CH-005

紙面の色と見開きの歪みを考慮した光学文字認識の実装と評価
Implementation and Evaluation of Optical Character Recognition Considering Color of Papers and Distortion of Spread Books

比留川 翔哉 *　　丸山 一貴 †
Shoya Hirukawa　　Kazutaka Maruyama

1　はじめに

近年 Google Cloud Vision API, Adobe Acrobat など, 文字を含む画像から文字コードに変換する光学文字認識 (Optical Character Recognition, 以下,「OCR」という.) サービスやアプリケーションが存在する. また, Evernote, Google Drive などのクラウドにアップロードすることで OCR を使用できるサービスも多く存在する. ほとんどの OCR は歪んだ画像 (図 1) を文字認識する場合, 文字認識精度を上げるために平滑化や歪み補正を行う.

平滑化とは, 認識精度低下につながる画像内のノイズを除去する手法である. 主にガウシアンぼかしや K-最近傍法を使用する. 平滑化を適切に行わなければ, 文字形状の特徴量欠落や, ノイズ増加など, 認識精度低下の原因になる.

歪み補正とは, 歪んだ画像を学習データや事前に取得した特徴量に近似させることである. 歪み補正を適切に行わなければ, 新たな歪みの原因や, 適切ではない文字抽出の原因, 違う文字に近似する原因になり, 認識精度が低下する. そのため, 画像ごとに手動での平滑化を行うことや, 歪み補正のパラメータを変更することで認識率が改善するが, 大量の画像に手動で補正をかけることは難しい. 加えて, 既存の OCR の補正は十分ではないため, 認識精度の向上は難しい.

本研究では, 誤認識の一因となる補正を行わずに, 書籍の厚みにより歪んだ文字を認識する OCR を実現することを目的とする. 本研究の OCR は, フォントデータから作成した原画像を生成型学習法を用いて歪ませ, 畳み込みニューラルネットワーク (以下,「CNN」という.) で学習した学習データを使用する.

* 明星大学大学院 情報学研究科
Graduate School of Information Science, Meisei University
† 明星大学 情報学部
School of Information Science, Meisei University

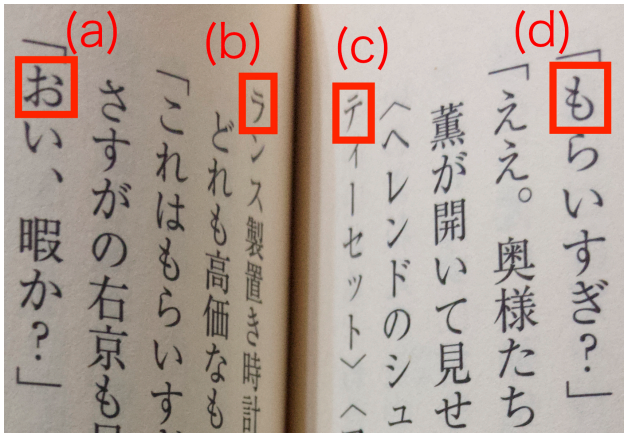


図 1　本の厚みで湾曲した画像 [1]

2　関連研究

荒木ら [2] は, 書籍を裁断せずにスキャンし, ページ外形や文字行を利用して, 本の厚みによる歪みや輝度・文字ボケの補正を行う手法を提案している. しかしこの手法では, スマートフォンのカメラで撮影した画像や, 白以外のページ色の書籍では正しく補正を行えない. OCR に使用する補正方法を提案しているため, 本研究とはアプローチが異なる.

志久ら [3] は, 傾斜文字を補正して文字認識を行う手法を提案している. この手法は, 傾斜文字の文字枠を作成し, その文字枠を正方形に補正する. その後に, CNN を用いて文字認識を行っている. しかしこの手法では, 三次元の変形や本の厚みによる湾曲した歪みなどの文字の認識は行えない. 成田ら [4] は, スキャナで取得した歪みのない英数字の文字画像を x,y,z 軸で回転させ, これらを学習し, 看板の文字を認識する手法を提案している. しかし, この手法では, スキャナで歪んでいない文字を取得させてから学習データを作成するため, 認識させる文字種を増やすことが困難である. また, 三次元の回転以外の湾曲した歪みなどの文字認識は精度が低下する. 本研究は, 補正を行わずに文字認識する手法を提案しており, 想定する歪みも異なる.

