令和２年度　特別研究報告書

CNNにおける未学習データ識別に関する検討

龍谷大学　理工学部　情報メディア学科

T170545　山田篤史

指導教員　三好力　教授

内容梗概

CNN を用いた画像識別において、入力された画像が、CNN が学習していない画像だった場合、CNN はその画像について学習していないことに気づけず、そのまま自分が学習したもので最も近い答えを出力する。本研究では、CNN に入力された画像が学習していない画像だった場合、その画像の特徴から、学習していない画像であることに気づき、識別が行えるように、学習した画像と学習していない画像それぞれの特徴を比較し、識別を可能にする基準値についての検討を行う。

目次

第1章 はじめに

1.1 研究背景

昨今、機械学習の発展により複雑な画像の識別が可能になっている。しかし主なニューラルネットワークでは学習していないデータについて妥当な識別結果を得ること、例えば、鳥や魚などを撮影した画像の識別を、機械学習を用いて行う場合、学習データがある既知の種類は正しく識別可能だが、学習データに含まれない種類についても既知の種類のいずれかとして識別してしまう。このことは自然画像中の対象の識別・分類にとって不都合である。そこで本研究では、意図的に学習させた画像と学習させなかった画像をニューラルネットワークに入力し特徴を抽出して比較することで、入力画像が学習した画像か学習していない画像かの識別をニューラルネットワークができる可能性を検討することを目的に実験を行った。

第2章 既存技術

2.1 CNN(Convolutional Neural Network)

2.1.1　畳み込みニューラルネットワーク

CNN（Convolutional Neural Network）とは畳み込み層とプーリング層が積み重なって構成されるニューラルネットワークであり、画像の深層強化学習に導入されている。畳み込みニューラルネットワークは畳み込み層、プーリング層、全結合層の 3 つの結合方法で構成される。

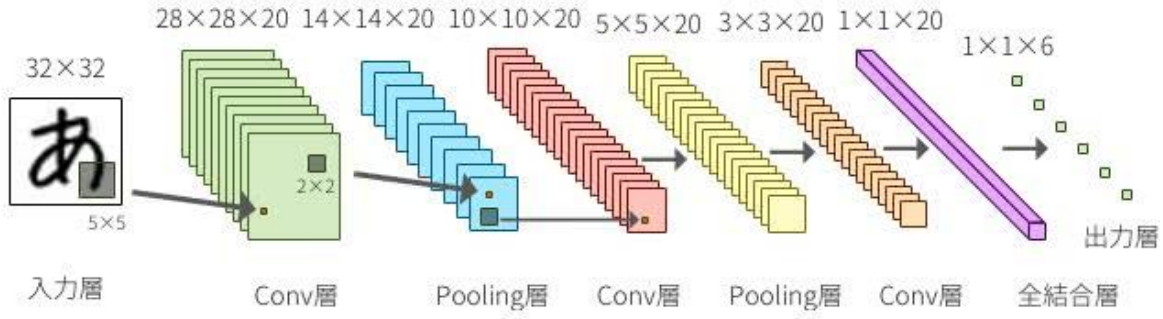

図 1:畳み込みニューラルネットワークの構成例(図は[2]より引用)

畳み込みニューラルネットワークの構成例を図 1 に示す。ここでは畳み込み層が Conv 層、プーリング層が Pooling 層である。畳み込みニューラルネットワークは、入力の局所的な特徴を抽出する役割を担う畳み込み層と、局所ごとの特徴をまとめ上げる役割を担うプーリング層を繰り返し、最後に全結合層を通して出力するような構成になっている。畳み込み層とプーリング層はそれぞれの層が異なる特徴を識別するように学習処理を繰り返す。全結合層では取り出された特徴量を一つのノードに結合し、活性化関数によって変換された値を出力する。出力層では全結合層からの出力を元に、活性化関数によって識別可能な形に変換し、分類を行う。

2.1.2　畳み込み層

畳み込み層では入力に対して畳み込み演算を行う。畳み込み演算では入力画像などの二次元配列と重みに相当するフィルタの２つのパラメータを用いる。フィルタと同じサイズの部分画像の要素同士を掛け合わせた後、それらすべてを合計して一つの数値に畳み込む。畳み込み演算の具体例を次に示す。

