Lab01 Report – DeepLearning

學生：梁皓鈞，學號：314580042，系所：前瞻半導體研究所晶片組

I. How to improve the accuracy (list your method)

1. Your network

如 Figure1 所示，網路的輸入最初為形狀 (Batch_size, 784) 的向量，會先轉換為 NCHW 格式的(Batch_size,1,28,28)灰階影像。接著，經由一層 3×3 卷積（Stride=1，輸出通道數=32），得到輸出(Batch_size,32,28,28)的特徵圖；接著進入中間的核心區域。

中間部分主要由 Block1 與 Block2 所組成，兩者皆採用 Pre-Act 架構，其核心單元為兩組 Batch Normalization + SiLU + 3×3 卷積，並搭配 Residual 結構。兩者的差異在於：Block1 的起始卷積採用 Stride=2 進行 Downsampling，Residual 分支則需透過一層 1×1 卷積（Stride = 2）來完成 Projection。Block2 則維持 Stride = 1，不進行下採樣。經過三次 Block1 與 Block2 的堆疊後，最初的 (Batch_size, 32, 28, 28) 特徵圖會逐步轉換為 (Batch_size, 128, 4, 4)。隨後，透過 Global Average Pooling，輸出被壓縮成 (Batch_size, 128, 1, 1)。最後，經過一層 Linear 全連接層與 Softmax，模型會輸出 10 個類別對應的機率分佈。

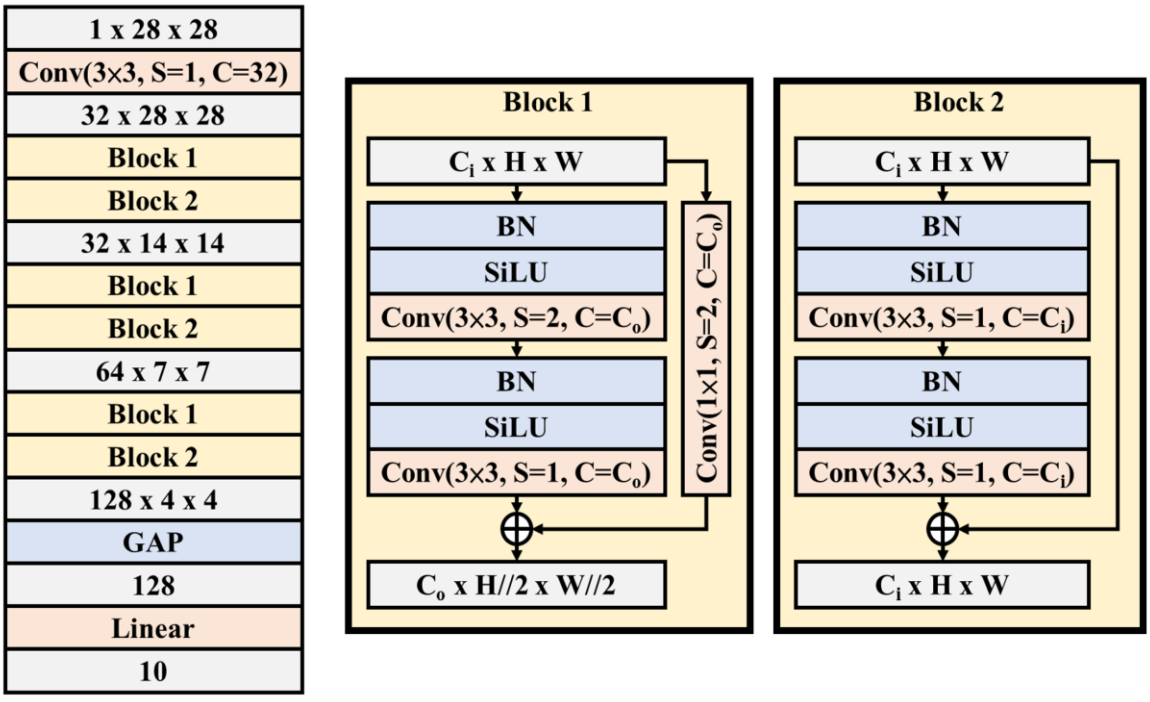

Figure1

2. *Loss function*

在此實驗中，我採用了 Cross Entropy 作為損失函數，其數學形式如下所示。其中，N 代表 Batch Size；$y_{(n,i)}$代表在 Batch 中第 n 個樣本第 i 個類別上的 One-Hot 標籤，$\hat{y}_{(n,i)}$則表示模型對 Batch 中第 n 個樣本在第 i 個類別上的預測機率值。

$$\text{Loss} = -\frac{1}{N}\sum_{n=1}^{N}\sum_{i=0}^{10} y_{n,i}\log(\hat{y}_{n,i})$$

