

クラウド・テクノロジー活用部会活動報告 実践機械学習システム 音声編

2016年9月16日

先端IT活用推進コンソーシアム
クラウド・テクノロジー活用部会
富士通株式会社 松井 唯

- 機械学習では画像や自然言語に対する処理が盛ん
- 画像分野
 - Googleの猫
- 画像＋言語
 - 画像からの説明文生成



People play soccer

※この発表は個人の見解であり、所属する組織の公式見解ではありません

音響分野における機械学習

- 音響分野では音声認識に対する適用が研究
 - Siri
 - Google音声認識
 - ロボット対話

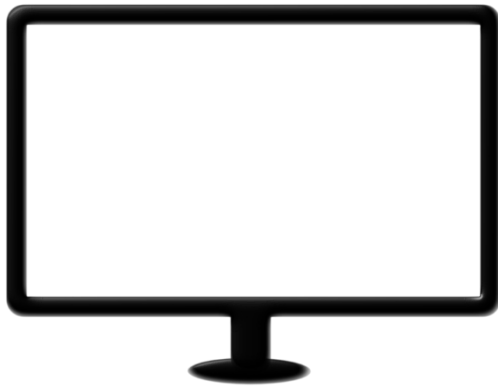
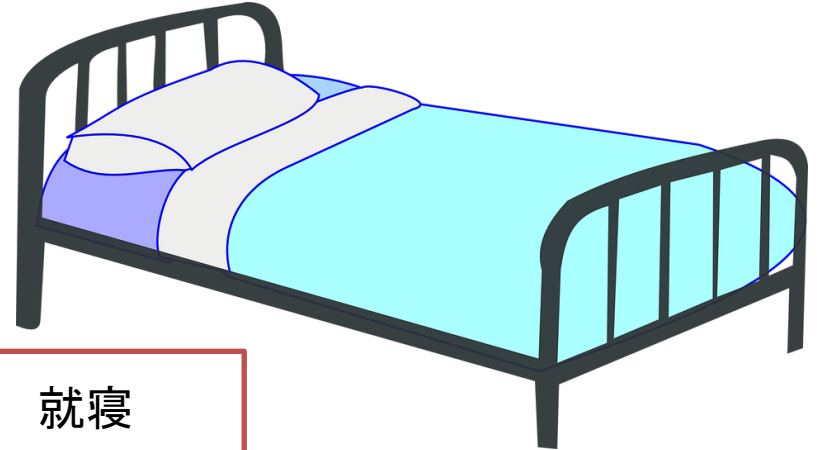
クラウド化により大量のデータを
処理可能となり性能向上



- 機械学習（特にDeepLearningの領域）では画像や自然言語に対する処理が盛ん
- 音声認識の分野も研究
- 「音声認識（文字の書き起こし）」以外の機械学習は注目度が低い

音声認識以外の適用先

- 環境音識別
– ライフログ管理



音声認識以外の適用先

- 危険音認識
 - 子供, 老人見守り



一人暮らしのお年寄りが
倒れた音を家族に連絡



泣き声を検知し両親に連絡

音声認識以外の適用先

- 話者識別



- その他にも

- 音声合成, 機械異常検査, 音楽分類, 感情分析
etc...

- 機械学習の足がかりとして音声プログラムで取り扱う必要
- 機械学習と親和性の高いpythonでの音声の取り扱いをクラウド・テクノロジー活用部会で実践してみました

- ~~声でドアの鍵を開けるシステム~~

まではいかなかったので

- 言っている合言葉があっていて、
話者がそれっぽいかを判定するシステム



1. 音声入力



(共通部分)

2. 特徴量抽出



3. 機械学習

(分野ごとに多様)

```
1. from scikits.talkbox.features import mfcc
2. import sys
3. import numpy as np
4. from scipy.io.wavfile import read
5. import math
6. argvs = sys.argv
```

