鳥鳴特徵頻率辨識

分析頻率組成並做鳥種預測

大綱

- 動機
- 訓練步驟

  Step1_取鳥類鳴叫聲作為資料庫

  Step2_前處理將時域訊號做傅立葉轉換為頻域訊號

  Step3_前處理將目標做結構化處理

  Step4_切分訓練集與測試集並執行模型訓練及測試

  Step5_準確率評估

- 結論

動機　訓練步驟　結論

動機

鳥類棲地觀測除現地踏查外，亦會架設錄音器材取得鳥類鳴叫聲音檔，做鳥種分析辨識。此專題試圖將鳥鳴頻率組成作為訓練依據，藉此作鳥種預測。

動機　訓練步驟　結論

Step1_取鳥類鳴叫聲作為資料庫(1/3)

從Kaggle找到鳥鳴錄音檔集合

🦜 Sound Of 114 Species Of Birds Till 2022 🦜

GET AMAZING 2161 SOUNDS🔊OF 114 UNIQUE BIRDS🐦& RUN YOUR EXPERIMENTAL KERNELS .

Data Card | Code (2) | Discussion (0)

About Dataset

IF YOU FIND THIS DATASET USEFUL THEN MAKE AN UPVOTE 👍 .

THIS DATASET HAVE **2** SECTION .

1.VOICE OF BIRDS:

- HERE YOU FIND 2161 AUDIO FILES OF 114 DIFFERENT SPECIES OF BIRDS .

2.Birds Voice.csv:

- HERE YOU FIND THE METADATA ABOUT 2161 AUDIO FILES & 114 BIRDS .

Usability ⓘ
10.00

License
CC0: Public Domain

Expected update frequency
Annually

Step1_取鳥類鳴叫聲作為資料庫(2/3)

搜尋資料夾中mp3音檔有達30個以上的前5個鳥種

```
#走訪特定目錄下的所有資料夾(資料夾皆以鳥種名稱命名)，列出前5個音檔達30個以上的資料夾名稱及數量。
#需確認音檔格式為mp3檔
allList = os.listdir('Voice of Birds')
birdspecies = []

for dirname in allList:
    if len(birdspecies) <= 4:
        count = 0
        subpath = os.path.join('Voice of Birds', dirname)
        sublist = os.listdir(subpath)
        for soundtrack in sublist:
            if soundtrack[soundtrack.index('.')+1:] == 'mp3':
                count += 1
            if count >= 30:
                birdspecies.append(dirname)
print(birdspecies)
```

```
['Andean Guan_sound', 'Andean Tinamou_sound', 'Band-tailed Guan_sound', 'Bartletts Tinamou_sound', 'Black-capped Tinamou_soun
d']
```

Step1_取鳥類鳴叫聲作為資料庫(3/3)

新建資料夾並將音檔複製到其中

```
#取出該5種鳥種之音檔並放置於新建的data資料夾中
if os.path.exists('data'):
    shutil.rmtree('data')
os.mkdir('data')

for name in birdspecies:
    sourcepath = os.path.join('Voice of Birds', name)
    file = os.listdir(sourcepath)
    for g in file:
        shutil.copyfile(os.path.join(sourcepath, g), os.path.join('data', g))
```

Step2_前處理將時域訊號做傅立葉轉換為頻域訊號(1/2)

收集目標集合與頻率數據集合

```
#收集target：由於音檔檔名包含鳥種名稱，故存取檔名做target集合
soundlist = os.listdir('data')
target = []
for s in soundlist:
    tg = s[0: s.index('.')-2]
    target.append(tg)

#收集data：收集各音檔之Octave Band頻域之規一化數據
octave_value = np.zeros([1,10])

for s in soundlist:
    T = AudioSegment.from_mp3(os.path.join('data', s)).duration_seconds
    Fs = AudioSegment.from_mp3(os.path.join('data', s)).frame_rate
    x = a2n.audio_from_file(os.path.join('data', s))
    x = np.ravel(x[0].transpose())[0:int(T*Fs)]
    X = fft.rfft(x)
    freq = fft.rfftfreq(len(x), d = 1 / Fs)
    amp = np.abs(X)
    f0 = 1/T
```

Step2_前處理將時域訊號做傅立葉轉換為頻域訊號(2/2)

