卒業論文

人物再識別システムの試作と評価

提出者　　飯塚　敦志

提出年月日　平成 31 年 1 月 25 日

指導教員　　金子邦彦　教授

人物再識別システムの試作と評価

情報工学科　飯塚敦志

研究概要

私たちの身の回りでは人物認証やカメラの撮影といった様々な場面で人工知能を利用した人物の識別システムは多く存在している. その中でカメラを使用した人物の再識別について取り組んだ. 本稿では, 複数カメラから動画撮影を行い, 取得した顔画像および関連情報のデータベースにおいての人工知能を用いた人物識別の試みについて報告する.

目次

1. まえがき

現在，人物の識別は多くの場面で利用されている．顔認証での利用にはスマートフォンのロックを解除する機能(Face ID)や，コンビニ，空港などの場所で顔認証による支払いや搭乗許可を行うシステムが導入されている．認証以外には，カメラで人物を撮影するときに顔部分を表示，監視システムなどで利用されている．私はそれらの技術を利用した人物の行動について研究に取り組んだ．行動に着目した理由は，人の行動情報は多くの活用ができると考えられるからである．観光地や似通った建物に行くとき，交通量や移動経路，移動時間などを人の行動情報として取得することでそれを元に人の行動予測を立てることが考えられる．行動予測は新しく店舗を新設するとき(マーケティング)，渋滞の起こりやすい時間帯，時間帯別で男性あるいは学生の方で交通量が変わる，というように様々な活用が考えらえる．人工知能を利用した人の行動予測をするシステムはすでに存在しており，防犯や来場者の人数予測などで利用されている．

しかし，複数のセンサー類から収集した画像や情報をデータベースに保管して，任意の画像と情報を自由に操作閲覧できるシステムは簡単ではない．本研究では，人物再識別における画像やその他の情報のデータベース試作を行い，いかに簡単にデータベース内のデータが管理できるかの評価を試みた．人物再識別とは，画像から人物識別を行うことである．画像には静止画と動画の 2 種類がある．識別された人物同士の照合を行い，同一人物の判別を行う．判別を行うための人物の特徴取得には，顔や衣類，身体的特徴(身長)などが使用されるが本研究ではカメラから撮影したファイルから顔部分を利用した人物再識別を行う．

2. 人物再識別システム

人物再識別システムは，カメラからの顔識別，取得した画像と情報を記録するデータベースの2つの機能から成り立っている．

観光地や駅など行動予測に利用できそうな場所にカメラを設置し，撮影を行い，撮影した動画から顔検知を行う．顔検知の結果として，顔の座標情報が得られる．これで，元画像から顔領域部分の画像である顔画像が抽出できる．現在，我々が実装した人物再識別システムでは， 顔画像がデータベースに保存されるとともに，ファイル名，撮影を行った日時の情報もデータベースに保存される．その他，顔画像から算出された特徴量や各種属性

(性別の推定値など)も保存できるようにデータベース設計している.

2.1 顔識別

顔識別には, Python 開発環境の Anaconda をインストールして, Python 環境化で Python パッケージ, GitHub, Dlib を利用した.

2.1.1 顔識別, 人物再識別にて使用した技術

Anaconda…python パッケージなどを提供, インストールすることで利用できるプラットフォームである. 多くのモジュールやツールのコンパイル済みバイナリファイルを提供している.

Spyder…オープンソースでクロスプラットフォーム Python 用統合開発環境である. Spyder には, Numpy, Scipy, Matplotlib, IPython など開発の手助けを行う機能が統合されている. 開発環境を変えて, 開発を行える.

Dlib…汎用目的のクロスプラットフォームソフトウェアライブラリである. 顔識別や機械学習のライブラリである.

GitHub…ソフトウェア開発のプラットフォームであり, ソースコードをホスティングするオンラインのサイトである. ホスティングをすることで複数人のソフトウェア開発者と協働してコードのレビュー, プロジェクトを管理しながら開発を行うことができる.

OpenCV…オープンソースのコンピュータビジョン向けのライブラリである. 画像処理や画像認識, 機械学習等の機能を持つ C/C++, Python, Java, MATLAB 用ライブラリ.

以下, 顔識別を行う手順を記載する.

手順① Anaconda, Dlib ソフトウェアなどの Python 開発環境の準備.

手順② GitHub のソースコードを利用し, 事前にカメラで撮影しておいた動画ファイルを静止画像へ変換.

手順③ 取得した画像に Dlib ソフトウェアを利用して顔検知をして, 顔の識別を行う.

