データマイニング班Group1
「機械学習による手書き文字の識別」

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

提出日：2021年1月26日

目次

1　はじめに
1.1　背景
1.2　関連事例
1.3　本実験の位置付け
1.4　実験の目的と達成目標

2　実験方法
2.1　実験目的
2.2　データセット構築
2.3　モデル選定
2.4　パラメータ調整
2.4.1　CNN
2.4.2　KNN
2.4.3　SGD

3　実験結果
3.1　CNN
3.2　特徴量選択

4　考察
4.1　それぞれのモデルのみで考察
4.2　3つのモデルの結果を踏まえて

5　意図していた実験計画との違い

6　まとめ

7　振り返り

要約

本実験では、手書き文字認識を用いて機械学習を行なった。用いた学習器は、畳み込みニューラルネットワーク(以下「CNN」と表記)とK近傍法、確率的勾配降下法の3つである。どのように処理を行えば精度を挙げられるのか、また、これら3つのうちどれが良い精度を上げるかを実験した。結果として、CNNは、4~7層の畳み込み及び10回程度の学習で約95%ほどの精度を出すことがわかった。K近傍法は画素毎の明度を特徴として学習した場合は約86%となり、LDC特徴を用いて学習した場合は約74%という結果であった。また確率的勾配降下法は画素毎の明度を特徴として学習した場合は約52%となり、LDC特徴を用いて学習した場合は約82%という結果となった。よりこれら3つの学習器ではCNNがより精度の高い学習が行える。

1. はじめに

1.1 背景

手書きで書いた文字をデジタルに取り込みたいと考えた。機械学習で手書き文字認式はよく取り扱われるので、我々でもできるのか興味をもった。

1.2 関連事例

手書き文字の画像をスキャナーや写真で取り込み、デジタルの文字コードに変換する代表のソフトウェアとして光学文字認識(Optical character recognition)[1]がある。また奈良工業高等専門学校で行われた採点記号の分離抽出と認識処理の高精度化[2]の研究などもある。機械学習を用いた文字認識に焦点を合わせると、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[3]や乳房 X 線像上の腫瘤影判別のための特徴量選択法の検討[4]、異なる特徴抽出法を併用した手書き類似文字の識別[5]などの文献を目にした。

1.3 本実験の位置づけ

機械学習を用いた文字認識は主にCNNを用いた方法と、文字の特徴を選択し、識別する方法があることがわかった。そのため本実験では、上記の2つの方法で実験を行い、どのような差があるのか調べる。また、どのような学習方法や処理を行うことで文字認識をしているのか、精度を上げる方法はあるのかを実験し調べる。

1.4 実験の目的と達成目標

CNNを用いた方法、そして、特徴量を選択する方法を用いて、それぞれ手書き文字をなるべく精度高く認識させる。

2. 実験方法

2.1 実験目的

CNNを用いた方法と特徴量選択の2 つの方法により手書き文字認識を行い、精度の良さなどを比較して、精度が良いものまたは順調に実装できた方法でさらに精度をあげれるのか調べた。

2.2 データセット構築

EMNIST[6]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。本実験では、データをダウンロードする手順をプログラム内に記述した。

2.3 モデル選定

用いた学習器は、CNN、K近傍法(以下「KNN」と表記)、確率的勾配降下法(以下「SGD」と表記)を用いた。KNNは特徴量選択で、Forward Selectionを行うため使用した。Forward Selectionは特徴を選んでいない状態から選択する特徴を増やしていく方法である。

2.4 パラメータ調整

2.4.1 CNN

4つのモデルパターンを作成し実行した。

1つ目は、2次元の画像に対して3×3の範囲のフィルタで畳み込みを2層と5×5の範囲のフィルタで畳み込みを1層行い、プーリング層で2×2のフィルタでMax Poolingにかけた。また過学習を防ぐためにKerasにある関数Dropoutを用いた。それらを2回行うモデルを作成し、20回繰り返して学習を行わせた。ここでのモデルを表1に示した。

