

知能情報実験 III（データマイニング班）  
指紋認証を用いた CNN の手法についての模索

185710A 金城海斗

185714C 石橋竜弥

185745C 上間翔

185752F 新垣裕二

185763B 草薙幸菜

提出日：2021 年 2 月 x 日

## 目次

1	はじめに	2
1.1	Convolutional Neural Network:CNN	2
1.2	テーマ指紋認証とは	3
2	実験方法	3
2.1	実験目的	3
2.2	データセット構築	3
2.3	モデル選定	4
2.4	パラメータ調整	4
3	実験結果	4
3.1	エポック数を 1 から 30 までに変化	4
3.2	epoch を 20 に固定し、batch_size を変更	4
3.3	活性化関数に変更	4
4	考察	6
5	意図していた実験計画との違い	6
6	まとめ	7

# 1 はじめに

データマイニングとはデータの中に埋め込まれている有用な知識を発掘することである。別の言い方では、データマイニングは、より良い意思決定をするために履歴データをうまく使って一般的な規則性を発見しようとする研究分野である。今回私たちのグループ3では、機会学習の基本的な考え方を実装、体験を通して学んだ。そしてその応用として、既存に存在する指紋データを使った指紋認証のプログラムの分析から、畳み込みニューラルネットワークの手法についての模索、改善を行い、結果の精度向上を目指し、考察したことについて発表する。

## 1.1 Convolutional Neural Network:CNN

Convolutional Neural Network（これより CNN と呼ぶ）は畳み込みニューラルネットワークという意味であり、機械学習で画像の深層学習といえば CNN であるというほどよく使われている識別手法である。これは、ニューラルネットワークに畳み込みという操作を導入したものである。CNN について、簡単な手順を記述する。まず手順1として、画像から特徴を抽出する。フィルタを使い、入力層データの中で位置を変えながらスキャンした部分のデータと、フィルタ自身の持つデータとの差異を畳み込みの結果として畳み込みそうに書き込んだものを特徴量といい、入力層の全データをスキャンしてできた畳み込み結果の値の集まりを特徴マップという。複数のフィルタを用意することで、入力層のデータ特徴を捉えやすくしている。下の図は用意した複数フィルタのうち、一つが完全に入力層のデータの一部と同じであることを示す。

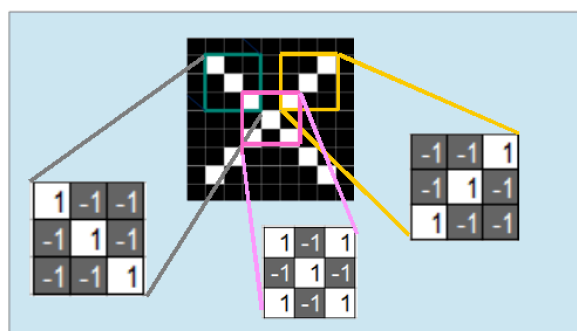


図1 CNN 解説手順1

手順2として、画像を畳み込みする。入力層のデータをフィルタのデータとピクセル毎に比較することで、畳み込み層にその類似度（特徴量）を書き込む。下記の図はフィルタを利用して特徴量を抽出し、特徴マップを作成した例である。

手順3として、画像をプーリングする。畳み込みの層の情報はプーリング層で集約する。出力に関しては、プーリング層のユニット全てと全結合し、計算結果を利用して、フィルタ、重み、バイアスを更新していく。