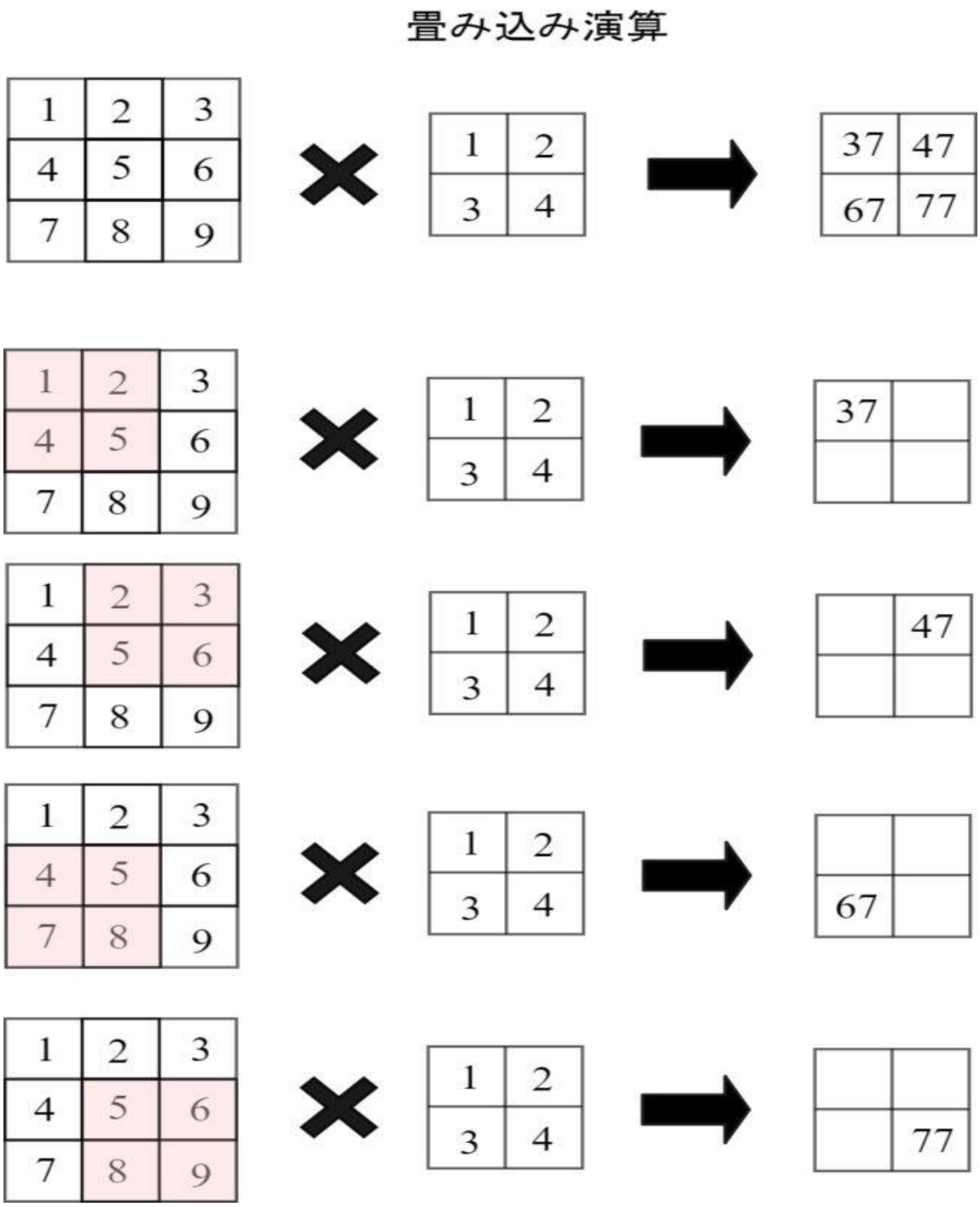

図 2:畳み込み演算の具体例(図は[3]より引用)

図のようにフィルタを入力に対して少しずつ動かし、入力とフィルタの対応している部分を掛け合わせたものの和を取っている。フィルタはストライドの値に沿って動き、ストライドを大きくすればするほど演算結果の大きさは小さくなる。

