機器學習系統設計實務與應用 HW2

學號：B093040051 姓名：劉世文 系所：資工 113

HW2-1：

在本次作業當中，會藉由與 Baseline model 比較以下幾種差異，來更深入了解不同結構、超參數、activation function 對於模型表現的影響。

1. Model Structure: NN v.s CNN
2. Activation Function: ReLU v.s. Softmax v.s. Sigmoid
3. Optimizer: Adam v.s. SGD
4. Learning Rate: 0.05 v.s. 0.01 v.s. 0.25
5. Batch size: 64 v.s. 4 v.s. 1024

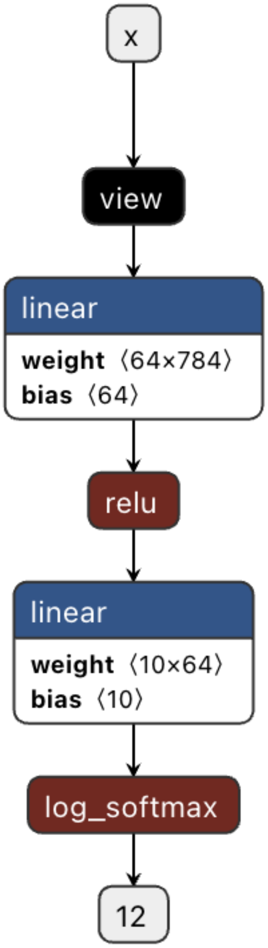
其中，Baseline model 會以紅色文字作為模型的訓練配置。

以下是實驗結果：

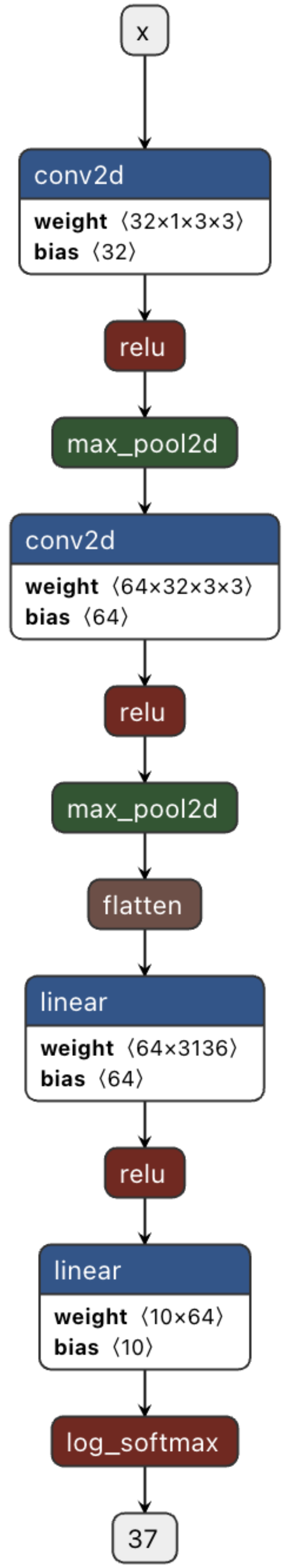
1. Model Structure: NN v.s CNN

- Activation Function: ReLU
- Optimizer: Adam
- Learning Rate: 0.05
- Batch size: 64
- Epoch: 10

NN_model_structure:



CNN_model_structure:



在這次的測試結果，因為分辨數字的任務相對簡單，所以不管是 NN 或是 CNN 都有非常良好的表現成果，NN 與 CNN，皆在第二個 epoch，train accuracy 就已經接近最後訓練結束的狀況。(epoch2:

- NN:

Epoch [2/10], phase: train, samples: 48000, Loss: 0.0039, Top-1 Accuracy: 92.49%, Top-3 Accuracy: 98.77%

- CNN:

Epoch [2/10], phase: train, samples: 48000, Loss: 0.0009, Top-1 Accuracy: 98.19%, Top-3 Accuracy: 99.87%

)

而在 train, validation, test 的準確度則是落差在 1%以內，並沒有出現過擬和的狀況。(epoch 10:

- NN:

Epoch [10/10], phase: train, samples: 48000, Loss: 0.0027, Top-1 Accuracy: 94.85%, Top-3 Accuracy: 99.33%

Epoch [10/10], phase: val, samples: 6000, Loss: 0.0038, Top-1 Accuracy: 93.55%, Top-3 Accuracy: 98.95%

Epoch [10/10], phase: test, samples: 10000, Loss: 0.0032, Top-1 Accuracy: 94.22%, Top-3 Accuracy: 99.08%

