**Deep Learning Lab1-Backpropagation Report**

**智能所311581006 林子淩**

**1. Introduction**

本次lab主題為實作backward propagation。Backpropagation是在訓練deep neural network不可或缺的步驟，在backpropagation的過程中，我們可以通過梯度下降(gradient descent)來由後至前更新neural network中每一層layer裡面的參數(weight, bias)，使model output prediction和ground truth label的error term逐漸下降。本次lab禁止使用常見的deep learning framework (例如：Pytorch, Tensorflow)，只能使用基本的numpy和python standard libraries去實作出至少涵括兩層hidden layer的neural network，並建構出forward pass和backward propagation的計算。本次lab的學習目的為讓我們能夠深入瞭解最基本的神經網路架構及訓練參數更新的原理和計算過程。

**2. Experiment Setups**

**A. Sigmoid Functions**

Sigmoid function是一種在deep learning model中常用的activation function，其提供非線性的轉換方式(non-linearity)，讓模型可以有更多的轉換空間去fit data。在計算上，sigmoid是一個簡單的映射函數，可以將數字映射到[0,1]的範圍內，為平滑單調遞增函數，整個函數以(0, 0.5)呈現中心對稱。

Sigmoid和sigmoid微分的公式如下，圖形如Figure 1：

Sigmoid function:

Derivative of sigmoid function:

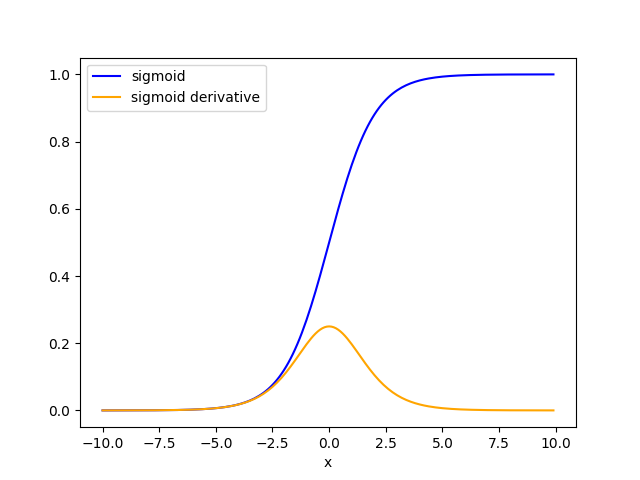


Figure 1 sigmoid和sigmoid derivative圖形

**B. Neural Network**

在本次lab中，我建構了一個neural network作為分類器(classifier)來將資料點(data points)分成兩個類別。整個neural network model包含一層input layer、兩層hidden layer以及一層output layer (總共四層)。每一層layer後面可以選擇接(sigmoid, ReLu)其中一種作為activation function或是選擇不用activation function (none)，除了output layer如果整體選擇none作為activation function，會自動改成sigmoid。Input layer每筆資料輸入維度為2，中間根據不同實驗選擇不同的hidden units數量(即hidden layer的node數量，default為50個)，output layer輸出維度為1。當data point屬於第一類時，output應該輸出0；屬於第二類則output應輸出1。在實驗中我選擇將threshold設成0.5 (考慮到使用default activation function為sigmoid，而sigmoid的映射範圍介於0和1之間)，因此output layer輸出的prediction value如果小於/等於0.5會被分作第一類，大於0.5則分作第二類。

**C. Backpropagation**

在執行backpropagation步驟，首先需要先計算Loss value來量化model prediction和ground truth label之間的差異，再利用gradient descent來循環更新model weight和bias (由最後一層layer開始向前更新)。本實驗選擇使用的loss function為Mean Square Error (MSE)，目標為最小化model prediction value和ground truth之間的差，其公式如下：

公式中，為training data筆數，和分別代表第筆資料的ground truth label和prediction value。

本次lab中使用的梯度下降法為Stochastic gradient descent (SGD)，SGD的精神為修改model參數來達到降低loss value 的目標(為loss的通稱，在本次實驗中為MSE)。通過計算loss 對參數的微分，可以得到梯度(gradient)，梯度的物理意義為參數沿著該方向變化會讓loss 變化最快、變化量最大。當參數沿著gradient的正向方向移動，loss 會上升；反之，當參數沿著gradient的負向方向移動，loss 會朝著最有效率的方向下降，即朝著local minima移動，達到minimize loss function的訓練目的。SGD參數更新法相當直觀，然而若學習率(learning rate，可以看作更新參數的幅度)太大，這種方法容易造成參數在local minima附近來回震盪，沒有辦法順利地降低loss 。實現SGD的方法為計算loss的梯度，根據微積分chain-rule可得到：