Ofusa ら [5] は, フォントのグリフを変形させ, 学習データを増やし, CNN の認識精度を向上させる手法を提案している. フォントデータを用いて CNN で文字を学習するという点では本研究と同じだが, この手法は補正後の文字認識の学習データの拡張のため, 本研究が対象にしている歪みには有効ではない. また, フォントデータ内のグリフデータを利用して変形させているため, 画像を用いて変形させる本研究の手法とはアプローチが異なる.

石田ら [6] は, 携帯電話のカメラの画像の文字を, 生成型学習法を用いて, 光学的ぼけ, ぶれ, 低解像度に対応する学習データを作成し, 部分空間法と複数フレーム認識, カメラ速度を用いた識別法を使用して文字認識を行う方法を提案している. 本研究とは, あらかじめ歪みなどの悪条件に対応する画像を生成し, 学習する点では共通するが, 学習方法に CNN を使用している点と, 対象にする歪みを湾曲にする点で異なる.

3 提案手法

本研究の提案手法は, 原画像をフォントデータから作成し, 生成型学習法を用いて歪んだ学習画像を生成する. その後 CNN を用いて学習し, テスト画像の認識精度を算出する. この章では, (1) 生成型学習法を用いた学習画像生成, (2)CNN を用いた学習, (3) 学習データを用いた文字認識の認識精度算出について説明する.

3.1 生成型学習法を用いた学習画像生成

3.1.1 生成型学習法とは

画像認識や文字認識を行う場合, 認識させる画像を想定して学習画像を用意し, 学習させなければならない. 文字認識や識別する種類が多い画像認識では, 学習画像の撮影や取得が難しい. 本研究では, 様々な条件を考慮するため, 最大 1277 万 5644 枚の学習画像を用意する必要がある. しかし, 1000 万枚以上の学習画像を手動で撮影するのは非常に時間がかかるため困難である. この問題点を解決する手法が生成型学習法 [7] である. 生成型学習法は, 原画像を加工して学習画像を生成する手法である. 本研究は生成型学習法を用いて学習画像を生成する.

3.1.2 学習画像生成方法

本研究では, フォントデータから文字画像を作成し, その文字画像に紙面の色と見開きの歪みを考慮した条件を付与した学習画像を生成する. 見開きの歪みは, OpenCV の remap メソッドを用いて再現する. OpenCV の remap メソッドは, 入力画像の各ピクセルの座標を指定した座標に変形するメソッドである. remap メソッドは以下の式 [8] を使用する. dst は出力画像, src は変換元の画像, map_x と map_y はそれぞれ x 軸・y 軸方向へ歪ませる座標群である.

$$dst(x, y) = src(map_x(x, y), map_y(x, y))$$

表 1　テスト画像の撮影に使用した書籍

書籍のレイアウト	枚数	書籍の分類名
明朝体・背景色薄橙・文字色黒	69 枚	書籍 1[1](図 2(a))
ゴシック体・背景色白・文字色赤	26 枚	書籍 2[10](図 2(b))
ゴシック体・背景色白・文字色赤	31 枚	書籍 3[11](図 2(c))
明朝体・背景色薄橙・文字色黒	49 枚	書籍 4[12](図 2(d))

(a) 書籍 1 の文字画像の一例　(b) 書籍 2 の文字画像の一例

(c) 書籍 3 の文字画像の一例　(d) 書籍 4 の文字画像の一例

図 2　テストに使用した文字画像の一例

3.2 CNN を用いた学習

CNN は, 入力層 (Input Layer) と畳み込み層 (Convolution Layer), プーリング層 (Pooling Layer), 全結合層 (Fully Connected Layer), 出力層 (Output Layer) で構成される機械学習の一種である. 畳み込み層は, 入力画像に対して指定したフィルタを使用し畳み込み演算を行う層である. プーリング層は, 指定した演算方法で特徴量を残しつつサイズを小さくする層である. 全結合層は, 前層の全ユニットと繋がっており, 特徴量から入力画像の識別を行う層である.