- CNN:

Epoch [10/10], phase: train, samples: 48000, Loss: 0.0003, Top-1 Accuracy: 99.28%, Top-3 Accuracy: 99.98%

Epoch [10/10], phase: val, samples: 6000, Loss: 0.0013, Top-1 Accuracy: 98.32%, Top-3 Accuracy: 99.88%

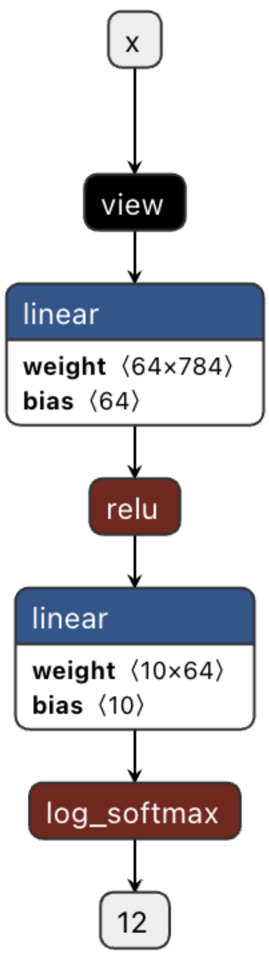
Epoch [10/10], phase: test, samples: 10000, Loss: 0.0010, Top-1 Accuracy: 98.71%, Top-3 Accuracy: 99.86%)

最後，比較兩模型的表現，可以發現 CNN 模型的表現略勝 NN 模型 4~5%，CNN 模型對於影像特徵的萃取以及提升模型效能上有顯著的效果。

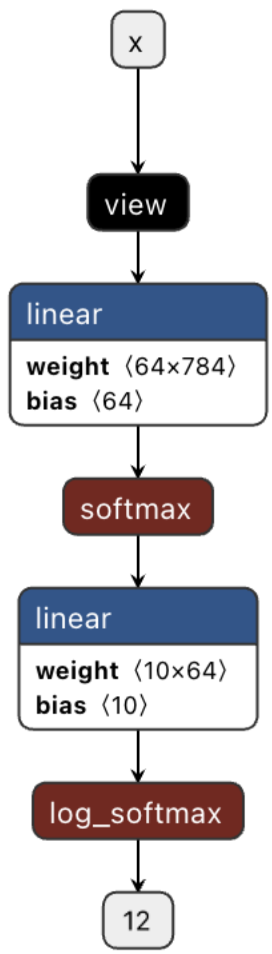
2. Activation Function: ReLU v.s. Softmax v.s. Sigmoid

- Model Structure: NN
- Optimizer: Adam
- Learning Rate: 0.05
- Batch size: 64
- Epoch: 10

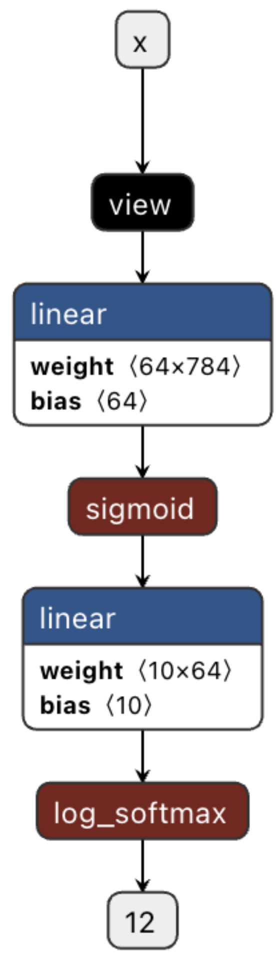
ReLU:



Softmax:



Sigmoid:



以下是實驗數據：

- ReLU:

Epoch [10/10], phase: train, samples: 48000, Loss: 0.0027, Top-1 Accuracy: 94.85%, Top-3 Accuracy: 99.33%

Epoch [10/10], phase: val, samples: 6000, Loss: 0.0038, Top-1 Accuracy: 93.55%, Top-3 Accuracy: 98.95%

Epoch [10/10], phase: test, samples: 10000, Loss: 0.0032, Top-1 Accuracy: 94.22%, Top-3 Accuracy: 99.08%

- Softmax:

Epoch [10/10], phase: train, samples: 48000, Loss: 0.0151, Top-1 Accuracy: 57.69%, Top-3 Accuracy: 94.87%

