自然語言處理 HW2

成功大學114資訊系 F74102022唐文蔚

|  |  |
| --- | --- |
| Environment | Ubuntu 20.04.6 LTS |
| CPU | Intel(R) Xeon(R) Gold 6242R CPU @ 3.10GHz |
| GPU | NVIDIA GeForce RTX 3090 |
| Python Version | Python 3.9.20 |

Report：

Q1 : What impact does using different learning rates have on model training?

Performance Comparison：

1. Optimizer Adam：

|  |  |  |  |
| --- | --- | --- | --- |
| Learning Rate | 0.005(附圖1) | 0.001(附圖2) | 0.0005(附圖3) |
| Loss | 0.64 | 0.174 | 0.124 |
| Accuracy | 28.4% | 83.5% | 86.0% |
| 最佳Acc epoch | epoch 1 | epoch 10 | epoch 10 |

1. Optimizer RMSCrop：

|  |  |  |  |
| --- | --- | --- | --- |
| Learning Rate | 0.005 | 0.001(附圖5) | 0.0005(附圖6) |
| Loss | 0.53 | 0.106 | 0.0414 |
| Accuracy | 38.3% | 87.4% | 90.8% |
| 最佳Acc epoch | epoch 6 | epoch 10 | epoch 9 |

A1：從上方結果可以發現，不同的 learning rate 對於模型訓練的 loss 和 accuracy 有顯著影響。當使用 Adam optimizer 時，較高的 learning rate（0.005）雖然初始收斂速度快，但模型的 accuracy 較低（28.4%），而較低的 learning rate（0.001 和 0.0005）在第 10 個 epoch 達到較高的 accuracy（分別為 83.5% 和 86.0%），且 loss 下降更多。對於 RMSprop optimizer，較低的 learning rate也可以有更低的 loss（0.0414）和更高的 accuracy（90.8%）。總而言之，適當的 learning rate 可以有效提高模型的 accuracy 並降低 loss。

Q2：If you use RNN or GRU instead of LSTM, what will happen to the quality of your answer generation? Why?

(實驗使用lr = 0.0005, RMSCrop Optimizer)

LSTM：



GRU：



RNN：



A2：在 CharRNN 模型中使用 RNN 或 GRU 來取代 LSTM，生成結果會下降，主要原因可能如下：

長期依賴：LSTM 使用記憶單元和門控機制來保留長期資訊，減少梯度消失問題，因此在處理長序列時比 RNN 表現更佳。RNN 缺乏這些機制，導致無法有效學習長距依賴。GRU 則是結構較簡單，計算效率較高的折衷方案，能部分保留長期依賴資訊，但在需要複雜長期記憶的任務上可能略遜於 LSTM。

因此，改用 RNN會大幅影響生成結果，而 GRU 則會有些微影響。

Q3：If we construct an evaluation set using three-digit numbers while the training set is constructed from two-digit numbers, what will happen to the quality of your answer generation?

A3：模型在訓練時採用2位數但驗證集使用3位數會導致模型幾乎完全無法正確回答驗證集問題，這會導致模型在驗證集上的loss值正常，但正確率極低，顯示模型在這種情況下缺乏泛化能力，無法適應未見過的數據範疇。

Q4：If some numbers never appear in your training data, what will happen to your answer generation?

A4：

隨機抽取超出訓練資料數字(50 - 100)當作input時的比較

範圍外加法運算正確率: 1.00%

範圍外減法運算正確率: 5.00%

範圍外乘法運算正確率: 0.00%

* 完全無法運算out of bound的input

Q5：Why do we need gradient clipping during training?

A5：gradient clipping 是為了防止梯度爆炸。由於 LSTM 處理長序列數據，gradient容易在back propagation中不斷增大，導致模型不穩定或發散。用 gradient clipping可以限制每次gradient的最大值，確保訓練過程穩定並加速收斂。

Q6(Extra)：更換Optimizer的差別

Optimizer Adam：

|  |  |  |  |
| --- | --- | --- | --- |
| Learning Rate | 0.005(附圖1) | 0.001(附圖2) | 0.0005(附圖3) |
| Loss | 0.64 | 0.174 | 0.124 |
| Accuracy | 28.4% | 83.5% | 86.0% |
| 最佳Acc epoch | epoch 1 | epoch 10 | epoch 10 |

Optimizer RMSCrop：

|  |  |  |  |
| --- | --- | --- | --- |
| Learning Rate | 0.005 | 0.001(附圖5) | 0.0005(附圖6) |
| Loss | 0.53 | 0.106 | 0.0414 |
| Accuracy | 38.3% | 87.4% | 90.8% |
| 最佳Acc epoch | epoch 6 | epoch 10 | epoch 9 |

A6：RMSCrop Optimizer 在各項指標上都優於 Adam，特別是在高學習率和較早期的訓練階段就能達到較高的準確率與更低的 Loss 值。因此我認為在這個實驗中，RMSCrop 比 Adam更適合用來做此模型的Optimizer。

Q7 (Extra)：Generate 正確率比較 (使用GPT將比較設想撰寫成程式碼測試)

A7：

1. 比較三種運算的正確率 ( 無括號，單純a @ b)

|  |  |  |
| --- | --- | --- |
| 加法運算正確率 | 減法運算正確率 | 乘法運算正確率 |
| 95.52% | 95.80% | 79.56% |

* 乘法運算的正確率明顯較低，顯示模型較無法學習乘法的運算

1. 比較減法運算時答案為正和為負的正確率比較(無括號，單純a - b)

|  |  |
| --- | --- |
| 正數結果正確率（a > b） | 負數結果正確率（a < b） |
| 95.84% | 95.76% |

* 無太大差異，模型在a – b 和 b – a時都可以算出結果

Q8 (Extra)：探討作業為何不包含除法的運算：

A8：

1. 整數運算：除法會涉及小數結果，若是加入除法運算的話複雜度會大幅上升
2. 穩定性：除法在分母小時變化非常迅速，且會有極端值或無法定義(除0)

Q9 (Extra)：檢查模型收斂效果

一張含有 行, 圖表, 文字, 繪圖 的圖片

自動產生的描述A9：從Loss和ACC曲線比較可以發現，模型在第9個epoch時表現最好，在最後一個epoch稍微有一點Overfitting問題，但Training和Validation在loss和ACC的表現都大致相同，穩步收斂。

<附圖1> Adam：lr = 0.005

一張含有 文字, 螢幕擷取畫面, 平行, 設計 的圖片

自動產生的描述

<附圖2> Adam：lr = 0.001( Default setting )

一張含有 文字, 螢幕擷取畫面, 設計 的圖片

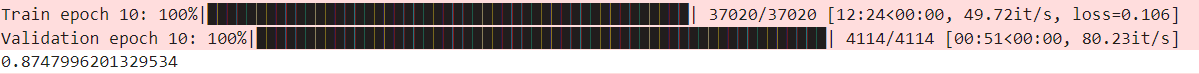
自動產生的描述

<附圖3> Adam：lr = 0.0005

一張含有 文字, 螢幕擷取畫面, 平行, 設計 的圖片

自動產生的描述

<附圖5> RSCROP：lr = 0.001



<附圖6> RSCROP：lr = 0.0005 (Best Performance)