手順④ 顔識別された画像から人物再識別を行う. 再識別には Dlib ソフトウェア付属の Ageitgey/face_recognition を使用する. これについては 3.実験で説明を行う. これらで使用された画像や画像情報はデータベースに保管される.

2.2 データベース

画像や情報などのデータを保存するデータベースとして Google 社の Firebase を利用した. Firebase には, 多くの機能があり, その中の Cloud Storage と Realtime Database の機能を利用した.

Cloud Storage はユーザーが生成した写真や動画などのコンテンツをアップロードまたはダウンロードすることができる機能を持つ. Realtime Database は NoSQL クラウドデータベースでデータの保管と同期を行うことができ, データはすべてのクライアントにわたってリアルタイムで同期される.

Firebase で行う手順は以下の通りである.

手順① Cloud Storage への画像アップロードに Python パッケージの thisbejim/pyrebase を使用する.

手順② Realtime Database のデータ保管に Python パッケージの python-firebase を使用する.

3. 実験

3.1 撮影した動画を静止画像に画像処理

人物の識別では, 前処理として, カメラで撮影した動画を複数の静止画像に変換する処理を行った. GitHub で公開されているソースコードを使用して, Python-opencv で指定フレームごとに画像として保存した.

```
Video to frames...
Save ../data/images/target/img_000001.png
Save ../data/images/target/img_000002.png
Save ../data/images/target/img_000003.png
Save ../data/images/target/img_000004.png
Save ../data/images/target/img_000005.png
Save ../data/images/target/img_000006.png
Save ../data/images/target/img_000007.png
Save ../data/images/target/img_000008.png
Save ../data/images/target/img_000009.png
Save ../data/images/target/img_000010.png
Save ../data/images/target/img_000011.png
Save ../data/images/target/img_000012.png
Save ../data/images/target/img_000013.png
Save ../data/images/target/img_000014.png
```

図 1 指定フレームごとに画像を分割

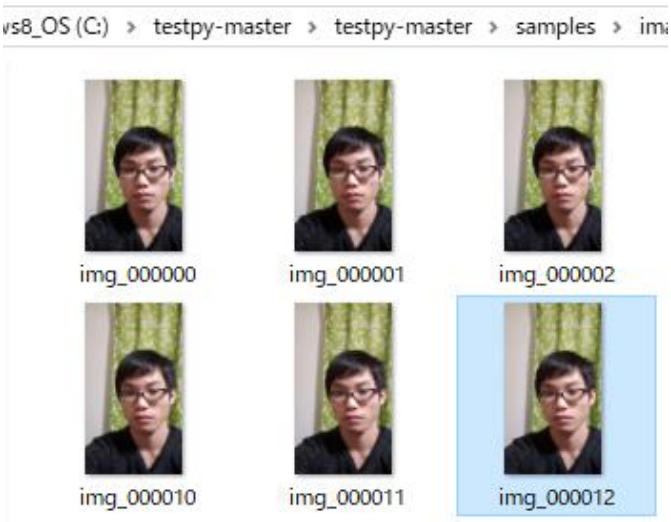

図 2 結果

3.2 Dlib ソフトウェアを使用した顔検知

3.1 で作成した画像で顔検知を行い, 顔部分の識別を行う. 結果として顔の位置情報である座標を取得する.

顔検知には Dlib の顔検知機能を利用する. 画像のイメージピラミッドを作り, 固定サイズのスライディングウィンドウ識別器を使用し, 顔検出を行う. イメージピラミッドから HOG 特徴量を抽出して, HOG ピラミッドを作成する. HOG とは局所領域における輝度の勾配方向の分布から特徴量の取得. HOG ピラミッドで線形識別機を移動させる. Dlib 付属の学習済みデータ mmod_human_face_detector を使用して実験を行った.

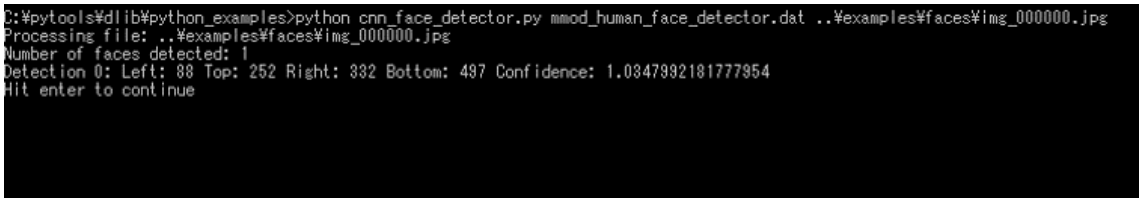

図 3 プログラムの実行と座標取得画面

実行結果から顔検知と顔情報(座標)の取得が行えている. 実行結果より,

Number of faces detected: 1…1 人の顔検知が行えている.