公式中，為layer 的weight，為layer 的output。

：根據forward pass， 。因此這一項偏微分的結果其實就是layer 的input。

：將每一層layer 的後面這項偏微分簡寫為，即backward propagation由後往前計算至layer 前更新的gradient。output layer的公式如下：

為loss derivative (loss value對model prediction y的微分)，則為sigmoid derivative，其輸入為output layer前一層layer的輸出。其他layer的公式如下：

可看作backward propagation前一個節點輸出的gradient，因為順序為由後向前更新，因此需要將此gradient 矩陣相乘上通往該層(即layer )的weight matrix ，再通過activation function derivative才可得到當前一層(layer )的。(注意此項為總梯度計算的後項偏微分結果，需要再和前項偏微分結果做dot product才能得到當前layer需要更新的gradient。另外必須考量矩陣形狀，因此在算時，必須做transpose；同理，在計算時，必須transpose。

**3. Results of your testing**

**A. Screenshot and Comparison Figure**

在default setting (hidden units=50, activation function=sigmoid, learning rate=1e-1)的設定之下，實驗的loss和accuracy結果如Table 1所示。可以看出隨著epoch升高，loss會逐漸下降，代表模型確實有逐漸學習到如何分類data，而accuracy則上升直到等於1 (完全分類正確)時停止訓練。比較Linear data與XOR data的訓練過程可以發現，XOR data需要的訓練epoch較少，比較容易訓練。

|  |  |
| --- | --- |
| Training loss and accuracy | |
| Linear data | **XOR data** |
|  |  |

Table 1 Training loss和accuracy截圖

Table 2展示了Linear和XOR dataset中每一個data point對應的model prediction結果。截圖是還沒有經過threshold分類的結果，因為本次實驗中分類threshold設定為0.5，因此model prediction小於等於0.5的會被分到第一類，大於0.5則分到第二類，以此得到最終的label。

|  |  |
| --- | --- |
| Model predictions | |
| Linear data | **XOR data** |
|  |  |

Table 2 Linear和XOR datasets中每一個data point對應的model predictions

Table 3為ground truth與predict result的分類結果視覺化。圖中的predict result為model訓練到accuracy=1的分類結果，因為本次實驗沒有區分training set和test set，預測資料與訓練資料一樣，所以可以訓練到跟ground truth完全一致。

|  |  |
| --- | --- |
| Comparison between Ground truth and Predict result | |
| Linear data | **XOR data** |
|  |  |

Table 3 比較Ground truth與Prediction result

**B. Show the Accuracy of your Prediction**

在Table 2中以每5000個epoch一次的頻率以文字顯示training accuracy，可以看出隨著loss降低，accuracy逐漸升高至1，模型成功訓練到能夠完全正確分類。Table 4進一步將accuracy紀錄視覺化，通過觀察training accuracy curve可以看出Linear data在訓練後期accuracy curve急遽上升(找到接近local minima的參數組合)。XOR data則是因為資料點較少所以accuracy curve比較崎嶇(每多分類對一個資料點accuracy就會上升一階)。同時也猜測可能是因為實驗選擇的learning rate 1e-1比較大，所以兩者訓練過程中可能一直在local minima附近來回震盪，直到剛好找到local minima後accuracy curve才上升至1。

|  |  |
| --- | --- |
| Accuracy Learning Curve | |
| Linear data | **XOR data** |
|  |  |

Table 4 Training accuracy curve

**C. Learning Curve**

Table 4和Table 5分別為Accuracy Learning Curve和Loss Learning Curve，通過比對兩者可以發現對應關係，當loss急遽下降代表模型找到接近local minima的參數組合，此時的accuracy對應也會急遽上升(分類正確性提高)。而loss在逐漸下降後突然上升可能是因為learning rate較大所以震盪到距離local minima比較遠的位置，此時對應的accuracy也會下降。但整體來說，大致上呈現隨著epoch升高，loss下降，accuracy上升的趨勢，代表模型隨著訓練次數上升逐漸學會如何正確分類。

|  |  |
| --- | --- |
| Loss Learning Curve | |
| Linear data | **XOR data** |
|  |  |

Table 5 Training loss curve

**4. Discussion**

第四部份探討實驗數據與其餘額外實驗的結果，以表格記錄不同參數實驗 (調整Learning Rate, Hidden Units, Activation Function) 之下，Linear data與XOR data訓練的收斂情況和Accuracy表現。本次實驗對每一個參數組合都進行了十次實驗，並回報十次實驗的平均結果。其中，Stop Epoch代表十次訓練平均在第幾個epoch第一次達到accuracy為1，即完全分類正確的情況。# of Converge Failure代表在十次實驗中，有幾次沒有在100000個epoch內達到accuracy為1，即沒有成功收斂(此時的Stop Epoch會被記錄成100000+)。Best Acc代表十次實驗中達到過最好的accuracy表現(單次實驗結果，非平均)。Avg of Best Acc則代表十次實驗Best Acc的平均。

