清華大學資工系大二 學號: A053095 林柏淵

 Introduction: 這次做的是要作 MNIST data set 的 classify 而目前可以 用的方法有 CNN 和 RNN 此次 lab 將以 recurrent neural networks，RNN 來做 classify。

 Experiment setup: Hyperparameter:

 sequence\_length = 28

 input\_size = 28

 hidden\_size = 128

 num\_layers = 2

 num\_classes = 10

 batch\_size = 50

 num\_epochs = 5

 learning\_rate = 0.01 detail of my model:

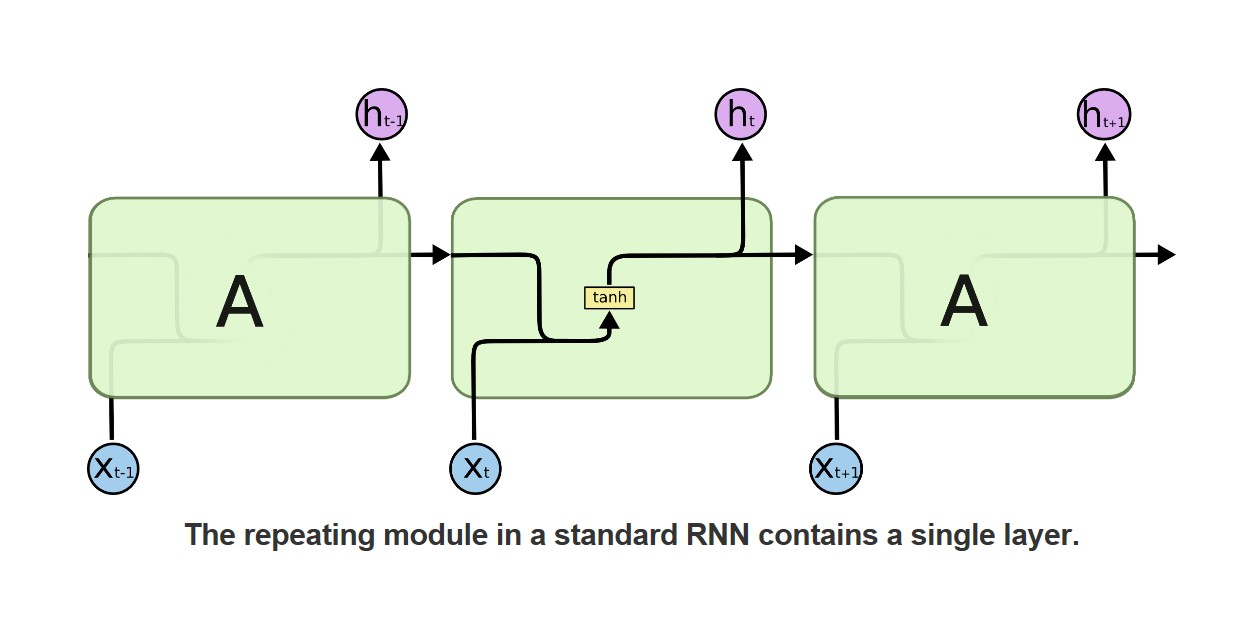
在 load data 方面跟上一次的 Resnet 類似使用 torchvision.datasets.MNIST 來將

data 轉 tensor load 進來再經過 batch\_size 的處理。在 RNN model 定義方面先將 model 初始值，將總共幾層、hidden size 以及 input\_size 做初始，然後使用 pytorch library 內建的 LSTM，最後在接一個 fully connect，初始化就告一段落。 接下來在 forward step 方面先初始化 parameters 再進去 LSTM forward propagate 再來做 linear computation，再來 loss function 方面採用 CrossEntropyLoss 來判斷 而在優化部分跟 lab1(lab1 採 SGD+momentum)不同我們採用 adam 來做優化， 在來就是跑 training 迴圈了，將圖片一組一組 load 進來 reshape 再來跑 forward 跑完後算 loss 再來將前次的 gradient 歸零重新算一次 backward，並做 optimize 直到 5 個 epoch 結束，train 結束後做 testing 並記錄 Acc，再把參數即 model 存

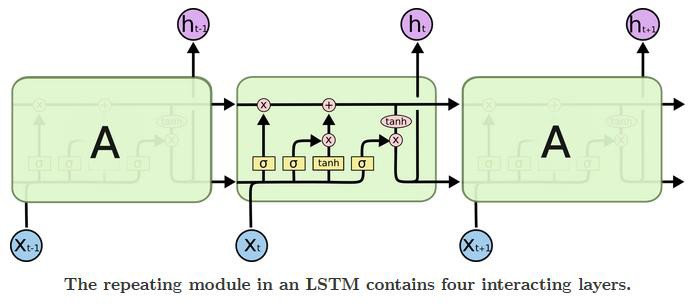
起來。

在此稍微說明一下 LSTM 的用處:

這是一般的 RNN



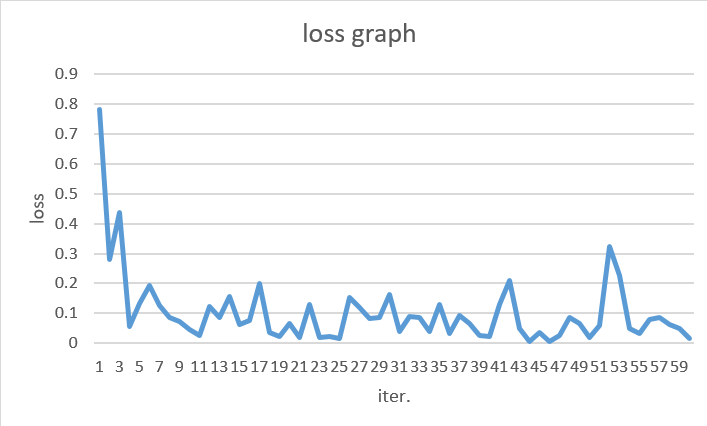
這是 LSTM



當前時刻的預測值要依賴之間時刻的信息，當兩個時間間隔較短時，RNN 可以 比較容易地利用先前時刻信息。但當這兩個時間間隔不斷變長時，簡單的循環 神經網絡有可能會喪失學習到距離很遠的時刻的信息的能力。在一些複雜語言 場景中，有用信息的間隔有大有小、長短不一，簡單的 RNN 網絡的性能會收到 限制。而 LSTM 網絡的設計就是為了解決該問題。

Final Test error:

100-97.3% = 2.7%



 Discussion: 這次 Loading 比前面一次比起來輕鬆很多，差最大的就在

training time，1~5 分鐘就 train 完真的是省了很多時間，所以額外做了很多 無聊的小實驗，像是如果用 CPU 驅動的話效率會差多少，結果計算時間差 了 10 倍真的差很多，而且從 7 月初到現在也發現了我研究方法需要做一些 調整，之後不會再把 code 了解得徹底，只要知道他的 function 是幹麼就足 夠，應該把時間花在 NN 本身的架構及數學的理解，往常我習慣會 trace code，因為不弄懂裡面再幹麼我會渾身不自在，但上了這堂課後發現【根本 沒時間】，難處有兩點

1. pytthon 包太多了:接觸這堂課之前是在寫 c/c++再怎麼樣也不會像 python 包那麼誇張，所以 要 dig in 會有困難。

2. pytorch 是之前碰都沒碰過的 library:其實這跟 1.類似要查的東西太多了，所

以常常花時間在了解 function 的功能上