3. *Activation function*

在本模型中，我選用了 SiLU (Sigmoid Linear Unit) 作為 Activation Function。其數學定義為

$$\text{SiLU}(x) = x \cdot \sigma(x) = \frac{x}{1+e^{-x}}$$

其中 $\sigma(x)$ Sigmoid 函數。與 ReLU 相比，SiLU 在零點附近較平滑且可導，如 Figure 2 所示，能夠避免因梯度不連續而造成的訓練不穩定問題。同時，對於負值輸入，SiLU 不會完全截斷，而是給予一定的抑制，這使得模型在保留訊息流動的同時，能避免老師課堂上所提到的梯度消失。研究結果顯示，採用 SiLU 的深度學習模型在訓練收斂速度與最終準確率上，通常能優於使用 ReLU 的模型。因此，在本實驗中使用 SiLU 有助於增強非線性表達能力並提升模型的整體效能。

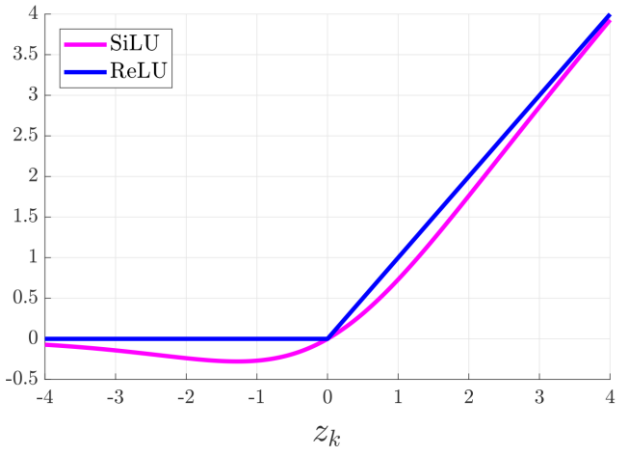

Figure 2[1]

4. *Hyperparameters*

在本實驗中，所使用的訓練資料共計 60000 筆，測試資料則為 10000 筆。模型訓練的 Epoch

[1] Sigmoid-Weighted Linear Units for Neural Network Function Approximation in Reinforcement Learning, url= https://arxiv.org/abs/1702.03118

數設定為 30，Batch Size 為 50，初始 Learning Rate 為 0.03。此外，學習率調整策略採用了 Step Learning Rate Scheduler，在第 15 個 Epoch 時將學習率降低為原本的十分之一，並於第 25 個 Epoch 再次降低至原本的十分之一，以逐步提升模型收斂效果。

5. Optimizations

由於輸入網路的資料最初為形狀 (Batch_size, 784) 的向量，其中每個元素的數值範圍介於 0 至 255 之間，因此我先將其除以 255 進行 Normalization，將像素值縮放至 [0, 1] 區間。除此之外，為了提升模型的泛化能力，我進行了 Data Augmentation，具體包括：在水平方向與垂直方向上隨機平移最多 2 個像素，以及隨機進行水平翻轉。

II. What differences do you find between the results of Task1 and Task2?

Task 1 以及 Task 2 的 Validation Accuracy 截圖分別是 Figure 3 以及 Figure 4，分別為 92.4400 %以及 92.3500 %，皆大於 80 %。他們之間的 Validation Accuracy 較沒有差別，比較有差別的地方在於前者的 Training Accuracy 比較差、Epoch Time 也較長，推測原因來自於 Pytorch 的算法相對於我自己手刻的 numpy 算法，具有較高的運算效能。

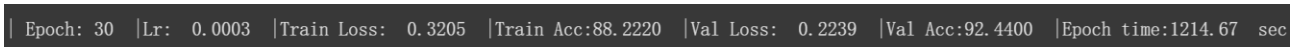

Figure 3

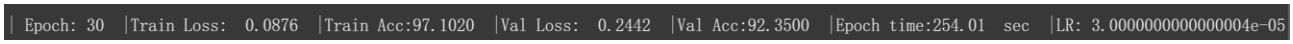

Figure 4