Study of transformer-based TTS and its embedded implementation

Sian-Yi Chen

Advisors : Tay-Jyi Lin and Chingwei Yeh

Outline

- **Action item**
  - 確定論文的架構 (三點貢獻、實驗方法、數據呈現方法)
- **論文題目：Study of transformer-based TTS and its embedded implementation**
  - **三點貢獻**
    1. 使用Transformer架構，並將使用的特徵從MFCC轉換成LPS，降低運算複雜度
    2. 調整Transformer的超參數，降低運算複雜度
    3. embedded implementation
  - **實驗方法**
    1. 先完成第一版Transformer的嵌入式系統
       - 找到一版可執行的Transformer C與python code，並將python版本的參數給C code在嵌入式系統上執行，並呈現相同結果
    2. x-vector
       - 比照Transformer同樣的方式完成x-vector
    3. 將Transformer與x-vector接起來，完成與VCC2020 baseline相同的架構
    4. 有了與baseline架構相同且可執行的嵌入式系統，再更改MFCC至LPS
    5. 最後調整Transformer的超參數，降低複雜度
  - **數據呈現方法**
    1. 若完成實驗，將會有四個版本的結果(MFCC, python、C)、(LPS, python、C)，使用MOS分數表示，並分析比較
    2. 實驗結果應與baseline相似，MOS分數應落於3~4之間，而相似度接近90%

論文大綱

Abstract

1. 序論
   - 背景
   - 研究動機
   - 三點研究貢獻
   - 論文章節架構
2. Transformer的文字語音轉換
   - 系統架構
   - 文字語音轉換流程
   - Transformer的文字語音轉換基於C code的實現
3. 運算複雜度分析
   - MFCC轉換成LPS的效益分析
   - 超參數調整的效益分析
4. Experimental results
   - 呈現四個版本的結果(MFCC, python、C)、(LPS, python、C)
   - 使用圖表呈現訓練結果(MOS分數、相似度)
5. Conclusion
   - 將問題、方法、結果用別人看完的角度重新描述一次
   - 關於這個研究，未來可以發展的方向

附錄

Status report

- **參數量評估與FPGA板選擇**
  - 原先在VCC2020 baseline專案中，對於添加計算模型函數一直失敗，但目前已成功計算出x-vector+transformer的參數量
    - 問題：在專案中建模的檔案，與執行的檔案沒有互相呼叫，執行的檔案是呼叫線上資源進行建模
    - 方法：將線上資源複製到與執行檔同一資料夾，再進行模型參數函數的添加
  - x-vector+transformer的參數量程式執行結果為 26,610,372
    - 問題：與原論文[1]顯示的數據 65,000,000 有一段差距
    - 推測：原論文[1]輸入為512維，而專案是384維，另外程式中似乎沒有計算到input embedding以及positional embedding，因此計算6層encoder(Multi-head ateention + Feed Forward)+6層decoder(Multi-head ateention*2 + Feed Forward)=27,879,168 與程式執行結果近似
  - 目前正在嘗試執行找到的transformer C++ github專案
    - **(已解決)**執行至編譯transformer階段，出現c++20錯誤，應為gcc版本不對，並從gcc官網下載正確版本
    - 目前判斷為沒有Clang/LLVM套件，因此錯誤，仍在解決

- **後續規劃**
  - **完成第一版Transformer的嵌入式系統**