CIFAR10 Classification

B083040029邱品諺

* 1. Dataset and Dataloader

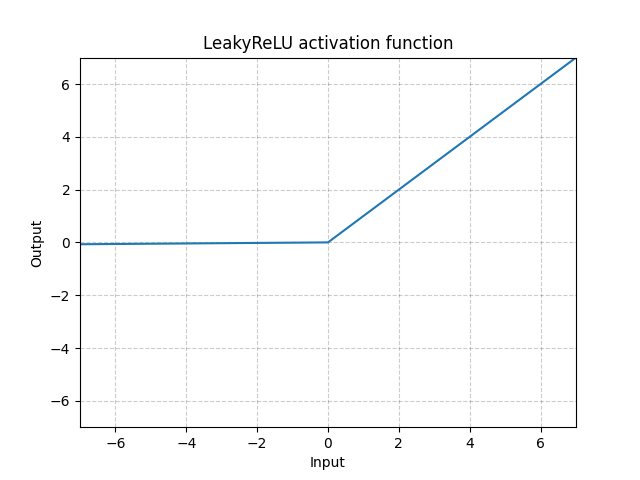
資料集處理及載入的部分和過去作業不同在於本次作業使用了FlattenTransform，也就是將以多維陣列表示的圖片展開成一維陣列表示，此方式為給予神經網路訓練最簡單的圖片儲存方式，但其無法表示圖片中各物件的空間關係。

* 1. Data Augmentation

模型的訓練需要大量的資料，但人為產生的資料再多也有限，再加上需要人類標注，因此資料數量無法滿足訓練的需求，因此需要Data Augmentation來增加資料的數量及廣度，如此模型可更加的general及robust。Data Augmentation的方式有以下幾種：圖片翻轉、旋轉、調整飽合度、亮度、Cropping等等。

* 1. Layers

Activation Function的部分除了Sigmoid及ReLU外，本次將介紹LeakyReLU和Tanh。首先，LeakyReLU和ReLU最大的差別在於小於0的部分不為0，因此不會有ReLU只有右半邊能計算Gradient的情況，且能保有ReLU可快速收斂且不會飽和的優點。以下為LeakyReLU的計算式及函式圖：

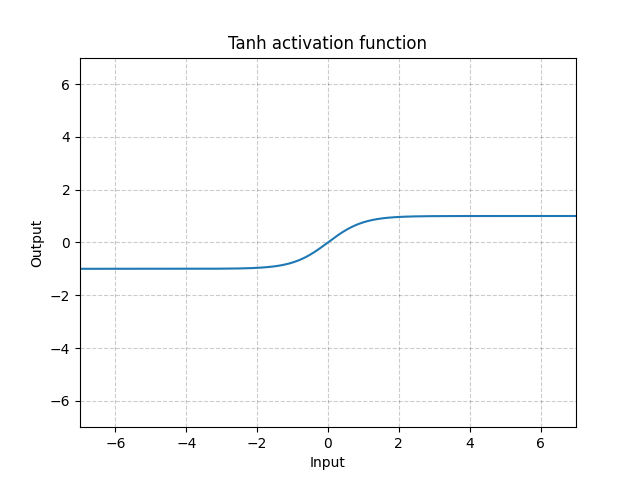


而Tanh和Sigmoid相似，但Tanh較Sigmoid好的一點是他的中心點為0，輸出範圍為

0~1，但仍然有和Sigmoid一樣會有飽和的問題，也就是在較大及較小的輸入值中，通

過此Activation Function後所產生的輸出無法進行微分，亦或是輸出值已趨近於0。以

下為Tanh的計算式及函式圖：



* 1. Regularization loss

計算Loss Function除了計算預測值與Ground Truth的差距外，還會加上Regularization Loss以防止模型Overfitting， 也就是訓練的結果過度地符合訓練資料。而Regularization Loss 之所以可以放止Overfitting是因為其可使模型變得較簡單、較General，像是在L2 Regularization，由於計算方式為將每個Weight取平方相加，因此如果有較大的Weight便會使得整體的Loss較大，因此經過更新後會使得weight間的變異較小，模型也不會為了要過度地學習Training set中某項偏差的資料而學到較極端、不符合整體情形的weight。

1. An overview of hyperparameters

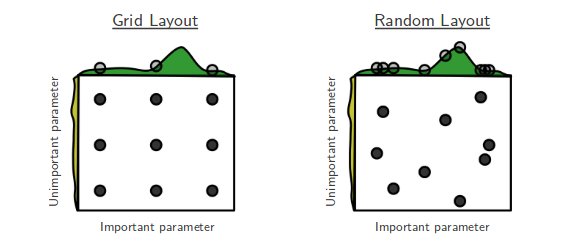
訓練神經網路除了讓model自己學習model parameters，像是weights, bias，另外，有些參數像是hyperparameters是在訓練前需要我們自己決定，並透過一次次的訓練結果來調整的，而這些hyperparameters包含網路的架構像是Activation Function、網路的層數、每一層要有幾個Neurons，Learning rate、epochs、Batch size、Regularization strength等等。