2.1.3　プーリング層

プーリング層では指定した範囲の要素をまとめ、入力の大きさを小さくする。Max プーリングではフィルタ内の要素の最大値を取り、Average

プーリングではフィルタ内の要素の平均値を取る。
プーリングの具体例を次に示す。

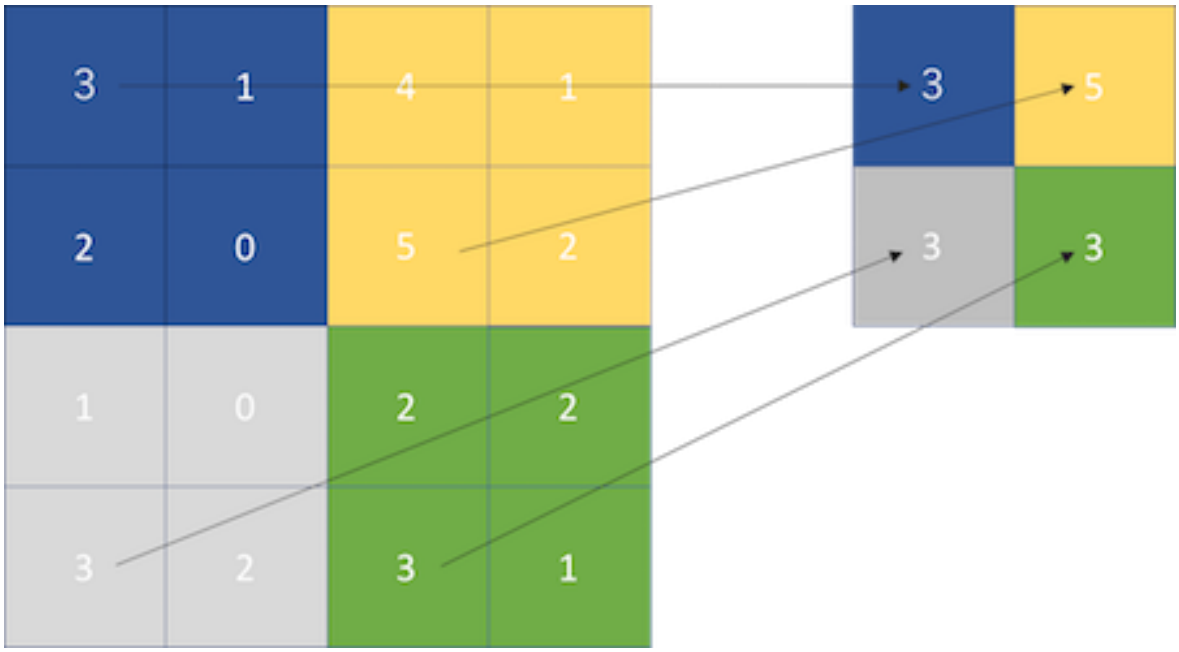

図 3:Max プーリングの具体例(図は[5]より引用)

