畳み込みニューラルネットワークを用いた
ひらがなのくずし字認識

筑波大学
図書館情報メディア研究科

2018年3月

原　皓

目 次

図 目 次

表 目 次

第1章　はじめに

1.1　本研究の背景

歴史を研究するには古文書の解読が不可欠である．近代以前における日本語の歴史的典籍の中で用いられた書体は、一般的に草書であり、それはくずし字と呼ばれる．くずし字とはその名のとおり、字を「崩し」ており、なおかつ複数の文字が連続して書かれているため、現代の一般人にとって認識することは難しい．したがって、古文書にあるくずし字は経験のある専門家によって解読を行う必要があるが、それ相応の費用と時間も必要となっている．

一方、機械学習の手法を用いた文字認識は、1980 年代から重要な研究領域になっている．それらはオンライン型文字認識とオフライン型文字認識に分けられている．オンライン型文字認識とは、書く人がコンピュータ、携帯電話などの機器を使用する際、キーボードではなく、専用のペンまたは指を使用して文字を入力し、機械がそれを認識するものである．オフライン型文字認識とは、手書き文字や印刷された文字を、イメージスキャナやデジタルカメラによって読みとり、二次元の画像に変換した後、機械が画像の中の文字を認識するものであり、いわゆる光学文字認識である．

オンライン型文字認識とオフライン型文字認識では識別対象が違うため、文字を識別する際、用いる手法も違う．オンライン型文字認識は書き順や位置によって文字を識別する．オフライン型文字認識では書き順の情報がなく、単純に二次元の画像から文字を認識することになる．その上、画像を獲得する環境によってノイズが入り、また元の媒体が損傷している場合もあるため、オフライン型文字認識はオンライン型文字認識と比べ、識別はより困難である．

オフライン型文字認識は印刷体文字認識と手書き文字認識に分けられる、手書き文字認識は個人差や字形が不規則などの原因で、また印刷体文字認識と比べ困難である．

オフライン型文字認識では、二つの段階に分けられる

1. 文字を画像の中から正確に切り出すこと
2. 切り出した文字を正確に識別すること

1 の文字の切り出しに対して、投影法、連結空間分析などの方法を用いてテキストを分割することがあり [1][2]、またウィンドウをある程度の幅で移動し、ウィンドウ内の文字を識別する手法もある [3]．2 の文字の識別に対して、特徴抽出の段階では Gabor 特徴 [4] や勾配に基づいた特徴抽出 [5] の手法があり、分類器としてサポートベクターマシン [6]、加重方向指数ヒストグラム法及びその改良版 [7][8] などの手法がある．

一般的な機械学習の手法を用いた手書きオフライン型文字認識は画期的な成果がなかったものの、近年、深層学習の発展とともに、手書きオフライン型文字認識にも新たな進歩がもたらされた．その中で、画像認識の分野に特化した畳み込みニューラルネットワークは、手書き数字認識 [9] を始め、画像の分類 [10][11]、オブジェクト検出 [12]、そして手書き漢字認識 [13] にも次々と成果が現れている．

1.2 本研究の目的

本研究の目的は畳み込みニューラルネットワークの手法を用いて、日本語のくずし字を認識することである．日本語のくずし字を機械学習の手法を用いて識別するにはいくつか難点がある．一つは言語としての難しさである．日本語は英語などの表音文字 (アルファベット) ではなく、表語文字 (漢字)、表音文字 (仮名) の混合システムである [14]．そのために英語のように 26 種のアルファベットを正確に識別することと比べ、日本語に使われる漢字と仮名の全文字種を正確に識別する必要がある．これは極めて困難なことである．もう一つはくずし字が一般的な手書き文字より認識が困難になっている．しかも古文書の保存状態により、獲得した画像はある程度のノイズや毀損が予想される．また、くずし字の特徴として、「仮名を表すときに漢字で書く」場合があり、「ふ」が不、婦、風、布などと書かれるように、仮名一種で文字として何パターンの可能性がある．またひらがなとカタカナを混用する場合もある．

本論文の着目点は、既に切りだされた古文書の日本語をまずひらがなの範囲で正確に認識することである．本研究は、人文学オープンデータ共同利用センターが提供する日本古典籍字形データセットを使用する．既に切り出したくずし字画像からひらがな画像データを研究対象として抽出し、畳み込みニューラルネットワークを用いてひらがなのくずし字を識別する手法を提案する．

1.3 本論文の構成

本論文 2 章以後の構成は以下のようになっている．2 章では畳み込みニューラルネットワークの基礎理論について述べる．3 章ではで本研究で使用するデータセットを紹介し、またデータの数を改変することで、データセットの中に各クラスの不均衡を改善する方法について述べる．4 章では、一般的なひらがなのくずし字を認識する基本手法を紹介した上で、データセットの不均衡問題を改善できる 3 種の提案手法について述べる．5 章では、実験環境及び実験プロトコルを述べた上、基本手法も含め、合計 4 種の手法に対し、実験を行い、考察を述べる．6 章ではまとめ、また今後の課題について述べる．

第2章　畳み込みニューラルネットワーク

本章では畳み込みニューラルネットワークの基礎理論について述べる．畳み込みニューラルネットワーク（Convolutional Nueral Networks、以下 CNN）とは、一種の特殊なニューラルネットワークであり、その特徴は畳み込み計算である．

畳み込みニューラルネットワークは階層的なモデルである．オリジナルのデータ、例えば画像のデータなどを入力し、それを畳み込みニューラルネットワークの畳み込み操作、プーリング操作、活性化関数などの操作を経由し、データの特徴を少しずつ取り出している．この過程は、順伝播（Feed-Forward）と呼ぶ．そして異なる操作は、畳み込みニューラルネットワークの中では異なる「層」で、畳み込み層、プーリング層などで表示されている、これらの「層」は畳み込みニューラルネットワークの中間層であり、層数を任意に指定できる、そして最後の層で具体的なタスク（分類、回帰）によって、違う目標関数（objective function、また Cost Function、Loss Function）に転換し、予測した値と本当の値の誤差（Loss）を計算し、誤差逆伝播法（Back-Propagation）によって、最後の層から順次に前に各層のパラメータを更新し、モデル全体が収束するまで反復し、モデルを訓練する．

本章の各小節では、畳み込みニューラルネットワークの各層について紹介する．

2.1　畳み込み層

畳み込み層（Convolutional Layer）は畳み込みニューラルネットワークの中で最も重要な層であり、この層で行っているのは畳み込み計算である．入力した画像が図 2.1 の真ん中の 5×5 の行列、対応したフィルタ（Convolution Kernel、Convolution Filter）は図 2.1 の左の 3×3 の行列に仮定し、同時に、一回畳み込み操作をした後、画素を一個移動する、すなわちストライド（Stride）は 1 にすると仮定する．

最初の畳み込み操作は画像の（0, 0）の画素から始め、フィルタの中のパラメータを画像の対応画素の値を相乗し、足すのが一回の畳み込み操作の結果となる、すなわち $1 \times 1 + 2 \times 0 + 3 \times 1 + 6 \times 0 + 7 \times 1 + 8 \times 0 + 9 \times 1 + 8 \times 0 + 7 \times 1 = 1 + 3 + 7 + 9 + 7 = 27$ である．また、ストライドは 1 であるため、フィルタは一つずつ右に移動し、計算する、最終的に図 2.1 の右の 3×3 の特徴が得られて、次の層の入力とする．

畳み込みは操作は、畳み込みのフィルタを使い、画像の局部の特徴を得られる．実際、この操作によって、画像中の物体のエッジを抽出することができ、畳み込みニューラルネットワークの中の異なる畳み込み層によって、異なる特徴を捉えられることができる．

