
Machine Learning HW7

ML TAs

ntu-ml-2020spring-ta@googlegroups.com

Outline

- Task Description
- Dataset
- Guidelines
- Kaggle & Regulations
- Report
- GitHub Submission
- Links

Task Description

- 用一個小 model 也可以模擬出大 model 的行為/正確率。
- 這次作業我們的任務是用非常小的 model 去完成 HW3 的內容:food11。

Dataset & Submission Format

- 與HW3 - CNN 相同 (food-11)。請參考 [hw3連結](#)。

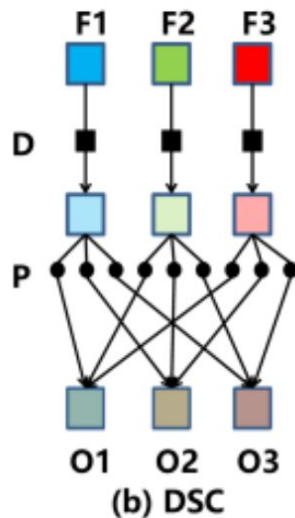
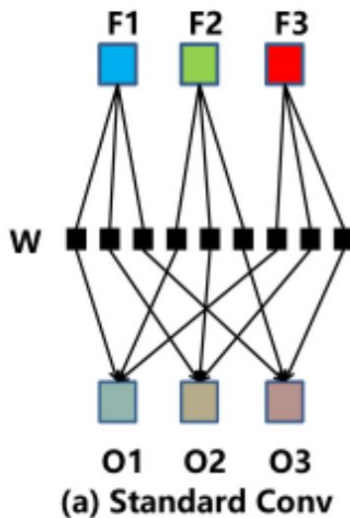


Guideline

- Network/Model Compression 有很多種門派, 這邊我們介紹四種:
 - **Knowledge Distillation:** 讓小 model 在學習任務的時候, 藉由觀察大 model 的行為來讓自己學得更好。(直譯: 讓小 model 萃取大 model 的知識)
 - **Network Pruning:** 將已經學習好的大 model 做剪枝, 讓整體 model 變小。
 - **Weight Quantization:** 用更好的方式來表現 model 中的參數, 以此降低運算量/消耗容量。
 - **Design Architecture:** 將原始的 layer 用更小的參數來表現。(例如 Convolution → Depthwise & Pointwise Convolution)