Epoch [10/10], phase: val, samples: 6000, Loss: 0.0155, Top-1 Accuracy: 57.27%, Top-3 Accuracy: 94.82%

Epoch [10/10], phase: test, samples: 10000, Loss: 0.0151, Top-1 Accuracy: 58.35%, Top-3 Accuracy: 95.07%

- Sigmoid:

Epoch [10/10], phase: train, samples: 48000, Loss: 0.0027, Top-1 Accuracy: 94.59%, Top-3 Accuracy: 99.22%

Epoch [10/10], phase: val, samples: 6000, Loss: 0.0032, Top-1 Accuracy: 93.90%, Top-3 Accuracy: 98.95%

Epoch [10/10], phase: test, samples: 10000, Loss: 0.0029, Top-1 Accuracy: 94.42%, Top-3 Accuracy: 99.09%

在這次的比較當中，ReLU 跟 sigmoid 的 activation function 對於最後模型的表現上相對接近，甚至幾乎一樣，因此可以了解在此任務及模型上，ReLU 與 Sigmoid 可能會帶來相近的效果。

不過因為老師上課有說過，這樣的現象可能僅限於較淺層的模型，若是層數增加，Sigmoid 的 activation function 便可能會帶來梯度消失的問題，屆時使用 ReLU 就會更有機會避免這樣的問題。

而在比較 Softmax 上，則可以看到使用 Softmax 的模型，並沒有在訓練階段做很好的收斂，因此 Softmax 會更適合作為最後輸出層在歸結預測結果的 function。

3. Optimizer: Adam v.s. SGD

- Model Structure: NN
- Activation Function: ReLU
- Learning Rate: 0.05
- Batch size: 64
- Epoch: 10

模型訓練效率比較：

- Adam:

Epoch [1/10], phase: train, samples: 48000, Loss: 0.0063, Top-1 Accuracy: 87.66%, Top-3 Accuracy: 96.96%

Epoch [2/10], phase: train, samples: 48000, Loss: 0.0039, Top-1 Accuracy: 92.49%, Top-3 Accuracy: 98.77%

Epoch [3/10], phase: train, samples: 48000, Loss: 0.0035, Top-1 Accuracy: 93.42%, Top-3 Accuracy: 99.02%

Epoch [4/10], phase: train, samples: 48000, Loss: 0.0032, Top-1 Accuracy: 93.88%, Top-3 Accuracy: 99.12%

Epoch [5/10], phase: train, samples: 48000, Loss: 0.0031, Top-1 Accuracy: 94.20%, Top-3 Accuracy: 99.14%

- SGD

Epoch [1/10], phase: train, samples: 48000, Loss: 0.0188, Top-1 Accuracy: 72.60%, Top-3 Accuracy: 89.86%

Epoch [2/10], phase: train, samples: 48000, Loss: 0.0081, Top-1 Accuracy: 86.81%, Top-3 Accuracy: 96.89%

Epoch [3/10], phase: train, samples: 48000, Loss: 0.0064, Top-1 Accuracy: 88.65%, Top-3 Accuracy: 97.39%

Epoch [4/10], phase: train, samples: 48000, Loss: 0.0058, Top-1 Accuracy: 89.60%, Top-3 Accuracy: 97.62%

Epoch [5/10], phase: train, samples: 48000, Loss: 0.0054, Top-1 Accuracy: 90.09%, Top-3 Accuracy: 97.79%

比較 Adam 跟 SGD 可以看到，模型在使用 Adam 的 optimizer 時，會更快的收斂，但不排除可能是模型在一開始有一個更好的起點，這樣的結果需要更多次實驗做驗

證。但不論是模型擬和的效率，或是最終模型的表現，Adam 都展現出更優且更穩定的訓練成效。

4. Learning Rate: 0.05 v.s. 0.01 v.s. 0.25

- Optimizer: Adam
- Model Structure: NN
- Activation Function: ReLU
- Batch size: 64
- Epoch: 10

模型訓練結果比較（前三個 epoch）：

- Learning Rate: 0.05

Epoch [1/10], phase: train, samples: 48000, Loss: 0.0063, Top-1 Accuracy: 87.66%, Top-3 Accuracy: 96.96%

Epoch [2/10], phase: train, samples: 48000, Loss: 0.0039, Top-1 Accuracy: 92.49%, Top-3 Accuracy: 98.77%

Epoch [3/10], phase: train, samples: 48000, Loss: 0.0035, Top-1 Accuracy: 93.42%, Top-3 Accuracy: 99.02%