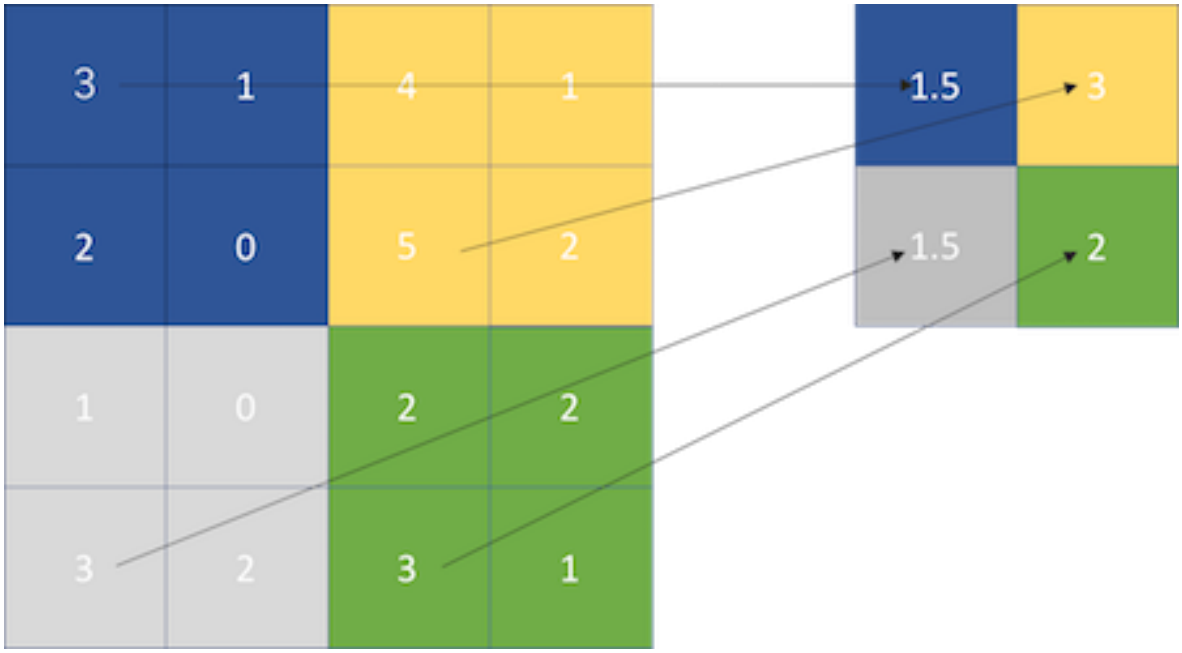

図 4:Average プーリングの具体例(図は[5]より引用)

図に示したように入力を小さく圧縮することで、微小な位置変化に対して頑健となる、過学習を抑制する、計算コストを下げるといった効果がある。

2.1.4　全結合層

全結合層は畳み込み層とプーリング層によって抽出した特徴から最終的な識別を行う、入力と出力がすべて結合している層である。出力の値

は入力(x_1, x_2)と接続の重み(w_1, w_2)の内積を取り、バイアス(b)を加える。次式によってあらわすことができる。

$$y = f(w_1 x_1 + w_2 x_2 + b)$$

2.1.5 ReLU(Rectified Linear Unit)

ReLU とは活性化関数の 1 つである。ReLU は入力された値が 0 以下の場合は 0 を出力し、0 より上の場合は入力された値と同じ値を出力する関数であり、次式によって定義される。

