多媒體內容分析 hw5

F74082141 資訊 112 王浩

0、介紹環境

使用 python 3.11.0, opencv 4.7.0, scikit-learn 1.2.2, librosa 0.10.0.post2

1、 使用的 Audio Features

1.1 MFCC

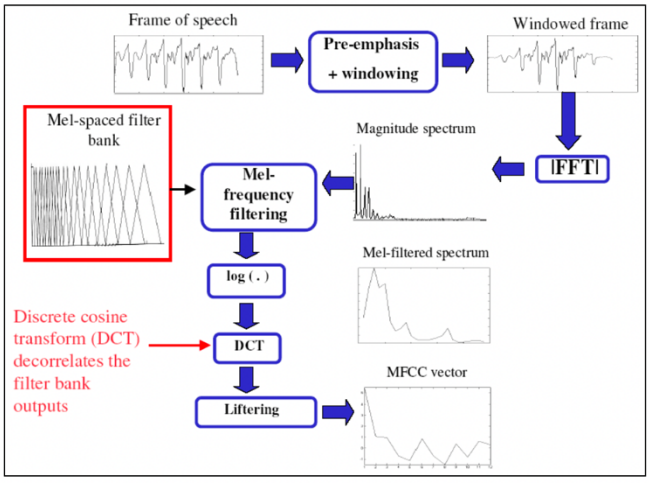
本次使用 librosa 的 mfcc 最為擷取 Features 的實作

```
mfcc_features = mfcc(y=signal, sr=sample_rate)
```

來作為套件使用，MFCC 廣泛應用於音樂和語音處理任務中，能夠很好的捕捉音訊的 frequency、energy features，特別是人類耳多感知的音高和音色。

在 mfcc 這個 function 中，y 是 input 的 audio signal，sr 是音訊的 sampling rate（每秒樣本數）。這個 function 會先進行 pre-emphasis 跟 windowing，pre-emphasis 可以加重 signal 的高頻部分，也就是 high pass filter，而 windowing 則是加強區段的性質。接著就會做 FFT 傅立葉轉換，將 time domain 性質轉成 frequency domain。Mel-frequency filtering 就是調整成「人類會感知」到的程度調整，最後包括將音訊轉換為 Discrete Cosine Transform（DCT）以獲取 MFCC。

預設情況下，它會計算出 20 個 MFCC 係數。而我們本次的 feature 全部都會使用到。



$$X_a[k] = \sum_{n=0}^{N-1} x[n] e^{-j2\pi nk/N}, \quad 0 \le k < N$$

$$S[m] = \ln\left[\sum_{k=0}^{N-1} |X_a[k]|^2 H_m[k]\right], \quad 0 < m \le M$$

$$c[n] = \sum_{m=0}^{M-1} S[m] \cos(\pi n(m-1/2)/M), \quad 0 \le n < M$$

M: the number of filters
N: the size of the FFT

1.2 BHF

本次使用 librosa 的 tempogram 最為擷取 Features 的實作

```
beat_features = tempogram(y=signal, sr=sample_rate, hop_length=hop_length, win_length=n_fft)
```

在這個 function 中，y 是 input 的 audio signal，sr 是音訊的 sampling rate（每秒樣本數）。

另外，tempogram 特徵的計算是基於 Short-time Fourier Transform(STFT)，因此需要提供相關的參數，還有兩個參數需要提供：hop_length 和 win_length。hop_length 表示 window 的「移動步長」（以樣本數表示），用於控制 tempo 特徵的時間解析度。win_length 則表示用於計算 tempo 特徵的 window「大小」（以樣本數表示），用於控制節奏特徵的頻率解析度。

tempogram 特徵描述了音訊中不同時間段內節奏的變化情況。通過計算不同時間 window 內的節奏強度分佈，可以獲得一個 time-tempo 強度的二維特徵表示。本次實作，有鑒於計算量大小，我們在有些情況下限制了 win_length(n_fft)的值。

1.3 PHF

本次使用 librosa 的 chroma_cqt 最為擷取 Features 的實作

```
pitch_features = chroma_cqt(y=signal, sr=sample_rate, hop_length=hop_length, n_chroma=12)
```

在這個 function 中，y 是 input 的 audio signal，sr 是音訊的 sampling rate（每秒樣本數）。

librosa.feature.chroma_cqt 函式用於從 signal audio 中提取音高特徵。音高特徵是一種描述音樂中「音高分佈」的表示方法，它將音樂中的聲音轉換為不同音高類別的強度分佈。這種特徵可以捕捉音樂中音高的模式和變化。

另外，他還需要設定 hop_length 和 n_chroma 兩個參數。hop_length 控制特徵的時間解析度，表示每次計算「音高特徵的樣本移動量」。n_chroma 則指定音高特徵的音高「類別數量」，通常設定為 12，對應於音樂中的 12 個音階。