CNN は, 合成性と移動不変性に強い特徴を持つ. 合成性は, 画像の特徴を層ごとに抽出し, 組み合わせて利用することである. これにより, 上位の層では輪郭を抽出し, 下位の層では詳細を抽出するなど層の構成を理解するのが容易である. 移動不変性は, フィルタを使用して特徴を抽出するため, 学習画像とテスト画像内にある検出物の位置変動に対して頑健であることを指す. そのため, 学習画像とテスト画像の位置を合わせる必要がない.

3.3 学習データを用いた文字認識の精度算出

本研究では, CNN を用いて作成した学習データと, ベースラインとして Tesseract[9] の認識精度を計測す

る．Tesseract は, セグメンテーションとコンテキストを使用しない設定を用いて計測する．

精度算出に使用するテスト画像は, 小説から撮影した文字画像と, 比較対象としてグレースケール化した画像, 2 値化した画像を使用する．文字画像の撮影に使用した書籍を表 1, 使用するテスト画像の文字の一例を図 2 に示す．書籍の文字を撮影した画像は, 文字の縦と横の最大値を 1 辺とする正方形に切り抜いた後, 128×128 ピクセルに正規化する．その後, グレースケール化と, 手動で最も良い 2 値化の閾値を決めて 2 値化を行う．以降, 撮影し正規化した画像をカラーテスト画像, グレースケール化した画像をグレースケールテスト画像, 2 値化した画像を 2 値テスト画像と呼ぶ．

4 実装

本章では, 目的となる書籍の歪みを無補正で認識する文字認識システムを実装するために, (1) 生成型学習法を用いた学習画像生成プログラム, (2)CNN を用いた学習プログラム, (3) 文字を認識する認識プログラムについて説明する．

4.1 生成型学習法を用いた学習画像生成プログラム

本手法では原画像を作成し, 指定した歪みを再現する式で作成した座標と OpenCV の remap メソッドを使用して歪み画像を生成する．原画像の作成に使用した設定を表 2 に示す．解像度とフォントサイズの対応は, 128×128 ピクセルの場合は 112pt, 64×64 ピクセルの場合は 56pt, 32×32 ピクセルの場合は 38pt とする．解像度は全て 128×128 ピクセルに正規化し, 低解像度に対応する．背景色と文字色の対応は, 白・黒, 黒・白, 薄橙・黒とする．未学習の文字色の認識率を調査するため, 文字色が赤の条件は追加しない．作成する歪み画像の文字種を表 3 に示す．本研究では, 作成した原画像と歪み画像の 2 種類を学習画像とする．

remap メソッドは, 座標をもとに変形するため座標を計算する必要がある．remap メソッドで使用した, 歪みを再現する式は以下の通りである．

$$map_x(x,y) = \frac{x^{\gamma_x+1}}{w^{\gamma_x}} \tag{1}$$

$$map_x(x,y) = \left|\frac{(w-x)^{\gamma_x+1}}{w^{\gamma_x}} - w + 1\right| \tag{2}$$

$$map_x(x,y) = x \tag{3}$$

$$map_y(x,y) = y - \frac{x^2}{h \cdot (n-\gamma_y)} \tag{4}$$

$$map_y(x,y) = y - \frac{(h-x)^2}{h \cdot (n-\gamma_y)} \tag{5}$$

$$map_y(x,y) = y \tag{6}$$

表 2　生成する学習画像の条件

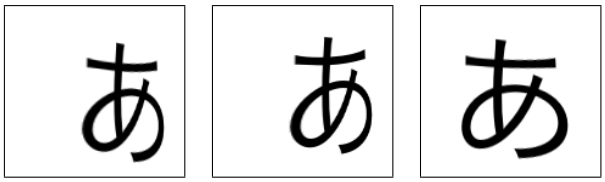
条件名	使用する条件
文字色	白・黒
背景色	白・黒・薄橙
フォント	ゴシック体・明朝体
フォントサイズ (pt)	112, 56, 38
解像度 (pixel)	128×128, 64×64, 32×32

