ラズベリー・パイで画像認識

修正事項があるため、修正版のテキストを作成しました。新たな SD カードで授業を行う場合には、修正事項は訂正済みです。その場合は、このテキストの「InceptionV3 を使った画像認識」から作業を行って下さい。

従来からの SD カードで授業を行う場合には、前回のテキストの「4 ディープラーニングのフレームワーク、その他のモジュールのインストール」か完了してから、以下に進んで下さい。

1 InceptionV3 のインストール

inception_v3 のモデルをインストールします。

```
cd ~/Lesson
git clone https://github.com/fchollet/deep-learning-models.git
```

keras のアップデートに伴い、一部修正が必要になっています。inception_v3.py の以下の部分を修正します。

プログラムの修正　inception_v3.py　１５７行目

（修正前）

```
    input_shape = _obtain_input_shape(
        input_shape,
        default_size=299,
        min_size=139,
        data_format=K.image_data_format(),
        include_top=include_top)
```

（修正後）「include_top=」を「require_flatten=」へ修正

```
    input_shape = _obtain_input_shape(
        input_shape,
        default_size=299,
        min_size=139,
        data_format=K.image_data_format(),
        require_flatten=include_top)
```

2 motion の停止

他の授業で motion というサービスがインストールされて、自動的に起動している場合があります。その場合、WEB カメラからの画像取得時にリソースが競合してエラーとなります。それを避けるために、サービスを停止します。

サービスの停止

```
sudo systemctl stop motion
```

再起動時に起動しないようにする

```
sudo systemctl disable motion
```

また、取得された画像や動画でディスク容量があふれる場合があります。画像や動画を削除します。

```
sudo rm /var/lib/motion/*
```

3 InceptionV3 を使った画像認識

inception_v3 のモデルを使って画像認識を行います。

```
cd ~/Lesson/deep-learning-models
wget https://upload.wikimedia.org/wikipedia/commons/6/63/
   African_elephant_warning_raised_trunk.jpg -O elephant.jpg
python inception_v3.py
Using TensorFlow backend.
Predicted: [[('n02504013', 'Indian_elephant', 0.87686646), ('n01871265
   ', 'tusker', 0.044712357), ('n02504458', 'African_elephant',
   0.02874627), ('n02398521', 'hippopotamus', 0.0072720782), ('
   n02092339', 'Weimaraner', 0.0020943223)]]
```

実行すると、学習済みの InceptionV3 のモデルをダウンロードします。デフォルトでは、象の画像の認識を行います。「elephant.jpg」というファイルを用意して、認識させます。画像を解析すると、「ラベル番号・ラベル名・予測値」が多次元配列で返ってきます。

```
wget ＜画像のURL＞ -O elephant.jpg
```

で画像を「elephant.jpg」というファイル名で保存できます。プログラムはデフォルトで、「elephant.jpg」というファイル名の画像を認識するようになっています。他の画像を探して、試してみましょう。

4 OpenCV 画像の解析

OpenCV で読み込んだ画像を認識するように、プログラムを修正します。if __name__ == '__main__以降を以下のように書き換えます。

```
cp inception_v3.py answerthings.py
nano answerthings.py
```

answerthings.py

```
import cv2
if __name__ == '__main__':
    model = InceptionV3(include_top=True, weights='imagenet')
    cam = cv2.VideoCapture(0)
    print("Start")

    while(True):
        ret, frame = cam.read()
        cv2.imshow("Show FLAME Image", frame)

        k = cv2.waitKey(1)
        if k == ord('s'):
            cv2.imwrite("output.png", frame)
            # img_path = 'elephant.jpg'
            img_path = "output.png"
            img = image.load_img(img_path, target_size=(299, 299))
            x = image.img_to_array(img)
            x = np.expand_dims(x, axis=0)

            x = preprocess_input(x)
            preds = model.predict(x)
            recognize = decode_predictions(preds)
            #print('Predicted:', decode_predictions(preds))
            print('Label:', recognize[0][0][1])

        elif k == ord('q'):
            break
    cam.release()
    cv2.destroyAllWindows()
```

cv2.imshow で、ビデオ画像を表示するため、VNC ビューワーで respberry pi に接続します。VNC ビューワーで、ターミナルを開き、以下のコマンドを実行します。

```
source ~/.profile
workon cv
cd ~/Lesson/deep-learning-models
```

```
python answerthings.py
```

これで、「s」キーを押すと、画像を「output.png」に保存し、そのファイルの解析を行います。「q」キーを押すと、プログラムを終了します。上記のキーを押すときは、WEB カメラからの画像が前面に表示されている状態で押して下さい。

5 英語ラベルの発話

画像を認識すると、英語のラベルが返ってきますので、それを発声します。英語の発声には、espeak を利用します。

```
sudo apt-get install espeak
sudo apt-get install pulseaudio
jack_control start
pulseaudio --start
espeak “Hello”
```

subprocess を使って、python で上記の espaek コマンドを実行します。上記のプログラムを以下のように書き換えます。

```
cp answerthings.py speakthings.py
nano speakthings.py
```

speakthings.py

```
1 # linuxコマンドを使うためにsubprocessをインポート
2 import subprocess
3
4 # ４２２行目以降に以下を追記
5 speak = "This is a " + recognize[0][0][1]
6 subprocess.check_output(["espeak", "-k5", "-s150", speak])
```

