嵌入式系統導論專題

AI Disco System

410885010 劉嘉蕎

指導老師 林伯星

目錄

一、 摘要

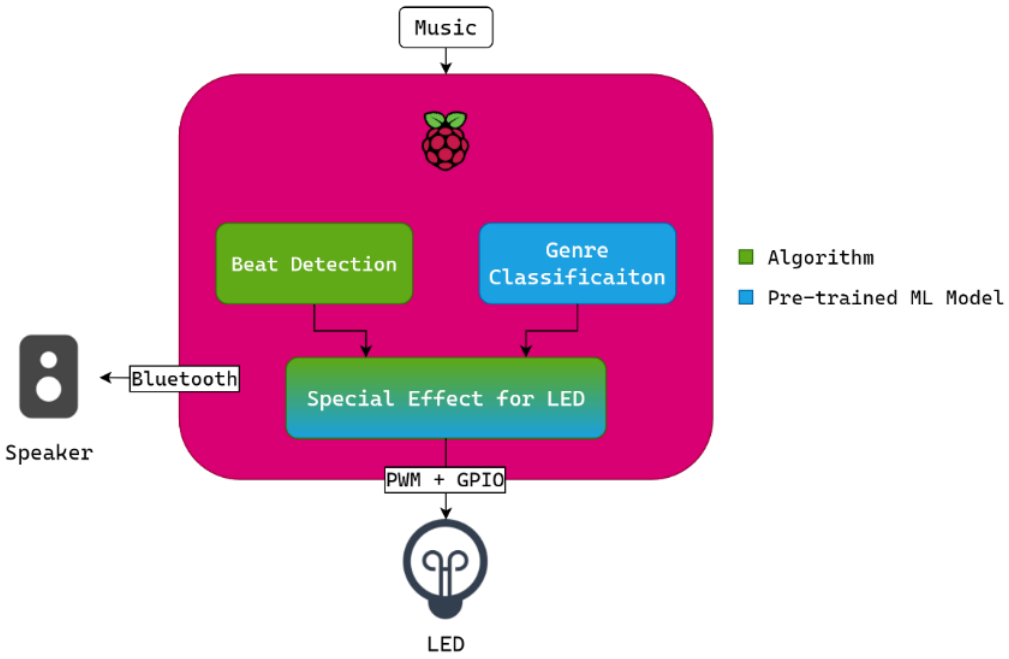
近年新冠肺炎疫情在全球爆發，世界各地普遍限制人民外出並保持社交距離，作為有效控制疫情的方式。原本以為，只要等到疫苗開發完成，就能大幅減輕新冠肺炎的威脅，疫情也能隨之結束，為無法外出的生活畫下句號。然而就在疫苗出現後，新變異株來勢凶猛，變異更多、傳播更快、感染更強的變種病毒嚴重威脅全球，疫苗的開發趕不上新變異株誕生的速度，我們的應對措施，不再是執意消滅病毒，也不是要清零，而是要重新思考如何與病毒共存。

如何在疫情之下維持原有的生活品質，成為需要被解決的當務之急。原本因為疫情，人們不如往常，在周末或慶功時，能與朋友在舞池、酒吧等地方慶祝放鬆，大大減少令人愉快的社交活動，生活變得枯燥乏味。因此，我想藉這次專題，運用課程所學，透過 AI 創造對應歌曲的光效，為大家打造在家裡也能蹦迪的環境，讓人們可以藉由邀請朋友到家的方式，維持原本的社交活動，卻享受原有的燈光氛圍。