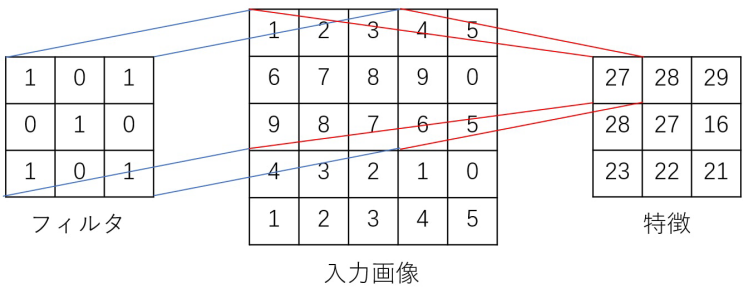

図 2.1: 畳み込み操作

2.2 プーリング層

プーリング層はプーリング操作を行う層である、プーリング操作は単純に、対象領域から最大値（もしくは平均値）を取る処理である．プーリング層には学習する必要があるパラメータがない、指定すべきパラメータはフィルタの大きさとストライドだけである、図 2.2 では、フィルタのサイズを 2、ストライドを 1 に設定した最大値を取るプーリング操作の例である．

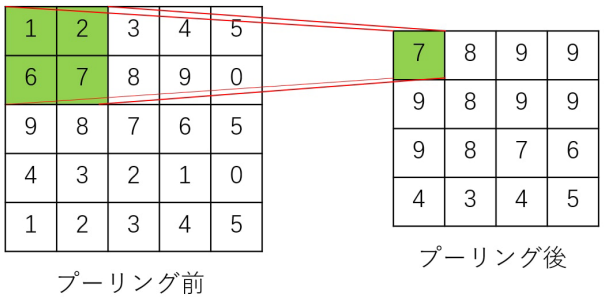

図 2.2: プーリング操作

プーリング層の操作は簡単であるが、その意義は大いにある．図で分かったことは、プーリングした出力が小さくなったが、元のデータの特徴を失ってはいない．また、データの次元数が減ることにより、全体的な計算量も同じく減らすことができる．かつ、ある程度の過学習を防ぐこともできる．

2.3 活性化関数

活性化関数（Activation Function）とは、学習するパラメータを非線形に変換するために使用する関数である．活性化関数を使用する理由は、モデルの表現力を強めることができる．活性化関数は多種ある、ニューラルネットワークの中で最も一般的に使われている活性化関数は

sigmoid(図 2.3) である．しかし、ネットワークの階層が深くなることにつれ、Sigmoid 関数には問題が生じる．誤差逆伝播の時、前の層にその誤差を伝播しにくくなることによって、モデル全体の更新が止まり、訓練できなくなる可能性がある．そのため、新しい活性化関数 ReLU（Rectified Linear Unit）[15]（図 2.4）が提案された．今の畳み込みニューラルネットワークの活性化関数には基本的に ReLU を使用している．本研究も ReLU を使用する．

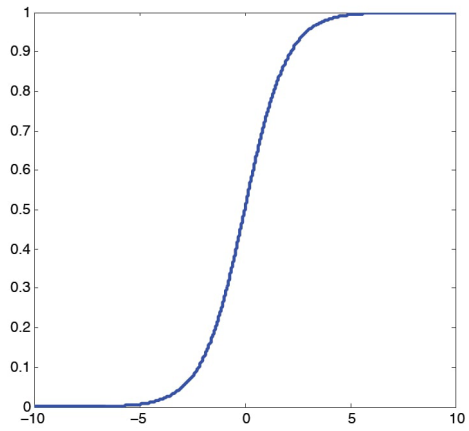

図 2.3: Sigmoid

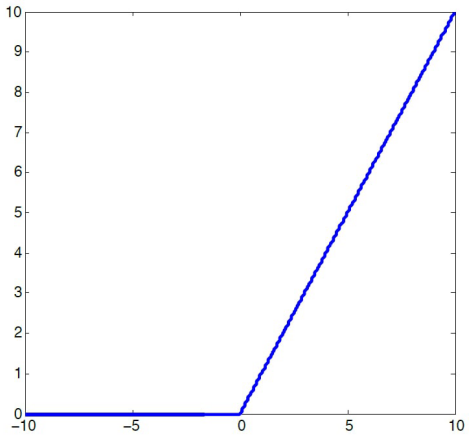

図 2.4: ReLU

2.4 全結合層

全結合層（Fully Connected Layers）は、一般的には畳み込みニューラルネットワークの最後に置き、「分類器」として働いている．畳み込み層、プーリング層、活性化関数の役割はオリジナルのデータから特徴を抽出することに例えると、全結合層の役割は、その特徴の結果を反映することである．一般的には、全結合層は普通のニューラルネットワーク層である．また、本研究のような分類タスクにおいて、畳み込みニューラルネットワークの最終層は、前の全結合層の後にクラス数と同数の全結合層を配置し、活性化関数はソフトマックス関数（Softmax）を用いる．そして、確率が一番高い値を取り、One-Hot Vector に変換し、1 になるクラスを、正答クラスとして予測する．

2.5 Dropout

ニューラルネットワークは表現力が高いモデルなので過学習しやすい、そのため過学習制御する必要がある．そして、Dropout は、過学習を防ぐために最も一般的に使われている方法であり、近年のニューラルネットワーク研究では基本的に Dropout を使用している．[16]

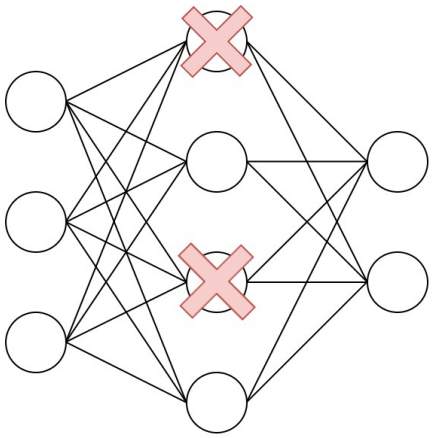

図 2.5: Dropout

Dropout は、ニューラルネットワークの中のニューロンを、ランダムに消去しながら学習する手法である．モデルを訓練する時、畳み込み層や全結合層のニューロンをランダムに消去する、そこで消去されたニューロンには、信号が伝達できないことによって、更新がその回では停止する（図 2.5）．テストの時には、すべてのニューロンの信号を伝達し、加算する．すなわち、訓練の段階では、わざとニューロンに忘れるということである．こうすることで、過学習を防ぐことができる．しかし Dropout には、モデルの収束速度を緩めるという欠点もある．

2.6 最適化方法

ニューラルネットワークの学習の目的が、目標関数の値をできるだけ小さくするパラメータを見つけることであり、すなわち最適なパラメータを見つける過程である．最適なパラメータを見つけるために、パラメータの勾配を用いて、勾配方向にパラメータを繰り返し更新し、徐々に最適なパラメータへと近づけてることは、最適化方法である．これまでの研究では様々な手法が提案されていた．最も一般的な方法は確率的勾配降下法（Stochastic Gradient Descent）という方法で、そのほかにも、Momentum、AdaGrad、Adam などの方法がある．本研究では、Adam[17], を使用する．

第3章　データの前処理

本章では、まず本研究で使用するデータセットについて紹介する．その上データの数を改変することによって、データセットの中に各文字種の不均衡問題を改善する方法について述べる．

3.1　データセットの紹介

本研究は、人文学オープンデータ共同利用センター [18] が公開した日本古典籍字形データセットを使用する．日本古典籍字形データセットは、日本古典籍データセット [19] で公開されるデジタル化された古典籍を中心として、翻刻テキストを制作する過程とともに、機械や人がくずし字を学習するために制作したデータセットである．