- Learning Rate: 0.01

Epoch [1/10], phase: train, samples: 48000, Loss: 0.0073, Top-1 Accuracy: 86.69%, Top-3 Accuracy: 96.55%

Epoch [2/10], phase: train, samples: 48000, Loss: 0.0044, Top-1 Accuracy: 91.68%, Top-3 Accuracy: 98.39%

Epoch [3/10], phase: train, samples: 48000, Loss: 0.0035, Top-1 Accuracy: 93.37%, Top-3 Accuracy: 98.87%

- Learning Rate: 0.25

Epoch [1/10], phase: train, samples: 48000, Loss: 0.0225, Top-1 Accuracy: 46.72%, Top-3 Accuracy: 81.77%

Epoch [2/10], phase: train, samples: 48000, Loss: 0.0191, Top-1 Accuracy: 55.03%, Top-3 Accuracy: 87.44%

Epoch [3/10], phase: train, samples: 48000, Loss: 0.0186, Top-1 Accuracy: 57.05%, Top-3 Accuracy: 87.98%

模型訓練結果比較（後兩個 epoch）：

- Learning Rate: 0.05

Epoch [9/10], phase: train, samples: 48000, Loss: 0.0027, Top-1 Accuracy: 94.76%, Top-3 Accuracy: 99.30%

Epoch [10/10], phase: train, samples: 48000, Loss: 0.0027, Top-1 Accuracy: 94.85%, Top-3 Accuracy: 99.33%

- Learning Rate: 0.01

Epoch [9/10], phase: train, samples: 48000, Loss: 0.0017, Top-1 Accuracy: 96.74%, Top-3 Accuracy: 99.62%

Epoch [10/10], phase: train, samples: 48000, Loss: 0.0016, Top-1 Accuracy: 96.90%, Top-3 Accuracy: 99.67%

- Learning Rate: 0.25

Epoch [9/10], phase: train, samples: 48000, Loss: 0.0177, Top-1 Accuracy: 62.42%, Top-3 Accuracy: 88.60%

Epoch [10/10], phase: train, samples: 48000, Loss: 0.0177, Top-1 Accuracy: 62.53%, Top-3 Accuracy: 88.58%

在前三個 epoch 當中，Learning Rate 0.05 有最好的表現，0.01 則是在模型學習上有更緩慢的成長，而 0.25 則沒有辦法很好地使模型收斂。

在後兩個 epoch 當中，Learning Rate 0.01 雖然花了更長的時間達到 0.05 的模型表現，卻在最後超越 0.05 的表現，達到更好的終點。而 0.25 直到最後也沒有辦法將模型收斂在一個很好的結果，0.25 的 Learning Rate 對於此次任務、模型、loss function 來說過大。

5. Batch size: 64 v.s. 4 v.s. 1024

- Learning Rate: 0.05
- Optimizer: Adam
- Model Structure: NN
- Activation Function: ReLU
- Epoch: 10

模型訓練結果比較：

- Batch size: 64

Epoch [1/10], phase: train, samples: 48000, Loss: 0.0063, Top-1 Accuracy: 87.66%, Top-3 Accuracy: 96.96%

Epoch [2/10], phase: train, samples: 48000, Loss: 0.0039, Top-1 Accuracy: 92.49%, Top-3 Accuracy: 98.77%

Epoch [3/10], phase: train, samples: 48000, Loss: 0.0035, Top-1 Accuracy: 93.42%, Top-3 Accuracy: 99.02%

Epoch [4/10], phase: train, samples: 48000, Loss: 0.0032, Top-1 Accuracy: 93.88%, Top-3 Accuracy: 99.12%

Epoch [5/10], phase: train, samples: 48000, Loss: 0.0031, Top-1 Accuracy: 94.20%, Top-3 Accuracy: 99.14%

- Batch size: 4

Epoch [1/10], phase: train, samples: 48000, Loss: 0.1476, Top-1 Accuracy: 82.29%, Top-3 Accuracy: 95.61%

Epoch [2/10], phase: train, samples: 48000, Loss: 0.1259, Top-1 Accuracy: 86.09%, Top-3 Accuracy: 96.52%

Epoch [3/10], phase: train, samples: 48000, Loss: 0.1223, Top-1 Accuracy: 86.76%, Top-3 Accuracy: 96.67%

Epoch [4/10], phase: train, samples: 48000, Loss: 0.1181, Top-1 Accuracy: 87.29%, Top-3 Accuracy: 96.84%