**A. Try different learning rates**

Table 6為使用不同learning rates，固定hidden units=50, activation function=Sigmoid的實驗結果。從表中可以看出，當learning rate為1e-1時，Linear和XOR兩種資料的訓練都會收斂最快(Stop Epoch最小)。而當learning rate減半至5e-2時，兩個資料集仍然能夠在100000個epochs內收斂，但收斂速度明顯較慢，需要的訓練epoch數量上升。當learning rate再降低至1e-2和5e-3時，因為更新梯度的幅度變小，需要超過100000個訓練epochs才能達到收斂，學習到如何正確分類所有data points，只訓練100000個epochs不足以讓模型的accuracy上升至1(灰色標示部分)。整體數據符合當learning rate越小，模型會需要更多epoch以達到收斂的邏輯。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Param. | | | Linear/XOR | | | |
| Learning  Rate | **Hidden**  **Units** | **Activation**  **Function** | **Stop**  **Epoch** | **# of**  **Converge Failure** | **Best Acc** | **Avg of**  **Best Acc** |
| 1e-1 | 50 | Sigmoid | 44344/24014 | 0/0 | 1/1 | 1/1 |
| 5e-2 | 50 | Sigmoid | 88683/62027 | 0/0 | 1/1 | 1/1 |
| 1e-2 | 50 | Sigmoid | 100000+/100000+ | 10/10 | 0.58/0.86 | 0.58/0.65 |
| 5e-3 | 50 | Sigmoid | 100000+/100000+ | 10/10 | 0.58/0.52 | 0.58/0.52 |

Table 6 Experiment of different learning rates

**B. Try different numbers of hidden units**

Table 7為使用不同hidden units，固定learning rate=1e-1, activation function=Sigmoid的實驗結果。從表中可以看出，對於Linear 和XOR兩種資料來說，當hidden units越多(上限100為例)，模型參數越多，其fit data的能力越好，需要的training epoch也較少。整體而言，hidden units在50以上的模型皆可完美分類Linear和XOR的資料，收斂達到accuracy=1的表現。當hidden units在40以下，模型會不夠複雜，沒辦法保證完美分類XOR資料點，在十次實驗中會有1~2次無法在100000個training epochs內成功收斂(灰色標示部分)。另外，Stop Epoch雖整體隨hidden units上升而逐漸下降的趨勢，但仍有幾個實驗不符合此規律(例如：hidden units=50時，XOR data的Stop Epoch為24014，少於hidden units=60時XOR data的Stop Epoch 26428)，推測當hidden units上升時模型雖參數增加可以fit更複雜的資料點，但穩定性會稍微降低。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Param. | | | Linear/XOR | | | |
| Learning  Rate | **Hidden**  **Units** | **Activation**  **Function** | **Stop**  **Epoch** | **# of**  **Converge Failure** | **Best Acc** | **Avg of**  **Best Acc** |
| 1e-1 | 10 | Sigmoid | 57100/59377 | 0/2 | 1/1 | 1/0.99 |
| 1e-1 | 20 | Sigmoid | 53253/49556 | 0/2 | 1/1 | 1/0.99 |
| 1e-1 | 30 | Sigmoid | 48832/47748 | 0/2 | 1/1 | 1/0.99 |
| 1e-1 | 40 | Sigmoid | 44899/39934 | 0/1 | 1/1 | 1/0.99 |
| 1e-1 | 50 | Sigmoid | 44344/24014 | 0/0 | 1/1 | 1/1 |
| 1e-1 | 60 | Sigmoid | 39880/26428 | 0/0 | 1/1 | 1/1 |
| 1e-1 | 70 | Sigmoid | 37824/20959 | 0/0 | 1/1 | 1/1 |
| 1e-1 | 80 | Sigmoid | 35516/20475 | 0/0 | 1/1 | 1/1 |
| 1e-1 | 90 | Sigmoid | 33543/15276 | 0/0 | 1/1 | 1/1 |
| 1e-1 | 100 | Sigmoid | 32807/16204 | 0/0 | 1/1 | 1/1 |

