グループ1
手書き文字認識

メンバー
185756J 松田一秀
185715B 比嘉信斗
185708J 安和良祐
185759C 高江洲壱星

目的、目標

目的

　　手書きで書いた英字文字を機械学習を用いて認識させたい。

目標

　　機械学習を用いて英字文字をなるべく精度高く認識させる。

アプローチの全体像

　手書きの文字が写されてある画像に対して、ディープラーニングを行うか、特徴量を自分達で選択して学習を行うかで悩んだ。

　結果、「深層学習班」と「特徴量選択班」に分かれて機械学習を行い、学習結果にどのような違いが出るのかを調べることにした。

データセットの構築方法

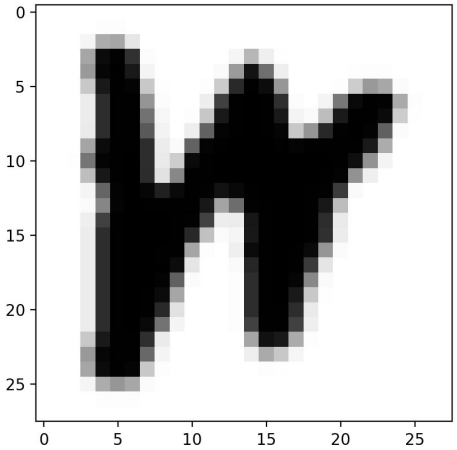
EMNISTの手書き文字データ(letters)を使用

EMNIST-Lettersのデータは、画像サイズが28x28ピクセル、

画像の数が訓練用に124,800枚、

評価用に20,800枚用意されている。

モジュールはKerasを用いた。




```
import extra_keras_datasets.emnist as emnist

# 訓練データ,ラベル(t_images, t_labels)とテストデータ,ラベル（v_images, v_labels)を取得する
# type="letters"の場合、28×28の3次元の画像データで trainが124800、testが20800である
(t_images, t_labels), (v_images, v_labels) = emnist.load_data(type="letters")
```

深層学習班

畳み込みニューラルネットワーク(CNN)

CNNのモデルの作成

モデルのコード

```
model = keras.Sequential(
    [
        Conv2D(32, (3, 3), activation="relu", padding="same", input_shape=(28,28, 1)),
        Conv2D(32, (3, 3), activation="relu", padding="same"),
        Conv2D(32, (5, 5), activation="relu", padding="same"),
        MaxPooling2D((2, 2), padding="same"),
        Dropout(0.4),
        Conv2D(64, (3, 3), activation="relu", padding="same"),
        Conv2D(64, (3, 3), activation="relu", padding="same"),
        Conv2D(64, (5, 5), activation="relu", padding="same"),
        MaxPooling2D((2, 2), padding="same"),
        Dropout(0.4),
        Flatten(),
        Dense(128, activation="relu"),
        Dropout(0.4),
        Dense(27, activation="softmax"),
    ]
)
```

モデルの概要

畳み込み層(conv2d)を3層に行い、プーリング層(MaxPooling)を1回の組み合わせを2回行い行なった。

畳み込み層は、3×3のフィルタを2層と5×5のフィルタを1層の組み合わせを用いた。(モデルのコードより)

過学習を防ぎ、汎化性能を向上させるためにKerasにある関数Dropoutを用いた。

今回のモデルではTotal params:598,107とあるように、約60万のパラメーターを訓練しならが調整した。

後半で、Flattenで直列化したあと完全結合し、最終的な出力はソフトマックスで27カテゴリに分類した。
(今回用いたデータセットlettersが27カテゴリのため)

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 28, 28, 32)	320
conv2d_1 (Conv2D)	(None, 28, 28, 32)	9248
conv2d_2 (Conv2D)	(None, 28, 28, 32)	25632
max_pooling2d (MaxPooling2D)	(None, 14, 14, 32)	0
dropout (Dropout)	(None, 14, 14, 32)	0
conv2d_3 (Conv2D)	(None, 14, 14, 64)	18496
conv2d_4 (Conv2D)	(None, 14, 14, 64)	36928
conv2d_5 (Conv2D)	(None, 14, 14, 64)	102464
max_pooling2d_1 (MaxPooling2	(None, 7, 7, 64)	0
dropout_1 (Dropout)	(None, 7, 7, 64)	0
flatten (Flatten)	(None, 3136)	0
dense (Dense)	(None, 128)	401536
dropout_2 (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 27)	3483