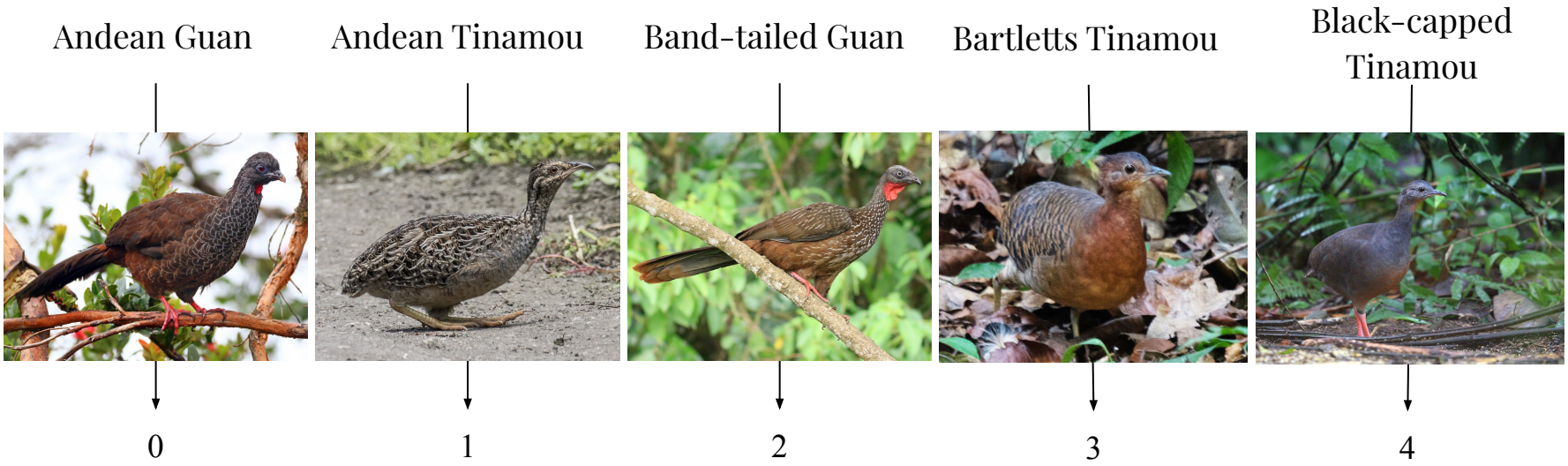
將目標集合與頻率數據集合合併存成csv檔

```
#將data及target合併，並存成csv檔
cols = ['16Hz','31.5Hz','63Hz','125Hz','250Hz','500Hz','1000Hz','2000Hz','4000Hz','8000Hz']
df_octave_value = pd.DataFrame(delete_octave_value, columns=cols)
df_target = pd.DataFrame(np.array(target), columns=['target'])

res = pd.concat([df_octave_value,df_target],axis=1)
res.to_csv('birdsoundrecog.csv', header=True, index=False)
```

	16Hz	31.5Hz	63Hz	125Hz	250Hz	500Hz	1000Hz	2000Hz	4000Hz	8000Hz	target
0	0.000000	0.013836	0.139232	0.199144	0.206934	0.202467	1.000000	0.980490	0.212575	0.207851	Andean Guan
1	0.000000	0.013836	0.139232	0.199144	0.206934	0.202467	1.000000	0.980490	0.212575	0.207851	Andean Guan
2	0.000000	0.013836	0.139232	0.199144	0.206934	0.202467	1.000000	0.980490	0.212575	0.207851	Andean Guan
3	0.000000	0.000050	0.001638	0.067103	0.233409	0.360025	0.963711	1.000000	0.048460	0.017908	Andean Guan
4	0.000000	0.000050	0.001638	0.067103	0.233409	0.360025	0.963711	1.000000	0.048460	0.017908	Andean Guan
...	...	...	...	...	...	...	...	...	...	...	...