表 3　使用文字種と文字数

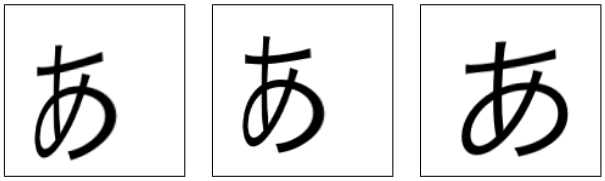
文字種		文字数
ひらがな カタカナ	清音	46 文字 ×2 種類
	濁音・半濁音	25 文字 ×2 種類
	拗音 (ゃゅょ)	3 文字 ×2 種類
	促音 (っ)	1 文字 ×2 種類
	合計	150 文字
アルファベット	大文字	26 文字
	小文字	26 文字
常用漢字 [13]		2136 文字
アラビア数字		10 文字
記号 、 。 ・ - & % 「 」( )		10 文字
合計		2358 文字

表 4　想定する歪みと使用した式と書籍の歪みの割り当て

歪みの種類	式番号	想定する歪み
右下に湾曲した歪み	(1),(4)	図 1(a),(b)
左下に湾曲した歪み	(2),(5)	図 1(c),(d)
右に寄せた歪み	(1),(6)	図 1(a),(b)
左に寄せた歪み	(2),(6)	図 1(c),(d)
右側のみ湾曲した歪み	(3),(4)	図 1(a),(b)
左側のみ湾曲した歪み	(3),(5)	図 1(c),(d)



(a) 右下に湾曲した歪み　(b) 右に寄せた歪み　(c) 右側のみ湾曲した歪み



(d) 左下に湾曲した歪み　(e) 左に寄せた歪み　(f) 左側のみ湾曲した歪み

図 3　生成した歪んだ学習画像の一例

w は画像の幅, h は画像の高さ, γ_x と γ_y は歪み定数である．歪み定数を変更することにより歪みの強弱を変えることができる．これらの数式は, remap を用いて生成した画像を目視で確認し, 書籍の歪みに近似しているものを使用した．作成した歪み画像の一例を図 3 に示す．実際の本の画像と想定した歪みを図 1 と表 4 に示す．

本研究では, 歪んだ文字の認識に必要な条件を選定す

表 5　生成に使用した条件と条件設定した学習データ名

使用する条件			学習データ名
背景色・文字色	フォント名	解像度	
白・黒	ゴシック体	128 × 128	学習データ 1
白・黒	ゴシック体	全通り	学習データ 2
白・黒	全通り	128 × 128	学習データ 3
白・黒	全通り	全通り	学習データ 4
白・黒, 薄橙・黒	ゴシック体	128 × 128	学習データ 5
白・黒, 薄橙・黒	ゴシック体	全通り	学習データ 6
白・黒, 薄橙・黒	全通り	128 × 128	学習データ 7
白・黒, 薄橙・黒	全通り	全通り	学習データ 8
全通り	ゴシック体	128 × 128	学習データ 9
全通り	ゴシック体	全通り	学習データ 10
全通り	全通り	128 × 128	学習データ 11
全通り	全通り	全通り	学習データ 12

るために, 様々な条件で生成した文字画像を学習データごとに割り当てる．生成に使用した条件と条件設定した学習データ名を表 5 に示す．表 5 の使用した条件は, 左から背景色・文字色, フォント, 解像度 (単位:ピクセル) の順である．またフォント名は, ヒラギノゴシック体 ProN W3 を「ゴシック体」, ヒラギノ明朝体 ProN W3 を「明朝体」と記述する．「全通り」と記述されている場合, 表 2 の全てのパラメータで作成することを指す．