Detection1…Left, Top, Right, Bottom から顔領域の座標取得を行

えている.

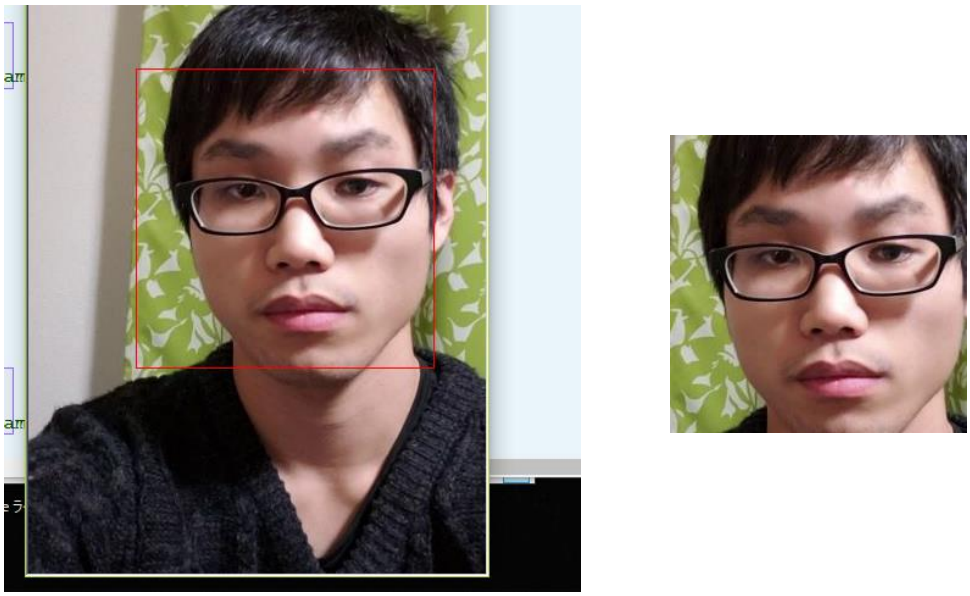

図 4 顔検知, 顔部分の抽出

1枚の画像から複数の顔検知を行うことを目的としているため, その実験と結果を図5に示す.

図 5 複数人写っている場合の顔識別と結果

実行結果から 3 人分の顔検知と顔情報(座標)の取得が行えている. 実行結果より,

Number of faces detected: 3…3 人の顔検知を行えている

Detection0~3…3 人の顔座標の取得が行えている.

3.3 人物再識別

スマートフォンを使用して, 研究室のメンバー4 名の各自の顔写真を撮影した. 撮影した環境(場所やスマートフォンの機体)は別々で行い, 研究室や各自の家で撮影日時を変えて, 撮影した. 人物の顔識別には Dlib の顔識別ライブラリを利用したソフトウェアである Ageitgey/face_recognition を用いて実験を行う. 以下, メンバーそれぞれを a, b, c, d と区別する. 各自の顔として識別されている画像を 1 枚ずつ計 4 枚選出し, 残りの画像(91 枚)との照合を行う. 使用した画像と実験結果を図 6～図 10 で示す.

先に選出した画像から顔識別をされず, 他の顔画像との識別が行えなかった例と原因を図 6 に挙げておく.