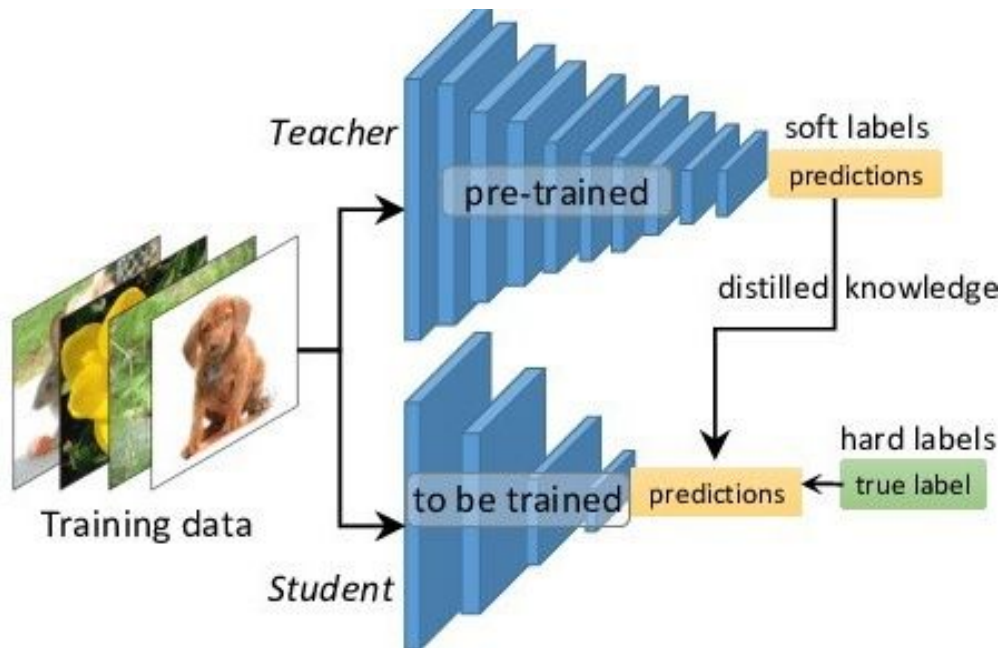
Guideline - Design Architecture

- Depthwise & Pointwise Convolution Layer (MobileNet 提出)
 - 原始的 Conv 你可以想像成它就是一個 Dense/Linear Layer, 但是每一條線/每一個 weight 都是一個 filter, 而原本的乘法會變成卷積運算。
 - 而 Depthwise 是讓每一個 Channel 都先過一個各自的 filter, 在對每個 pixel (Pointwise) 過一層 share weight 的 Dense/Linear。
- 強烈建議大家使用類似這樣的技巧去設計你的 model。
- [colab tutorial](#)



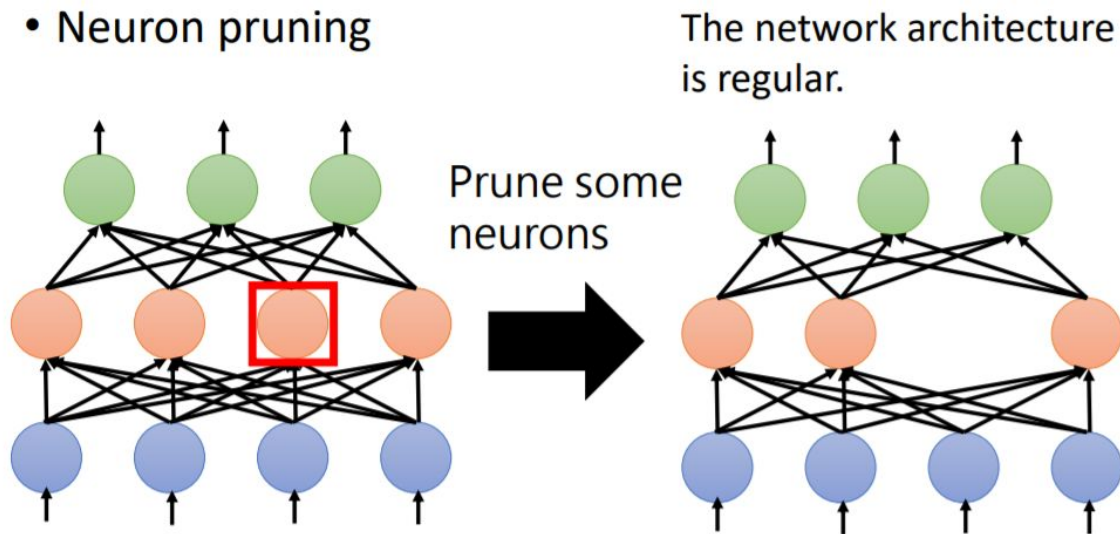
Guideline - Knowledge Distillation

- 在 train 小 model 的時候, 加入大 model 的一些資訊(例如預測的機率分布)來幫助小 model 學習。
- 我們有提供已經 train 好的 ResNet18 模型幫助大家做 Knowledge Distillation (Acc ~ 88.4), 各位在寫作業時請注意只能只用我們提供的 pre-trained model。
- [colab tutorial](#)



Guideline - Network Pruning

- 將已經 train 好的 model 做剪枝使其變小。
- 這次作業建議如果 model 大小差一點點再 prune 就好(因為比起其他方法更難實作)。
- [colab tutorial](#)
(rank by batchnorm's γ)



Guideline - Weight Quantization

- 用更好的方法來存取 model 的參數。
- 有些 quantization 是邊跑 train 邊 quantize (例如因為 int8 的運算比較快), 但這裡我們只做壓縮大小的操作。(省儲存空間)
- [colab tutorial](#) (壓縮已經 train 好的 model)

