國立臺灣大學生物資源暨農學院生物機電工程學系

人工智慧實作期中報告

Department of Biomechatronics Engineering
College of Bioresources and Agriculture
National Taiwan University

Artificial Intelligence Implementation Midterm Report

WhisperAI 介紹及實作

Introduction and Implementation of WhisperAI

陳品勳

Pin-Hsun Chen

授課教授：李明達 兼任副教授

Teacher: Ming-Da Li, Adjunct Associate Professor.

中華民國 114 年 3 月

March 2025

一、 Whisper 的技術原理

Whisper 是由 OpenAI 所開發的一款開源語音辨識模型，其設計基礎為 Transformer 架構中的編碼器－解碼器（Encoder-Decoder）模型。該模型的主要任務是將語音訊號轉換為對應的文字，屬於典型的序列到序列（Sequence-to-Sequence）模型。當輸入語音訊號時，Whisper 首先會將整段語音切分為最多 30 秒的片段，並將其轉換成對數梅爾頻譜圖（log-Mel spectrogram）。這種特徵表示保留了語音的頻率與時間資訊，並透過模仿人耳感知機制的 Mel-scale，以及使用對數尺度進行穩定化處理，讓模型能夠更有效率地學習語音內容中的語意與語調變化。

進入模型核心後，Transformer 編碼器會接收這些頻譜圖作為輸入，並透過多層的自注意力機制與前饋神經網路，將其轉換為一系列的隱藏向量，這些向量即為語音訊號的抽象語意表示。接著，這些表示會被送入 Transformer 解碼器。解碼器在每一步都會根據已生成的文字內容與來自編碼器的上下文資訊，逐步預測出下一個語詞（token），最終生成完整的文字序列。這種編碼－解碼架構類似於機器翻譯系統能夠有效處理長距離依賴與語境資訊，提升語音轉文字的準確率。

值得一提的是，Whisper 不僅僅執行單一任務，它是一個多任務模型，能同時執行語言識別、語音轉錄、語音翻譯以及時間戳記預測。當模型接收到語音輸入時，能自動判別該語音所使用的語言，並根據輸入的任務提示決定是否進行直接轉錄或翻譯為英文。此外，模型還能為每段轉錄文字標註對應的時間區段，實現高精準度的字幕對齊與後製需求。

二、 實作成果

2.1 Youtube 影片語音辨識

實作動機

近期知名 Youtube 頻道「眾量級」爆出收益爭議，引發大量關注與討論。其中成員之一家寧所發表的回應影片因為口齒不清而被網友批評如圖 1-1，基於此，本報告選擇以該影片為素材，探討 Whisper 模型在辨識咬字不清方面的表現，並用網友提供的資料如圖 1-2 對照結果。

圖 1-1、網友批評之貼文

圖 1-2、網友提供之發音錯誤表

實作過程

本報告採用開源的 Whisper 語音辨識模型進行實作，相較於透過 API 的方式，本地部署具備可離線運作之優勢，無須仰賴網路連線即可執行語音轉文字任務，亦無須支付任何費用。然而，Whisper 模型於本機運算時對圖形處理單元(GPU) 性能有較高之需求，若在硬體資源不足的情況下，將可能導致執行速度緩慢甚至失敗。為克服此限制，本報告選擇以 Google Colab 為執行環境，借助其雲端提供之 GPU 資源，順利完成模型推論流程。

此外，為了應用 Whisper 於實際影音資料，本報告整合了與 YouTube 平台相關之輔助套件，實現從線上影片中擷取音訊並進行自動轉錄之功能。完整的實作步驟與流程詳見圖 2-1 至圖 2-7，展示了從資料獲取、語音處理到文字輸出之各階段作業。