2つ目は、1つ目のモデルを300回繰り返し学習させた。ここでのモデルを表2に示した。

3つ目は、3×3の範囲のフィルタで畳み込みを4層行い、1つ目と同様のプーリング及びDropoutを用いた。それらを3回行うモデルを用いて、20回学習を行なった。ここでのモデルを表3に示した。

最後4つ目は、1つの目のモデルにに対してフィルタの範囲を9×9に変更し、Max Poolingのフィルタの範囲を5×5に変更して学習を行なった。ここでのモデルを表4に示した。

2.4.2 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。

2.4.3 SGD

SGDは最急降下法の派生。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。

3. 実験結果

3.1 CNN

表1は1つ目のモデルと実行結果を示している。Test lossが0.1528、Test accuracyが0.9511という結果になった。また、図1のグラフにEpochごとのテスト精度を示した。x軸がEpoch数で、y軸にテスト精度である。

表1 1つ目のモデルと実験結果

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

Test loss: 0.1528
Test accuracy: 0.9511

図1 1つ目のモデルのEpochごとのテスト精度

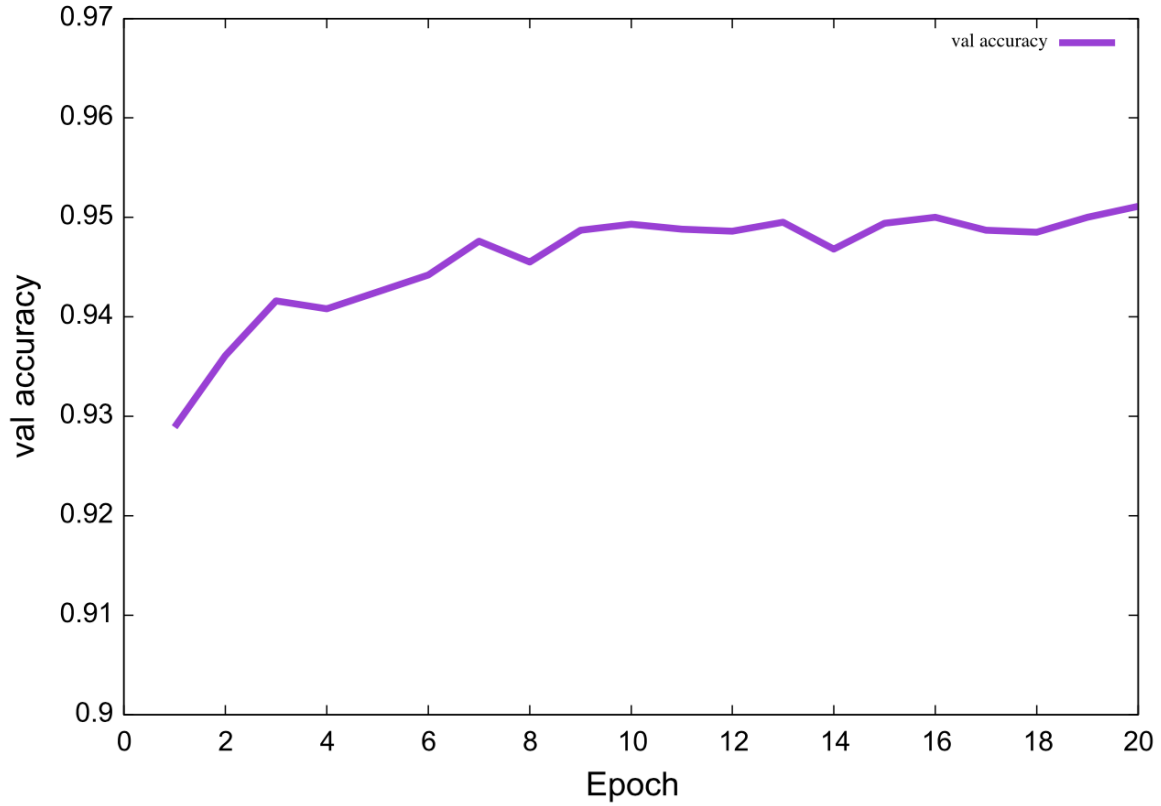


表2は2つ目のモデルと実行結果を示している。Test lossが0.2056、Test accuracyが0.9519という結果になった。また、図2のグラフにEpochごとのテスト精度を示した。x軸がEpoch数で、y軸にテスト精度である。