Guideline - Suggestion

- 使用 Knowledge Distillation Distill 我們給予的 model。(Network Pruning fine-tune 的時候也可以使用)
- 如果 train 到一個不錯的 model 差一點就可以過 size 的限制, 再用 pruning。(因為 pruning 比起其他方法比較難實作, 如果你對實作很有信心也可以:。)
- 基礎架構可以參見 MobileNet, ShuffleNet, DenseNet, SqueezeNet 等等。
- Quantization 壓最後的 model 很重要, 比較能壓過 size 限制 (20000bytes)。

Kaggle & Regulations

- 在上傳 submission 的時候, 請注意你的上繳的 state_dict 是否有 **<= 20000 bytes**, 如果沒有請不要上傳污染 Scoreboard。
- 請你的 code 要 **接近 reproduce 你上傳中最高的** public score, 避免有人沒有壓好就上傳進行威嚇作用。
- **禁止手標 label 或上網尋找 label (有標或找 dataset 但沒用在 model 上也一樣), 一被發現或檢舉, 該作業以 0 分計。**
- **除了我們給的 model 以及 hw3 你們自己的 model 以外, 禁止使用外來的 pre-trained model, 即使是 torchvision 的也一樣。**

Report

1. 請從 Network Pruning/Quantization/Knowledge Distillation/Low Rank Approximation/Design Architecture 選擇兩者實做並詳述你的方法, 將同一個大 model 壓縮至接近相同的參數量, 並紀錄其 accuracy。
2. 請嘗試比較以下 accuracy (兩個 Teacher Net 由助教提供)以及 student的總參數量以及架構, 並嘗試解釋為甚麼有這樣的結果。你的 Student Net 的參數量必須要小於 Teacher Net 的參數量。
 - x. Teacher net architecture and # of parameters: torchvision's ResNet18, with 11,182,155 parameters.
 - y. Student net architecture and # of parameters:
 - a. Teacher net (ResNet18) from scratch: 80.09%
 - b. Teacher net (ResNet18) ImageNet pretrained & fine-tune: 88.41%
 - c. Your student net from scratch:
 - d. Your student net KD from (a.):
 - e. Your student net KD from (b.):

Report

3. 請使用兩種以上的 pruning rate 畫出 X 軸為參數量, Y軸為 validation accuracy 的折線圖。
你的圖上應會有兩條以上的折線。
4. 請嘗試比較以下 validation accuracy:
 - a. 原始 CNN model (用一般的 Convolution Layer) 的 accuracy
 - b. 將 CNN model 的 Convolution Layer 換成總參數量接近的 Depthwise & Pointwise 後的 accuracy
 - c. 將 CNN model 的 Convolution Layer 換成總參數量接近的 Group Convolution Layer (Group 數量自訂, 但不要設為 1 或 in_filters)

2,3,4 擇二寫即可。

GitHub Submission

- GitHub 上 hw7-<account> 請至少包含：
 - report.pdf
 - hw7_test.sh (用以 reproduce 你的 public score, 誤差容忍 0.01)
 - model.bin (你 model 的 state_dict)
 - 各種 training 時需要的 Python 檔案
- 請不要上傳 dataset, 並且確定你 inference 是用 model.bin, 違者 0 分。
- 請確定你的 model.bin ≤ 20000 bytes, 違者 0 分。

Links

1. [torchvision ResNet18 fine-tuned from ImageNet \(state_dict\)](#) Accuracy ~ 88.41
2. [torchvision ResNet18 trained from scratch \(state_dict\)](#) Accuracy ~ 80.09
3. [Design Architecture colab tutorial](#)
4. [Knowledge Distillation colab tutorial](#)
5. [Network Pruning colab tutorial](#)
6. [Weight Quantization colab tutorial](#)
7. [state_dict of pre-trained model in Weight Quantization tutorial](#) Accuracy ~ 81.31