	書名	文字種	文字数
1	好色一代男	1,729	63,959
2	おらが春	1,119	11,197
3	雨月物語	1,969	44,832
4	当世料理	417	4,871
5	養蚕秘録	1,758	32,525
6	万宝料理秘密箱	843	24,480
7	膳部料理抄	704	11,397
8	料理物語	580	19,575
9	かてもの	430	5,599
10	日用惣菜俎不時珍客即席庖丁	595	9,046
11	料理方心得之事	330	3,003
12	新編異国料理	693	4,259
13	料理秘伝抄	255	9,545
14	物類称呼	2,197	75,462
15	比翼連理花迺志満台	1,972	83,492
	合計	3,999	403,242

表 3.1: 日本古典籍字形データセットが利用した書名

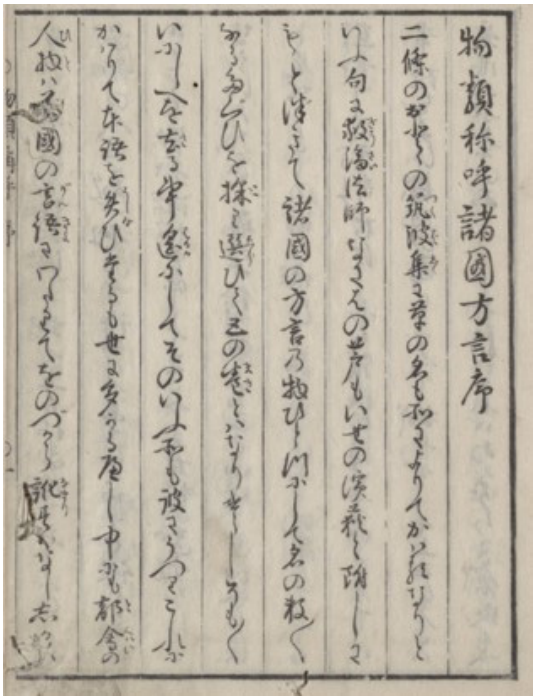

図 3.1: 原本補正画像データ

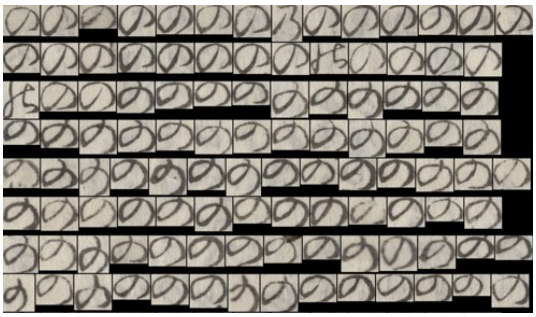

図 3.2: 字形画像データ

2017 年 6 月の時点で、日本古典籍字形データセットの規模は、国文学研究資料館所蔵で日本古典籍データセットにて公開する古典籍、および国文学研究資料館の関係機関が公開する古典籍 15 点の画像データから切り取ったくずし字で、3,999 文字種の字形データ、403,242 文字である．表 3.1 に日本古典籍字形データセットに含まれる古典籍の書名、各古典籍の文字種及び文字数を示す.

日本古典籍字形データセットは、以下 4 種のデータから構成される

1. 原本補正画像データ（図 3.1）
2. 文字座標データ
3. 字形画像データ（図 3.2）
4. 作業報告文書

1 の原本補正画像データは元の古典籍の画像に対して、翻刻作業を容易にするための前処理として、ページごとに分離するとともに、回転や正立などの処理を加えた画像である．2 の文字座標データは各文字を取り囲む長方形の座標、文字の Unicode、本の ID 及び文字の ID を記録したデータである．3 の字形画像データは 2 に基づいて、画像から各文字ごと切り抜いて、古典籍を単位として、文字種ごとに分類した画像である．4 の作業報告文書は翻刻作業で読めなかった文字に関する情報や、その他の注意事項が記録されている．

3.2 データの全体像

unicode	平仮名	画像数	unicode	平仮名	画像数	unicode	平仮名	画像数
U+3042	あ	3,697	U+305F	た	5,684	U+3079	べ	1,181
U+3044	い	7,022	U+3060	だ	1,356	U+307A	ぺ	18
U+3046	う	3,662	U+3061	ち	1,274	U+307B	ほ	1,122
U+3048	え	440	U+3062	ぢ	227	U+307C	ぼ	460
U+304A	お	2,734	U+3064	つ	3,618	U+307D	ぽ	17
U+304B	か	8,439	U+3065	づ	646	U+307E	ま	4,610
U+304C	が	3,204	U+3066	て	12,976	U+307F	み	1,586
U+304D	き	5,078	U+3067	で	1,349	U+3080	む	967
U+304E	ぎ	556	U+3068	と	11,068	U+3081	め	1,886
U+304F	く	5,447	U+3069	ど	1,622	U+3082	も	7,491
U+3050	ぐ	406	U+306A	な	7,892	U+3084	や	3,292
U+3051	け	2,519	U+306B	に	15,982	U+3086	ゆ	980
U+3052	げ	525	U+306C	ぬ	1,165	U+3088	よ	3,047
U+3053	こ	3,322	U+306D	ね	1,100	U+3089	ら	5,581
U+3054	ご	840	U+306E	の	14,337	U+308A	り	8,656
U+3055	さ	3,927	U+306F	は	8,725	U+308B	る	6,797
U+3056	ざ	639	U+3070	ば	2,137	U+308C	れ	3,897
U+3057	し	13,386	U+3071	ぱ	45	U+308D	ろ	1,257
U+3058	じ	1,060	U+3072	ひ	2,713	U+308F	わ	1,383
U+3059	す	3,946	U+3073	び	779	U+3090	ゐ	145
U+305A	ず	1,220	U+3074	ぴ	11	U+3091	ゑ	204
U+305B	せ	2,263	U+3075	ふ	4,742	U+3092	を	9,208
U+305C	ぜ	185	U+3076	ぶ	599	U+3093	ん	3,386
U+305D	そ	1,909	U+3077	ぷ	3			
U+305E	ぞ	507	U+3078	へ	4,121			

表 3.2: ひらがなのくずし字画像数

本研究では字形画像データ中の濁点・半濁点両方なしのひらがな 48 種、濁点ありのひらがな 20 種、半濁点ありのひらがな 5 種計 73 種のひらがなのくずし字画像データを使用する．字形画像データでは、切り出したくずし字の大小が異なるため、画像のサイズも全然違うである．また、全部の画像はカラー画像で、フォーマットは jpg である．

まず、前処理として、各古典籍の中から 73 種のひらがなのくずし字画像を抽出し、同じ文字種を 1 つのクラスにまとめる必要がある．表 3.2 では 15 点の古典籍からまとめた全 73 種のひらがなのくずし字画像の数を示した．また、図 3.3 では 73 種のひらがなのくずし字画像データの分布を示した．

これらより、一番データ数の多いクラス「に」の画像は 15,982 個、一番データ数の少ないクラス「ぷ」の画像はたっだの 3 個であり、各ひらがなのくずし字文字種の画像数は、極めて不均衡であることが分かる．