実際に動くかためして見ましょう。

```
python speakthings.py
```

6 日本語の発話

①英語ラベルを日本語に変換②日本語を発声というプロセスで、日本語の発声をするようにします。

6.1 英語ラベルを日本語に変換

英語ラベルの数が限られているので、変換用の辞書を使って変換します。

```
git clone https://gist.github.com/PonDad/4
    dcb4b242b9358e524b4ddecbee385e9
cp 4dcb4b242b9358e524b4ddecbee385e9/imagenet_class_index.json .
```

imagenet_class_index.json

```
[
    {
        "num": "n01440764",
        "en": "tench",
        "ja": "テンチ"
    },
    {
        "num": "n01443537",
        "en": "goldfish",
        "ja": "金魚"
    },
・・・
]
```

という構造になっています。

mydic.py

```
1  ###mydic.py###
2  import sys
3  import json
4
5  def en_to_ja(en_text):
6      with open('imagenet_class_index.json', 'r') as f:
7          obj = json.load(f)
8          for i in obj:
9              if i['en'] == en_text:
10                 return i['ja']
11     return ""
12
13 if __name__ == '__main__':
14     argvs = sys.argv
15     argc = len(argvs)
16     if (argc != 2):
17         print 'Usage: # python %s english_text' % argvs[0]
18         quit()
19     en_text = argvs[1]
```

```
20      ja_text = en_to_ja(en_text)
21      print(ja_text)
```

```
python mydic.py goldfish
金 魚
```

6.2 日本語を発声

open-jtalk を使って発声します。open-jtalk をインストールします。

```
sudo apt-get install open-jtalk open-jtalk-mecab-naist-jdic hts-voice-
   nitech-jp-atr503-m001
```

発声に使う音声ファイル（.htsvoice）をダウンロードし、抽出します。

```
wget https://sourceforge.net/projects/mmdagent/files/MMDAgent_Example/
   MMDAgent_Example-1.6/MMDAgent_Example-1.6.zip/download -O
   MMDAgent_Example-1.6.zip
unzip MMDAgent_Example-1.6.zip MMDAgent_Example-1.6/Voice/*
```

抽出した音声ファイルを設置します。

```
sudo cp -r MMDAgent_Example-1.6/Voice/mei/ /usr/share/hts-voice
```

jtalk.py

```
1  #coding: utf-8
2  import subprocess
3  from datetime import datetime
4
5  def jtalk(t):
6      open_jtalk=['open_jtalk']
7      mech=['-x','/var/lib/mecab/dic/open-jtalk/naist-jdic']
8      htsvoice=['-m','/usr/share/hts-voice/mei/mei_normal.htsvoice']
9      speed=['-r','1.0']
10     outwav=['-ow','open_jtalk.wav']
11     cmd=open_jtalk+mech+htsvoice+speed+outwav
12     c = subprocess.Popen(cmd,stdin=subprocess.PIPE)
13     c.stdin.write(t)
14     c.stdin.close()
15     c.wait()
16     aplay = ['aplay','-q','open_jtalk.wav']
17     wr = subprocess.Popen(aplay)
```

```
18
19 def say_datetime():
20     d = datetime.now()
21     text = '%s月%s日、%s時%s分%s秒' % (d.month, d.day, d.hour, d.minute
           , d.second)
22     jtalk(text)
23
24 if __name__ == '__main__':
25     say_datetime()
```

```
python jtalk.py
```

python モジュールとして利用する場合

```
import jtalk
jtalk.jtalk(u'日本語を話します')
```

6.3 ２つをまとめる

見たものの英語ラベルを日本語へ変換し、それを日本語で発声します。speakthings.py を speakzou.py にコピーして、上記の２つをまとめて以下の通りに修正します。mydic.py と jtalk.py はインポートするので、speakzou.py と同じディレクトリにおきます。

```
cp speakthings.py speakzou.py
nano speakzou.py
```

speakzou.py

```
1  import subprocess
2  import json
3  import sys
4  import jtalk
5  import mydic
6
7  if __name__ == '__main__':
8      model = InceptionV3(include_top=True, weights='imagenet')
9      cam = cv2.VideoCapture(0)
10     print("Start")
11
12     while(True):
13         ret, frame = cam.read()
14         cv2.imshow("Show FLAME Image", frame)
```

```
15
16          k = cv2.waitKey(1)
17          if k == ord('s'):
18              cv2.imwrite("output.png", frame)
19              # img_path = 'elephant.jpg'
20              img_path = "output.png"
21              img = image.load_img(img_path, target_size=(299, 299))
22              x = image.img_to_array(img)
23              x = np.expand_dims(x, axis=0)
24
25              x = preprocess_input(x)
26              preds = model.predict(x)
27              recognize = decode_predictions(preds)
28              recognize_rabel = recognize[0][0][1]
29              recognize_rabel_ja = mydic.en_to_ja(recognize_rabel)
30              ja_text_to_speak = u'これは' + recognize_rabel_ja + u'だよ'
31              jtalk.jtalk(ja_text_to_speak.encode('utf-8'))
32
33          elif k == ord('q'):
34              break
35      cam.release()
36      cv2.destroyAllWindows()
```

VNC ビューワーの中のターミナルから、

```
python speakzu.py
```

として、実行してみましょう。