1 つの条件で生成する学習画像の枚数は, 1 つの歪みの種類につき, 歪み定数が異なる画像を 50 枚ずつ作成するため, 6 つの歪みの種類と原画像 1 枚の合計 301 枚である．最小の条件の画像を学習する学習データ 1 の学習画像枚数は, 1 文字に対して 903 枚なので, 212 万 9274 枚, 全条件の画像を学習する学習データ 12 の学習画像枚数は, 1 文字に対して 5418 枚なので, 1277 万 5644 枚である．

4.2　CNN を用いた学習プログラム

本研究で使用した CNN を用いた学習プログラムは, Keras(Ver.2.2.4) と TensorFlow(Ver.1.11.0) を用いて CNN を実装する．TensorFlow を GPU を用いて演算する Cuda のバージョンは, 9.0.176 である．本研究で使用した層は VGG16[14] のモデルである．VGG16 は, 畳み込み層 13 層と総結合層 3 層の計 16 層で構成されたモデルである．VGG16 は 1000 クラスを分類するモデルだが, 学習を行うことにより 2358 クラスの分類が可能である．VGG16 の重みの初期値は, ImageNet の大規模な画像データセットを使用して訓練した重みを使用した．epoch 数は 35000 回, 学習率は 0.1%, Minibatch は 64 とする．epoch 数と学習率と Minibatch の値は, 事前実験で最も認識率が高い値を使用した．学習させる層は全層である．

4.3　文字を認識する認識プログラム

本手法の文字認識を行うプログラムは, 第 4.2 節で学習を行った学習データ 1 から学習データ 12 と Tesseract を使用して認識精度を計測する．使用する画像は第 3.3 節で記述した書籍の文字画像を使用する．CNN の学習データを用いた文字認識では, 累積認識率 (第 3 位累積認識率) も算出する．第 3 位累積認識率とは, 精度の上位 3 位までに正しい文字認識が行えた割合である．Tesseract は, 第 3 位累積認識率まで算出できないため, 提案手法のみ第 3 位累積認識率を算出した．

表 6　カラーテスト画像を用いた提案手法と Tesseract の文字認識の精度

文字認識	書籍 1	書籍 2	書籍 3	書籍 4	平均
学習データ 1	26.1%	15.4%	0.0%	12.2%	13.4%
学習データ 2	29.0%	23.1%	9.7%	16.3%	19.5%
学習データ 3	33.3%	15.4%	3.3%	12.2%	16.1%
学習データ 4	81.2%	73.1%	16.1%	53.1%	55.9%
学習データ 5	40.6%	23.1%	3.3%	8.2%	18.8%
学習データ 6	33.3%	30.8%	12.9%	18.4%	23.8%
学習データ 7	46.4%	19.2%	6.5%	18.4%	22.6%
学習データ 8	66.7%	50.0%	16.1%	28.6%	40.3%
学習データ 9	23.2%	23.1%	6.5%	6.1%	14.7%
学習データ 10	37.7%	23.1%	12.9%	14.3%	22.0%
学習データ 11	33.3%	11.5%	3.2%	12.2%	15.1%
学習データ 12	66.7%	34.6%	12.9%	30.6%	36.2%
Tesseract	76.8%	65.4%	80.6%	79.6%	75.6%

表 7　2 値テスト画像を用いた提案手法と Tesseract の文字認識の精度

文字認識	書籍 1	書籍 2	書籍 3	書籍 4	平均
学習データ 1	50.7%	42.3%	35.5%	44.9%	43.4%
学習データ 2	31.9%	34.6%	32.3%	32.7%	32.9%
学習データ 3	62.3%	42.3%	25.8%	28.6%	39.8%
学習データ 4	84.1%	76.9%	58.1%	53.1%	68.0%
学習データ 5	65.2%	53.8%	45.2%	36.7%	50.2%
学習データ 6	60.9%	46.2%	48.4%	46.9%	50.6%
学習データ 7	75.4%	57.7%	48.4%	38.8%	55.1%
学習データ 8	84.1%	73.1%	58.1%	44.9%	65.0%
学習データ 9	53.6%	38.5%	45.2%	34.7%	43.0%
学習データ 10	52.2%	46.2%	58.1%	34.7%	47.8%
学習データ 11	69.6%	53.8%	38.7%	38.8%	50.2%
学習データ 12	66.7%	61.5%	45.2%	42.9%	54.1%
Tesseract	74.0%	57.7%	87.1%	77.6%	75.6%