Total params: 598,107
Trainable params: 598,107
Non-trainable params: 0

精度向上に向けた実験

1. 学習回数を20回(20Epoch)行なった。

2. 学習回数を300回(300Epoch)行なった。

3. 畳み込み層を3×3の4層とプーリング層1層の組み合わせを3回行うモデルに変更し行なった。

(変更の一部)

```
Conv2D(32, (3, 3), activation="relu", padding="same",
input_shape=(28,28, 1)),
Conv2D(32, (3, 3), activation="relu", padding="same"),
Conv2D(32, (3, 3), activation="relu", padding="same"),
Conv2D(32, (3, 3), activation="relu", padding="same"),
MaxPooling2D((2, 2), padding="same"),
```

4. 畳み込み層のフィルタの範囲を9×9に統一し、プーリング層を5×5に変更したモデルで行なった。

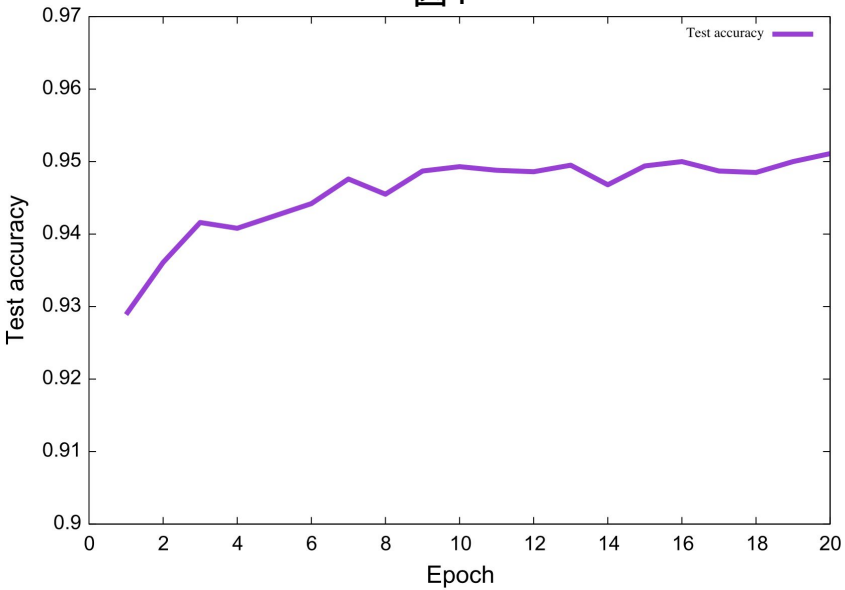
(変更の一部)

```
Conv2D(32, (9, 9), activation="relu", padding="same",
input_shape=(28,28, 1)),
Conv2D(32, (9, 9), activation="relu", padding="same"),
Conv2D(32, (9, 9), activation="relu", padding="same"),
MaxPooling2D((5, 5), padding="same"),
```