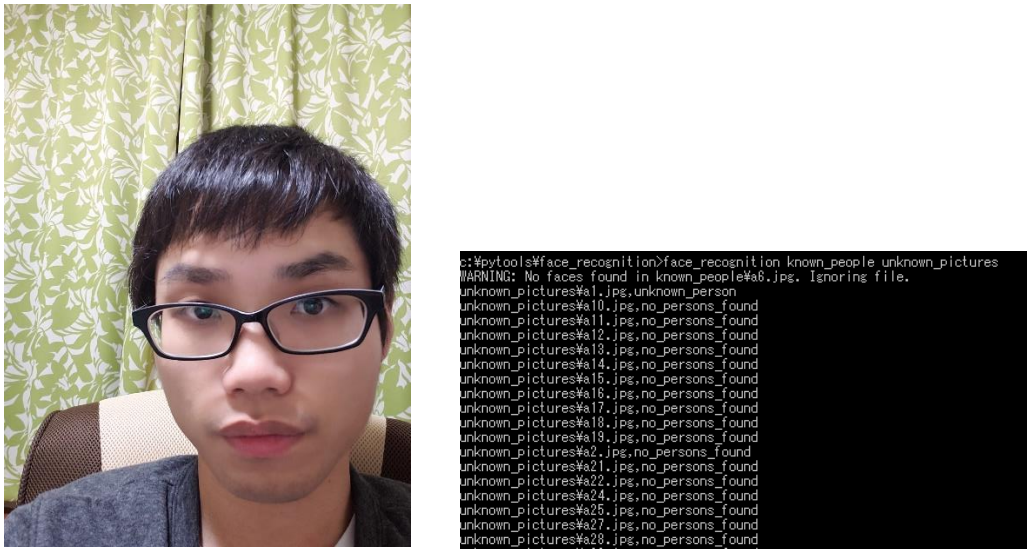

図 6 a から選出した画像と結果

結果としては, 選出した画像が顔識別されておらず, 他の画像との照合が行えなかった. 原因は端末による画像サイズの大きさが問題であった. よって, 全ての画像サイズをほぼ幅 480, 高さ 620pixel に統一して再度実験を行っ

た. 選出した画像と実行結果を図 7～図 10 に示す.

図 7　a から選出した画像と結果

結果, a の同一人物として識別された画像は 27 枚中 24 枚. 他の 3 名と誤って識別された画像は 64 枚中 0 枚. 同一人物として識別されなかった画像は 27 枚中 3 枚. 顔識別されなかった画像は 27 枚中 1 枚であった.

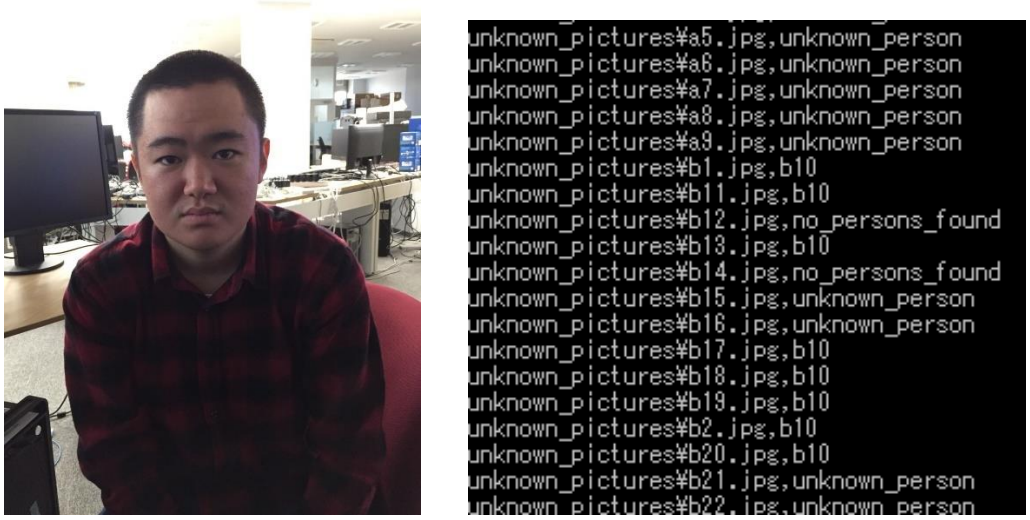

図 8　b から選出した画像と結果

結果, b の同一人物として識別された画像は 22 枚中 12 枚. 他の 3 名と誤って識別された画像は 69 枚中 1 枚. 同一人物として識別されなかった画像は 22 枚中 7 枚. 顔識別されなかった画像は 22 枚中 3 枚であった.

```
unknown_pictures¥b5.jpg,unknown_person
unknown_pictures¥b6.jpg,no_persons_found
unknown_pictures¥b7.jpg,unknown_person
unknown_pictures¥b8.jpg,unknown_person
unknown_pictures¥b9.jpg,unknown_person
unknown_pictures¥c1.jpg,unknown_person
unknown_pictures¥c10.jpg,c16
unknown_pictures¥c11.jpg,c16
unknown_pictures¥c12.jpg,unknown_person
unknown_pictures¥c13.jpg,unknown_person
unknown_pictures¥c14.jpg,c16
unknown_pictures¥c15.jpg,unknown_person
unknown_pictures¥c17.jpg,c16
unknown_pictures¥c18.jpg,c16
unknown_pictures¥c19.jpg,unknown_person
unknown_pictures¥c2.jpg,unknown_person
unknown_pictures¥c20.jpg,c16
unknown_pictures¥c21.jpg,unknown_person
unknown_pictures¥c22.jpg,c16
unknown_pictures¥c23.jpg,c16
unknown_pictures¥c24.jpg,unknown_person
unknown_pictures¥c25.jpg,unknown_person
```

