# Deep Learing HW2 - LeNet/Computational Graph

R76111206 曾凱揚
April 19, 2023

***Abstract - This report is an description document for HW1. And the GitHub link is as follow***

***Github Link, folder HW2***
***https://github.com/yangtseng219/DeepLearning***

## I. Introduction

本次作業將延續作業一的圖片萃取特徵與結果透過實作進行圖片的分類任務，將圖片數據套用至各個模型訓練，最終比較各類模型的成效。其中作業一的圖片特徵萃取是採用方向梯度直方圖（Histogram of Oriented Gradient, HOG），為一維的向量，維度不符合原始Lenet5輸入的(32, 32)，因此重新抓取二維的特徵，並於image_feature.ipynb中實作。

作業二將要求以下三種模型:

*1)* Two Layer Perceptron
*2)* LeNet 5
*3)* Improved LeNet5

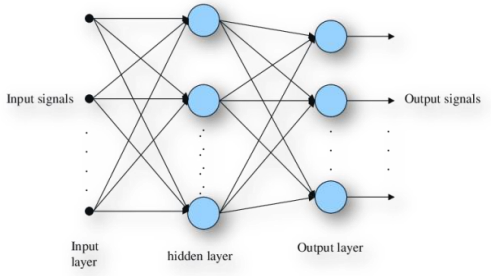
## II. Method

### A. *Two Layer Perceptron*

*1) Hyperparameter*

*a)* Input layer: 2516
*b)* Hidden layer: 50
*c)* Output layer: 50
*d)* Batch size: 32
*e)* Epoch: 50
*f)* Learning rate: 1e-6

*2) Model*




*3)*

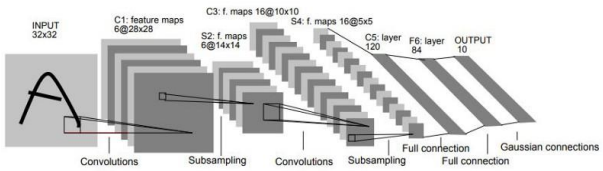
### B. *LeNet5*

使用Github上由chuanqi305所提出的開原程式碼，並將作者原先使用ReLU激活函數更換成Sigmoid激活函數。

*1) Hyperparameter*

a) Batch size: 32
b) Epoch: 10
c) Learning rate: 0.001

*2) Model*

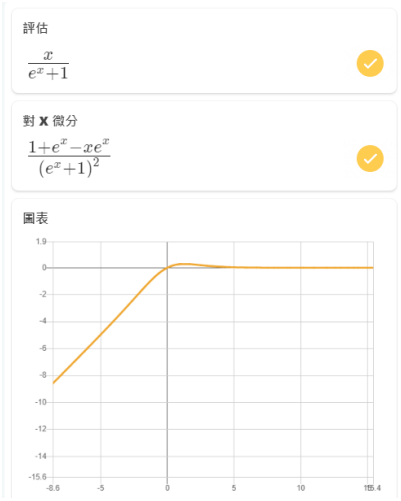



*3)* Optimization: SGD

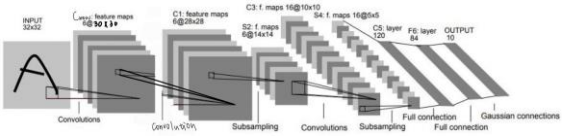
### C. *Improve LeNet5*

*1) Improved*

a) *Convolution kernel 5*5 > 3*3*
b) 增加一個Convoulution layer
此處新增於原input與第一層卷積層中間
c) 激活函數改為x*sigmoid(x)