$$f(x) = \max(0, x)$$

ReLU 関数のグラフは次のようになる。

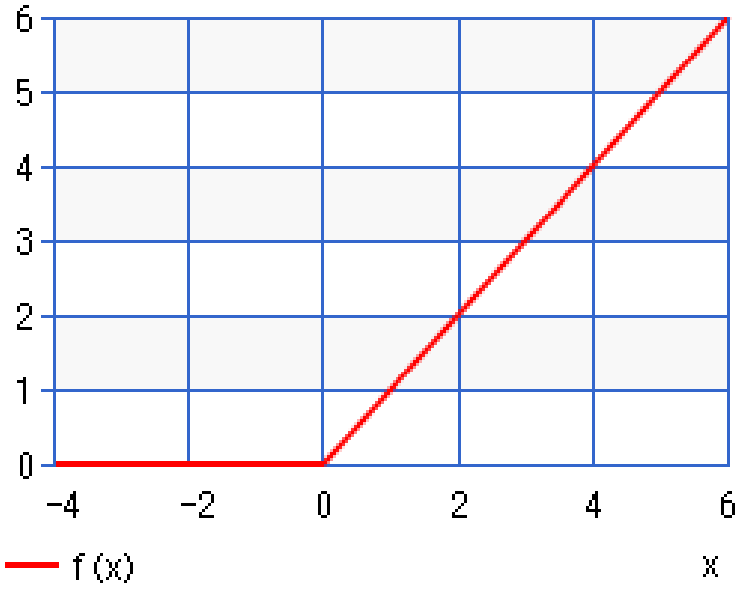

図 5:ReLU 関数

2.1.6 シグモイド関数

シグモイド関数とは活性化関数の 1 つである。シグモイド関数はあらゆる入力値を 0.0～1.0 の範囲に変換して出力する関数であり、次式によって定義される。

$$f(x) = \frac{1}{1 + e^{-x}}$$

シグモイド関数のグラフは次のようになる。

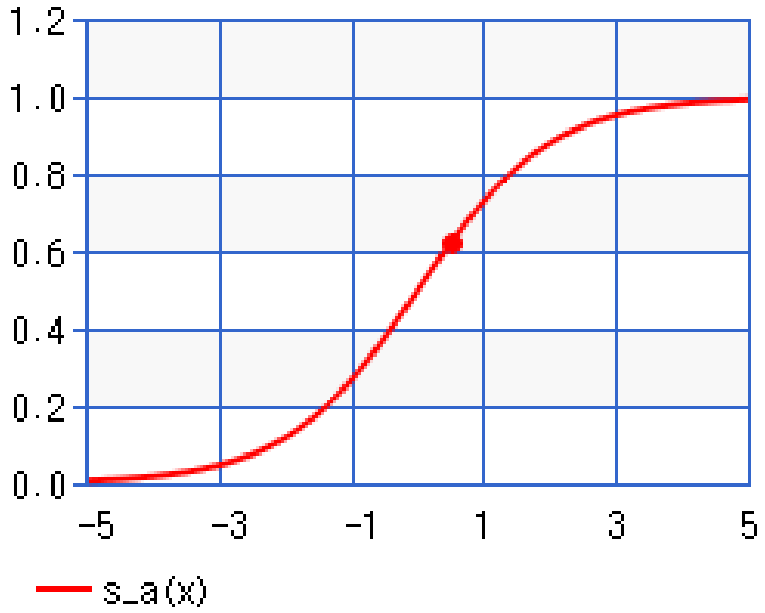

図 6:シグモイド関数

2.1.7　ソフトマックス関数

ソフトマックス関数とは活性化関数の 1 つである。ソフトマックス関数は複数の出力値の合計が 1.0 になるように変換して出力する関数であり、次式によって定義される。

$$y_i = \frac{e^{x_i}}{\sum_{k=1}^{n} e^{x_k}} (i = 1,2,\dots,n)$$

2.1.8　確率的勾配降下法

確率的勾配降下法は最適化アルゴリズムの 1 つで、学習データの中からランダムに 1 つを取り出して誤差を計算しパラメータを更新する。収束に長い時間を要するが、最急降下法に比べ再学習のための計算量が少なくなる。確率的勾配降下法は次式で表すことができる。

$$w_{t+1} = w_t - \eta_t \nabla l(w, z_t)$$

ここで l は損失関数、w はパラメータ、$\nabla l(w,z)$は確率的勾配、η は学習率を表す。

2.2 MNIST データセット

機械学習においてチュートリアル的なサンプルデータとして頻繁に使われているデータであり、手書きで書かれた数字を画像にした画像データ(image)と、その画像に書かれた数字を表すラベルデータ(label)から構成される。5 万枚の training_data と 1 万枚の test_data を含んでいる。各画像のフォーマットは 8bit グレースケール、幅 28×高さ 28 フィールド。MNIST データセットに含まれる手書き数字の例を以下に示す。

図 7: MNIST に含まれる「手書き数字」の例(図は[9]より引用)

第3章 提案手法

3.1 概要

本研究では CNN を用いて MNIST データセットの手書き数字の識別を行うことに加え、学習済み画像とは異なる特徴を示す画像について未学習画像であるという識別を行うプログラムを提案する。

[1]より引用したプログラムを改良したプログラムで学習済み画像と未学習画像をそれぞれ入力した際の出力から特徴を抽出する。本論文ではこの特徴のことを識別の迷い方と呼ぶことにする。

さらにプログラムを改良し未学習画像を入力した際に現れる迷い方をしている場合はそれを未学習画像として分類し、そうでない場合は学習済み画像として通常通りの識別を行う。

第4章 実験

4.1 学習済み画像の減少に伴う正解数の低下

4.1.1 実験概要

本実験では MNIST データセットの中から学習しないラベルの画像を設定し通常の識別を行うことで正解数が低下することを確認する。この正解数は後に行う改良において、正解数が増加したかどうかの基準となる。

4.1.2 実験方法

まず training_data、test_data に何も変更を加えず通所の識別を行い、正解数を確認する。次に MNIST データセット内の training_data に含まれる 0～9 の任意の 1 種類のラベルを持つ画像を除去し通常の識別を行い、正解数を確認する。

4.1.3 実験結果

プログラムを実行した結果を以下の表に示す。

未学習画像のラベル	正解数
0	8578
1	8406
2	8576
3	8585
4	8650
5	8767
6	8642
7	8591
8	8576
9	8593
なし	9501

表 1: 学習済み画像の減少に伴う正解数の低下

表に示したように training_data から 1 種類のラベルデータを持つ画像が削除されたことにより、削除を行わなかった時より正解数が約 900 前後減少した。