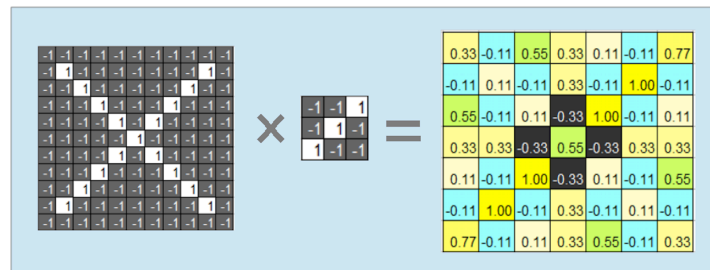


図 2 CNN 解説手順 2

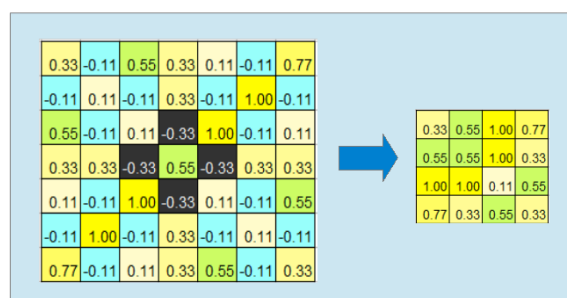


図 3 CNN 解説手順 3

## 1.2 テーマ指紋認証とは

本グループでは、授業の中でデータマイニングについて学び、それらの応用実験として、画像認識について分析しようと考えた。そして、データマイニングを行う識別手として CNN に目をつけ、データの分類ラベルがはっきりしており、複雑である指紋認証の分析、精度改善を行うこととなった。CNN は畳み込み処理を利用したニューラルネットワークであり、どのくらい畳み込み処理を行うのか、どのくらいニューラルネットワークを深くするのは定義されていない。

## 2 実験方法

### 2.1 実験目的

実験を通して明らかにしたいこと、確認したいこと、検証したいことを述べよう。

### 2.2 データセット構築

Kaggle より、指紋のデータセットを利用している。<https://www.kaggle.com/ruizgara/socofing>

## 2.3 モデル選定

アルゴリズムは序章で述べたとおり、CNN を利用しており、以下のコードを参考にしている。  
<https://www.kaggle.com/brianzz/subjectid-finger-cnnrecognizer>  
本アルゴリズムの運用において、指紋認証の正答率が十分に高い点、可読性に優れており改善手法を模索しやすい点に特に優れているため使用することにした。

## 2.4 パラメータ調整

CNN のパラメータ調整では、epoch 数・画像データ数・batchsize や活性化関数である LeakyReLU を変更していった。

# 3 実験結果

実験結果として、以下の結果が得られた。

## 3.1 エポック数を 1 から 30 までに変化

epoch	subjectID accarcy	fingerNum accarcy
1	01.5333333052694798 %	63.01666498184204 %
3	61.283332109451294 %	89.48333263397217 %
5	96.35000228881836 %	98.15000295639038 %
10	99.71666932106018 %	99.71666932106018 %
20	99.73333477973938 %	99.88333582878113 %
30	99.73333477973938 %	99.90000128746033 %

## 3.2 epoch を 20 に固定し、batch\_size を変更

batch_size	subjectID accuracy	fingerNum accuracy
32	99.73333477973938 %	99.90000128746033 %
64	99.73333477973938 %	99.88333582878113 %
128	99.71666932106018 %	99.88333582878113 %

## 3.3 活性化関数に変更

活性化関数	subjectID accracy	fingerNum accracy
LeakyReLU (alpha=-0.5)	99.43333268165588 %	99.88333582878113 %
LeakyReLU (alpha=0.3)	99.71666932106018 %	99.73333477973938 %
LeakyReLU (alpha=0.5)	99.6999979019165 %	99.6333360671997 %
sigmoid	99.73333477973938 %	99.86666440963745 %
tanh	99.73333477973938 %	99.88333582878113 %

## 4 考察

初めに、元のサンプルコードの条件はエポック数 20 のバッチサイズ 64、活性化関数は中間層で ReLU 関数を用いて、出力層に対して softmax 関数を用いている。結論から言うと、どの条件下においても大きな変化は見られなかった。精度を向上させることができた条件は、エポック数を 20 から 30 に増やすこと、バッチサイズを 64 から 32 に変更することの 2 つであった。また、両者とも精度が上がったのは fingerNum accuracy、つまりどの指であるかの識別するものであった。この原因と、その他の条件において精度が上昇しなかった原因について考える。

初めに、fingerNum accuracy のみ精度が向上したことについて考える。まず、エポック数とは学習する世代のことであり、これを多くしていくことで細かい識別が可能になる。直感的に考えれば、10 本のうちどの指かを識別するよりも、600 人のうち誰の指紋であるかを識別する方が難しいはずであるが、このことから fingerNum accuracy の方が細かい識別が必要になると考える。