關鍵字：後疫情生活、蹦迪、頻譜分析、機器學習

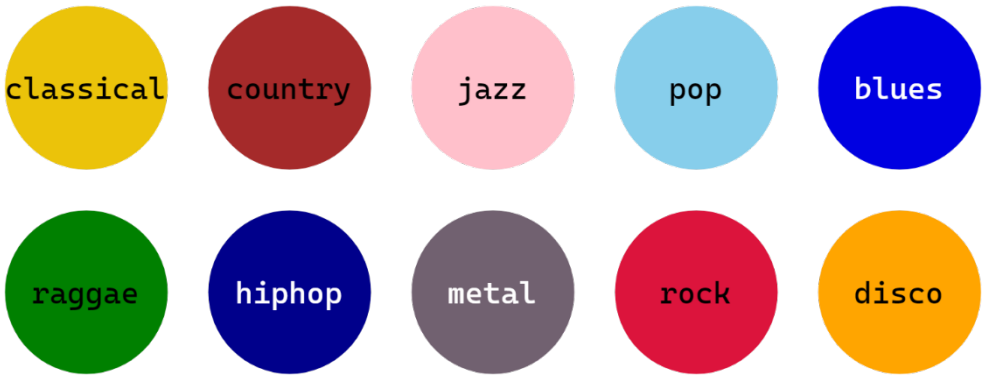
二、 架構

Raspberry Pi 4 通過藍芽輸出音樂、pwm 和 gpio 控制 LED 燈條。節奏偵測與生成燈光特效以演算法完成，而音樂分類的部分則通過 Tensorflow Lite 模型預測。



使用者要先上傳音樂檔案到 Raspberry Pi 4，並與藍芽音響連接。程式會分析每首歌曲的節奏點位置、從歌曲中間提出一小段音樂產生梅爾譜圖(Mel Spectrogram)，將譜圖輸入機器學習模型(CNN)，預測歌曲類型。模型將音樂類型分為十種，分別是藍調、古典、鄉村、迪斯可、嘻哈、爵士、金屬、流行、雷鬼和搖滾。

預測結果是十種音樂的機率，也可以視為是歌曲組成的比例。例如，如果一首歌是鄉村的機率為 79%、爵士 21%，那麼也可以將這首歌視為 79%的鄉村樂加上 21%的爵士樂。所以參考網路資料[1]，選出這十種音樂的代表色，作為蹦迪燈光特效的顏色依據：



分析完畢後，程式會開始播放音樂，同時將歌曲的組成比例當作抽中機率，抽出一種燈光特效，以兩個 pwm 通道與一個 gpio 通道分別控制 G、B、R 的燈光，每個節奏更換一次燈光特效。(由於 Raspberry Pi 4 只有兩個 pwm 通道，所以必須選擇一個顏色以 gpio 輸出。在紅綠藍三色當中，紅色是出現最多的顏色，也就是說，最無法從紅色區別出當前的音樂類型；而綠與藍的強弱，反而最能顯示燈光特效代表的音樂，因此選擇紅色以 0 和 1 輸出，綠藍則使用 pwm 通道輸出。)