表2 2つ目のモデルと実験結果

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
_________________________________________________________________

Test loss: 0.2056
Test accuracy: 0.9519
```

図2 2つ目のモデルのEpochごとのテスト精度

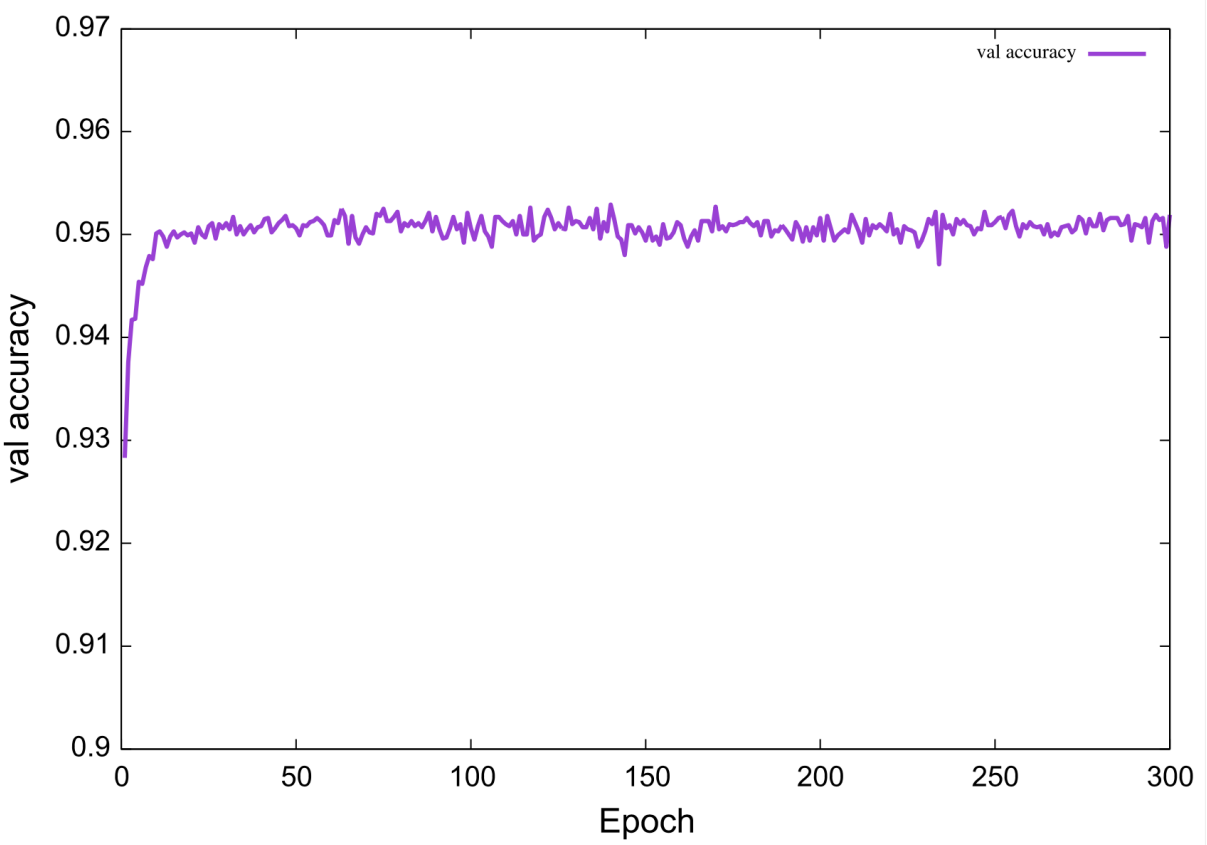


表3は3つ目のモデルと実行結果を示している。Test lossが0.1677、Test accuracyが0.9489という結果になった。また、図3のグラフにEpochごとのテスト精度を示した。x軸がEpoch数で、y軸にテスト精度である。

表3 3つ目のモデルと実験結果