Epoch [5/10], phase: train, samples: 48000, Loss: 0.1168, Top-1 Accuracy: 87.44%, Top-3 Accuracy: 96.92%

- Batch size: 1024

Epoch [1/10], phase: train, samples: 48000, Loss: 0.0009, Top-1 Accuracy: 72.73%, Top-3 Accuracy: 89.02%

Epoch [2/10], phase: train, samples: 48000, Loss: 0.0004, Top-1 Accuracy: 89.21%, Top-3 Accuracy: 97.61%

Epoch [3/10], phase: train, samples: 48000, Loss: 0.0003, Top-1 Accuracy: 90.74%, Top-3 Accuracy: 97.95%

Epoch [4/10], phase: train, samples: 48000, Loss: 0.0003, Top-1 Accuracy: 91.38%, Top-3 Accuracy: 98.19%

Epoch [5/10], phase: train, samples: 48000, Loss: 0.0003, Top-1 Accuracy: 91.88%, Top-3 Accuracy: 98.35%

最後要比較的是不同的 Batch size，batch size 除了表示模型在每次更新前，參考及計算多少資料的 loss，也會影響到每個 epoch 的 iteration。

在 batch size 4 的時候，雖然在相同的 epoch 時，會使模型相對於 batch size 64 的模型 16 倍的 iteration，但是因為每次更新的參考資料數量更小，所以相對的 loss 中噪音的比例會更容易被放大，使得模型訓練過程不穩定，也因此達不到 batch size 64 訓練出來的模型效果。而當 batch size 為 1024，雖然能夠大幅縮小每次更新的噪音比例，但相對的，在每個 epoch 當中所經歷的 iteration 也會大幅減少，但若比較相同的 iteration，大的 batch size 則會有更好的泛化能力。

HW2-2:

在本次的作業，我們要來比較不同預處理及模型結構，對於辨識效果的影像，以下先介紹本次做的預處理方法，以及使用到的模型結構。

預處理方法：

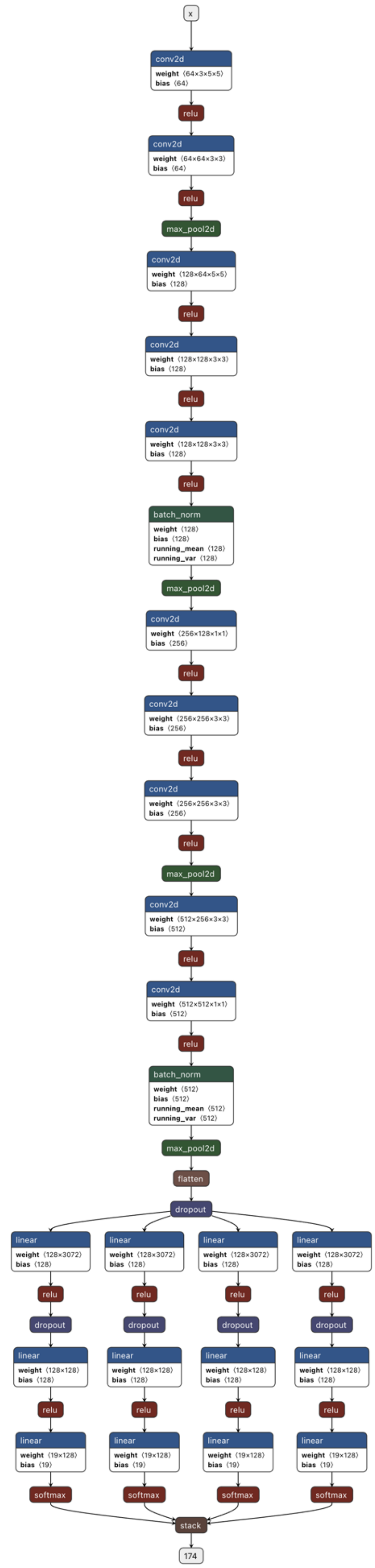
- Train Original: 將影像 Reshape 成(50, 130, 3)，並且正規化。
- Denoise: 將影像去除回歸線、雜訊後，Reshape 成(50, 130, 3)，並且正規化。
- Denoise 2: 將影像去除雜訊後，Reshape 成(50, 130, 3)，並且正規化。