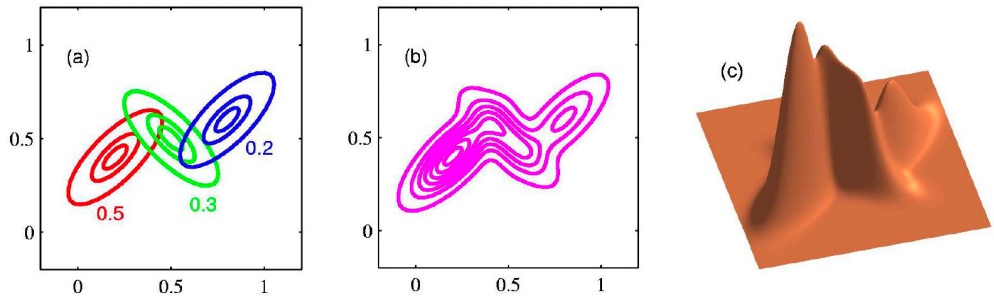
2、 分類的方法、模型

本次根據 paper 的結論

TABLE I
CLASSIFICATION ACCURACY MEAN AND STANDARD DEVIATION

	Genres(10)	Classical(4)	Jazz(6)
Random	10	25	16
RT GS	44 ± 2	61 ± 3	53 ± 4
GS	59 ± 4	77 ± 6	61 ± 8
GMM(2)	60 ± 4	81 ± 5	66 ± 7
GMM(3)	61 ± 4	88 ± 4	68 ± 7
GMM(4)	61 ± 4	88 ± 5	62 ± 6
GMM(5)	61 ± 4	88 ± 5	59 ± 6
KNN(1)	59 ± 4	77 ± 7	57 ± 6
KNN(3)	60 ± 4	78 ± 6	58 ± 7
KNN(5)	56 ± 3	70 ± 6	56 ± 6

直接以效果最好的 GMM，有三個 component 的方式來作為模型。



而 Gaussian Mixture Model 簡單來說就是 GMM 的主要思想是將資料視為由多個高斯分佈組成的混合，每個高斯分佈對應於資料中的一個群體或類別。模型的目標是通過學習分佈的參數，即每個高斯分佈的均值、協方差和權重，來描述資料的統計特性。

在學習過程中，GMM 使用 Expectation-Maximization, (EM)來估計模型的參數。該算法 iteratively 進行兩個步驟：期望步驟（E-step）和最大化步驟（M-step）。在 E-step 中，根據當前的模型參數，計算每個資料點屬於每個高斯分佈的概率；在 M-step 中，根據這些概率重新估計模型的參數。通過反覆進行 E-step 和 M-step，模型逐漸收斂並學習到資料的分佈。

3、 分類準確率

3.1 結果

mfcc	tempo	pitch	hop_length	n_fft	max_length	score
v	v	v	64	128	500	0.119
v	v	v	64	256	700	0.112
v			64	256	700	0.076
v			64	256	500	0.110
v			64	256	100	0.158
v			64	256	800	0.088

(max_length 指所有餵入的影音檔得到的 feature，經過 truncate 後的大小)

3.2 結論

3.2.1 tempo, pitch 有助於將精準度提高

由上述的內容可以得知，在第二與第三項實驗中，若我們將 tempo、pitch 的 BHF、PHF 類型資訊餵入模型中，在數據上，那就會有助於將精準度提升 0.046。

3.2.2 可能因為 n_fft 的數量限縮，倒致分數偏低

因為 n_fft 表示 Fast Fourier Transform, (FFT)的 window size。主要用途就是在 time-domain signal 上進行 frequency-domain 的分析。通過將音訊信號分成短時間的 clip，然後對每個 clip 都進行 FFT，可以獲得該片段的 frequency 資訊。所以 n_fft 決定了每個片段的長度，較大的 n_fft 可以提供更高的「解析度」。較小的 n_fft 可以提供的頻譜解析度就會比較低，但計算速度快、成本也比較低。更仔細說，n_fft 在 MFCC 的計算中，需要先進行 Short-Time Fourier Transform, (STFT)來獲得頻譜。

所以本次實驗因為硬體的關係，將 n_fft 調整在 128、256 兩個數量上，有可能因此，導致 n_fft 也許沒辦法達到適當的 1000~2000 的解析度情況，造成本次實驗的總體分數較低。

3.2.3 max_length 的大小沒有統一何者較好

max_length 指所有餵入的影音檔得到的 feature，經過 truncate 後的大小，根據實驗通常得到的結果會是 1000 多，但經過第 3、4、5、6 次實驗，score 雖然可以從 max_length 700 到 500 到 100 的過程中逐步上

升，但實驗在 max_length 800 時又重新掉回 0.088，我認為可能是因為 3.2.2 的實驗先天問題，導致 max_length 在目前的參數搭配下，較難看出其特定的規律。