結果

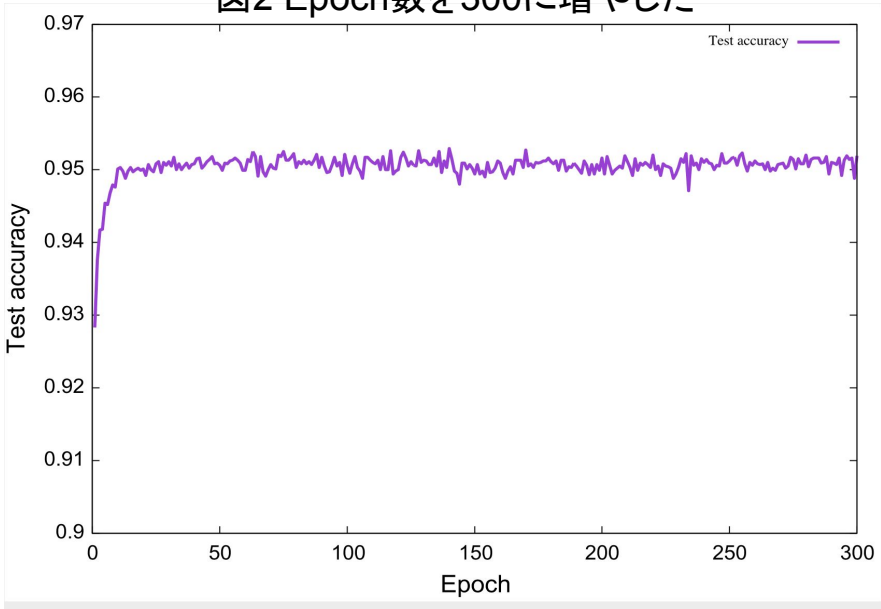
1. 正答率　約95.1%

図1



2. 正答率　約95.1%

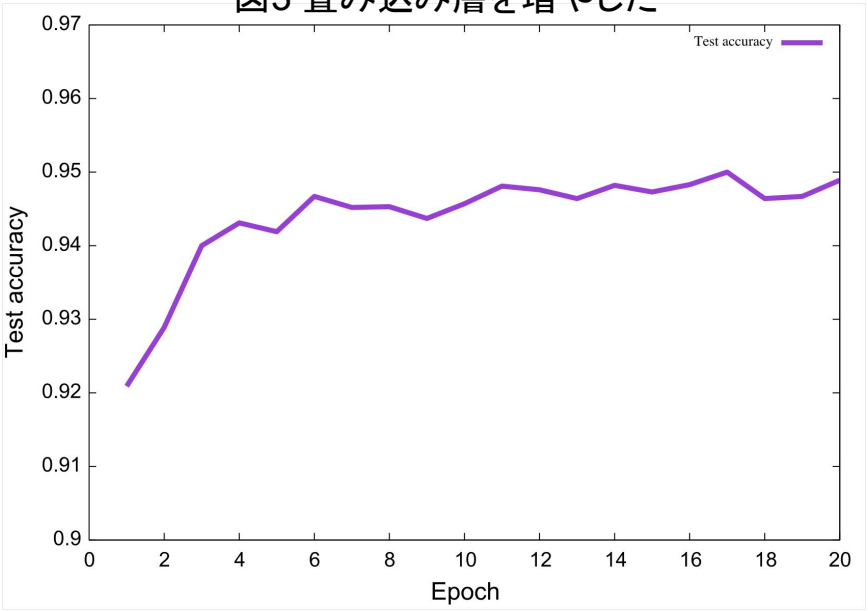
図2 Epoch数を300に増やした



結果2

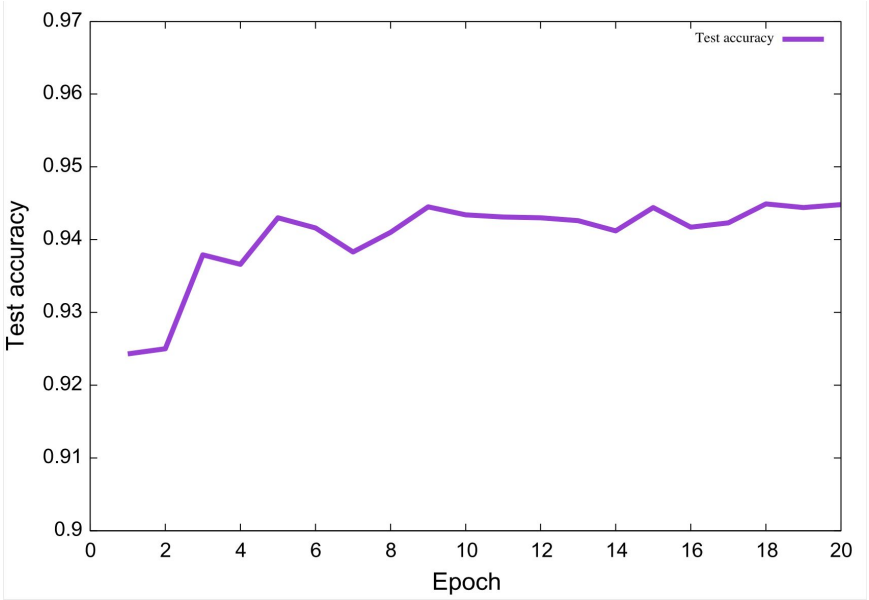
3. 正答率　約94.8%

図3 畳み込み層を増やした



4. 正答率　約94.4%

図4 畳み込み層とプーリング層のパラメータ値を変更した



CNNにおいての考察

- 本実験では1つ目のモデルを踏まえて、2つ目、3つ目は文字認識の精度をあげる予定で、モデルのパラメータ値を変更したが、1つ目とほぼ同じ認識精度となった。また、4つ目のモデルは1つ目のモデルの畳み込み層を3×3から9×9に、プーリング層を2×2から5×5に変更したため、大幅に認識精度が下がるのという予想をして行ったが、結果は認識精度は1つ目のモデルが0.9511で4つ目のモデルが0.9448と下がりはしたものの、誤差の範囲程度のものだった。