模型結構：

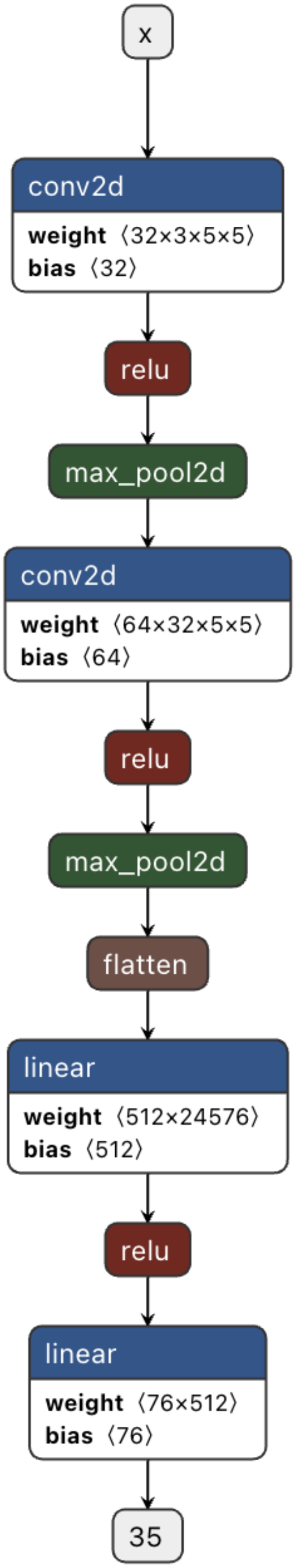
1. 助教提供的深度學習模型 – Baseline model，參數量 = 4,845,196。
2. 簡單的 CNN 模型 – SimpleCNN，參數量 = 12,674,508。
3. ResNet 模型 – ResNet18，參數量 = 11,209,228。
4. 改良後的模型 – ResNet18_mod，參數量 = 95,164。

以下是模型結構的圖像化：

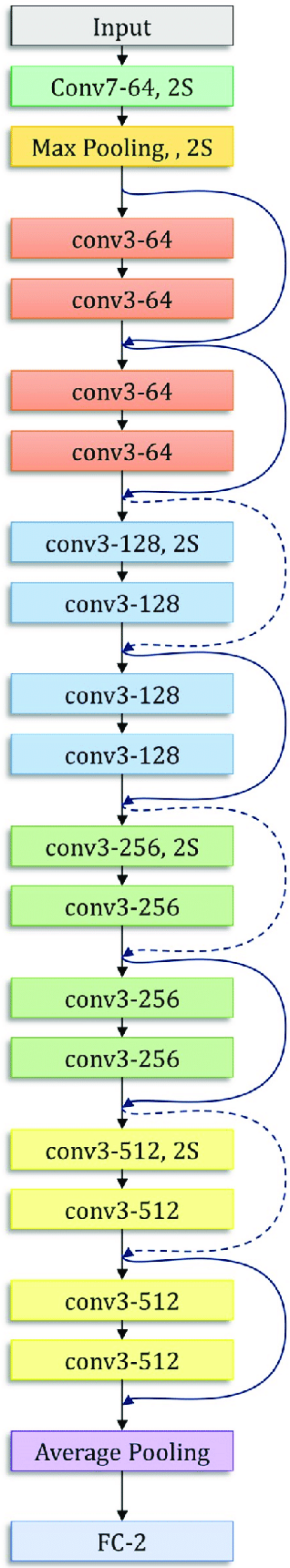
1. Baseline model


x
conv2d
weight (64×3×5×5)
bias (64)
relu
conv2d
weight (64×64×3×3)
bias (64)
relu
max_pool2d
conv2d
weight (128×64×5×5)
bias (128)
relu
conv2d
weight (128×128×3×3)
bias (128)
relu
conv2d
weight (128×128×3×3)
bias (128)
relu
batch_norm
weight (128)
bias (128)
running_mean (128)
running_var (128)
max_pool2d
conv2d
weight (256×128×1×1)
bias (256)
relu
conv2d
weight (256×256×3×3)
bias (256)
relu
conv2d
weight (256×256×3×3)
bias (256)
relu
max_pool2d
conv2d
weight (512×256×3×3)
bias (512)
relu
conv2d
weight (512×512×1×1)
bias (512)
relu
batch_norm
weight (512)
bias (512)
running_mean (512)
running_var (512)
max_pool2d
flatten
dropout
linear
weight (128×3072)
bias (128)
linear
weight (128×3072)
bias (128)
linear
weight (128×3072)
bias (128)
linear
weight (128×3072)
bias (128)
relu
relu
relu
relu
dropout
dropout
dropout
dropout
linear
weight (128×128)
bias (128)
linear
weight (128×128)
bias (128)
linear
weight (128×128)
bias (128)
linear
weight (128×128)
bias (128)
relu
relu
relu
relu
linear
weight (19×128)
bias (19)
linear
weight (19×128)
bias (19)
linear
weight (19×128)
bias (19)
linear
weight (19×128)
bias (19)
softmax
softmax
softmax
softmax
stack
174