。

圖 2-1 、實作一步驟一

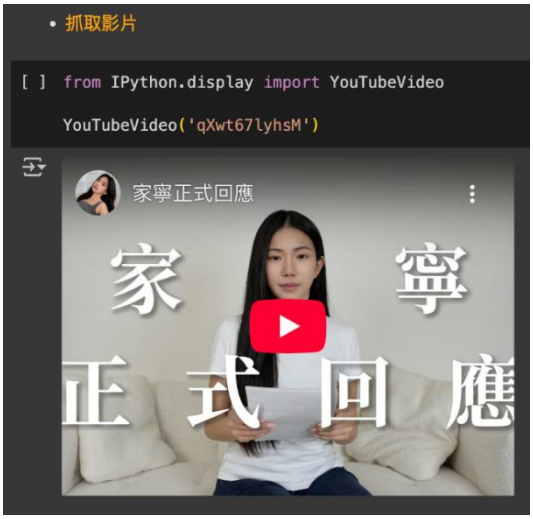

圖 2-2、實作一步驟二

圖 2-3、實作一步驟三

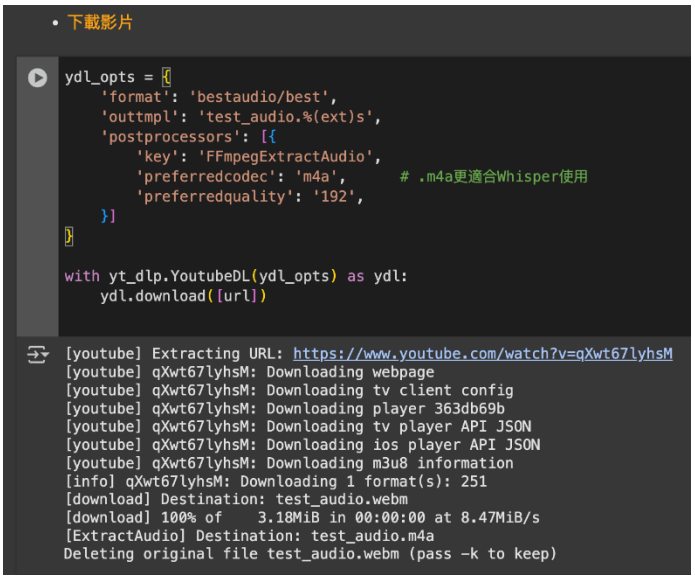

圖 2-4、實作一步驟四

圖 2-5、實作一步驟五

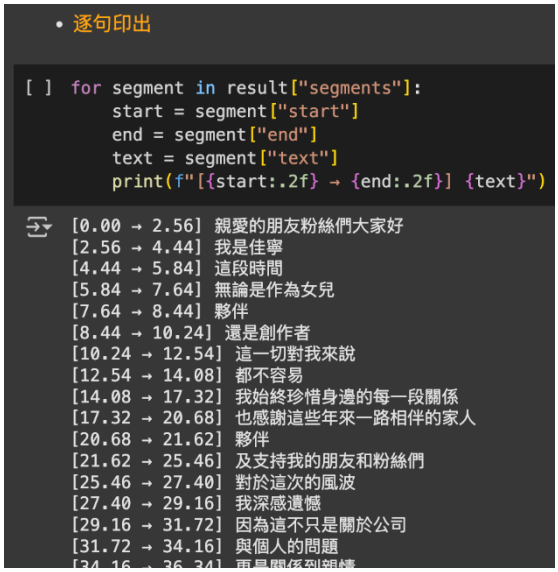

圖 2-6、實作一步驟六

• 存檔：存成逐句分段的文字

```
[ ] with open("segments.txt", "w", encoding="utf-8") as f:
        for seg in result["segments"]:
            f.write(f"[{seg['start']:.2f} → {seg['end']:.2f}] {seg['text']}\n")
```

• 存檔：存成JSON（含完整結構）

```
[ ] import json

    with open("result.json", "w", encoding="utf-8") as f:
        json.dump(result, f, ensure_ascii=False, indent=2)
```

圖 2-7、實作一步驟七

結果討論

本報告共設置兩種輸出格式，藉由存取 Whisper 模型所產出的 JSON 檔案，可觀察語音辨識結果的詳細結構。其中，"text" 欄位提供完整的辨識文字結果，為未經分段的純文字輸出；而 "segments" 則為分段後的語句資訊，包含每一段落的文字內容及其對應的時間軸等進階參數如圖 3。