```
_________________________________________________________________
```

```
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)        9248
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 28, 28, 32)        9248
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 14, 14, 32)        0
_________________________________________________________________
dropout (Dropout)            (None, 14, 14, 32)        0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 14, 14, 64)        18496
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 14, 14, 64)        36928
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 14, 14, 64)        36928
_________________________________________________________________
conv2d_7 (Conv2D)            (None, 14, 14, 64)        36928
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 7, 7, 64)          0
_________________________________________________________________
dropout_1 (Dropout)          (None, 7, 7, 64)          0
_________________________________________________________________
conv2d_8 (Conv2D)            (None, 7, 7, 128)         73856
_________________________________________________________________
conv2d_9 (Conv2D)            (None, 7, 7, 128)         147584
_________________________________________________________________
conv2d_10 (Conv2D)           (None, 7, 7, 128)         147584
_________________________________________________________________
conv2d_11 (Conv2D)           (None, 7, 7, 128)         147584
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 4, 4, 128)         0
_________________________________________________________________
dropout_2 (Dropout)          (None, 4, 4, 128)         0
_________________________________________________________________
flatten (Flatten)            (None, 2048)              0
_________________________________________________________________
dense (Dense)                (None, 256)               524544
_________________________________________________________________
dropout_3 (Dropout)          (None, 256)               0
_________________________________________________________________
dense_1 (Dense)              (None, 27)                6939
=================================================================
Total params: 1,205,435
Trainable params: 1,205,435
Non-trainable params: 0
_________________________________________________________________