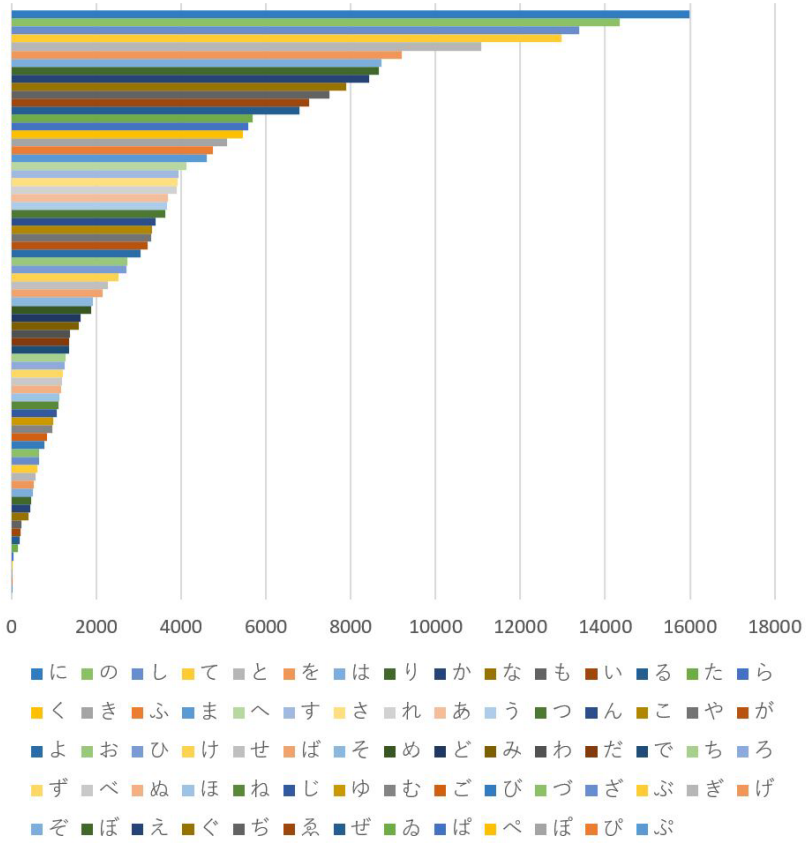

図 3.3: ひらがなのくずし字画像データ分布

Buda ら [20] の論文によれば、各クラスのデータが不均衡であることは、CNN を用いた分類タスクの性能に悪影響を及ぼすと言われている．また不均衡であればあるほど、与える悪影響も大きくなる．故に CNN を設計する前に、まずデータ不均衡の問題を解決する必要がある．

3.3 Under Sampling

Under Sampling は分類タスクのデータ不均衡問題に対して最も簡単な解決法である．Under Sampling とは、全体のデータが相対的に均等になるまで、データの多いクラスからランダムにデータを取り出すことである．しかし、Under Sampling はデータの多いクラスからランダムにデータを取り出すことで、有益なデータを損失する可能性がある．Under Sampling は分

類タスクのデータ不均衡問題に対して有効であるが、データが多いクラスの特徴を失うこともある．

また本研究においても、一方的に Under Sampling を行うのは現実的ではない．データ数が少ないクラスのデータは極めて少ないことである．もし他全てのクラスのデータをそのクラスの数に合わせて減少させると、分類することが難しくなる．

3.4 Over Sampling

Over Sampling は、Under Sampling と正反対の手法であり、データの少ないクラスから、全体のデータが相対的に均等になるまで、何らかの手法を用いて、データを増やすことである．深層学習の分野においては、Over Sampling は最も一般的な方法であり、データを増やす手法を総じて Data Augmentation と呼ぶ．[10][11][21][22] など CNN を使用した代表的な画像認識の研究では基本的に Data Augmentation を使用している．

Data Augmentation の中で最も簡単な手法はデータの少ないクラスからランダムにデータを取り出し、複製することであり、その有効性も証明されている、しかし過度のデータの複製は過学習を招くこともある [23][24] ため、画像認識における Data Augmentation は、複製だけでなく、画像を様々な方法で加工し、増やすことが一般的である．加工の手法として、画像のミラーリングがよく用いられる．その他、ランダムに画像を切り取り、画像の回転、画像の一部を変形するなどの手法もある．また、カラー画像であれば、チャンネル毎にランダムに数値を加えることも可能である．主成分分析の手法を用いて、より複雑な色変化を加えること [10] もできる．

しかし Data Augmentation にも限界が存在する．その原因は同じく過学習である．例え画像を加工しても、元の画像の数は限られており、少数のデータを過度の加工で増加させることは同じく過学習を招く可能性がある．また、本研究において、Data Augmentation の手法の大半は使用できない．文字認識において、色の情報は不要であり、全ての画像をグレースケール化し、認識しているため、色を改変する Data Augmentation は意味がない．そのほか、画像のミラーリング、過度の切り取り、回転、変形もできない．なぜなら、手書き文字の認識は一般的な物体認識と違い、画像の複雑度は高くないが、文字は形状だけで区別しているため、手書き文字のような個人差や年代差の原因で、変化が生じることがあるから、判別を間違ったことが起きやすい．画像のミラーリングを例にすると、猫の画像を増やすため、画像をミラーリングしても、その結果は猫の画像であるが．手書き文字だと、「さ」の字の画像をミラーリングすると、「ち」になる可能性が極めて高い．切り取りや回転にも、文字を判別する最も重要なところを失う可能性のあるため、過度な操作はできない．また、「あ」と「お」、「ね」と「ぬ」や「わ」と「れ」など元から似ているひらがなに過度な Data Augmentation を行うと、逆にデータに悪影響を与え、分類器の精度が下がる可能性がある．

3.5 Hybrid of Methods

Hybird of Methods とは、Under Sampling と Over Sampling を混合した手法で、多数派クラスのデータを Under Sampling でデータを減少させ、少数派クラスのデータを Over Sampling でデータを増加させることである．データセット不均衡の状況から考えて、Hybird Method は本研究において一番適切な方法であると判断した．しかし Hybrid of Methods はトレードオフが存在する．どの程度まで多数派クラスのデータを減少させるか、またどの程度まで少数派クラスのデータを増加させるか、そのバランスを取ることが一番重要である．

第4章　提案手法

本章では、第三章で紹介した手法を用いて、データセットを改善したことを前提に、ひらがなのくずし字を識別する基本手法を紹介した上、データセットの不均衡問題に対して改善を行う 3 種の手法を提案する．

4.1　基本手法

基本手法は、全 73 種のひらがなのくずし字を直接分類する手法であるこの手法を基本手法として設計した CNN の性能を、他の提案手法と対比する．図 4.1 では基本手法の CNN の訓練フローを示した．

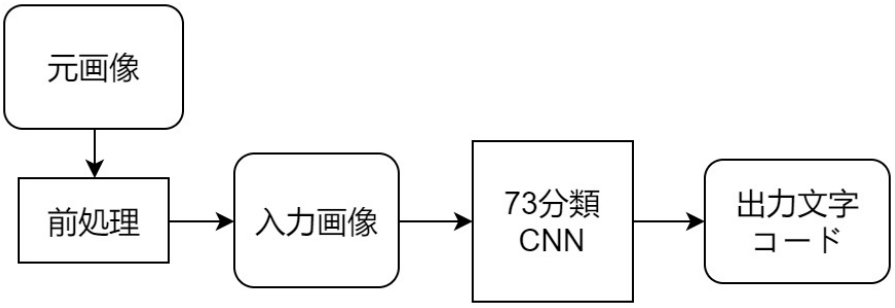

図 4.1: 基本手法の訓練フロー

元画像を前処理し、ラベルを付けた上で、データベースを作る、そして 73 分類の CNN がデータベースから画像を読み込んで、訓練する．これは、ひらがなのくずし字の分類タスクにとって基本的な手法であるが、欠点が存在する、この手法の前提条件として、全 73 種の文字種クラスのデータ数のバランスを保つ必要がある．しかし、第 3 章で述べたように、データの数を改変するだけでは、不均衡問題に対しての改善に限度がある．例として、データ数が 3 個しかない「ぷ」のクラスとデータ数が 15982 個ある「に」のクラスのバランスを取るのはあまりにも難しい．仮に各クラスのバランスを取っても、Under Sampling による特徴の損失と Over Sampling による過学習の可能性が大きい．