図 9　c から選出した画像と結果

結果，c の同一人物として識別された画像は 28 枚中 16 枚．他の 3 名と誤って識別された画像は 63 枚中 0 枚．同一人物として識別されなかった画像は 28 枚中 12 枚．顔識別されなかった画像は 28 枚中 0 枚であった．

```
unknown_pictures¥c25.jpg,unknown_person
unknown_pictures¥c26.jpg,unknown_person
unknown_pictures¥c27.jpg,unknown_person
unknown_pictures¥c28.jpg,unknown_person
unknown_pictures¥c29.jpg,unknown_person
unknown_pictures¥c3.jpg,unknown_person
unknown_pictures¥c4.jpg,unknown_person
unknown_pictures¥c5.jpg,unknown_person
unknown_pictures¥c6.jpg,unknown_person
unknown_pictures¥c7.jpg,unknown_person
unknown_pictures¥c8.jpg,unknown_person
unknown_pictures¥c9.jpg,unknown_person
unknown_pictures¥d1.jpg,no_persons_found
unknown_pictures¥d10.jpg,d11
unknown_pictures¥d12.jpg,no_persons_found
unknown_pictures¥d2.jpg,no_persons_found
unknown_pictures¥d3.jpg,no_persons_found
unknown_pictures¥d4.jpg,d11
unknown_pictures¥d5.jpg,d11
unknown_pictures¥d6.jpg,no_persons_found
unknown_pictures¥d7.jpg,d11
unknown_pictures¥d8.jpg,d11
unknown_pictures¥d9.jpg,d11
```

図 10　d から選出した画像と結果

結果，d の同一人物として識別された画像は 11 枚中 6 枚．他の 3 名と誤って識別された画像は 79 枚中 0 枚．同一人物として識別されなかった画像は 11 枚中 0 枚．顔識別されなかった画像は 5 枚であった．

人物再識別の実験結果から，人物再識別は良好であったことが確認できた．顔検知がされない原因としては，顔がしっかりと画像に写っていないこと(正面以外の方向に顔を向けている，マスクをつけているなど)が問題にあると画像から確認できた．

3.4 データベース

Google 社の Firebase を用いて，Cloud Storage 機能と Realtime Database 機能を利用した実験を行う．

Cloud Storage は実験で使用した画像のアップロードを行う．アップロード方法には Firebase の Python ライブラリ Pyrebase を利用する．Pyrebase を利用するにあたり，必要な準備を次に記載する．