而在訓練網路一開始，應該從少量的資料開始訓練，從1、10、1000慢慢增加資料的數量，一開始我們會發現在訓練資料極少的情形下訓練的準確率都是100%，那是因為模型並沒有學到東西，他只是背下來而已，因此在Validation set的準確率非常差。資料量越來越多後，會發現training和validation的loss都開始下降，但總在半路上就開始反升，此時一大原因是因為資料量的不足，因此除了需要增加資料量外，其實我們可以觀察訓練的結果找出規律，像是通過哪些Activation Function較好、幾層的神經網路結果較好等等，以調整Hyperparameters，但是手動調整這些參數實在是一個不太有效率的做法，因此本作業介紹了兩種除了手動挑整參數外的參數調整方式。

1. Hyperparameter Tuning

首先為Grid Search，透過指定每個參數的可能組合進行訓練，像是使用者指定三種learning rate，regularization strength有兩種，因此將有3x2=6種訓練組合，如此我們可以觀察訓練的結果來調整hyperparameters。

但有時候我們無法準確的知道確切的參數進行訓練並比較各個參數組合的好壞，我們只約略知道大概的範圍，因此有了Random Search的方法。Random Search使得我們可以決定各參數的範圍，在每次訓練隨機抽樣範圍中的數值進行訓練，並觀察多次的訓練結果以調整參數。

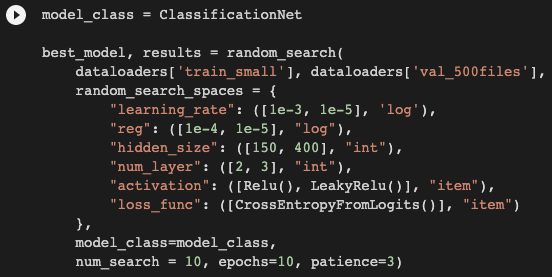


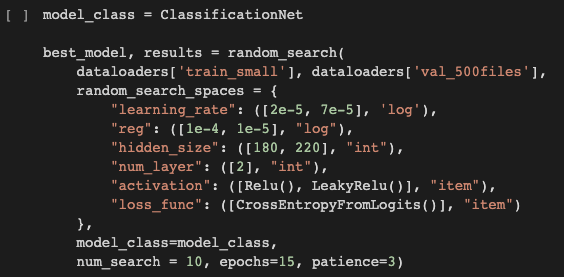
1. Let’s find the perfect model

依照上述訓練的方式，資料量由小到大進行訓練，並觀察參數及loss的變化以調整hyperparameters。

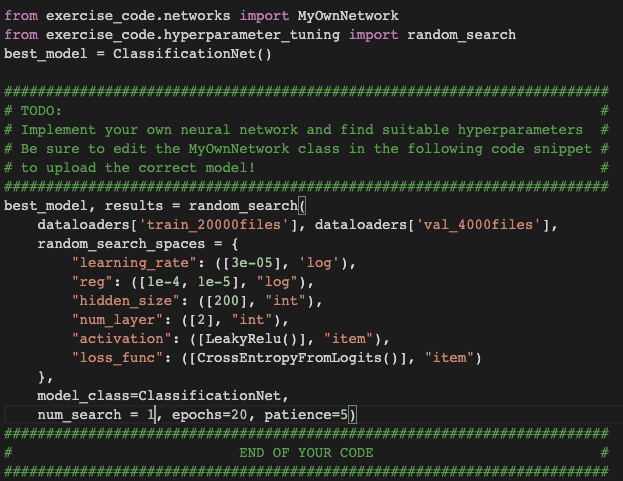
從1000筆開始訓練增加到10000筆，再到20000筆，在1000筆時發現learning rate在1e-3~1e-5時的訓練情形較好，太大的話會loss有可能會不降反升，而太小的話可能會下降的太慢。另外，神經網路的層數在2、3層的訓練結果較好，層數再多結果非但沒有比較好且耗時。而在Activation Function的部分，ReLU及LeakyReLU的表現較好。而由於資料量的不足加上early stopping的機制，因此增加epoch數也無法透過觀察出loss的變化調整參數。

資料數增加後，會發現learning rate應該在小一點，限縮在1e-4~1e-5間，另外，層數在2層時，且neuron數在200上下時普遍表現較好，因此也將epoch數量加大以觀察loss的變化。





(由於中間10000筆資料的訓練過程被20000筆的覆蓋因此未顯示)







由訓練結果可以發現已經overfitting，因此或許需要增加regularization的強度，亦或是利用Data Augmentation增加訓練資料，或是增加訓練及驗證的資料數量加以調整參數，才可改進準確率。