- 1つ目のモデルから3つ目のモデルのように畳み込み層の層を増やし、実験を行っても結果はほとんど変わらないことから、12層は過剰であったと考えられる。Platinum Data Blogの検証結果[7]だと128×128の画像データに対して4層で85%近くの正答率を出している。このことから本実験で使用した28×28の画像データに対しては、畳み込み層の層は4~7がベストだと推測できる。

- グラフから、Epoch数を14回目付近で、最終結果と近い数値を出していること、そして、2つ目のモデルでEpoch数300回行っても、1つ目のモデルと2つ目のモデルで正答率に差がほとんどないことから、シンプルな画像ほど少ないEpoch数で学習が完了すると考えられる。本実験に関してもEpoch数10回ほどで正答率の上昇が止まったことを確認できる。

参考文献

[7] Platinum Data Blog　深層学習は画像のどこを見ている !?CNNで「お好み焼き」と「ピザ」の違いを検証，
https://blog.brainpad.co.jp/entry/2017/07/10/163000

特徴量選択班

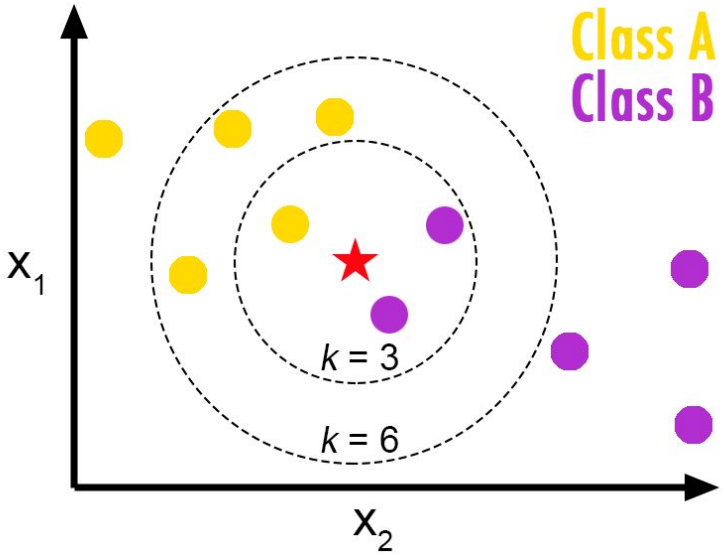
機械学習の進め方

　画素毎の明度を特徴量とした場合と、局所方向寄与度特徴(LDC特徴)を用いた場合の精度を学習器を変えて比較する。

使用した学習器 (1/2)

K近傍法(KNN)

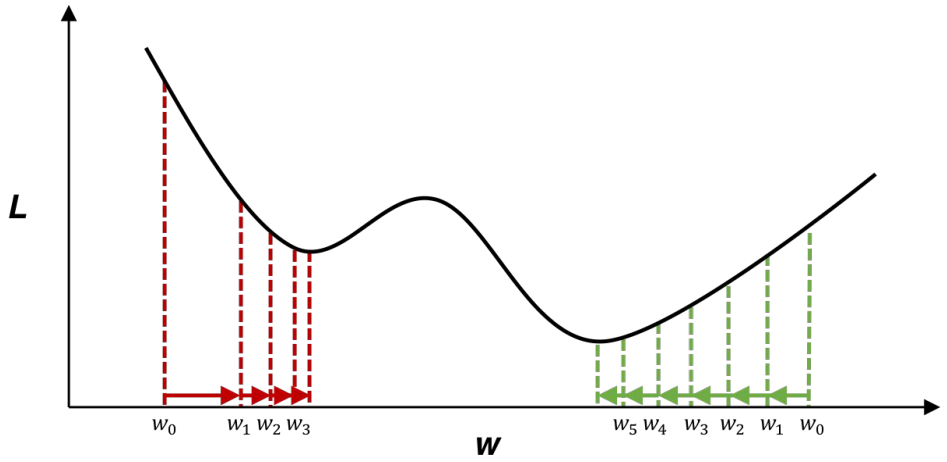
k近傍法とはデータを特徴ベクトルにし、特徴ベクトルが近いk個のデータのラベルを見て最も多かったラベルをデータに割り当てる方法である。手動で変更するパラメータは近傍数k、選択する特徴量の数である。kと選択数は整数である必要がある。kの数を指定しない場合はkにはデフォルトの値(k=5)が入っているとする。



使用した学習器 (2/2)

確率的勾配降下法(SGD)

最急降下法の派生。ランダムなデータ1つを使って、最急降下法を適用する。手動で変更するパラメータは特になし。



実験結果 (1/2)