5　実験

本研究の実験では, 提案手法と Tesseract の精度を算出する．使用するテスト画像はカラーテスト画像とグレースケールテスト画像, 2 値テスト画像の 3 種類を使用する．精度は, 小数点以下第 2 位を四捨五入したものを使用する．カラーテスト画像を提案手法と Tesseract で文字認識した精度を表 6, 2 値テスト画像の認識結果を表 7, カラーテスト画像の提案手法の第 3 位累積認識率を表 8, 2 値化テスト画像の第 3 位累積認識率を表 9 に示す．グレースケールテスト画像は, カラーテスト画像の結果と平均 1% 未満の差しか確認できなかった．

表 8　カラーテスト画像を用いた提案手法の第 3 位累積認識率

文字認識	書籍 1	書籍 2	書籍 3	書籍 4	平均
学習データ 1	42.0%	19.2%	3.2%	24.5%	22.2%
学習データ 2	46.4%	34.6%	12.9%	32.7%	31.6%
学習データ 3	52.2%	19.2%	3.2%	14.3%	22.2%
学習データ 4	92.8%	76.9%	29.0%	65.3%	66.0%
学習データ 5	53.6%	30.8%	6.5%	18.4%	27.3%
学習データ 6	68.1%	42.3%	19.4%	28.6%	39.6%
学習データ 7	55.1%	23.1%	6.5%	28.6%	28.3%
学習データ 8	81.2%	65.4%	19.4%	44.9%	52.7%
学習データ 9	34.8%	26.9%	6.5%	10.2%	19.6%
学習データ 10	52.2%	38.5%	19.4%	26.5%	34.1%
学習データ 11	44.9%	19.2%	6.5%	14.3%	21.2%
学習データ 12	85.5%	57.7%	25.8%	42.9%	53.0%

表 9　2 値化テスト画像を用いた提案手法の第 3 位累積認識率

文字認識	書籍 1	書籍 2	書籍 3	書籍 4	平均
学習データ 1	71.0%	53.8%	51.6%	53.7%	57.4%
学習データ 2	50.7%	42.3%	45.2%	44.9%	45.8%
学習データ 3	73.9%	53.8%	38.7%	44.9%	52.8%
学習データ 4	91.3%	84.6%	64.5%	71.4%	78.0%
学習データ 5	75.4%	61.5%	58.1%	53.1%	62.0%
学習データ 6	72.5%	61.5%	64.5%	57.1%	63.9%
学習データ 7	85.5%	73.1%	54.8%	53.1%	66.6%
学習データ 8	88.4%	76.9%	83.9%	57.1%	76.6%
学習データ 9	73.9%	50.0%	58.1%	46.9%	57.2%
学習データ 10	65.2%	53.8%	74.2%	63.3%	64.1%
学習データ 11	82.6%	65.4%	67.7%	51.0%	66.7%
学習データ 12	79.7%	73.1%	58.1%	53.1%	66.0%

6　考察

6.1　学習データ間の精度の違い

本研究では, 学習画像を全てコンピュータで生成した学習画像を用いて文字認識を行った．その結果である表 6 から, 最も精度が高いものは学習データ 4 であった．学習データ 4 は背景色白, 文字色黒, フォント 2 種類, 解像度 3 種類の画像を VGG16 で学習させたものである．背景色と文字色のバリエーションを増やした学習データ 8 と学習データ 12 より精度が高いことから, 不必要な背景色と文字色のバリエーションの画像は学習するべきではないことがわかった．