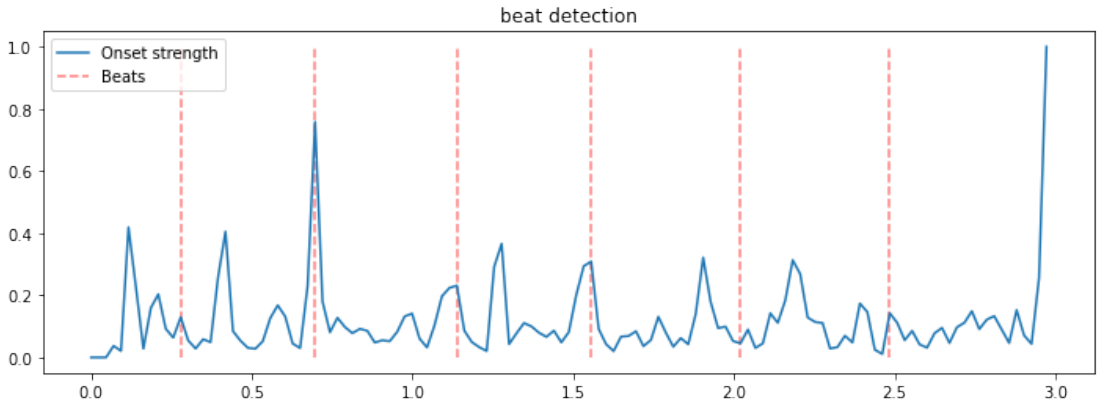
三、 方法

示範歌曲：**J Balvin, Willy William - Mi Gente ft. Beyoncé**

A. 偵測節奏

使用 Python 函式庫 Librosa [2] 的函式 `beat_track()` 偵測整首歌曲的節奏點位置，並計算節奏點之間的間隔，作為燈效停頓的時長。

為方便觀察，圖片僅輸出三秒內的節奏。



B. 機器學習

GitHub 上的專案 thomas-bouvier / music-genre-recognition [3]，以 Librosa 函式 `melspectrogram()` 產生的梅爾譜圖作為輸入資料，使用 GTZAN dataset 訓練 CNN 模型。

該專案模型的準確度約為 80%，表現普通，但 CNN 的速度較快、檔案較小，且產生輸入資料同樣使用 Librosa 的函式，所以本專題選擇使用此模型預測音樂類型。

由於 Keras 模型轉換成 Tensorflow Lite 模型對於版本有較多要求[4]，為避免環境錯誤產生問題，轉換模型的步驟會在 Google Colab 上進行，再將 tflite 模型上傳到 Raspberry Pi 4。

```
from google.colab import drive
drive.mount('/content/drive')
saved_model_dir = "/content/drive/MyDrive/model.h5"
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/conten
```

```
!pip install q keras==2.0.5
```

```
%tensorflow_version 1.x
```

```
TensorFlow 1.x selected.
```

```
import tensorflow as tf
print(tf.version.VERSION)
```

```
1.15.2
```

```
!pip install 'h5py==2.10.0' --force-reinstall
```

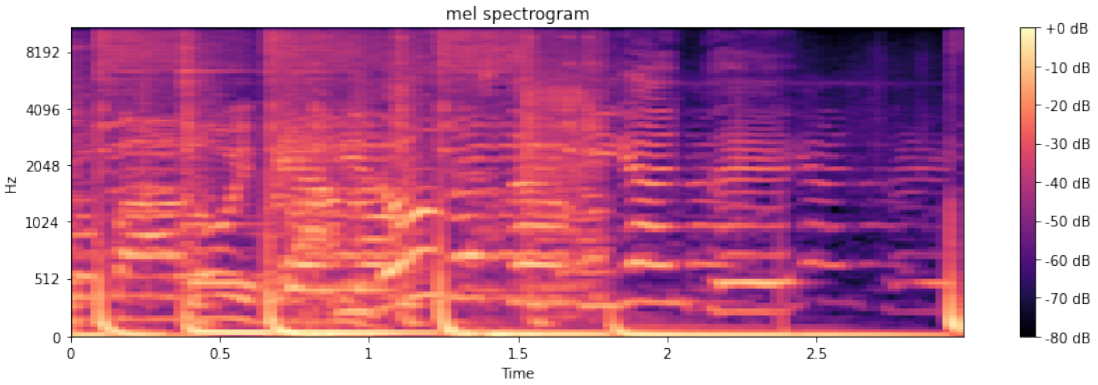
```
# Convert the model.
converter = tf.compat.v1.lite.TFLiteConverter.from_keras_model_file(saved_model_dir)
tflite_model = converter.convert()

# Save the model.
with open('model.tflite', 'wb') as f:
  f.write(tflite_model)
```

```
WARNING:tensorflow:From /tensorflow-1.15.2/python3.7/tensorflow_core/python/ops/resource_variable_
Instructions for updating:
If using Keras pass *_constraint arguments to layers.
WARNING:tensorflow:From /tensorflow-1.15.2/python3.7/tensorflow_core/lite/python/util.py:249: conv
Instructions for updating:
Use `tf.compat.v1.graph_util.convert_variables_to_constants`
WARNING:tensorflow:From /tensorflow-1.15.2/python3.7/tensorflow_core/python/framework/graph_util_i
```