Test loss: 0.1677
Test accuracy: 0.9489
```

図3 3つ目のモデルのEpochごとのテスト精度

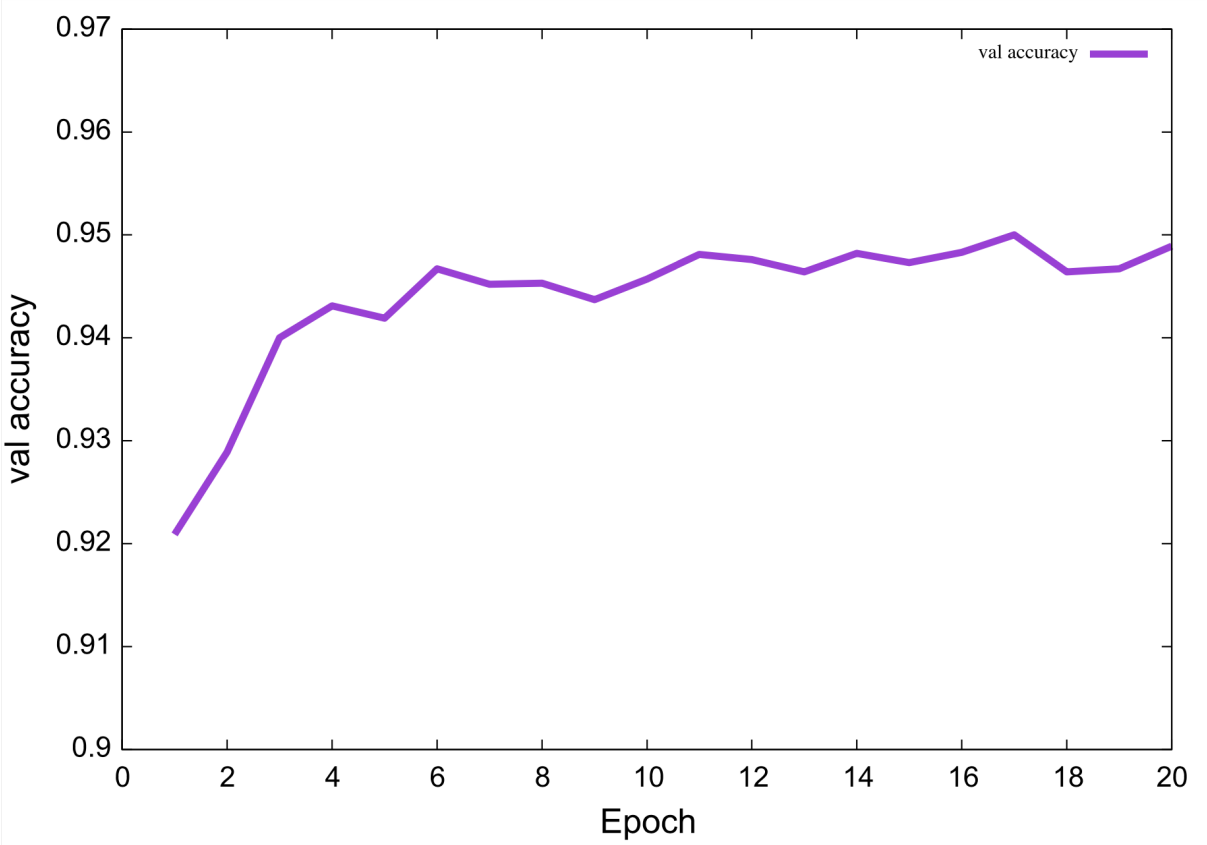


表4は4つ目のモデルと実行結果を示している。Test lossが0.1851、Test accuracyが0.9448という結果になった。また、図4のグラフにEpochごとのテスト精度を示した。x軸がEpoch数で、y軸にテスト精度である。

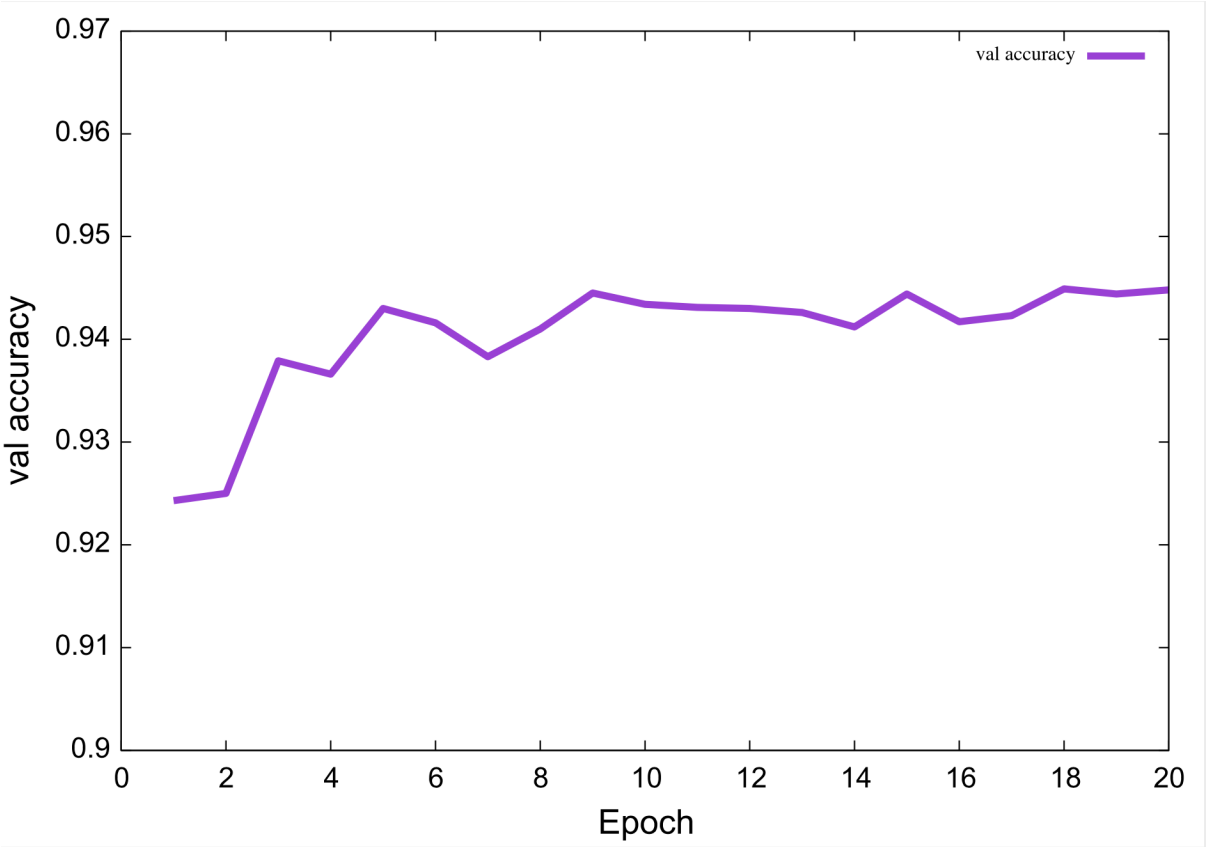
表4 4つ目のモデルと実験結果

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 28, 28, 32)	2624
conv2d_1 (Conv2D)	(None, 28, 28, 32)	82976
conv2d_2 (Conv2D)	(None, 28, 28, 32)	82976
max_pooling2d (MaxPooling2D)	(None, 6, 6, 32)	0
dropout (Dropout)	(None, 6, 6, 32)	0
conv2d_3 (Conv2D)	(None, 6, 6, 64)	165952
conv2d_4 (Conv2D)	(None, 6, 6, 64)	331840
conv2d_5 (Conv2D)	(None, 6, 6, 64)	331840
max_pooling2d_1 (MaxPooling2	(None, 2, 2, 64)	0
dropout_1 (Dropout)	(None, 2, 2, 64)	0
flatten (Flatten)	(None, 256)	0
dense (Dense)	(None, 128)	32896