学習データ 4 と Tesseract の精度を比較すると, 書籍 1 と書籍 2 では, 学習データ 4 の精度が高いが, 書籍 3 と書籍 4 では Tesseract のほうが精度が高い．このことから生成型学習法で生成する学習画像に使用した条件が不足している可能性がある．表 6 から考えられる生成画像に使用する条件は, 以下の通りである．

- 背景色・文字色を適切な色のみ学習すること
- 縦書き横書きの文字位置や角度を想定して画像の生成すること
- 様々なフォントで画像を生成すること
- 作成する画像の解像度を増やすこと
- 文字画像のデータ拡張を行うこと

書籍 3 と書籍 4 は白色の背景色, 赤色の文字色を使用しているため, 背景色白・文字色赤の条件を追加することが考えられる．しかし, この条件を追加したことにより, 書籍 1 と書籍 2 の精度が低下する可能性がある．そのため背景色・文字色の条件を追加や削除して精度に影響するか実験する必要がある．

縦書き・横書きによって位置が変わる, 拗音や句読点は全てフォントのデフォルト位置を使用して原画像を生成している．また, 書籍の文字画像は, 文字の縦と横の大きい方を 1 辺とする正方形に切り抜いた後, 正規化するため, 文字の位置は精度に影響しないと考えられる．しかし, 鍵括弧や長音などの縦書き横書きで向きが変わる記号は, 認識できないため, 原画像を回転させる, 縦・横書き用フォントを追加するなどの条件を追加して精度を比較する必要がある．

現在, 作成する画像の解像度は, 128×128・64×64・32×32 ピクセルで作成しているが, 小説の文字画像の一部は想定より解像度が低い画像あった．そのため, 32×32 ピクセルより低い解像度で学習画像を生成し, 精度を比較して必要な条件かどうかを判断する必要がある．またテスト画像の一部には, 手ブレによる文字の輪郭がはっきりしない文字画像が確認できた．そのためピントボケに対する画像を生成する手法 [6] を使用して実験すべきである．

画像認識の場合, 学習画像にノイズや変形, ヒストグラムの変更などのデータ拡張 (Data Augmentation) を行うのが一般的である．本研究では, 生成した画像のみ使用したため, データ拡張後の学習画像を用いて精度の変化を確認し, 必要なデータ拡張の条件も検証する必要がある．

6.2　CNN の層と学習数による精度の違い

本手法では, 1000 種類の物体検知に使用される VGG16 という層を使用した．文字認識は, 物体検知より画像の特徴がシンプルである．そのため文字認識に使用するには層が最適化されていないため, 最適な層を使用することで学習速度や, 認識速度・精度が向上する可能性がある．

6.3　各テスト画像による精度の違い

カラーテスト画像の精度である表 6 とグレースケールテスト画像を比較した場合, 平均 1% 未満の差しか確認できなかった．この結果から, グレースケールを用いる事は精度に影響しないことがわかった．原因として, CNN は, カラー画像を用いて畳み込んでいるが, 出力層ではグレースケール (チャンネル数が 1) であるためこのような結果になると考える．

カラーテスト画像の精度である表 6 と 2 値テスト画像である表 7 を比較した場合, 2 値化画像を用いた場合の精度のほうがカラーテスト画像より精度が高い．この結

果から, 生成型学習法による生成した学習画像の条件が不足していると考えられる. 例えば, ノイズを含む背景や, 背景色と文字色の境界線がピントボケして畳み込みに影響する画像の条件を追加するべきである.

本研究で使用した 2 値テスト画像は, 背景色白色, 文字色黒色にして使用した. 2 値化画像は黒以外のデータを使用するため, 背景の特徴を使用したことになる. そのため, 2 値化画像の白色と黒色を入れ替えた文字色の条件を追加するべきである. また, データ拡張を使用して学習画像を増やす必要がある.

6.4 文字種による精度の違い

精度が低い学習データは, 多くの文字種を記号と誤認識した. 提案手法の中で最も精度が良い学習データ 4 の誤認識は, ひらがなや画数の少ない漢字に多い. しかし, 歪み方やピントのボケ具合による影響が大きいため, 文字種による精度差は確認できなかった.