4.2　手法一　並列手法

基本手法の欠点を踏まえて、手法一の並列手法では、濁点のあるひらがな、半濁点のひらがな及び両方がないひらがなの判別と、ひらがなのくずし字の判別を分けて行う．詳細な理由は以下の 3 つである．

- 濁点や半濁点のあるひらがなは全体的にデータ数が少ない．例えば、「は」の画像データは 8,725 個に対して、「ば」は 2,137 個、「ぱ」は 45 個である．濁点のあるひらがな、半濁点のひらがな及び両方がないひらがなの判別と、ひらがなのくずし字の判別を分けることにより、データの不均衡を改善ができる．

- 濁点や半濁点のあるひらがなと、濁点や半濁点のないひらがなの違いは濁点、半濁点の有無だけであるから、形は似ている．そのため濁点のあるひらがな、半濁点のひらがな及び両方がないひらがなの判別と、ひらがなの判別を分けることは、基本手法のように直接分類のとくらべやさしいと考えられる．

- 分類タスクにおいて、クラスが多いほど、タスクも難しくなる．全 73 種のひらがな分類タスクを、濁点あり、半濁点ありと両方なしの 3 分類タスクと、濁点・半濁点を無視し、48 種のひらがなの分類タスクを分けることにで、ひとつの難しいタスクが 2 つの比較的簡単なタスクになり、全体としての精度が上がる可能性がある．

こうすることで、全体的にクラスの種類を減らし、1 クラス中のデータを増やすことができる．

この手法には、濁点と半濁点両方なし、濁点あり、半濁点あり、両方なしの 3 分類の CNN と、ひらがなを判別する 48 分類の CNN を両方訓練する必要がある．訓練のフローを図 4.2 に示した．

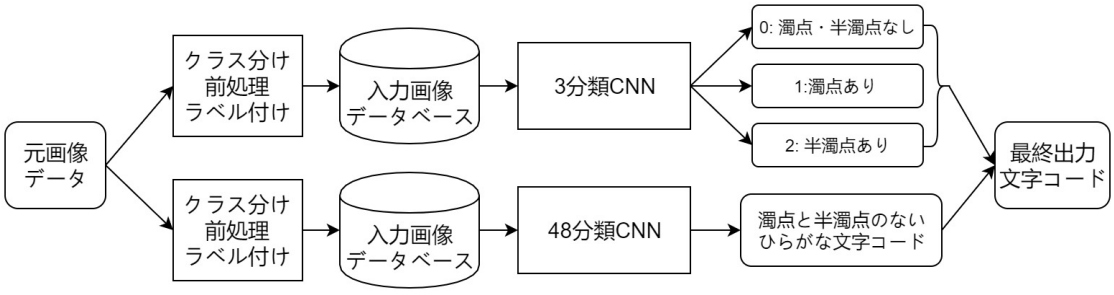

図 4.2: 並列手法の訓練フロー

元画像を前処理の段階で、2 個分のデータセットを良いし、それぞれ濁点あり、半濁点あり、両方なしの 3 分類の CNN とひらがなを判別する 48 分類の CNN を訓練するために異なるクラス分けを行う．その後ラベルを付けた上で、データベースを作る．そして両方の CNN

がデータベースから画像を読み込で、訓練する．最後に、両方の判別結果を合わせて、最終の出力とする．

この手法において、前処理段階の処理が最も重要である．48 種のひらがなを分類する CNN を訓練するには、濁点と半濁点のあるひらがなを、そのひらがなの濁点、半濁点がないひらがなのクラスにまとめる必要がある．例えば、「が」を全部「か」のクラスにまとめ、「ば」と「ぱ」を全部「は」のクラスにまとめる必要がある．

なぜなら、濁点や半濁点の有無の判別は 3 分類の CNN が行っているため、例え 48 分類の CNN に入れた画像が濁点や半濁点のあるひらがな画像でも、濁点や半濁点のない対応するひらがなに判別させれば良い．それ以外は、基本手法の 73 分類の CNN と基本的に同じである．

一方、濁点や半濁点の有無の判別は 3 分類の CNN を訓練するには、濁点のない全てのひらがなを濁点なしのクラス、すなわち「あ」行、「か」行、「さ」行、「た」行、「な」行、「は」行、「ま」行、「や」行、「ら」行、「わ」行及び「ん」が含めた全文字種を 0 のクラスにまとめる必要がある．同様に、濁点のある全てのひらがなを濁点ありのクラス、すなわち「が」行、「ざ」行、「だ」行、「ば」が含めた全文字種を 1 のクラスにまとめる必要がある．最後に半濁点のある全てのひらがなを半濁点のクラス、すなわち「ぱ」行が含めた 5 種のひらがな文字種を 2 のクラスにまとめる必要がある．

ひらがなを判別する 48 分類の CNN のクラス分けは、単純に文字種に基づいて分けているが、濁点と半濁点両方なし、濁点あり、半濁点ありの 3 分類の CNN のクラス分けは、表 4.1 に示した．

クラス名	濁点と半濁点両方なし	濁点あり	半濁点あり
ラベル	0	1	2
文字種	あ、い、う、え、お か、き、く、け、こ さ、し、す、せ、そ た、ち、つ、て、と な、に、ぬ、ね、の は、ひ、ふ、へ、ほ ま、み、む、め、も や、ゆ、よ、ら、り る、れ、ろ、わ、ゐ ゑ、を、ん	が、ぎ、ぐ、げ、ご ざ、じ、ず、ぜ、ぞ だ、ぢ、づ、で、ど ば、び、ぶ、べ、ぼ	ぱ、ぴ、ぷ、ぺ、ぽ
文字種数	48	20	5
データ数	228,683	19,498	94

表 4.1: 並列手法　3 分類 CNN の各クラスの文字種数及びデータ数

しかし、表を通じて、文字種数及びデータ数はまだ不均衡の問題が存在していることが分かった．そのため、3 分類する CNN を設計する前提は、3 つのクラスのデータ数をある程度

のバランスを保ちながら、1 クラス内の各文字種のデータ数もある程度のバランスを保つ必要がある．

また、1 クラスにおいて、特徴を捉えられない可能性が存在する、ラベル 0 のクラスを例として、このクラスの中のひらがなには統一した特徴は存在しない、唯一の特徴は濁点がないということである．CNN はうまくこのことを判別し、分類できるかどうかは、実験を行う必要がある．

4.3 手法二　順次手法 1

手法二の順次手法 1 は、手法一の並列手法のような、濁点と半濁点両方なし、濁点あり、半濁点ありの 3 分類 CNN と、濁点と半濁点を無視し、48 種のひらがなの分類 CNN を並列に訓練し、合わせた結果を出力する手法ではない．順次手法 1 では、まず 48 種のひらがなを判別した後、濁点と半濁点両方なし、濁点あり、半濁点ありを判別する手法であり、48 種のひらがなを判別した後、出力した文字コードに基づいて、濁点の有無を判別するか否かを決定する．

手法 2 には、48 種のひらがなを分類する CNN は並列手法と同様であるため、並列手法の CNN をそのまま使用する．濁点と半濁点両方なし、濁点あり、半濁点ありの判別を、ひらがなのくずし字を判別した後で行い、48 分類の CNN が、「あ」行、「な」行、「ま」行、「や」行、「ら」行、「わ」行及び「ん」の文字コードを出力したら、そのまま最終的な出力文字コードになる.「か」行、「さ」行、「た」行の文字コードを出力したら、2 分類の CNN に入れ、濁点の有無を判別する.「は」行の文字コードを出力したら、3 分類の CNN に入れ、濁点あり、半濁点あり、また両方なしかを判別する．

