

# 顔認証システムの構築

山口大輝

## 研究動機

新しいUIを作りたい

→手を使わなく手も操作可能なもの

当初、アイトラッキングを使用したものを作ろうと計画

顔を使用したシステムを作り、基礎研究とし、ハードルを下げる

顔認識、顔認証システムに注目した

## 特徴量抽出AIについて

《ディープ・ニューラルネットワークの構築》

N人それぞれ複数枚の顔画像を学習させる

それぞれの人物のみ反応するN人分のニューロンを出力層に用意する

推論時は出力層の一つ前の層に属するニューロンの出力値が特徴量として抽出される

## 実験

文字認識AIの作成

言語: python3.11

ライブラリ: OpenCV, Chainer

サンプルデータ: MNIST

```
def main():
    model = MyMLP()
    serializer = Serializer(model, model_path="update/model_main/", strict=False)
    capture = cv2.VideoCapture(0)
    if not capture.isOpened():
        raise Exception("no error")
    while True:
        ret, image = capture.read()
        if not ret:
            continue
        cv2.imshow("image", image)
        cv2.waitKey(1)
        key = cv2.waitKey(1)
        if key == ord('e'):
            img = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
            num = model(img)
            print(num.data)
            if key == ord('q'):
                cv2.destroyAllWindows()
                break
    cv2.destroyAllWindows()
    return
```

- ・カメラの映像を取り込む。
- ・画像内にデータ化して格納する範囲を赤枠で表示する
- ・特定のキー（今回はe）を押したときに画像を認識しやすいように変換する。
- ・MyMLPで作成したmodelで画像の認識をする。
- ・num.dataで0～9のそれぞれに対する認識結果、np.argmax(num.data)で最も確率の高い数字を表示する

## 実験

顔認証AIの作成

言語: python3.11

ライブラリ: OpenCV, Chainer, dlib

## 今後の展望

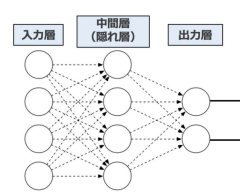
類似度から判別するAIの作成をする

顔認証AIを完成させ、精度をあげる

## 研究目的

- ・顔認証システムを構築し、PCのセキュリティを強固なものにする
- ・画像処理について学ぶ

## ニューラルネットワーク



- ・複数のデータからクラスを分類するもの
- ・出力したデータをターゲットと一致させるためのシナプス強度を調整する

```
class MyMLP(nn.Module):
    def __init__(self, n_in=784, n_hidden=100, n_out=10):
        super(MyMLP, self).__init__()
        self.linear1 = nn.Linear(n_in, n_hidden)
        self.linear2 = nn.Linear(n_hidden, n_out)

    def forward(self, x):
        h = F.relu(self.linear1(x))
        y = self.linear2(h)
        return y

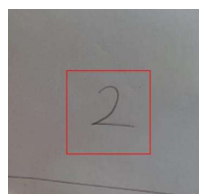
def main():
    pass
```

- MyMLPクラスで学習モデルを設定する
- ・Chainer.links.Linear関数で層を線形結合する。
- 今回は入力層、中間層×2、出力層の4層
- ・F.relu関数で入力xに対する出力yを作成する

```
optimizer = optim.Adam()
for epoch in range(100):
    print(epoch)
    for i in range(100):
        x = Variable(torch.randn(1, 1, 28, 28))
        y = model(x)
        loss = F.softmax_cross_entropy(y, target)
        loss.backward()
        optimizer.update()
```

- Optimizerを最適化する
- ・loss.backward関数で勾配を計算する
- ・F.softmax\_cross\_entropy関数で正解ラベルと予測との損失値を計算する
- ・loss.backward関数で逆伝播計算をする
- ・optimizer.update関数で最適化する

## 結果



↑認識中の映像

```
[[ 0.83724512  0.06471065  0.14416175  0.01819398  0.08380161 -0.06467605
  0.00047821  0.00828181 -0.03458825  0.11753891]]
2
```

文字を認識して結果を表示することができたしかし、誤った結果を出すこともあった

改善するには、学習するデータ量を増やすまたは学習モデルの中間層を増やす

```
import cv2
import dlib

detector = dlib.get_frontal_face_detector()
predictor = dlib.shape_predictor("shape_predictor_68_face_landmarks.dat")

image = cv2.imread("face.jpg")
faces = detector(image)

for face in faces:
    landmarks = predictor(image, face)
    for i in range(68):
        x, y = landmarks.part(i).x, landmarks.part(i).y
        cv2.circle(image, (x, y), 2, (0, 0, 255), -1)
    cv2.imshow("Face Landmarks", image)
    cv2.waitKey(0)
    cv2.destroyAllWindows()
```

- ・68個の顔のランドマークを検出し、それぞれの座標に円を描画する。
- ・ランドマークの座標を収集して特徴量ベクトルとして使用できるようにする。

## 参考文献・引用文献

※1 NEC."顔認証とは:顔認証".NEC.2023-03-10.<https://jpn.nec.com/biometrics/face/about.html>

※2 日本コンピュータビジョン."顔認識システムとは?仕組みやメリットから活用シーンまで詳しく解説".日本コンピュータビジョン株式会社.2023-03-10. <https://www.japancv.co.jp/column/2920/>

・※3 中野淳「日経ソフトウェア2021年5月号」日経BPマーケティング発行

・今岡仁 「顔認証の教科書」株式会社プレジデント社発行 2021/12/1

・北山洋幸 「pythonではじめるopencv4プログラミング」株式会社カットシステム社発行 2019/3/20