KNNを用いて、画素毎の明度を特徴量とした場合と、局所方向寄与度特徴(LDC特徴)を用いた場合の精度を比較する。

明度を特徴として用いた場合	0.865625
LDC特徴を用いて場合	0.7446634615384615

実験結果 (2/2)

　SGDを用いて画素毎の明度を特徴量とした場合と、局所方向寄与度特徴(LDC特徴)を用いた場合の精度を比較する

明度を特徴として用いた場合	0.5298076923076923
LDC特徴を用いて場合	0.8259134615384616

考察 (特徴量選択)

　画素毎の明度を特徴量とした場合と、LDC特徴を用いた場合の精度を学習器を変えて比較したところ、KNNでは画像毎の明度を特徴量として用いた方が精度が高かったが、SGDではLDC特徴を用いた方が精度が高かった。このことから、学習器によって適切な特徴量が異なることがわかる。KNNは比較的予測精度は高かったが特徴選択にかかる時間が3~4倍かかったので次元数が多いデータから特徴選択を行うにはSGDの方が適している。2つの学習器の性能を比べると画素明度・LDCの両方で引けを取らない精度と処理速度を見るとSGDの方が良い。

全体を通しての考察

本実験で作成したモデルでの結果だと、CNNを用いた手書き文字認識の精度は約95%。特徴量選択のKNNを用いた場合、明度を特徴とした時は約85%となり、LDC特徴を用いると約74%である。またSGDでは明度を特徴とした時は約52%となり、LDC特徴を用いると約82%という結果になった。このことからCNNがより高精度な学習が行える。

また学習能力の安定性を考えた場合、KNNだとkの値によっても学習結果が変化するほか、SGDにおいても出た結果が毎回最高値を出すとは限らないことから、確実に学習するのならCNNを用いるのが良い。

予定していた実験計画

1週目
- データをとってきて、中身を確認する。

2~4週目
- pythonを用いてコードの作成。

5,6週目
- 1回目の実験開始、改善など。

7,8週目
- 複数回実行をし、改善を繰り返す。

9週目
- レポート&プレゼン資料作成。（成果物整理）

10週目
- 最終発表。

ある程度予定通りに進めることができたが、調べて理解を深めることと、 pythonコードの作成などに時間を使ってしまったため、モデルの改善にあまり力を注ぐことができなかった。

実験計画を立てた時点では、2つの班に分かれることを想定していなかった。

時間の都合上省いた項目（実験計画にも含めていなかった項目等があれば、残された課題等）

モデルの改善に力が注げなかったこと。

理由として、予定していた実験計画の中に調べて理解を深める時間を加えていなかったため、計画が後ろ倒しになってしまったこと。また、pythonコードの作成で想定していたより時間がかかったことも理由の一つだと考える。

2つの班に分かれることになった時点で再度班ごとに計画の練り直しを行う必要があった。また、こまめに元の計画とのズレを確認し、再計画を行うべきだったと考える。

自己評価や振り返り

　実験を行う前に立てた計画では、調べて理解する時間を加えていなかったが、実際にはその部分に大半の時間を使ってしまったため、次に実験を行う際は調べて理解を深める時間を計画の中に含めようと思う。
　またモデルをより細部化して変更を行い、学習精度を上げられるようにする。そしてアルファベットのみだけでなく、数字やひらがなを含めて学習を試したい。

参考文献

[1] Wikipedia 光学文字認識　https://ja.wikipedia.org/wiki/光学文字認識, 2020/12/15

[2] 採点記号の分離抽出と認識処理の高精度化　https://www.nara-k.ac.jp/nnct-library/publication/pdf/h20kiyo7.pdf, 2020/12/15

[3]畳み込みニューラルネットワーク (CNN)を用いた手書き日本語文字認識システムの試作
https://www.jstage.jst.go.jp/article/jceeek/2015/0/2015_348/_pdf/-char/ja, 2021/1/28

[4] 乳房 X 線像上の腫瘤影判別のための特徴量選択法の検討
https://www.ieice.org/publications/conference-FIT-DVDs/FIT2003/pdf/H/H_014.pdf, 2021/1/28

[5] 異なる特徴抽出法を併用した手書き類似文字の識別
異なる特徴抽出法を併用した手書き類似文字の識別［ PDF：0.2MB］, 2021/1/28

[6] EMNIST https://www.nist.gov/itl/products-and-services/emnist-dataset, 2021/1/28

[7] Platinum Data Blog　深層学習は画像のどこを見ている !?CNNで「お好み焼き」と「ピザ」の違いを検証 ,
https://blog.brainpad.co.jp/entry/2017/07/10/163000, 2021/1/26