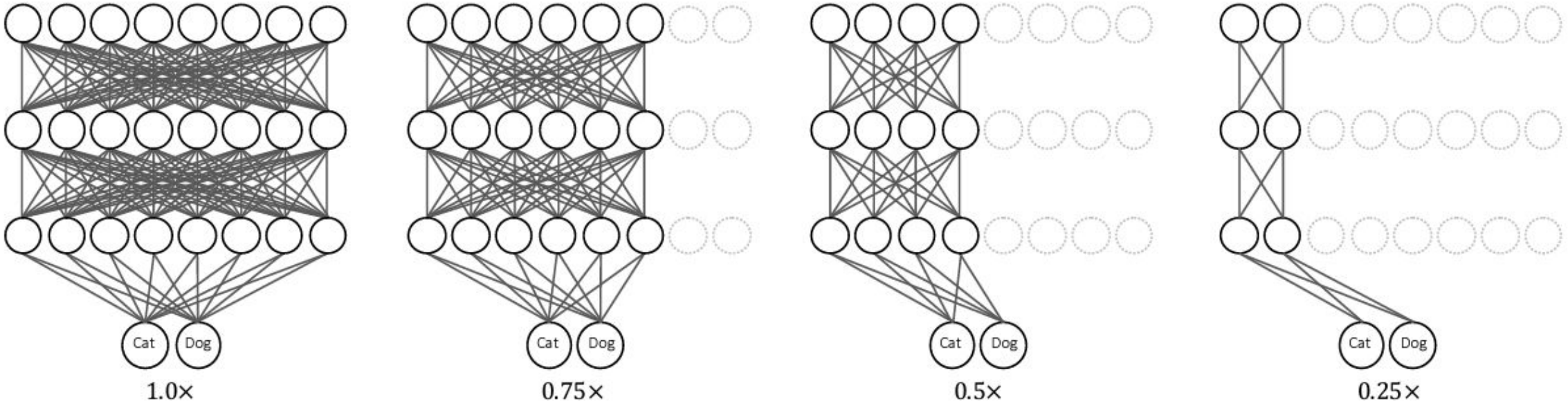
Why Early Exiting effective?

- 實際上難以分類(需要更多層運算)的樣本只有綠色直線內的區域而已，因此不需要為了這些樣本每次都調用這麼多資源



Dynamic Width

- 神經網路根據輸入資料的特性動態調整寬度。
  - 在每層中，根據需求動態啟用或禁用一部分通道或神經元。
  - 當輸入較簡單時，模型可以選擇只激活一部分通道，從而減少計算量。
  - 訓練時會把所有不同寬度產生的結果loss做加權計算，使其最小化



所有壓縮方式比較

技術	主要目標	優點	缺點
Network Pruning	移除不重要的神經元或權重	減少參數量與計算量，能顯著降低模型大小	可能造成準確率下降，需小心選擇被移除的參數
Knowledge Distillation	使用輕量模型學習大型模型的知識	保持模型精度的同時顯著減少參數和計算量	需要額外的教師模型訓練，且知識傳遞效果受限於教師模型
Post-Training Quantization	將權重縮小到較低位元（如 8-bit）	簡單易用，快速實現量化	可能造成較大的準確度損失，特別是對小型數據集
Quantization Aware Training	在訓練過程中模擬量化影響	減少量化對準確度的負面影響，適用於精度敏感應用	訓練成本增加，且需要大量資料和計算資源
Weight Clustering	將權重分組並共享權重	減少模型參數存儲需求，適用於低存儲應用	對模型精度可能有影響，且需要額外的處理來支援權重共享
Dynamic Computation	根據輸入動態調整計算	適用於變動需求的應用，能動態節省計算量	需要設計特別架構，且訓練和推理時複雜度較高

組合技

- Network Pruning + KD
  - 通過 KD可以使用較大的教師模型來引導被剪枝的輕量模型學習，幫助恢復或維持一定程度的準確性
- (QAT +) PTQ + Weight Clustering
  - 進行假量化訓練並做後量化及權重聚類，可以進一步減少模型的存儲需求和計算量。量化保證了計算效率，而權重聚類則進一步壓縮儲存大小。
- Network Pruning + PTQ + KD
  - 先使用 Network Pruning 移除冗餘參數，再用 PTQ 進行量化來降低模型大小，最後使用KD來提升被壓縮模型的準確率。

Implementation

Download notebook at:
https://github.com/albert831229/nchu-computer-vision/tree/main/113/day