この手法を実行するには、3 分類の CNN、2 分類の CNN と 48 分類の CNN の 3 つを訓練する必要がある、提案手法の訓練フローは図 4.3 で示した．

元画像を前処理の段階で、 全ての元画像データから 2 分類と 3 分類の CNN を訓練するための画像（表 4.2、表 4.3）を抽出し、それぞれ異なるクラスに分ける．その後ラベルを付けた上で、データベースを作る.（48 分類の CNN は並列手法と同様のため、直接使用する．図で表示するが、実際訓練しない）そして両方の CNN がデータベースから画像を読み込で、訓練する．最後に、もし濁点・半濁点両方ないの文字コードが 48 分類の CNN から出力したら、直接最終の出力コードとする．濁点ありの文字コードが 48 分類の CNN から出力したら、濁点の有無を判別する 2 分類の CNN に入れ判別し、最終の文字コードを出力する．半濁点ありの文字コードが 48 分類の CNN から出力したら、濁点あり、半濁点あり、また両方なしを判別する 3 分類の CNN に入れ判別し、最終の文字コードを出力する．

手法 2 での 2 分類 CNN と 3 分類 CNN は基本的に提案手法 2 の 3 分類 CNN と同じであるが、各クラスの中の文字種数が違う．2 分類の CNN を訓練するために抽出した画像は、濁点なしのクラスでは「か」行、「さ」行、「た」行及び「は」行でのひらがな文字種計 20 種あり、濁点ありのクラスでは「が」行、「ざ」行、「だ」行、「ば」行の同じくひらがな文字種計 20 種ある．3 分類の CNN を訓練するために抽出した画像は、濁点と半濁点両方なしのクラスでは「は」行のひらがな文字種 5 種あり、濁点ありのクラスでは「ば」行のひらがな文字種 5 種

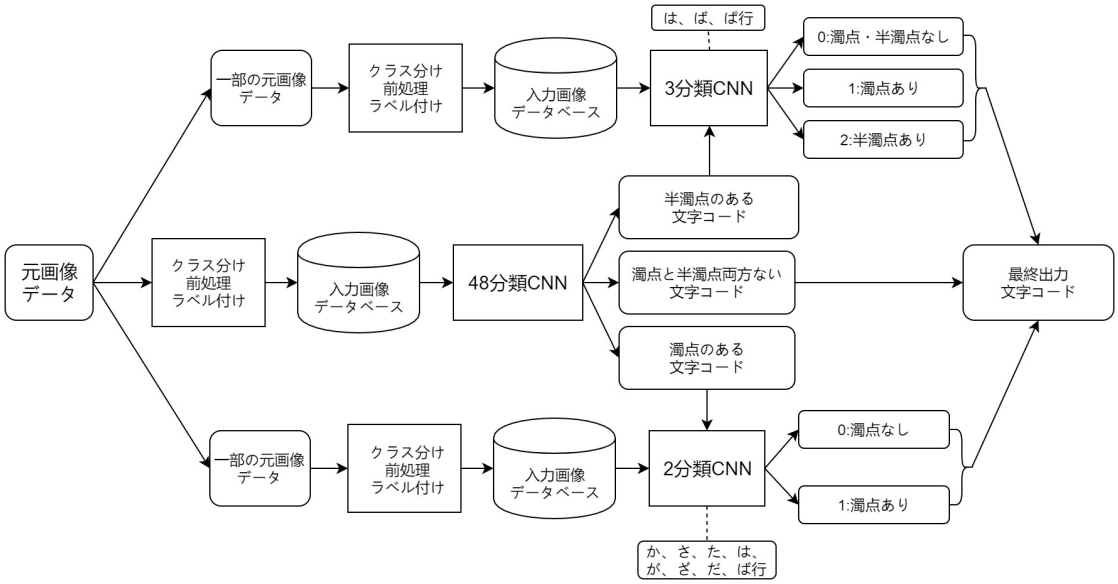

図 4.3: 順次手法 1 の訓練フロー

あり、半濁点ありのクラスでは「ぱ」行のひらがな文字種 5 種ある．表 4.2 と表 4.3 では、2 分類と 3 分類の CNN の中の各クラスのデータ数を示した．順次手法 1 と並列手法を比べ、利点は以下にある．

- 並列手法と比べ、順次手法 1 の 2 分類タスクの濁点なしの文字種が 48 種から 20 種に減り、濁点ありの文字種数と一致になった．また、3 分類タスクの濁点あり、半濁点あり、両方なしの文字種数は全部 5 種で、文字種数のバランスが良くなり、総体的なデータ数のバランスも良くなった．
- 「か」「さ」「た」「が」「ざ」「だ」行の各文字種が、3 分類から 2 分類になるため、タスクとして簡単になる．

クラス名	濁点なし	濁点あり
ラベル	0	1
文字種	か、き、く、け、こ さ、し、す、せ、そ た、ち、つ、て、と は、ひ、ふ、へ、ほ	が、ぎ、ぐ、げ、ご ざ、じ、ず、ぜ、ぞ だ、ぢ、づ、で、ど ば、び、ぶ、べ、ぼ
文字種数	20	20
データ数	106,279	19,498

表 4.2: 順次手法 1　 2 分類 CNN の各クラスの文字種数及びデータ数

クラス名	濁点と半濁点両方	濁点あり	半濁点あり
ラベル	0	1	2
文字種	は、ひ、ふ、へ、ほ	ば、び、ぶ、べ、ご	ぱ、ぴ、ぷ、ぺ、ぽ
文字種数	5	5	5
データ数	21,423	5,156	94

表 4.3: 順次手法 1　 3 分類 CNN の各クラスの文字種数及びデータ数

- 2 分類の濁点なし、濁点ありの判別も、3 分類の濁点あり、半濁点あり、両方なしの判別も、ひらがなのペア数は全て 1 対 1 になったため、特徴の特定は簡単になっていると推測できる.
- 2 分類の CNN では、「は」と「ば」の判別は行っていないが、CNN の訓練データとして入れるのは可能であり、2 分類の CNN の訓練データ数を増やすことができる.

4.4　手法三　順次手法 2

並列手法と順次手法 1 では、同じひらがなのくずし字に対して、そのひらがなの濁点や半濁点の有無を判別するのが難しいと仮定した上、提出した手法である．実際、同じひらがなのくずし字に対して、濁点や半濁点があっても、判別できる可能性のが高い、第 5 章の実験 5.4 の結果もこのことを証明した．しかし、基本手法のように直接 73 種のひらがなを分類することは現実ではない．そのため、基本手法の 73 種のひらがなを直接分類の方法と、並列手法と順次手法 1 のように、濁点や半濁点の有無を判別するの分けて分類の方法の間を取り、濁点のあるひらがな及び濁点と半濁点両方なしのひらがなを同じ段階で、すなわち合計 68 個の文字種を分類する、そして半濁点のみを分けて判別するのが、順次手法 2 である.

この手法を実行するには、2 分類の CNN と 68 分類の CNN を両方訓練する必要がある、この手法の訓練フローは図 3.2 で示した.

