105061254 林士平 數位訊號處理實驗報告 Lab8

1. Abstract/Introduction

本實驗的目標為提高嬰兒哭聲辨識率，database 共有五種 label：**Canonical, Crying, Junk, Laughing 和 Non-canonical**。training set 有 3996 筆資料，testing set 則有 3617 筆資料。

利用 **MFCC** 作為聲音特徵，選用適合的演算法，然後使用 training set 來訓練模型，調整出適當的 hyperparmeter 讓 model 的 performance 達到最好。評估 performance 的方式為 **k-fold cross-validation**，並將最後對 testing set 的預測結果上傳到 kaggle 評分。

關於嬰兒哭聲辨識的研究，網上可以查到相當多的論文，目的往往是因為嬰兒無法用言語表達情感，然而光從哭聲我們無法理解他是為何而哭，是肚子餓嗎？抑或是生氣？還是難過？或是想睡覺？藉由機器學習甚至是深度學習，選用適當的聲音特徵，並選擇適合的演算法，有機會可以分辨出上述的不同，並以此預測結果對症下藥，讓寶寶惱人的哭聲不再繼續。

2. Goals of this lab

The baby sound challenge(嬰兒哭聲辨識挑戰)

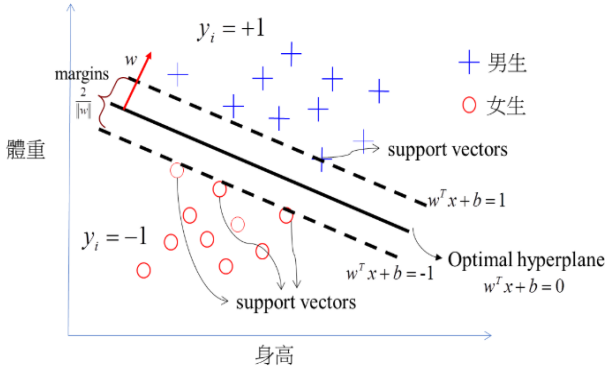
- 擷取聲音特徵。
- 選用適當的演算法建立 model。
- 配合適當的 preprocessing 和評估 performance 的方式。
- 期望可以建立出高辨識率的模型。

3. Method

首先我們使用 Lab5 教的 MFCC 來做為 feature。取得 MFCC 的過程就是在**去蕪存菁**：從 Pre-emphasis、STFT、Mel-scale Filtering、Log Energy 到 Discrete Cosine Tranform，由這些程序可以將聲音的精華留下來，而這些精華是重要的**語音特徵**。本次實驗我使用 **librosa 函數庫**，可以更輕鬆地由音檔取得 MFCC。

接下來針對取得的 feature 做**標準化**，標準化是在做機器學習前很常會使用的一個方法，可以讓訓練出來的模型有更好的 performance。

然後選用演算法並取用適當的 hyperparameter。本次我採用 SVM(Support vector machine)。SVM 的概念非常簡單，就是找到一個決策邊界(decision boundary)**讓兩類之間的邊界(margins)最大化**，使其完美區隔開來。



由上頁的示意圖可以看到男生和女生之間有一條明顯的界線，SVM 的目標就是讓 margin 越大越好。

其中 SVM 有個重要的參數 C，**C 越大 model 會有 high accuracy 但 poor generalization，相對的 C 越小 model 會有 low accuracy 但 good generalization**。

在做 training 前我把 Data 分成 **training set 和 testing set**。Training set 用來訓練出模型的參數，然後用 testing set 來確認這個模型的好壞。Cross-validation 即是重複這個過程，其中 K-fold 是比較常用的交叉驗證方法。做法是將資料**隨機平均分成 k 個集合**，然後將某一個集合當做「測試資料(Testing data)」，剩下的 k-1 個集合做為「訓練資料(Training data)」，如此**重複進行直到每一個集合都被當做「測試資料(Testing data)」**為止。最後的結果(Predication results)再和真實答案(ground truth)進行成效比對(Performance Comparison)。