圖 3、辨識結果 JSON 檔案內容

每個 segment 為一個語句片段，包含以下主要欄位說明：

- "id"：段落編號，從 0 開始遞增。
- "seek"：模型從音訊中此時間點開始分析。
- " start" / "end"：該段語音的起始與結束時間（以秒為單位）。
- "text"：此段落辨識出的文字內容。
- "tokens"：此段語音對應之語音 token 序列，為模型內部運算使用。
- "temperature"：語音解碼時所使用之溫度參數，影響生成的隨機程度。
- "avg_logprob"：模型對該段語音之平均對數機率，數值越接近 0 表示模型辨識信心越高。
- "compression_ratio"：文字壓縮比，用於偵測重複字詞情況，數值過高可能代表語音品質不佳或辨識異常。
- "no_speech_prob"：模型判定該段為「無語音」的機率，越接近 1 表示可能為靜音片段。

透過此分段資訊，並將辨識結果與實際逐字稿進行比對後，進一步整理出辨識錯誤的情形，並標註其對應的平均對數機率 (avg_logprob) 數值如表 1。

表 1、辨識結果與文稿對比

編號	錯誤時間	語音辨識結果	實際文稿	avg_logprob
1	1:22	我能為獲得完整的數據	我未能獲得完整的數據	-0.20
2	2:11	有更清新的規劃與管理	有更清晰的規劃與管理	-0.21
3	2:34	我想誠實的請求大家	我想誠摯地請求大家	-0.07

結果可見，Whisper 模型在口齒不清的語音辨識中仍然展現了良好的準確度。從表 1 的比對可觀察到，雖然部分字詞仍出現偏差，但其 avg_logprob 數值能有效反映辨識信心，提供後續判斷與修正的依據。搭配 JSON 檔的段落資訊，不僅能追蹤錯誤來源，也能作為字幕製作與語音分析的參考。

2.2 字幕 .srt 檔生成工具

實作動機

有鑑於課堂中老師展示之 AI 線上工具能夠透過上傳影片自動產生字幕檔，使影音資料更具可讀性與可檢索性，本報告也想嘗試實作一套類似的系統。期望透過 Whisper 模型將影片中的語音內容自動轉換為文字，並進一步整理為符合字幕格式 (.srt) 的檔案。

實作過程

本實作同樣採用 Google Colab 作為執行環境，藉由其提供的雲端運算資源與檔案上傳、下載功能，讓使用者僅需透過網頁介面上傳影音檔案，即可完成語音辨識與字幕檔 (.srt)生成之流程，無需在本地端安裝額外套件或配置執行環境。本報告亦將產出的字幕檔實際應用於 YouTube 平台，驗證其在實際上字幕流程中的可行性。操作流程如圖 4-1 至 4-4。

