

Embedded system of Transformer-based TTS implementation

Sian-Yi Chen

Advisors : Tay-Jyi Lin and Chingwei Yeh

Outline

Action item

- 完成 Transformer-based TTS 的嵌入式系統 (尚未完成)

Status report

- 先前進度
 - 熟悉大部分的encoder架構，除了embedding層中的ScaledPositionalEncoding
 - 解決「因為pytorch的高度模組化，因此隱藏層的資料無法直接查看」的問題
- 本周進度
 - Encoder層的種類(Embedding、ScaledPositionalEncoding、Linear、ReLU、LayerNorm)，已使用C完成大部分底層功能
 - 完成：Embedding、Linear、ReLU、LayerNorm
 - 未完成：ScaledPositionalEncoding、將實際資料(維度、bias和weight)擺上去驗證答案
 - 目前實現ScaledPositionalEncoding過程中正在解決的問題是「陣列維度的變換」，其餘遇到的問題接刃而解，但因小問題繁多而降低了實作速度
 - ScaledPositionalEncoding層目的是因為transformer中的attention層同步執行序列資料沒有位置資訊，因此需要額外添加

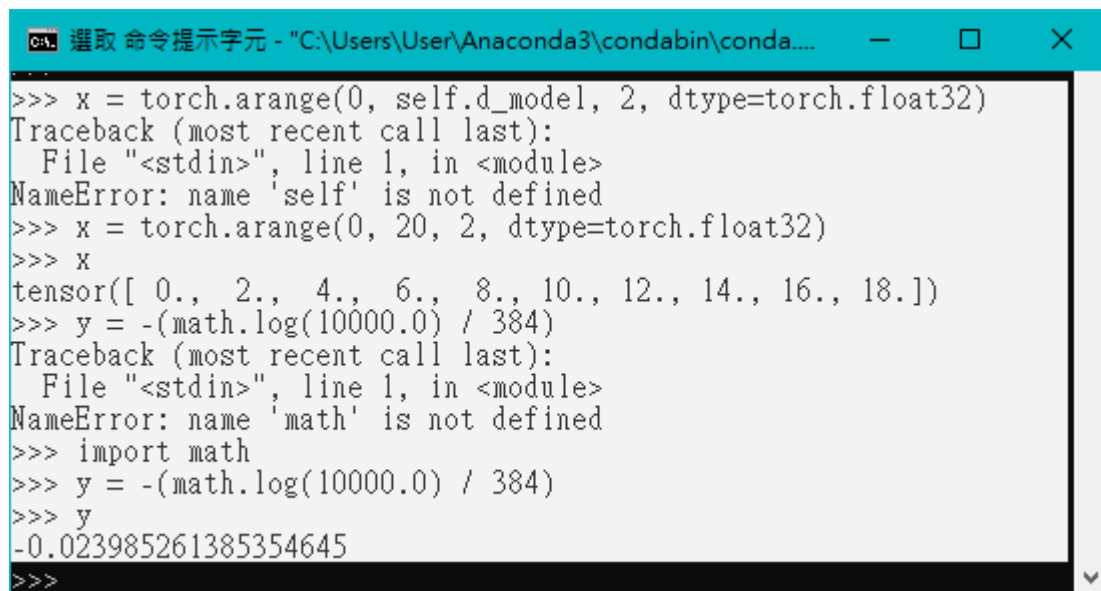
附錄

請教學長問題時，發現更有效率的做事方法

- 直接在**CMD**快速查看執行結果，而非以往習慣開檔案，寫完再全部執行
- 底層程式常出現一大串函式呼叫，如下圖所示，

```
torch.exp(  
    torch.arange(0, self.d_model, 2, dtype=torch.float32)  
    * -(math.log(10000.0) / self.d_model)  
)
```

將所有函數都獨立操作，一邊看函數document，一邊執行看結果



```
C:\Users\User\Anaconda3\condabin\conda...  
>>> x = torch.arange(0, self.d_model, 2, dtype=torch.float32)  
Traceback (most recent call last):  
  File "<stdin>", line 1, in <module>  
NameError: name 'self' is not defined  
>>> x = torch.arange(0, 20, 2, dtype=torch.float32)  
>>> x  
tensor([ 0.,  2.,  4.,  6.,  8., 10., 12., 14., 16., 18.])  
>>> y = -(math.log(10000.0) / 384)  
Traceback (most recent call last):  
  File "<stdin>", line 1, in <module>  
NameError: name 'math' is not defined  
>>> import math  
>>> y = -(math.log(10000.0) / 384)  
>>> y  
-0.023985261385354645  
>>>
```

- 以上兩點可以大幅降低以往在看程式碼的思考時間