Cross validation 是選用 hyperparmeter(C)的重要根據。最後利用 train 出來的模型做預測，將預測的結果上傳 kaggle。

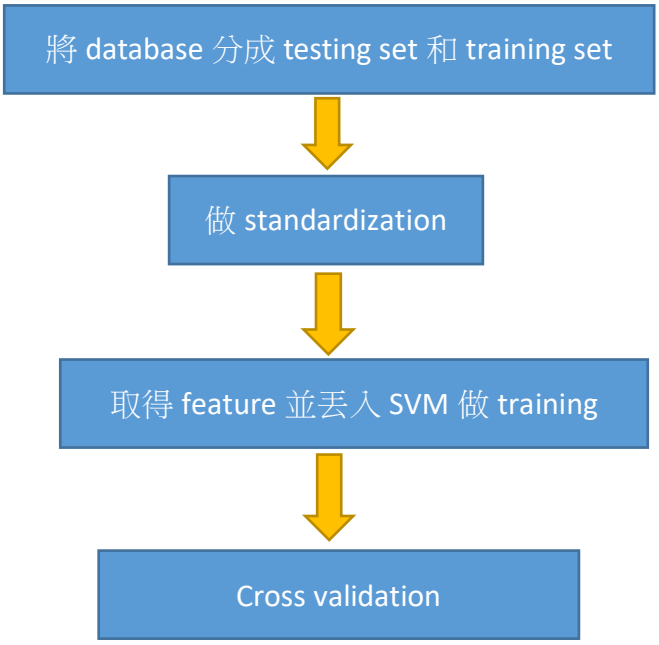
4. Pseudo code

(0) 定義 pre emphasis 函數	*def pre_emphasis(signal,coefficient=0.95):* *return np.append(signal[0],signal[1:]-coefficient*signal[:-1])* **說明：pre-emphasis 將語音訊號 s(n) 通過一個 High pass filter：** $$H(z) = 1 - a * z^{-1}$$ **其中 a 介於 0.9 和 1.0 之間。若以 time domain 的運算式來表示，pre-emphasis 後的訊號 $s_2(n)$ 為** $$s2(z) = s(n) - a * s(n-1)$$
(0) 定義 MFCC_feat 函數取得 feature	*def MFCC_feat(file):* *y, sr = librosa.load(file)* *y2 = pre_emphasis(y)*

	# mfcc *mfcc = librosa.feature.mfcc(y=y2, sr=sr, hop_length=512, htk=True)* *feature = np.mean(mfcc, axis=1)* *feature = np.hstack((feature, np.std(mfcc, axis = 1)))* *feature = np.hstack((feature, np.max(mfcc, axis = 1)))* *feature = np.hstack((feature, np.min(mfcc, axis = 1)))* *return feature* 說明：在取得 **mfcc** 前先將音檔丟到 **pre emphasis** 函數做預處理。接著由 **librosa.feature.mfcc** 取得 **mfcc** 聲音特徵，其中每個音檔會有 **20** 個特徵，我將不同音框的這些特徵做平均，得到前 **20** 個 **feature**，再將不同音框的這些特徵取標準差，再得到 **20** 個 **feature**，接著把不同音框的這些特徵取最大最小值，便可以得到每個音檔各 **80** 個 **feature**。
(0) 定義 cross_val 函數做交叉驗證評估模型 performance	*def cross_val(cv, c, train_data, train_target):* *'''* *You can do cross validation here to find the best 'c' for training.* *'''* *RANDSEED = 0* *Kf = KFold(n_splits=cv, shuffle=True, random_state=RANDSEED)* *y_test_cv = []* *y_predict_cv = []* *for cvIdx, (trainIdx, testIdx) in enumerate(Kf.split(range(len(train_data)))):* *# split data into Train & Test* *X_train, X_test = train_data[trainIdx], train_data[testIdx]* *y_train, y_test = train_target[trainIdx], train_target[testIdx]* *# perform the same train / testing process in IRIS-TrainingTest* *clf = SVC(kernel='rbf', random_state=0, C = c)* *# Note that u have to build a new classifier too!* *clf.fit(X_train, y_train)* *y_predict= clf.predict(X_test)* *# collect the predict results and ground truths from each folds* *y_test_cv.extend(y_test)* *y_predict_cv.extend(y_predict)* *# Evaluation* *unweighted_averaged_recall = recall_score(y_test_cv, y_predict_cv,*

	average='macro') *cm = confusion_matrix(y_test_cv, y_predict_cv)* *# Visulization* *print(cm)* *print('UAR = ', unweighted_averaged_recall)* *pass*
(1) 對 feature 做標準化	*sc = StandardScaler()* *sc.fit(X_train)* *X_train_std = sc.transform(X_train)* *X_test_std = sc.transform(X_test)*
(2) training	*C_me = 0.8* *model = SVC(kernel='rbf', random_state=0, C = C_me)* *model.fit(X_train_std, y_train)* *y_pred = model.predict(X_test_std)* 說明：**training，經過反覆測試取用 hyperparameter C = 0.8，SVM kernal 為 rbf。**
(3) cross validation 確認模型的 performance	*#cross validation* *fold = 10* *cross_val(fold, C_me, X_train_std, y_train)*

5. Flow chart



6. Results

```
[[ 104    1   86    0  253]
 [   1   14   15    0  213]
 [  29    0 1549    0  248]
 [   1    0   11    0   34]
 [  38   12  254    0 1133]]
UAR =  0.38571957178966987
```

↑ confusion matrix 和 UAR

↑ kaggle 目前評分為 0.65210