6.5 第 3 位累積認識率による精度向上

文字認識後に前後の文字と文脈の情報を用いて, 誤認識した文字を修正する手法がある. しかし, 正しい結果を誤修正する可能性があるため精度が低下する可能性がある. 表 8 から本研究の提案手法の学習データ 4 と, カラーテスト画像の第 3 位累積認識率は 92.8% である. そのため, 認識の信頼度が高い 3 文字を対象として自然言語処理による文脈情報と組み合わせることにより, 現在の認識精度より高い精度を出す可能性がある.

7 結論

本論文では, 本の厚みで湾曲した文字画像を無補正で認識するため, 生成型学習法を用いて学習画像を生成し, CNN を用いて文字認識する手法を提案した. 本手法は, フォントデータから作成した画像から, 本の厚みで湾曲した文字と近似した学習画像を作成し, 学習と文字認識を行った. 認識精度は, 書籍 1 と書籍 2 ではベースラインとした Tesseract より高い精度で認識できたが, 書籍 3 と書籍 4 では Tesseract より劣る結果となった.

今後の課題は, 生成型学習法の条件の追加や, CNN で使用する層の最適化, 前後の文字と文脈の情報を用いた修正を行い, 精度向上を目指すことである.

参考文献

[1] 脚本・興水泰弘ほか, ノベライズ・碇卯人, ” 相棒 season 4 下”, 朝日新聞社, pp.222–223, 2009.

[2] 荒木禎史, 関海克, 小島啓嗣, ”製本原稿スキャン画像の歪み補正技術”, Ricoh Technical Report, No.29, pp.31–40, 2003.

[3] 志久修, 手島裕詞, 内田誠一, ”傾斜文字認識のための正規化方法”, 電子情報通信学会論文誌 D, Vol.J100-D, No.10, pp.902–906, 2017.

[4] 成田了, 大山航, 若林哲史, 木村文隆, ”3 次元回転不変カメラベース文字認識”, 電気学会論文誌 C (電子・情報・システム部門誌), 133 巻, 4 号, pp.876–882, 2013.

[5] Kenichiro Ofusa, Tomo Miyazaki, Yoshihiro Sugaya, Shinichiro Omachi, ”Glyph-Based Data Augmentation for Accurate Kanji Character Recognition”, 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), Kyoto, pp.597–602, 2017.

[6] 石田皓之, 高橋友和, 井手一郎, 目加田慶人, 村瀬洋, ”携帯カメラ入力型文字認識におけるぼけやぶれに対処するための生成型学習法”, 電子情報通信学会論文誌 D, Vol.J89D, No.9, pp.2055–2064, 2006.

[7] 村瀬洋, ”画像認識における生成型学習法”, 情報処理学会研究報告コンピュータビジョンとイメージメディア, 91(2004-CVIM-145) 号, pp.211–218, 2004.

[8] OpenCV 2.4.13.7 documentation, Geometric Image Transformations, <https://docs.opencv.org/2.4/modules/imgproc/doc/geometric_transformations.html#remap> (参照 2019/06/18).

[9] GitHub, tesseract-ocr, <https://github.com/tesseract-ocr/> (参照 2018/11/29).

[10] 成美堂出版, ”リキュール・スピリッツでひけるカクテル BOOK300” 若松誠志監修, 成美堂出版, pp.120–121, 2008.

[11] 学科試験問題研究所, ”普通免許これだけマスター問題集”, 永岡書店, pp.126–127, 2011.

[12] 別冊宝島編集部, ”新装版 わかりたいあなたのための心理学・入門”, 宝島社, pp.204–205, 2007.

[13] 文化庁, 常用漢字表 (平成 22 年内閣告示第 2 号), <http://www.bunka.go.jp/kokugo_nihongo/sisaku/joho/joho/kijun/naikaku/kanji/index.html>(参照 2018/08/22).

[14] Visual Geometry Group, Very Deep Convolutional Networks for Large-Scale Visual Recognition, <http://www.robots.ox.ac.uk/~vgg/research/very_deep/> (参照 2018/11/28).