4.1.4 考察

test_data は 0～9 のランダムな手書き数字 10000 個で構成されているため 1 種類の数字あたりの個数は約 1000 個と考えられる。training_data から 1 種類学習しない数字を作ったためその数字に対する正解数が下る。よって正解数 900 前後の減少は約 1000 個の test_data の正解数が下がったものと同じと考えられる。

4.2 迷い方の抽出と基準値項目の決定

4.2.1 実験概要

本実験では、本研究で提案する未学習画像識別のために、未学習画像を入力した場合のみに見られる識別の迷い方を抽出し、識別に使用する基準となる値を得られる項目を考察する。

4.2.2 実験方法

はじめに 4.1 節と同様に training_data に含まれる 0～9 の任意の 1 種類のラベルを持つ画像を除去する。次に test_data を training_data から除去したラベルを持つ画像、持たない画像の二つに分類する。これにより前者を未学習画像群、後者を学習済み画像群とできる。

本プログラムが通常の識別を行った際の出力層における結果の例は(array([[1.87233707e-04],[5.77079658e-05],[2.73467832e-05],[2.89149165e-05],[3.45731199e-01],[1.96559649e-05],[5.13649980e-04],[7.64525369e-03],[2.13526145e-02],[4.83130574e-01]]), 9)のような十次元配列とラベルデータの組み合わせになっており、最終的な識別では十次元配列の中で最大値を持つインデックスが識別結果となり、ラベルデータと一致するかどうかで正解不正解の判別を行っている。

この識別を未学習画像群、学習済み画像群のそれぞれで行うことで出力される十次元配列から迷い方を見つける。

本実験では十次元配列内の 10 個の要素の平均値、不偏分散、不偏標準分散、最大値、最小値、中央値、最大値と最小値の差を計算しそれぞれの画像群での平均を算出し未学習画像の識別に利用可能な指標とその特徴を検討する。

4.2.3 実験結果

プログラムによる算出結果を以下の表に示す。

	未学習(1)	学習済(1)	未学習(2)	学習済(2)
平均	0.081838004	0.097815842	0.0914894	0.099154146
不偏分散	0.058456954	0.083666932	0.062553976	0.085370995
標準偏差	0.224246312	0.285192056	0.22943779	0.28924307
最大値	0.738866526	0.948661835	0.740372721	0.961749346
最小値	2.50E-08	5.86E-09	2.38E-09	3.09E-09
中央値	4.83E-05	3.47E-05	4.66E-05	2.32E-05
最大値と最小値の	0.738866501	0.948661829	0.740372719	9.62E-01

差				

表 2:算出結果(1,2)

	未学習(3)	学習済(3)	未学習(4)	学習済(4)
平均	0.081369279	0.098477566	0.084377625	0.099148576
不偏分散	0.056139219	0.085342328	0.066438275	0.085549453
標準偏差	0.212904992	0.288892591	0.23944472	0.289276666
最大値	0.693748348	0.960770586	0.792940733	0.961644468
最小値	9.25E-09	6.56E-09	1.13E-08	5.17E-08
中央値	6.33E-05	1.44E-05	4.07E-05	2.81E-05
最大値と最小値の差	0.693748339	0.960770579	0.792940721	0.961644416

表 3:算出結果(3,4)

	未学習(5)	学習済(5)	未学習(6)	学習済(6)
平均	0.086635255	0.099877062	0.095122201	0.098647377
不偏分散	0.061791619	0.086310718	0.066645597	0.084756644
標準偏差	0.227362409	0.29137543	0.238711722	0.287748014
最大値	0.74096104	0.968566023	0.768422498	0.956899052
最小値	1.41E-09	4.47E-09	2.43E-09	3.12E-09
中央値	5.10E+00	1.18E-05	0.000161	2.26E-05
最大値と最小値の差	0.740961039	0.968566019	0.768422495	0.956899049

表 4:算出結果(5,6)

	未学習(7)	学習済(7)	未学習(8)	学習済(8)
平均	0.075405938	0.098332771	0.060719136	0.098104231
不偏分散	0.054910068	0.084655717	0.038409313	0.084346316
標準偏差	0.205860819	0.287196449	0.161915554	0.286929684