*2) Model*



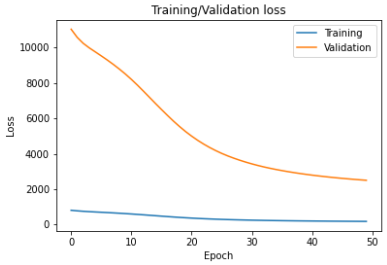

*3) Hyperparameter*

a) Batch size: 32
b) Epoch: 10
c) Learning rate: 0.001

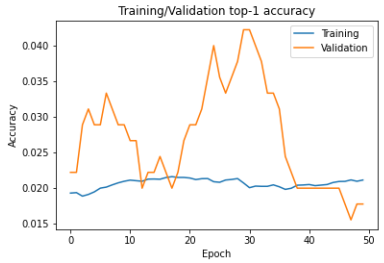
## III. RESULT

### A. *Two Layer Perceptron*

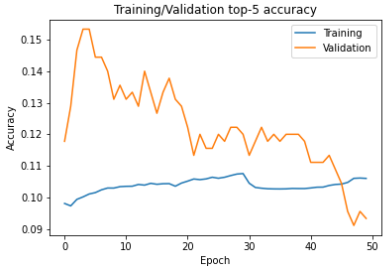
#### 1) *Loss*




#### 2) *Top-1 Accuracy*

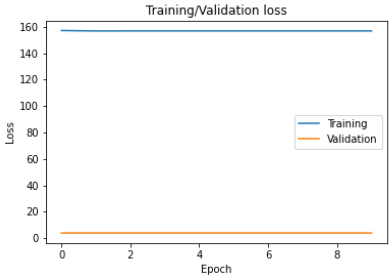



#### 3) *Top-5 Accuracy*

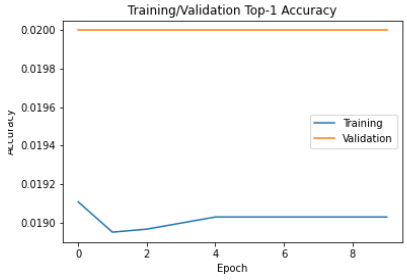



### B. *LeNet5*

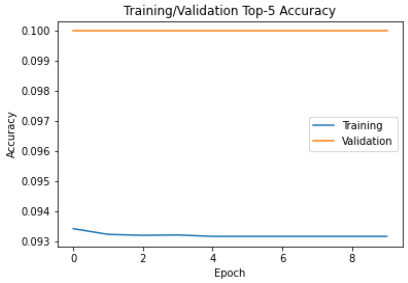
#### 1) *Loss*




#### 2) *Top-1 Accuracy*

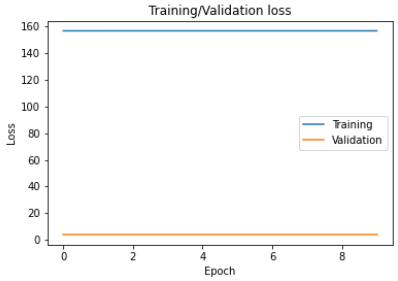



#### 3) *Top-5 Accuracy*

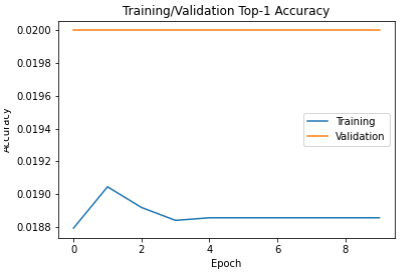



### C. *Improve LeNet5*

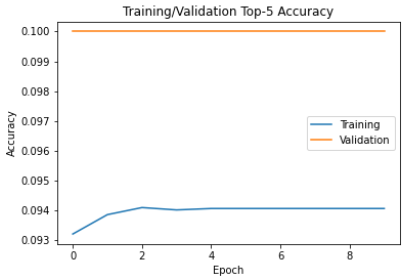
#### 1) *Loss*




#### 2) *Top-1 Accuracy*




#### 3) *Top-5 Accuracy*




## IV. COMPARISON

在此小節比較準確度，以下由Top-1之Accuracy來進行比較，表一比較One layer perceptron以及Two layer perceptron，表二比較Lenet5以及Improve Lenet5:

表一 Perceptron 之比較

| Model | Train | Validation | Test |
|---|---|---|---|
| One layer | 0.03657 | 0.0311 | 0.9667 |
| Two layer | 0.0211 | 0.0178 | 0.0311 |

表二 LeNet5 之比較

| Model | Train | Validation | Test |
|---|---|---|---|
| Lenet5 | 0.0190 | 0.020 | 0.020 |
| Improved | 0.0189 | 0.0020 | 0.1022 |

## V. Conclusion

透過本次的作業了解到前向與反向傳播的算法，然而在實作本次作業中，各階層傳遞的參數維度對應是本次作業中遇到的最大問題。

## References


[1] LeNet5: https://github.com/chuanqi305/LeNet5
[2] Two layer Perceptron: https://www.youtube.com/playlist?list=PLqXS1b2lRpYTpUIEu3oxfhhTuBXmMPppA
[3] ChatGPT