最後參考 tf.lite 官方文件[5]啟用 interpreter，得到預測結果。

C. 頻譜分析

由於 GitHub 專案為整首歌曲切出重疊率 50%的窗口，再產生每個窗口的頻譜進行訓練，過程相當耗費時間，所以本專題修改輸入資料，僅僅切出一個約三秒的窗口產生頻譜做為 CNN 的輸入資料，實測發現預測結果仍然相當可靠，不須輸入整首歌曲的頻譜。



D. 模型預測

實測 Mi Gente 的音樂類型最有可能是嘻哈(0.41)、流行(0.37)和雷鬼(0.20)，顯示預測結果相當符合。

```
playing Mi Gente.mp3
metal: 7.1529320848640054e-06
disco: 0.003077058820053935
classical: 1.1028521385014756e-07
hippop: 0.41392970085144043
jazz: 0.002825259929522872
country: 0.0007476953323930502
pop: 0.3745923638343811
blues: 0.00012432316725607961
reggae: 0.20468877255916595
rock: 7.589549568365328e-06
```

E. 燈光特效

音樂的 RGB 數值依照前述「音樂代表色」計算得到。

```
GENRES = ['metal', 'disco', 'classical', 'hippop', 'jazz',
          'country', 'pop', 'blues', 'reggae', 'rock']
COLORS = [(35,30,35), (61,39,0), (53,44,3), (0,0,100), (39,30,31),
          (66,17,17), (23,36,41), (0,0,100), (0,100,0), (73,7,20)]
```

綠色的燈效由 pwm0 控制、藍色由 pwm1 控制，紅色則是設定閥值 50，若數值介於 51 到 100 則燈亮，反之燈暗。

```
for d in duration:
    effect = random.choices(range(10), output_data)[0]
    print('special effect is ' + GENRES[effect])
    print('last for ' + str(d.item()) + ' sec\n')
    pwm0.ChangeDutyCycle(COLORS[effect][1])
    pwm1.ChangeDutyCycle(COLORS[effect][2])
    if COLORS[effect][0] > 50:
        led.on()
    else:
        led.off()
    time.sleep(d.item())
```

四、 實測結果

執行程式後，首先會偵測整首歌曲的節奏，準備預測資料，啟用 interpreter，得到預測結果。完畢後，使用 Python 的函式庫 vlc 的函式 `play()` 播放音樂，同時將音樂類型的預測結果當作抽中機率，抽出一種燈光特效，以兩個 pwm 通道與一個 gpio 通道分別控制 G、B、R 的燈光，每個節奏更換一次燈光特效。

每次抽選燈效與間隔時間都會在終端機輸出，如下圖所示：

```
pi@raspberrypi:~/c9sdk/environment $ python3 project.py
press Ctrl-C to stop
playing Mi Gente.mp3
special effect is hippop
last for 0.278564453125 sec

special effect is hippop
last for 0.5810546875 sec

special effect is hippop
last for 0.580078125 sec
```

或是也可以從以下連結觀看含 LED 燈效的實測結果：

嵌入式系統導論專題實作結果

五、 結論

AI Disco System 通過藍芽輸出音樂，結合頻譜分析與機器學習的技巧，為音樂打造專屬的燈光特效。如果購買額外的燈光設備連接，就能打造更加完整的蹦迪環境，對使用者而言，無疑是疫情下與朋友在舞池放鬆的最佳替代方法。使用者只需自行建立播放清單，即可在符合防疫規定的同時，享受由電腦為歌曲產生的專屬燈光特效，盡情與朋友們蹦迪。對於本專題，如果繼續深入研究舞池 DJ 的工作內容，以機器學習等方式實作，或許能夠完全複製 DJ 的角色，為 AI Disco System 的音樂自動混音，製造獨一無二的音樂 remix，進一步提升使用者在家蹦迪的體驗。

六、 參考資料

[1] 維基百科：音樂類型顏色

[2] Librosa 0.8.1 Documentation

[3] GitHub Project：music-genre-recognition

[4] TensorFlow Lite 轉換工具

[5] Module: tf.lite | TensorFlow Core v2.7.0