最大値	0.675167712	0.955039922	0.532545044	0.954606839
最小値	1.46E-09	3.69E-09	1.16E-08	1.00E-08
中央値	1.01E-05	1.34E-05	0.000185877	2.23E-05
最大値と最小値の差	0.67516771	0.955039918	0.532545032	0.954606829

表 5:算出結果(7,8)

	未学習(9)	学習済(9)	未学習(0)	学習済(0)
平均	0.091497221	0.098123692	0.063804919	0.099961358
不偏分散	0.062665868	0.084436023	0.043712526	0.085666447
標準偏差	0.23676404	0.286962906	0.172253834	0.28974801
最大値	0.775067129	0.954535949	0.566850007	0.962432607
最小値	1.23E-08	1.09E-08	4.97E-09	2.17E-08
中央値	8.34E-05	2.75E-05	6.89E-05	2.94E-05
最大値と最小値の差	0.775067117	0.954535938	0.566850002	0.962432585

表 6:算出結果(9,0)

各行の項目は各画像群の平均値となっている。括弧内の数値はその時に除去していた入力画像の数値である。例えば未学習(1)は 1 を未学習の状態で test_data が 1 の場合の出力から算出された値が入り、学習済(1)には 1 を未学習の状態で test_data が 1 以外の場合の出力から算出された値が入る。

4.2.4 考察

7 つの項目について未学習画像群、学習済画像群の迷い方を比べたが 2 つの項目での迷い方に注目した。

一つ目は不偏分散である。学習済画像群ではすべての場合で約 0.085 だが未学習画像群では 0.067 以下で、低いものでは未学習(8)の 0.038409313 が挙げられる。分散が小さく要素にばらつきがないということは十次元配列内の要素それぞれの値が近く、自信をもって識別できていないと考えられる。

二つ目は最大値である。学習済画像群ではすべての場合で約 0.95 だが未学習画像群では 0.77 以下となっている。十次元配列内の要素の最大値は 1 でこの要素はそのまま%に置き換えることができる。つまり学習済

画像群では平均として配列内の最大値が 0.95 なため 95%の自信をもって識別できていることになる。これは 4.1 節の実験で確かめた、未学習画像がない場合の正解率 9501/10000 と正解率が一致すると考える。未学習画像群において、逆にその最大値が低いということは識別の自信がなく、数値としてみても 77%以下になると考えられる。

この 2 つの項目を、入力画像を未学習画像と識別する基準値にすることが可能と考える。

4.3 基準値を設定しての識別

4.3.1 実験概要

4.2 節の実験で不偏分散と最大値を基準値にすることに決定した。4.2.3 節の表から大体の値で基準値を設定し、基準値より大きければ学習済画像として通常の識別、小さければ未学習画像として識別するようにプログラムし正解数が増加するか確かめる。

4.3.2 実験方法

4.2.3 節の表を見て基準値を設定する。本実験では学習済画像を未学習画像として識別してしまわないように 4.2 節で求めた平均値よりも低めに、最大値の基準値を 0.70、不偏分散の基準値を 0.040 と設定する。これで、十次元配列の要素の最大値が 0.70 未満の場合は未学習画像として識別するようになり、不偏分散が 0.06 未満の場合も未学習画像として識別するようになる。完成したプログラムを実行し正解数の変化を確認する。

4.3.3 実験結果

プログラムを実行した結果を以下の表に示す。

	正解数	
未学習画像のラベル	識別なし	識別あり
0	8578	8889
1	8406	8832
2	8576	8844
3	8585	8739
4	8650	8514
5	8767	8864
6	8642	8854
7	8591	8566
8	8576	8987
9	8593	8575

表 7:改良したプログラムの識別結果

識別なしのデータは比較用として表 1 と同じものを示している。未学習画像が 0、1，2，3，5，6，8 の場合は正解数が増加したが、4，7，9 の場合は正解数が減少する結果となった。

4.3.4 考察

識別の基準値を低めに設定したため正解数のわずかな増加を予想していたが中には 400 以上正解数が増加する場合があり、テスト毎の誤差ではないと考えられるため最大値と不偏分散を識別に関わる要素とすることで未学習画像の識別ができる可能性があることが示された。一方で正解数が減少する場合もあり増加した場合と比べて、増加幅が少なくかつテスト毎の誤差で減少してしまったとは考えにくい。よって、識別の項目が最大値と不偏分散だけでは不十分、もしくは、最大値と不偏分散の基準値は特定の場合において誤った識別をしてしまう可能性があることが示された。