```
_________________________________________________________________
dropout_2 (Dropout)          (None, 128)               0
_________________________________________________________________
dense_1 (Dense)              (None, 27)                3483
=================================================================
Total params: 1,034,587
Trainable params: 1,034,587
Non-trainable params: 0
_________________________________________________________________

Test loss: 0.1851
Test accuracy: 0.9448
```

図4 4つ目のモデルのEpochごとのテスト精度



3.2 特徴量選択

画素毎の明度を特徴として学習した場合と、局所方向寄与度特徴(LDC特徴)を用いて学習した場合にどちらが高い精度を出すかKNNを用いて比較する。

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

画素毎の明度を特徴として学習した場合と、LDC特徴を用いて学習した場合にどちらが高い精度を出すかSGDを用いて比較する。

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

4. 考察

4.1 それぞれのモデルのみで考察

・CNN

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。

表1で示した1つ目のモデルから表3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。

図1、3、4のグラフから、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。

・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。

4.2 2つ方法の結果を踏まえて

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。

モデルのコードの最適度にもよるが、本実験で作成したCNNのモデルは約2時間半、長いモデルで約13時間程度で学習が終了したのに対し、特徴量選択で作成したモデルは学習時間が長くなり、数日間の学習時間になることから、文字認識の学習にはCNNが効率的だと言える。

5. 意図していた実験計画との違い

実験を行う前に立てた計画

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。

実際に進行した実験スケジュール

1週目

- 教師データを探して、データの中身を確認した。深層学習班と特徴量選択班に分かれた。

2, 3週目
- 班ごとに分かれて調べ、でてきた参考コードのコード読みを行った。

4週目
- 調べながらpythonコードを書き始めた。

5週目
- pythonコード書きの続き。エラーの解消。

6週目
- pythonコード完成、singularityでの環境構築。

7,8週目
- 実行結果を元にモデルの改善、再実行。

9週目
- レポート、プレゼン資料作成。

実験を実際に行ってみて、調べて、理解するまでに想定より時間がかかったこと、そして、pythonコードの作成でエラーの解消などに時間がかかったことによって、計画とのズレがでたと考える。
また、最初は2つの班に分かれることを想定していなかったため、2つの班に分かれることが決まった時点で、再度班ごとに計画の練り直しを行ってもよかったと考える。

6. まとめ

CNNの精度が約95%あり、KNNは2種類の特徴量のうち最高精度で約86%となり、SGDの2種類の特徴量のうち最高精度で約82%という結果から文字認識の学習においてはCNNが効率的な学習器であることがわかる。
また2種類の特徴量を用いた結果から、学習器の違いによって、精度や学習時間に大きな差が出る。今実験では、KNNに対しては明度特徴量を用いて、SGDに対してはLDC特徴を用いることでより高精度の学習が行える。

7. 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。
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。また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