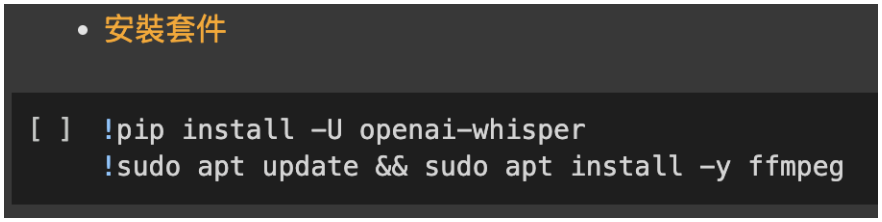

圖 4-1、實作二步驟一

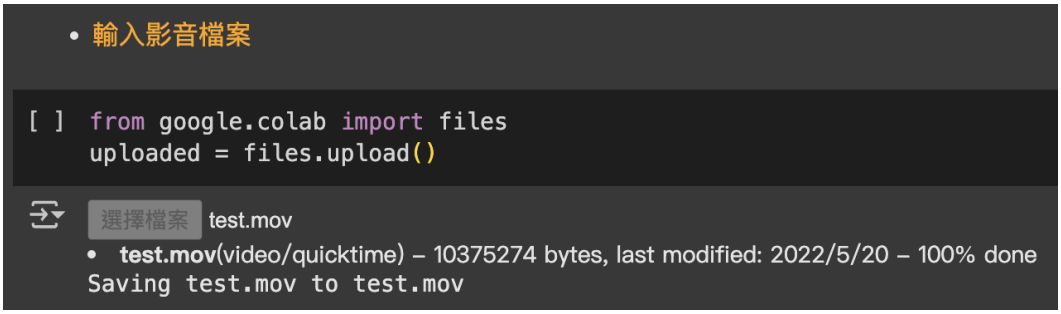

圖 4-2、實作二步驟二

- 字幕生成

```
[ ]  import whisper
     import os
     from datetime import timedelta
     import subprocess
     from google.colab import files

     # 輸入影片檔名
     filename = list(uploaded.keys())[0]
     base_name = os.path.splitext(filename)[0]

     # 轉成 mp3 音訊（確保 Whisper 能辨識）
     audio_file = f"{base_name}_audio.mp3"
     !ffmpeg -y -i "{filename}" -vn -acodec libmp3lame "{audio_file}"

     # 進行語音辨識
     model = whisper.load_model("large")
     result = model.transcribe(audio_file, language="en")  # zh=中文, en=英文

     # 產出 SRT 字幕檔
     srt_filename = f"{base_name}.srt"

     def format_srt_timestamp(seconds):
         milliseconds = int(seconds * 1000)
         hours = milliseconds // (3600 * 1000)
         minutes = (milliseconds % (3600 * 1000)) // (60 * 1000)
         secs = (milliseconds % (60 * 1000)) // 1000
         ms = milliseconds % 1000
         return f"{hours:02}:{minutes:02}:{secs:02},{ms:03}"

     with open(srt_filename, "w", encoding="utf-8") as f:
         for i, seg in enumerate(result["segments"], 1):
             if seg["avg_logprob"] < -1.0 or seg["no_speech_prob"] > 0.6:
                 continue
             start = format_srt_timestamp(seg["start"])
             end = format_srt_timestamp(seg["end"])
             text = seg["text"].strip()
             f.write(f"{i}\n{start} --> {end}\n{text}\n\n")
```

圖 4-3、實作二步驟三

- 存檔：字幕.srt檔

```
[ ]  from google.colab import files
     files.download(srt_filename)
```

圖 4-4、實作二步驟四

結果討論

完成語音辨識與字幕轉換後，將產出的 .srt 字幕檔案下載至本地如圖 5 ，並實際上傳至 YouTube 平台進行字幕測試。結果顯示字幕內容可成功套用於影片播放，對應時間軸準確、語句清晰如圖 6 。

testSubtitle.srt

```
00:00:13,000 --> 00:00:17,000
I have participated in high school academic events contest of geoscience.

5
00:00:17,000 --> 00:00:22,000
I won the first place in semi-final and got the qualification for final contest.

6
00:00:22,000 --> 00:00:24,000
That's one of these.

7
00:00:24,000 --> 00:00:29,000
But geologists need to do a lot of field geological surveys which require professional
financial.

8
00:00:29,000 --> 00:00:33,000
I happen to be a nature lover and enjoy various outdoor activities.

9
00:00:33,000 --> 00:00:40,000
Overall, I believe I'm a strong candidate for your department and hope I can be your student in
the next autumn semester.

10
00:00:40,000 --> 00:00:43,000
That's my introduction. Thank you.
```