第5章 終わりに

本研究では、画像を CNN に入力し識別するプログラムの、出力された多次元配列の要素を用いて最終的な識別をする部分において、未学習の画像を未学習な画像であると識別する基準となる項目とその値について検討を行った。その結果、多次元配列の最大値の基準値を 0.70、不偏分散の基準値を 0.040 と設定することで未学習画像の識別ができる可能性が示された。

今後の課題としては、同時に示された、識別の項目が最大値と不偏分散だけでは不十分なこと、もしくは、最大値と不偏分散の基準値は特定の場合において誤った識別をしてしまう可能性があることを改善すべく、追加の基準項目とその値を模索することや、CNNのネットワーク構成、エポック数やバッチ数の組み合わせなどを模索していくことが挙げられる。

謝辞

本論文の作成にあたり、多くの助言、指導をしてくださった三好力教授に心からお礼申し上げます。また、議論に協力してくださった三好研究室や学友の皆様に心から感謝致します。

参考文献

[1]“ニューラルネットワークと深層学習”
http://nnadl-ja.github.io/nnadl_site_ja/index.html
[2]“定番の Convolutional Neural Network をゼロから理解する”
https://deepage.net/deep_learning/2016/11/07/convolutional_neural_network.html#convolution%E5%B1%A4
[3]“畳み込みニューラルネットワークの基礎”
https://screwandsilver.com/cnn_convolutional_net/#i-2
[4]“ 畳み込みニューラルネットワーク_CNN(Vol.16)”
https://products.sint.co.jp/aisia/blog/vol1-16
[5]“畳み込みニューラルネットワーク（CNN）”
https://www.renom.jp/notebooks/tutorial/basic_algorithm/convolutional_neural_network/notebook.html
[6]藤岡優也,“機械学習を用いた野鳥の鳴き声分類手法に関する研究”, 令和元年度龍谷大学修士論文(2020).
[7]“第 4 回 CNN（Convolutional Neural Network）を理解しよう（TensorFlow 編）”
https://www.atmarkit.co.jp/ait/articles/1804/23/news138.html
[8]“高精度計算サイト”
https://keisan.casio.jp/
[9]” MNIST：手書き数字の画像データセット”
https://www.atmarkit.co.jp/ait/articles/2001/22/news012.html

付録　開発したソースコード

```
def load_data():
    f = gzip.open('../data/mnist.pkl.gz', 'rb')
    training_data, validation_data, test_data = cPickle.load(f, encoding='latin1')
    f.close()
    training_data_a,training_data_b = training_data
    notlearn = np.where(training_data_b == 9)
    new_training_data_a = np.delete(training_data_a, notlearn, 0)
    new_training_data_b = np.delete(training_data_b, notlearn, 0)
    new_training_data = new_training_data_a,new_training_data_b
    return (new_training_data, validation_data, test_data)
def evaluate(self, test_data):
        var_sum = 0
        test_max_sum = 0
        test_results = [(self.feedforward(x), y) for (x, y) in test_data]
        for i in range(len(test_results)):
            test_results_right = test_results[i][0]
            var = np.var(test_results_right)
            test_max = np.max(test_results_right)
            test_results_left = test_results[i][1]
            if var < 0.040 or test_max < 0.80:
                new_test_results = [(9,test_results_left)]
            else:
                new_test_results = [(np.argmax(test_results_right),test_results_left)]
            res = res + sum(np.int(x == y) for (x,y) in new_test_results)
        return res
def SGD(self, training_data, epochs, mini_batch_size, eta,test_data=None):
        if test_data:
            n_test = sum(1 for _ in test_data)
        n = sum(1 for _ in training_data)
        for j in range(epochs):
            random.shuffle(training_data)
            mini_batches = [
                training_data[k:k+mini_batch_size]
                for k in range(0, n, mini_batch_size)]
            for mini_batch in mini_batches:
                self.update_mini_batch(mini_batch, eta)
            if test_data:
                print ("Epoch {0}: {1} / {2}".format(j, self.evaluate(test_data), n_test))
```

```
else:
    print ("Epoch {0} complete".format(j))
```