Step3_前處理將目標做結構化處理(1/2)



*圖片來自The Cornell Lab of Ornithology, eBird

Step3_前處理將目標做結構化處理(2/2)

各頻段規一化後的數據, 1代表該頻段占比最多, 0代表該頻段占比最少

	16Hz	31.5Hz	63Hz	125Hz	250Hz	500Hz	1000Hz	2000Hz	4000Hz	8000Hz	target
0	0.000000	0.013836	0.139232	0.199144	0.206934	0.202467	1.000000	0.980490	0.212575	0.207851	0
1	0.000000	0.013836	0.139232	0.199144	0.206934	0.202467	1.000000	0.980490	0.212575	0.207851	0
2	0.000000	0.013836	0.139232	0.199144	0.206934	0.202467	1.000000	0.980490	0.212575	0.207851	0
3	0.000000	0.000050	0.001638	0.067103	0.233409	0.360025	0.963711	1.000000	0.048460	0.017908	0
4	0.000000	0.000050	0.001638	0.067103	0.233409	0.360025	0.963711	1.000000	0.048460	0.017908	0
...	...	...	...	...	...	...	...	...	...	...	...
145	0.009033	0.026994	0.022312	0.043509	0.014654	0.013201	1.000000	0.393745	0.058640	0.000000	4
146	0.036702	0.014212	0.005409	0.000840	0.000000	0.010328	1.000000	0.343887	0.056378	0.008303	4
147	0.000376	0.000105	0.000000	0.000282	0.001807	0.053563	1.000000	0.354200	0.030050	0.005231	4
148	0.000376	0.000105	0.000000	0.000282	0.001807	0.053563	1.000000	0.354200	0.030050	0.005231	4
149	0.000376	0.000105	0.000000	0.000282	0.001807	0.053563	1.000000	0.354200	0.030050	0.005231	4

150 rows × 11 columns

Step4_切分訓練集與測試集並執行模型訓練及測試

切分訓練集與測試集

```
#將數據(含鳥種名稱結構化)切分為訓練級與測試級
x = df_data[['16Hz','31.5Hz','63Hz','125Hz','250Hz','500Hz','1000Hz','2000Hz','4000Hz','8000Hz']]
y = df_data[['target']]
X_train, X_test, Y_train, Y_test = train_test_split(x, y, test_size=0.3, random_state=100)
```

模型訓練

```
#取訓練級做模型訓練
## 建立邏輯迴歸模型
model = model = linear_model.LogisticRegression()
## 擬和數據
model.fit(X_train, Y_train)
```

模型測試

```
#取測試級做模型測試
prediction = model.predict(X_test)

print('Real Result: ', Y_test)
print('Model Predict: ', prediction)
```

Step5_準確率評估

訓練準確率約 70.48 %, 測試準確率約 73.33 %

```
#評估模型表現
score_train = model.score(X_train, Y_train)
score_test = model.score(X_test, Y_test)
print('Training Accuracy :' + str(score_train * 100) + '%')
print('Testing Accuracy :' + str(score_test * 100) + '%')
```

```
Training Accuracy :70.47619047619048%
Testing Accuracy :73.33333333333333%
```

動機　訓練步驟　結論

結論

1. 本次只使用邏輯迴歸模型，欲提高準確率尚須考慮其他迴歸或分類方式。

2. 音檔中的環境雜音會影響模型學習，為提高準確率需先濾除雜音。

	16Hz	31.5Hz	63Hz	125Hz	250Hz	500Hz	1000Hz	2000Hz	4000Hz	8000Hz	target
0	0.000000	0.013836	0.139232	0.199144	0.206934	0.202467	1.000000	0.980490	0.212575	0.207851	0
1	0.000000	0.013836	0.139232	0.199144	0.206934	0.202467	1.000000	0.980490	0.212575	0.207851	0
2	0.000000	0.013836	0.139232	0.199144	0.206934	0.202467	1.000000	0.980490	0.212575	0.207851	0
3	0.000000	0.000050	0.001638	0.067103	0.233409	0.360025	0.963711	1.000000	0.048460	0.017908	0
4	0.000000	0.000050	0.001638	0.067103	0.233409	0.360025	0.963711	1.000000	0.048460	0.017908	0
5	0.000000	0.000050	0.001638	0.067103	0.233409	0.360025	0.963711	1.000000	0.048460	0.017908	0
6	0.000181	0.000000	0.000104	0.000546	0.020744	0.236513	0.863480	1.000000	0.234535	0.124922	0
7	0.130692	1.000000	0.759840	0.075508	0.044478	0.070921	0.105272	0.066852	0.010263	0.000000	0
8	0.787597	1.000000	0.244368	0.278105	0.680571	0.124588	0.050791	0.028054	0.010572	0.000000	0

3. 除頻率組成分析外，也有研究是將鳴叫聲轉為聲紋(圖像化)，用影像分析做訓練及預測。為使提升準確率可多嘗試不同訓練方式。