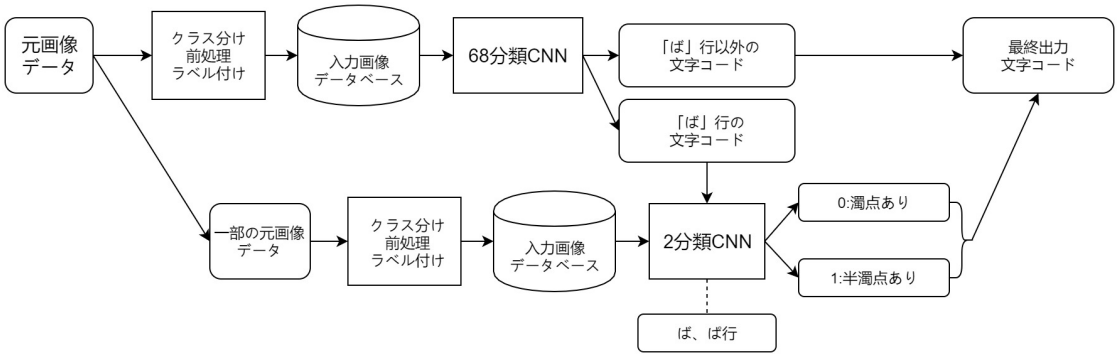

図 4.4: 順次手法 2 の訓練フロー

順次手法 2 と順次手法 1 の異なる点は 2 つある．1 つは、順次手法 1 の 48 分類 CNN と違い、順次手法 2 では、濁点のある 20 種のひらがなもひらがなのくずし字を判別する CNN に入れ、全体的に 68 分類の CNN を訓練することになった．もう 1 つは、半濁点のある「ぱ」行の 5 種のひらがなクラスを、対応な濁点のある「ば」行の 5 種のひらがなクラスにまとめる．そのため、2 分類 CNN の前処理の段階で、 全ての元画像データから訓練するための画像（表 4.3 の濁点ありのクラスと半濁点ありのクラス）を抽出し、クラスに分ける．その後ラベルを付けた上で、データベースを作る．68 分類の CNN の前処理では、「ぱ」行のデータを対応の「ば」行のデータにまとめた後、ラベルを付けて、データベースを作る．すなわち 68 分類の段階で、CNN は「ば」と「ぱ」、「び」と「ぴ」、「ぶ」と「ぷ」、「べ」と「ぺ」、「ぼ」と「ぽ」を同じであると判別させる．そして両方の CNN がデータベースから画像を読み込で、訓練する．その後、もし 68 分類の CNN が「ば」行の 5 種のひらがな文字コードを出力したら、また濁点あると半濁点のあるを判別する 2 分類の CNN に入れ、判別する．それ以外の文字コードが 68 分類の CNN から出力したら、直接最終の出力コードとする．

第5章　実験と考察

本章では、実験環境及び実験プロトコルを述べ、その上、第 4 章で提案した基本手法も含め各手法の実験を行う．

5.1　実験環境

表 5.1 に実験環境を示す．深層学習用ライブラリに関して、本研究の実験は主に Tensorflow[25] を使用した、Tensorflow は Google が開発しオープンソースで公開している機械学習ソフトウェアライブラリである．Tensorflow 以外に、予備実験及び Data Augmentation の部分においては、Keras[26] を使用した．

表 5.1: 実験用計算機環境

CPU	Intel(R) Core(TM) i5-4460 3.20GHz 4cores
メモリ	8GB
GPU	NVIDIA(R) TITAN X(Pascal) 12GB
GPU 並列計算環境	CUDA 9.0 , cudnn 6
プログラム言語	Python 2.7.12
深層学習用ライブラリ	Tensorflow 1.3.0 , Keras 2.0.8

5.2　実験プロトコル

実験プロトコルでは、実験をするために必要なデータ処理の方法と実験の評価方法を述べ、また各実験で共通したパラメータも述べる．

5.2.1　Data Augmentation

Data Augmentation の部分では、Keras の ImageDataGenerator を使用した．本実験では、ImageDataGenerator の以下 6 つの方法を使用する．

- 画像角度の回転: 0 から指定された数値の範囲でランダムに画像の角度を回転する．

- 画像を水平移動: 0 から指定された数値の範囲でランダムに x 軸方向に移動する．
- 画像を垂直移動: 0 から指定された数値の範囲でランダムに y 軸方向に移動する．
- 画像のせん断変換: 0 から指定された数値の範囲でランダムにせん断変換する．（せん断変換とは各々の点がある方向へ、その方向と平行な定直線からの符号付き距離に比例して移動することである）．
- 画像の拡大縮小: 画像を指定された範囲に拡大や縮小する．1 より小さい場合は拡大、1 より大きい場合は縮小．
- 空白部分の補充手法: 画像を変化させることにより生じた空白部分の補充手法であって、本実験では空白部分の画素を最も近い非空白の画素と同じ数値を取る

本実験では、以上の 6 つの方法をランダムに組み合わせ、少数派クラスのデータを増加した．増加する上限は基本手法以外の実験を除いて、60 倍に設定した．

5.2.2 K-Fold 交差検証

K-Fold 交差検証とは、全体のデータを k 個の部分に分割した後、まず最初の 1 部分をテストデータとし、残りの k-1 部分を訓練データとして訓練を行う．これが一回の訓練であり、次の段階では、２つ目の部分をテストデータとし、残りの k-1 部分を訓練データとして訓練を行う．こうやって k 回を繰り返し、最後に k 回の正答率の平均値を求め、検証の結果とする方法である．K-fold 交差検証をすることによって、過学習、及びデータの偏りによる正答率の間違いを防ぐことが出来る．

本研究では、Baseline の再現実験を除き、他の実験用データは全て 4 : 1 の比率で訓練データとテストデータに分け、5-fold 交差検証で実験を行う．

5.2.3 評価方法

本研究は正答率 (Accuracy) を実験の評価方法とする．すなわちテストデータの中、全画像に対して、正しく識別できたひらがなのくずし字画像の割合である．

5.2.4 実験の共通パラメータ

本研究の各実験において、以下のような共通のパラメータが設定した

- 入力画像サイズ: 28×28 ; 入力画像サイズとして、最初は 28×28、56×56 の二種類で予備実験を行ったが、56×56 の場合、正答率はほぼ変わらなかった．また、56×56 の画素数が 4 倍になったことで、計算量も増えた．そのため、本実験では、Baseline と同じ入力画像サイズ、28×28 とする．

- 入力画像チャンネル: 1（グレースケール画像）
- 畳み込み層のフィルタサイズ：3×3
- 畳み込み層のフィルタストライド：1
- 活性化関数種類: ReLU
- プーリングの種類：Max プーリング
- プーリングのフィルタサイズ：2×2
- プーリングのストライド：2
- 最適化方法：Adam

  以上のハイパーパラメータは、[10][11][21][22] など関連する論文で設定した数値に参照し、また複数の予備実験に得られた結果によって設定した数値である．

また、各実験な具体的ネットワークの構造及びハイパーパラメータは付録に記載されている．

5.3 予備実験

本研究の実験を行う前に、まず予備実験を行う、予備実験では、サンプルプログラムが提供したベースラインの再現実験及び最適な CNN 構造を検証するための実験がある．

5.3.1 Baseline の再現

人文学オープンデータ共同利用センターはデータセットを提供した上、Python と Keras で書いたサンプルプログラムを Baseline として提供している．Baseline とは、提案した手法と比較対照となる手法である．本研究の実験を始める前に、まずベースラインの再現の実験を行った．

サンプルプログラムでは、出現頻度の高い 10 種のひらがなの分類を行い、96 %の正解率を得た [27]．表 5.2 に 10 種のひらがな及びそれぞれの数を示した．サンプルプログラムはこのデータセットを 85:15 の比率で訓練データとテストデータに分け（訓練データ：19,909 個、テストデータ：3,514 個）、サンプルプログラムが設計した CNN を使用し訓練したものである．サンプルプログラムの CNN 構造及びハイパーパラメータは付録の図 6.5 と表 5.6 で示した．本研究の実験を行う前に、まず予備実験としてサンプルプログラムを実行し、確実に 96%の正解率という結果を得て、Baseline の再現に成功した．

文字種	し	に	の	て	り	を	か	く	き	も
データ数	3,929	3,147	2,908	2,398	2,193	2,021	1,910	1,739	1,715	1,463