圖 5、 字幕 srt 檔案內容

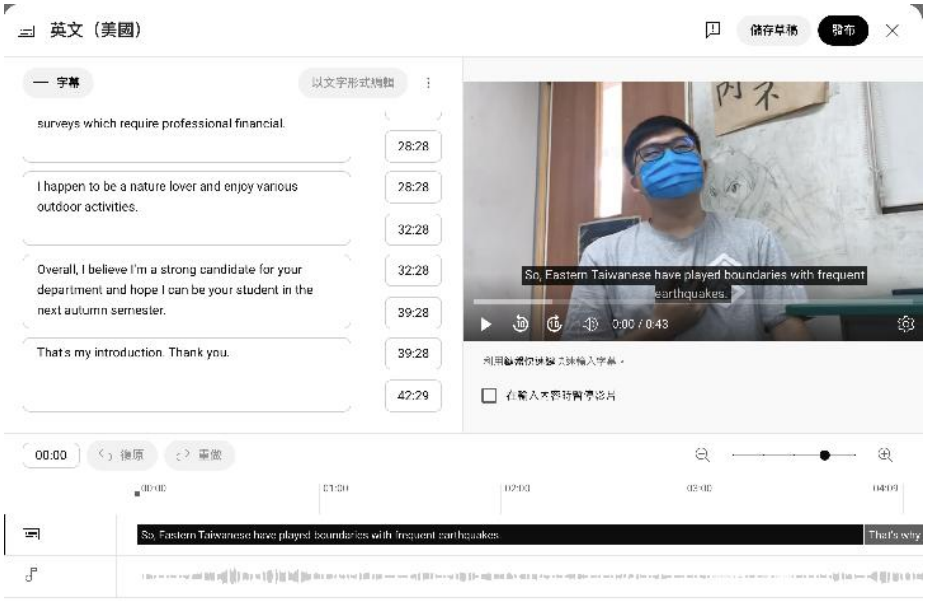

圖 6、實際 Youtube 字幕測試結果

三、結論

Whisper 作為一款支援多語言與多任務的語音轉文字模型，其準確度表現穩定，甚至對於口齒不清的語音資料都具備良好的辨識能力。此外，其開源性質與完整的模型權重釋出，使研究者能在本機端或雲端環境中靈活部署與實驗，無須依賴外部 API，也無需支付額外費用，提升了開發的自由度。

在實務操作上，Whisper 的使用相對直觀，只需輸入音訊檔案，即可快速獲得純文字結果與分段時間資訊。其輸出的 JSON 檔案結構清晰，特別是 segments 欄位提供的逐句時間戳與模型信心分數（如 avg_logprob、no_speech_prob 等），對於進一步分析辨識品質、標示錯誤或進行字幕對齊等應用極具幫助。

不過，在實作過程中也觀察到若使用者機器未配備具備足夠運算效能之 GPU，模型推論速度可能較慢甚至無法順利執行。為解決此問題，採用 Google Colab 作為執行平台。

整體而言，Whisper 不僅展現出優異的辨識能力，也具備高可擴充性與應用潛力。未來若能結合特定領域之語料進行微調（fine-tuning），或搭配自然語言處理模組進行摘要與關鍵字萃取，其應用範圍將更為廣泛。

參考文獻

Abbo, P. M., Kawasaki, J. K., Hamilton, M., Cook, S. C., DeGrandi-Hoffman, G., Li, W. F., Liu, J., & Chen, Y. P. (2017). Effects of Imidacloprid and Varroa destructor on survival and health of European honey bees, Apis mellifera. *Insect science, 24*(3), OpenAI. (2022). *Introducing Whisper*. OpenAI. https://openai.com/index/whisper/

OpenAI. (2023). *ChatGPT can now see, hear, and speak*. OpenAI. https://openai.com/index/chatgpt-can-now-see-hear-and-speak/

KoshurAI. (2024). *Unpacking OpenAI's Whisper: The game-changing AI transcription tool with real-world risks*. Medium. https://medium.com/@TheDataScience-ProF/unpacking-openais-whisper-the-game-changing-ai-transcription-tool-with-real-world-risks-0797e5afee99

AMIA. (2024). *Whisper AI transcription, human implementation* [Conference session]. AMIA 2024 Annual Symposium. https://amia2024.sched.com/event/1rKZ2/whisper-ai-transcription-human-implementation

Mollick, E. (2024). *Giving AI ears: AI roleplay feedback in oral presentations*. One Useful Thing. https://www.oneusefulthing.org/p/giving-ai-ears

Solos. (2024). *AirGo3 smart glasses expand live translate at CES 2024* [Press release]. Solos. https://www.solosglasses.com/