Assignment-2 Pytorch/Tensorflow in LeNet5

*1 Shao-Ning, Chen* (陳劭寧)

1 Instute of Data Science,

National Cheng Kung University, Taiwan,

E-mail: re6101037@gs.ncku.edu.tw

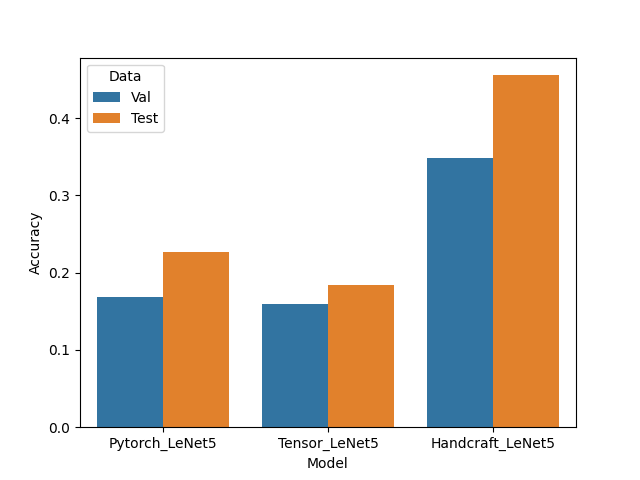
#### Abstract

本次作業需要找出並修改臉部驗證的程式跟架構，本次作業的程式碼跟結果的部分如[1]所示

**Keywords:** Face verification, Tensorflow, LFW dataset

**1. Introduction**

本次作業主軸仍是圖像分類，在上次作業[7]中使用了Single Layer Perceptron及Multi Layer Perceptron來做分類，獲得準確度接近0的結果，基本上跟亂猜沒甚麼差別。本次在Tensorflow[6]、Pytorch[3,4,5] 以及Handcraft[2]的基底下各自實現LeNet，並且在Tensorflow[6]、Pytorch[3,4,5]方法中獲得了約0.2的準確率，並且在Handcraft[2]方法中更是獲得了約0.4的準確率，視覺化結果如Fig. 1所示。



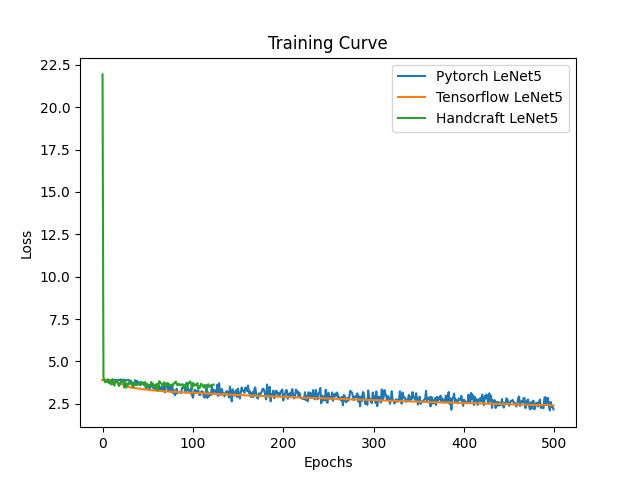
**Fig. 1.** Validation and test accuracy comparison between Tensorflow[6], Pytorch[3,4,5] and Handcraft[2] three methods.

**2. Experiment**

總圖片數共有64225張，其中63325張圖片用來做訓練，450張圖片用於驗證，剩下450張圖片用來做測試，要分的類別數共有50個。公平起見，所以三個方法的learning rate都設為0.0001，momentum設為0.9，迭帶次數方面，tensorflow[6]跟pytorch都是設500，handcraft的部分，因為只能用CPU訓練的關係，所以訓練時間過長，折衷之下只訓練125次。其中tensorflow[6]只花了1個小時出頭，pytorch[3,4,5]花了將近6小時，而handcraft[2]花了13個多小時才完成。此外，本次訓練驗證測試所使用的硬體規格為24核TR 3960X CPU、384 GB RAM和Nvidia RTX 3090 GPU的主機上進行的。

**3. Result**

在訓練過程中，雖然三者[2,3,4,5,6]的loss都有下降趨勢，並且training accuracy一直不斷提升，但驗證跟測試結果都收斂於0.5以下，其中pytorch[3,4,5]的準確度來的比tensorflow[6]好些，不過令我意外的是，handcraft[2]版雖然訓練次數遠少於其他兩者，但結果明顯優於其他兩者。雖然時間上花的比較多，但至少它是有比較好，也符合效益原則。



**Fig. 2.** Training curve comparison between Tensorflow[6], Pytorch[3,4,5] and Handcraft[5] three methods.

**4. conclusion**

先前作業所使用的Single Layer Perceptron及Multi Layer Perceptron方法，基本上是不太可用的。不過本次作業所使用的LeNet，準確度從低於0.05提升到了0.4左右，雖然就結果論而言可能還不夠好，加上訓練跟測試所花的時間非常久，但也算是成功跨越一大步了。

**References**

[1] https://github.com/johnny373/Assignment-2-Pytorch-Tensorflow-in-LeNet5

[2] Huang, M., Zhang, W., Pan, L., Yan, X., Zhao, G., Chen, H., ... & Li, J. (2021). Supporting Information for “Self-adaptive Learning in Memristor Convolutional Neural Networks”. Authorea Preprints.

[3] Pytorch : LeNet神經網路結構

https://jennaweng0621.pixnet.net/blog/post/403588244-%5Bpytorch%5D-lenet%E7%B6%B2%E8%B7%AF%E7%B5%90%E6%A7%8B-%2B-mnist%E6%95%B8%E5%AD%97%E8%A8%93%E7%B7%B4

[4] PyTorch 學習筆記 : 模型建立步驟與 nn.Module

https://iter01.com/524139.html

[5] Pytorch official website

https://pytorch.org/tutorials/beginner/blitz/neural\_networks\_tutorial.html

[6] Github

https://gist.github.com/AFAgarap/86b2ba313d4a61c5470721dfad0c09d4

[7] Github

https://github.com/johnny373/Assignment-1-Backpropagation-and-Image-Recognition/tree/main