Lab3 補充說明

在我們做 NLP 相關的 task 時，像是這一次 Lab3 的例子，因為 ground truth 的句子可能不一樣長，例如我們有三個不一樣長的句子:

1. "天"、"氣"、"真"、"好"
2. "下"、"雨"、"天"
3. "彩"、"虹"

那我們在讀檔做訓練時我們會進行以下處理:

1. pad\_sequence

(官網連結 : https://pytorch.org/docs/stable/generated/torch.nn.utils.rnn.pad\_sequence.html#torch.nn.utils.rnn.pad\_sequence)

Dataloader 在 return 一個 batch 時，這個 batch 會是一個高維的 tensor，其中每個長度必須是依樣的長度，那在我們舉的例子中，這三個句子都不一樣長，所以我們會以最長的句子長度為準，每個長度不足這個的句子後面補 "pad" 直到長度一致，補完的結果會是:

1. "天"、"氣"、"真"、"好"
2. "下"、"雨"、"天"、"pad"
3. "彩"、"虹"、"pad"、"pad"

再舉個例子，實例上若我們有個 tensor 如下:

from torch.nn.utils.rnn import pad\_sequence

train\_x = [torch.tensor([1, 1, 1, 1, 1, 1, 1]),

torch.tensor([2, 2, 2, 2, 2, 2]),

torch.tensor([3, 3, 3, 3, 3]),

torch.tensor([4, 4, 4, 4]),

torch.tensor([5, 5, 5]),

torch.tensor([6, 6]),

torch.tensor([7])]

那麼經過 pad\_sequence

我們可以得到 x 為

tensor([[1, 1, 1, 1, 1, 1, 1],

[2, 2, 2, 2, 2, 2, 0],

[3, 3, 3, 3, 3, 0, 0],

[4, 4, 4, 4, 0, 0, 0],

[5, 5, 5, 0, 0, 0, 0],

[6, 6, 0, 0, 0, 0, 0],

[7, 0, 0, 0, 0, 0,,0]])

1. pack\_padded\_sequence

那為何在有了 pad\_sequence 之後，還需要一個 pack\_padded 呢?這是因為在 RNN 訓練中，我們常常會設 batch size 大於 1 。

* 1. 我們先考慮如果 batch size 為 1 是如何訓練：我們每次 model 吃進一個 time step 的數據+該數據對應的 hidden state，然後一個 output，再繼續吃進去第二個 time step 的數據 + hidden state，再 output，以此類推。
  2. 但如果換成 mini-batch 的訓練模式則是： Model 每次吃進去一組同樣 time step 的數據，也就是mini-batch 中所有 sequence 中相同時間下標的數據，加上它們對應的 hidden state，獲得一個 mini-batch 的 output，然後再移到下一個 time step，再讀入 mini-batch 中所有該 time step 的數據，再output

那這樣的話，我們在讀入數據時就會像是這樣去讀進 model 去計算:

1. 第一次讀下方黃色:
   1. "天"、"氣"、"真"、"好"
   2. "下"、"雨"、"天"、"pad"
   3. "彩"、"虹"、"pad"、"pad"
2. 第二次去讀下方綠色:
   1. "天"、"氣"、"真"、"好"
   2. "下"、"雨"、"天"、"pad"
   3. "彩"、"虹"、"pad"、"pad"

以此類推。那 pack\_padded\_sequence 可以回傳資料方便讓我們使用。那我們先 來看這個函數(pack\_padded\_sequence 的 pytorch 官網連結為:<https://pytorch.org/docs/stable/generated/torch.nn.utils.rnn.pack_padded_sequence.html>)

pack\_padded\_sequence 有三個參數：input, lengths, batch\_first 。input 是上一步加過 padding 的數據，lengths 是各個 sequence 的實際長度，batch\_first是數據各個 dimension 按照 [batch\_size, sequence\_length, data\_dim]順序排列。所以在我們這三句話的例子裏我們需要設 length = [4,3,2]，那麼我們把這三句話的例子經過 pack\_padded\_sequence 之後可以得到以下結果，是一個 PackedSequence:

PackedSequence(

data=tensor(["天", "下", "彩", "氣","雨", "虹", "真", "天", "好"]),

batch\_sizes=tensor([3, 3, 2, 1]))

其中， data 即為原來的數據按照 time step 重新排列，而 padding 的部分則直接跳過，不過在真實 case 如 lab3 中，字會先轉成 word vector 等，所以 “data” 裡不會是原本的文字，我們這邊為了解釋方便所以先用原本的字表示。 batch size 則表示每一次該讀入的量。所以我們有如上的 PackedSequence 我們該如何撈出 data 呢? 可以參考以下步驟:

Step.1: batch\_szie 中第一個值為 3，則第一次去撈出 data 中第 0 ~ 2 (共 3 個) 項 : "天", "下", "彩"

Step.2: batch\_szie 中第二個值為 3，則第二次去撈出 data 中第 3 ~ 5 (共 3 個) 項 : "氣","雨", "虹"

Step.3: batch\_szie 中第三個值為 2，則第三次去撈出 data 中第 6 ~ 8 (共 2 個) 項 : "真", "天"

Step.4: batch\_szie 中第四個值為 1，則第四次去撈出 data 中第 9 (共 1 個) 項 : "好"

以上為 Lab3 中關於 pack\_padded\_sequence 的解釋，若有問題歡迎詢問。