次に、活性化関数を変更した場合について考える。今回は画像の識別であるため、活性化関数も多くの値を表現できるものが良いと考えた。0 と 1 だけで識別するのと、1 から 100 までは用いて識別するのは後者の方がより細かいものを表現できるはずである。実際、元のサンプルコードは正の値はそのまま使用する ReLU 関数を用いていた。そこで、負の値も一定の割合で使用する LeakyReLU 関数を用いたが結果は精度がやや落ちる程度だった。また、sigmoid 関数と tanh 関数も用いた。これらは入力された値を、0 から 1、-1 から 1 の実数にする関数である。これまでの考えだと精度は下がると予測したが、LeakyReLU 関数を用いた結果とあまり大差はなかった。これらのことを見ると活性化関数に精度との相関が見られない。活性化関数の他に精度をあげている要因があると考えられる。今回の実験では最適化関数について変更を行っていたため、最適化関数に原因があるのではないかと考える。

以上より、エポック数による精度の向上はやや見られたが、活性化関数が識別にどのような効果をもたらしているのかは読み取ることができなかった。

hoge hoge	subjectID accuracy	fingerNum accuracy
hoge	hoge %	hoge %
hoge	hoge %	hoge %

## 5 意図していた実験計画との違い

当初予定していた実験計画は以下の通りとなる。

- 6 週目 (11/17) : テーマ決め、データセット探し、あと色々
- 7 週目 (11/24) : 特徴ベクトルの抽出とかコード探して理解したりとか
- 8 週目 (12/1) : 画像認識をコードに落とし込む

- 9 週目 (12/8) : 実験開始? 出来上がったコードを動かす段階?
- 10 週目 (12/15) : コードや特徴ベクトルの調整 1
- 11 週目 (12/22) :
- 12 週目 (1/5) :
- 13 週目 (1/12) : 改善実験?
- 14 週目 (1/19) :
- 15 週目 (1/26) : レポート・プレゼン資料作成
- 期末テスト日 (2/2) : 最終発表

上記の予定は 11/17 日に作成されたもの。これ以降に追加された締め切り予定に、

- 15 週目 (1/26) : レポート初期版の提出日
- (2/16) : Github 公開

がある。

実際の進捗の経過をまとめると、下記の通りとなる。

- 6 週目 (11/17) : テーマ決定 (画像認証/fingerprint)・データセット探し、画像認証に対しての学習
- 7 週目 (11/24) : 指紋の特徴量の分析、CNN の画像認証コードの検索
- 8 週目 (12/1) : 参考する CNN コードの決定、コードの内容の分析・実行
- 9 週目 (12/8) : コードの内容の分析・実行 2nd
- 10 週目 (12/15) : コードの内容の分析 3rd・amane による実行開始・実験目的の草案
- 11 週目 (12/22) : コードの内容の分析 4th・実行 2nd。また amane で画像データの出力は難しいと判断。来年度に向けたの引継ぎ
- 12 週目 (1/5) : コードの内容の分析 5th・epoch 数を変えた大規模実験・Python/Keras 環境の統一
- 13 週目 (1/12) : コードの内容の分析 6th・batch\_size や LeakyReLU を変更した実験
- 14 週目 (1/19) : レポートの作成・amane による実験
- 15 週目 (1/26) :
- 期末テスト日 (2/2) :

## 6 まとめ

データマイニング班の達成目標を振り返り、選んだテーマに対する機械学習の適用を通して得られた知見や学んだことをまとめよう。また今後やるべきことや後進に伝えたいこと等あれば自由に述べよう。



## 参考文献

- [1] CNN 解説,  
<https://udemy.benesse.co.jp/data-science/ai/convolution-neural-network.html>, 2021/01/026.
- [2] 指紋データセット, <https://www.kaggle.com/ruizgara/socofing>
- [3] アルゴリズム, <https://www.kaggle.com/ruizgara/socofing>
- [4]