2. SimpleCNN


x
conv2d
weight ⟨32×3×5×5⟩
bias ⟨32⟩
relu
max_pool2d
conv2d
weight ⟨64×32×5×5⟩
bias ⟨64⟩
relu
max_pool2d
flatten
linear
weight ⟨512×24576⟩
bias ⟨512⟩
relu
linear
weight ⟨76×512⟩
bias ⟨76⟩
35

3. ResNet18


Input
Conv7-64, 2S
Max Pooling, , 2S
conv3-64
conv3-64
conv3-64
conv3-64
conv3-128, 2S
conv3-128
conv3-128
conv3-128
conv3-256, 2S
conv3-256
conv3-256
conv3-256
conv3-512, 2S
conv3-512
conv3-512
conv3-512
Average Pooling
FC-2

4. ResNet18_mod：是將 ResNet18 的 Channel 由 [64, 64, 128, 256, 512]，改變為[16, 16, 16, 32, 32]，以此降低模型餐數量。

以下為實驗數據：

1. Baseline model:

- Data: Train origin

Epoch [25/25], phase: train, samples: 4000, Loss: 0.2789, Top-1 Accuracies: ['68.62%', '82.47%', '85.22%', '83.53%']

Epoch [25/25], phase: val, samples: 1000, Loss: 0.2854, Top-1 Accuracies: ['64.60%', '84.80%', '87.40%', '83.30%']

Epoch [25/25], phase: test, samples: 3000, Loss: 0.2818, Top-1 Accuracies: ['64.37%', '82.40%', '85.43%', '80.90%']

Elapsed time: 72.57933807373047 seconds

- Data: Denoise

Epoch [25/25], phase: train, samples: 4000, Loss: 0.2929, Top-1 Accuracies: ['75.05%', '47.10%', '71.75%', '81.08%']

Epoch [25/25], phase: val, samples: 1000, Loss: 0.3000, Top-1 Accuracies: ['74.80%', '46.40%', '72.00%', '81.10%']

Elapsed time: 58.91046595573425 seconds

- Data: Denoise2

Epoch [25/25], phase: train, samples: 4000, Loss: 0.2861, Top-1 Accuracies: ['67.42%', '74.65%', '76.08%', '78.57%']

Epoch [25/25], phase: val, samples: 1000, Loss: 0.2945, Top-1 Accuracies: ['65.50%', '73.50%', '76.90%', '76.40%']

Elapsed time: 59.257035970687866 seconds

2. SimpleCNN:

- Data: Train origin

Epoch [25/25], phase: train, samples: 4000, Loss: 0.3672, Top-1 Accuracies: ['6.65%', '6.30%', '6.25%', '6.40%']

Epoch [25/25], phase: val, samples: 1000, Loss: 0.3768, Top-1 Accuracies: ['5.50%', '6.20%', '5.60%', '5.60%']

Epoch [25/25], phase: test, samples: 3000, Loss: 0.3695, Top-1 Accuracies: ['6.47%', '5.17%', '5.47%', '5.20%']

Elapsed time: 25.39776062965393 seconds

- Data: Denoise

Epoch [25/25], phase: train, samples: 4000, Loss: 0.0053, Top-1 Accuracies: ['99.48%', '98.47%', '97.38%', '99.05%']

Epoch [25/25], phase: val, samples: 1000, Loss: 0.2166, Top-1 Accuracies: ['73.40%', '66.80%', '65.30%', '84.10%']

Elapsed time: 23.59954285621643 seconds

- Data: Denoise2

Epoch [25/25], phase: train, samples: 4000, Loss: 0.3673, Top-1 Accuracies: ['5.62%', '6.30%', '6.05%', '5.58%']

Epoch [25/25], phase: val, samples: 1000, Loss: 0.3766, Top-1 Accuracies: ['7.40%', '6.20%', '7.10%', '5.00%']

Elapsed time: 23.52213764190674 seconds

3. ResNet18:

- Data: Train origin

Epoch [25/25], phase: train, samples: 4000, Loss: 0.0000, Top-1 Accuracies: ['100.00%', '100.00%', '100.00%', '100.00%']

Epoch [25/25], phase: val, samples: 1000, Loss: 0.0030, Top-1 Accuracies: ['99.60%', '99.70%', '100.00%', '99.80%']