Table 7 Experiment of different numbers of hidden units

**C. Try without activation functions**

Table 8為嘗試不加入activation function的實驗結果(Activation Function=None)，實驗固定learning rate=1e-1、hidden units=50，與使用Sigmoid作為activation function做比較。由表中可以看出，當不使用activation function時，在針對XOR data的十次實驗皆無法在100000個epochs內收斂(灰色標記部分)，最高accuracy只達到0.52。因為分類類別只有兩類，avg accuracy=0.58只比隨機猜測label的accuracy期望值略高。分類效果差的原因在於如果不加入activation function，模型會缺乏非線性關係(non-linearity)，單靠線性組合很難去fit XOR data，因為XOR屬於non-linear data distribution。然而此次的實驗設定為模型output layer必須接上一個activation function (若設定為None則output layer自動接Sigmoid)，因此即使設定activation function為None，在最後輸出時仍會經過Sigmoid function映射到[0,1]範圍，所以accuracy至少有0.58的水準。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Param. | | | Linear/XOR | | | |
| Learning  Rate | **Hidden**  **Units** | **Activation**  **Function** | **Stop**  **Epoch** | **# of**  **Converge Failure** | **Best Acc** | **Avg of**  **Best Acc** |
| 1e-1 | 50 | Sigmoid | 44344/24014 | 0/0 | 1/1 | 1/1 |
| 1e-1 | 50 | None | 2991/100000+ | 0/10 | 1/0.76 | 1/0.58 |

Table 8 Experiment of activation functions (Sigmoid and None)

**5. Extra**

**B. Implement different activation functions**

在加分項目的部分，本實驗加入了ReLU作為Sigmoid以外的activation function選擇。ReLU同樣作為一種deep learning model常用的activation function，可以提供模型非線性組合能力(non-linearity)， ReLU和ReLU微分的公式如下，圖形如Figure 2：

ReLU function:

Derivative of ReLU function:

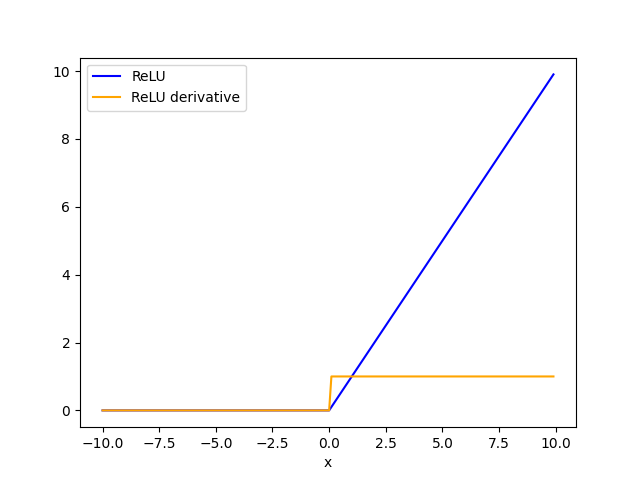


Figure 2 ReLU和ReLU derivative圖形

Table 9為比較Sigmoid, ReLU, None三種activation function choices的實驗數據，實驗參數固定learning rate=1e-1、hidden units=50。從表中可以觀察到使用ReLU作為activation function的話，在Linear和XOR data的十次實驗中皆無法收斂。這可能是因為本次實驗output只有一個維度，分類以0或1作為target label，而使用ReLU無法將output value映射到[0,1]範圍之內，因此模型在output value離0,1很遠的情況下不容易收斂。從數據中可以很明顯看出使用Sigmoid作為activation function的訓練效果較ReLU好(所需epoch較少，accuracy較高)，同樣也是因為實驗設定label介於[0,1]之間，使用Sigmoid作為mapping function更合適。None雖然只在最後output layer後接上Sigmoid，在其餘layer皆沒有加入non-linearity導致在針對non-linear distribution XOR data的訓練效果差於Sigmoid，但其在Linear和XOR data的訓練效果仍比ReLU好，這也是因為在最後輸出時有將output value控制在[0,1]範圍內的緣故。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Param. | | | Linear/XOR | | | |
| Learning  Rate | **Hidden**  **Units** | **Activation**  **Function** | **Stop**  **Epoch** | **# of**  **Converge Failure** | **Best Acc** | **Avg of**  **Best Acc** |
| 1e-1 | 50 | Sigmoid | 44344/24014 | 0/0 | 1/1 | 1/1 |
| 1e-1 | 50 | ReLU | 100000+/100000+ | 10/10 | 0.42/0.52 | 0.42/0.52 |
| 1e-1 | 50 | None | 2991/100000+ | 0/10 | 1/0.76 | 1/0.58 |

Table 9 Experiment of activation functions (Sigmoid, ReLU and None)