インポート

```
7. def calc_mfcc(data, fs):
8.     mfcc_data, trush, trush = mfcc(data, nwin=256, nfft=512, fs=fs, nceps=13)
9.     meanceps = np.zeros(mfcc_data[0].size)
10.    for mc in mfcc_data:
11.        meanceps += mc
12.    return meanceps/mfcc_data[0].size
```

特徴量抽出

```
13. def build_model(datas):
14.     mue=np.zeros(len(datas[0]))
15.     sigma=np.zeros(len(datas[0]))
16.     for d in datas:
17.         mue = mue+d
18.         mue = mue/len(datas)
19.     for d in datas:
20.         sigma = sigma+(mue-d)*(mue-d)
21.         sigma = sigma/len(datas)
22.     return mue, sigma
23. def gaussian(x,mue,sigma):
24.     delda = 0.001
25.     return 1/math.sqrt(2*math.pi*sigma+delda)*(math.exp(-1*pow(x-mue,2)/(2*sigma+delda)))
```

機械学習
モデル学習

```

26. def Likelihood_average(ary):
27.     ave=0
28.     for n in ary:
29.         ave+=n
30.     return n/len(ary)
31. def ditect(model_mue,model_sigma,target):
32.     likelihood = []
33.     for i in range(len(target)):
34.         likelihood.append(gaussian(target[i],model_mue[i],model_sigma[i]))
35.     return Likelihood_average(likelihood)
36. def judge(val):
37.     if val > -2:
38.         print "Open"
39.     else:
40.         print "Close"
41. if __name__ == '__main__':
42.     mfccs = []
43.     fnames = ['hirake5.wav','hirake2.wav','hirake3.wav','hirake4.wav']
44.     target = argvs[1]
45.     for fname in fnames:
46.         fs, data = read(fname,'rb')
47.         mfccs.append(calc_mfcc(data, fs))
48.     mue, sigma = build_model(mfccs)
49.     trush, t_data = read(target,'rb')
50.     t_mfcc = calc_mfcc(t_data, fs)
51.     val=math.log10(ditect(mue, sigma, t_mfcc))
52.     judge(val)

```

機械学習
認識部分

入力等
メイン部分

1. モデル構築用ファイル読み込み
2. 特徴量抽出
3. モデル構築
4. 被判定ファイル読み込み
5. 特徴量抽出
6. モデルを使って尤度を算出
7. 尤度から最終判定

今回のデモの内部処理

- 入力 : scipy
- 特徴量抽出 : scikits.talkbox
- 機械学習(自作)

- C言語などではメモリ確保やファイルオープン, 開放など大変
- Pythonの場合は一行で記述可能
 - `scipy.io.read(ファイル名)`

- MFCC(メル周波数ケプストラム係数)
 - 一般的に音声認識に用いられる特徴量
 - 人間の聴覚に基づいた特徴
- 計算手順
 1. FFT
 2. 高域強調
 3. メル軸射影
 4. フィルタバンク処理
 5. 離散コサイン変換

- scikits.talkboxを使うと
- 計算手順

1. `ceps, mspec, spec = scikits.talkbox.feature(
input, nwin=256, nfft=512, fs=fs, nceps=13)`

入力データ

分析パラメータ

出力次元数

- 4つの音声ファイルを元にMFCC算出
 - 13次元それぞれに正規分布に従うモデルを学習
 - 入力音声に対する尤度を計算
-
- ここはまだまだ工夫の余地がある

今回のデモの内部処理

- 入力や前処理はライブラリが行ってくれる
- 難しく、工夫のしがいがある学習部分だけを頑張れば良い
- 一番難しいところ
 - 環境構築: windows上でpythonを行うとライブラリが動かなかったり非常に困難

- 音声認識以外でも音の機械学習の余地が多い
- 音声の特徴量にはMFCCが使用されている
- 実践機械学習を元に声を認識してみた
- 音声入力や特徴量抽出はpythonを使えば簡単