Epoch [25/25], phase: test, samples: 3000, Loss: 0.0114, Top-1 Accuracies: ['99.30%', '99.23%', '99.47%', '99.00%']

Elapsed time: 49.64688324928284 seconds

- Data: Denoise

Epoch [25/25], phase: train, samples: 4000, Loss: 0.0000, Top-1 Accuracies: ['100.00%', '100.00%', '100.00%', '100.00%']

Epoch [25/25], phase: val, samples: 1000, Loss: 0.0042, Top-1 Accuracies: ['99.60%', '99.20%', '99.80%', '99.90%']

Elapsed time: 43.79024338722229 seconds

- Data: Denoise2

Epoch [25/25], phase: train, samples: 4000, Loss: 0.0000, Top-1 Accuracies: ['100.00%', '100.00%', '100.00%', '100.00%']

Epoch [25/25], phase: val, samples: 1000, Loss: 0.0036, Top-1 Accuracies: ['99.30%', '99.60%', '99.70%', '100.00%']

Elapsed time: 44.71420383453369 seconds

4. ResNet18_mod:

- Data: Train origin

Epoch [25/25], phase: train, samples: 4000, Loss: 0.0011, Top-1 Accuracies: ['100.00%', '100.00%', '100.00%', '100.00%']

Epoch [25/25], phase: val, samples: 1000, Loss: 0.0137, Top-1 Accuracies: ['97.40%', '96.90%', '97.20%', '98.90%']

Epoch [25/25], phase: test, samples: 3000, Loss: 0.0195, Top-1 Accuracies: ['97.33%', '95.77%', '97.27%', '98.17%']

Elapsed time: 46.727765798568726 seconds

- Data: Denoise

Epoch [25/25], phase: train, samples: 4000, Loss: 0.0018, Top-1 Accuracies: ['100.00%', '100.00%', '99.98%', '100.00%']

Epoch [25/25], phase: val, samples: 1000, Loss: 0.0257, Top-1 Accuracies: ['96.40%', '90.30%', '93.20%', '98.10%']

Elapsed time: 42.23278546333313 seconds

- Data: Denoise2

Epoch [25/25], phase: train, samples: 4000, Loss: 0.0014, Top-1 Accuracies: ['100.00%', '100.00%', '100.00%', '100.00%']

Epoch [25/25], phase: val, samples: 1000, Loss: 0.0162, Top-1 Accuracies: ['97.10%', '92.80%', '95.90%', '99.10%']

Elapsed time: 41.3394889831543 seconds

以本次的實驗結果來說，比較 original、denoise、denoise2 對於各個模型的影響，可以發現 original 在各個模型所需要花費的時間都會高出其他兩種資料預處理方法，這是因為 original 的資料有三個 channel，而 denoise、denoise2 則只有一個 channel，因此在計算上需要花費更多的時間，因此以時間作為考量的話，應從 denoise 及 denoise 2 之間擇一。

比較 denoise 及 denoise2 對於模型表現的影響，在 Baseline model, ResNet18, ResNet18_mod 的效果接近，所以就單論結構簡單的 SimpleCNN，在這個模型來看，可以發現 denoise 可以最大程度改善模型辨識的能力，而 denoise2 跟 original 的資料則是使模型表現幾乎一樣，模型並沒有辦法在這種資料中找到有用的辨識特徵以達到收斂。

接下來，需要設計新的模型架構來滿足 1M 以下的參數量同時維持 80%以上的準確度。首先 Baseline model 雖然表現良好，但結構複雜，因此設計了一個結構簡單的 SimpleCNN 作為測試，但結果並不好。而後測試非常擅長影像任務的 ResNet18 模型，並得到非常良好的成績，因此接下來就可以嘗試基於 ResNet18 改良模型。

一開始測試將 ResNet18 的層數縮減，改為 ResNet6、ResNet10、ResNet14，但是模型的表現都不如預期，並且無法有效降低模型的參數量，因此改將設計策略轉為減少每層 Conv 的 channel 數，將 channel 從原先的[64, 64, 128, 256, 512]，改變為[16, 16, 16, 32, 32]，在保持模型深度的同時，使參數量大幅下降，並且維持接近於 ResNet18 本身的準確度。

最後，透過 ResNet18_mod 的結構，讓模型使用 95K 的參數量，在 Original 的 Test Data 的驗證碼辨識達到 97.33%、95.77%、97.27%、98.17%的準確率。