・Python 仮想環境の作成，今回 Python のバージョンを 2.7 とする．

・Pyrebase のインストールとその他必要なパッケージをインストールする．

・Firebase の Web ページから新規プロジェクトを作成し，データベースを作成．プロジェクトの設定から WebAPI キーとプロジェクト ID を取得．

・作成した仮想環境で Firebase との認証を行う．

実験結果を図 11 に示す．

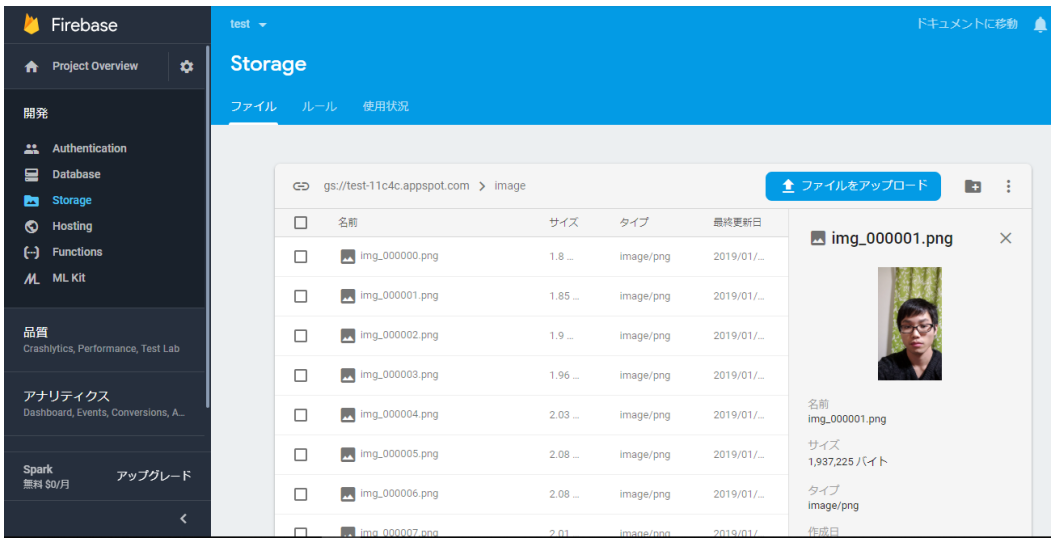

図 11　Cloud Storage で画像の保管

Realtime Database は実験で使用した画像から取得できるデータの保存と保存したデータ取得を行う．今回は，ファイル名，形式，撮影日時，場所(仮)のデータを保存した．実験結果を図 12 と図 13 に示す．

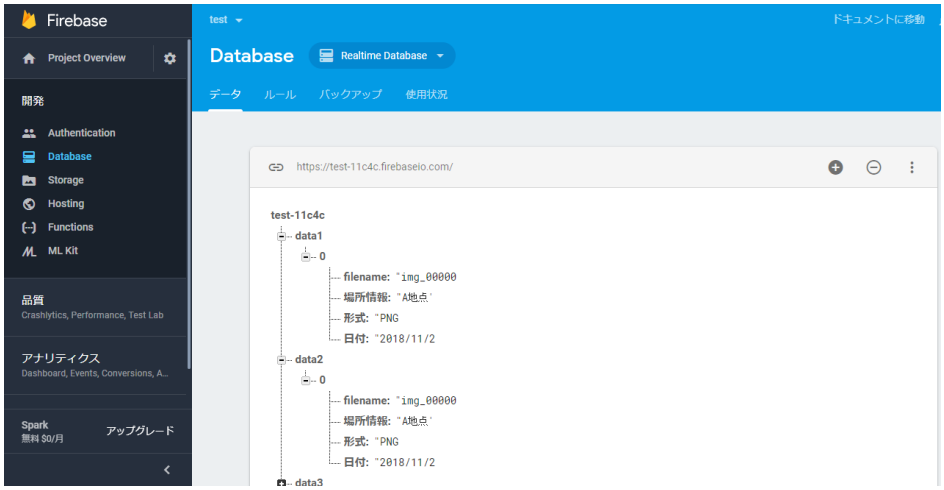

図 12　Realtime Database でデータの保存

```
In [4]: runfile('C:/Users/User/imgdata-update.py', wdir='C:/Users/User')
{'場所情報': 'A地点', 'filename': 'img_000002', '日付': '2018/11/21', '形式': 'PNG'}
```

図 13　Realtime Database からデータの取得

4. むすび

近年進歩している人工知能を用いた人物再識別システムと各種情報を管理するデータベースシステムを試作した．データベースシステムでは，データのアップロードとダウンロードが容易に行うことができた．Python の開発環境を変えて実験を行うことがあり，1 つに統合することができればさらに容易にデータの受け渡しを行えると考えている．また，手作業で行う部分の自動化も課題である．人物再識別システムでは，顔検知，識別の面でも良好な結果が得られた．

これからは，画像，カメラ，人物再識別から様々な特徴や情報を収集する手段が増えれば，前に述べた活用(行動予測など)をできる見通しが今回の研究から得られた．

謝辞

本研究の実施にあたり，卒業論文指導教員の情報工学科・金子邦彦教授にご指導賜りました．金子邦彦研究室の井上氏，田坂氏，半田氏には，実験の協力，研究室や実験の場での議論等を通して，知識や示唆の提供をいただきました．ここに感謝の意を表します．

参考文献

(1) GitHub - davisking_dlib_ A toolkit for making real world machine learning and data analysis applications in C++

https://github.com/davisking/dlib

(2) GitHub - ageitgey_face_recognition_ The world's simplest facial recognition api for Python and the command line

https://github.com/ageitgey/face_recognition

(3) testpy_samples_image_clustering at master · iShoto_testpy · GitHub

https://github.com/iShoto/testpy/tree/master/samples/image_clustering

(4) GitHub - thisbejim_Pyrebase_ A simple python wrapper for the Firebase API.

https://github.com/thisbejim/Pyrebase

(5) python-firebase · PyPI

https://pypi.org/project/python-firebase/