表 5.2: Baseline の各文字種及びデータ数

5.3.2 最適 CNN 構造検証

最適 CNN 構造検証実験はひらがなのくずし字を識別するために最適な CNN 構造を判別する実験である．この実験では、Baseline の CNN 構造も含め、5 種の CNN 構造を設計した（各 CNN 構造は付録で示した）．この 5 種の CNN を、同一なデータセットで訓練し、評価する．5 種の CNN 構造の中に、最も正解率が高い CNN を実験用 CNN にする．

この実験に用いたデータセットは、並列手法及び順次手法 1 で提案した 48 種のひらがなのくずし字を分類するために Under Sampling と Over Sampling を行い、用意した 1 クラス 1000 個程度のデータセットであり、48 種計 47,736 個ある．このデータセットを 5-fold 交差検証によって順番に分割し（訓練データ：38,188 個、テストデータ：9,548 個）、損失関数は 40 回前後で収束しているため、反復回数を 40 回に設定し、実験を行い、5-fold の平均値を求めた．

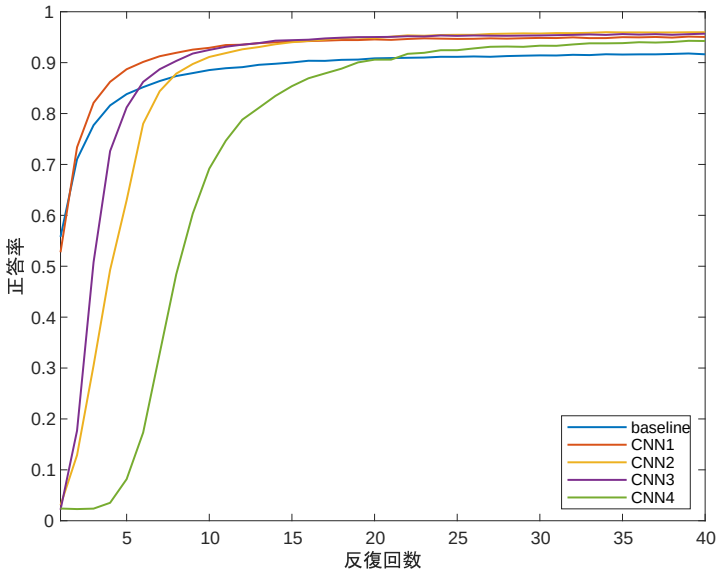

図 5.1: 最適 CNN 構造検証実験　平均正答率推移 ; x 軸 : 反復回数 ; y 軸 : 正答率 ; 四つの CNN モデルで比較実験を行い、40 回の反復によって黄色曲線の CNN2 の正答率は最も高い

結果を図 5.1 と表 5.3 に示す．この結果により、Baseline の CNN と比べ、本実験で設計した各 CNN 構造は正解率として 2.61%〜4.34%高くなっている．CNN2 は結果から見れば最も優秀な構造だと考えられる．CNN3 と CNN4 は構造から複雑になったが、正解率は CNN2 と比べ、逆に下がっている．これからの実験では基本的に CNN2 を使用する．

CNN 構造	平均正答率
Baseline	91.64%
CNN 1	95.05%
CNN 2	95.98%
CNN 3	95.69%
CNN 4	94.25%

表 5.3: 最適 CNN 構造検証実験　平均正答率

5.4　基本手法

濁点と半濁点のあるひらがなも含め、全 73 種のひらがなのくずし字（表 3.2）を Under Sampling と Over Sampling を行い、用意した 1 クラス 500 個程度のデータセットであり、73 種計 36,545 個ある．最適 CNN 構造検証実験の CNN2 を用い、このデータセットを 5-fold 交差検証によって順番に分割し（訓練データ：29,236 個、テストデータ：7,309 個）、損失関数は 40 回前後で収束しているため、反復回数を 40 回に設定し、実験を行い、5-fold の平均値を求めた．

本実験では 1 クラス 500 個程度のデータセットのみを作成、実験した理由は、データ数が過少なクラスを過度な（約 160 倍に）Data Augmentation によって、500 個以上のデータを生成しても、過学習を招く可能性があり、正答率の信頼性はある程度欠けていると考えられる．

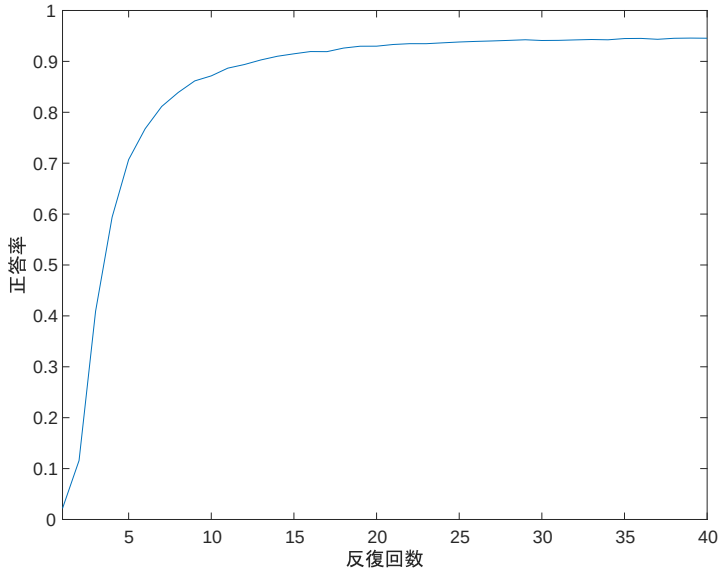

図 5.2: 基本手法　73 種ひらがなのくずし字の分類平均正答率推移；x 軸：反復回数；y 軸：正答率；40 回の反復によって、最終の正答率はおよそ 94.55%

クラス	データ数	クラス	データ数	クラス	データ数	クラス	データ数
あ	3,697	す、ず	5,166	の	14,337	ゆ	980
い	7,022	せ、ぜ	2,448	は、ば、ぱ	10,907	よ	3,047
う	3,662	そ、ぞ	2,416	ひ、び、ぴ	3,503	ら	5,581
え	440	た、だ	7,040	ふ、ぶ、ぷ	5,341	り	8,656
お	2,734	ち、ぢ	1,501	へ、べ、ぺ	5,302	る	6,797
か、が	11,643	つ、づ	4,264	ほ、ぼ、ぽ	1,582	れ	3,897
き、ぎ	2,817	て、で	14,325	ま	4,610	ろ	1,257
く、ぐ	5,853	と、ど	12,690	み	1,586	わ	1,383
け、げ	3,044	な	7,892	む	967	ゐ	145
こ、ご	4,162	に	15,982	め	1,886	ゑ	204
さ、ざ	4,566	ぬ	1,165	も	7,491	を	9,208
し、じ	14,446	ね	1,100	や	3,292	ん	3,386

表 5.4: 48 分類の各クラス文字種及びデータ数

結果を図 5.2 に示す．最終の平均正解率は 94.55%である．

5.5 並列手法

本実験では、ひらがなのくずし字を 48 分類する実験と、濁点・半濁点両方なし、濁点あり、半濁点ありを 3 分類する実験がある．

5.5.1 ひらがな 48 分類

ひらがなのくずし字を 48 分類する実験では、5 種のデータセットを用いられた．まず第 4 章で述べたように、48 分類できるよう、半濁点ありのひらがなと濁点ありのひらがなを、両方なしの対応ひらがなにまとめる．その後、各クラスのデータ数（表 5.4）により、Under Sampling または Over Sampling を施し、1 クラスをそれぞれ 500、1,000、2,000、4,000、8,000 個程度のデータ数に確保し、この 5 種のデータセットを作成した．また、5-fold 交差検証によって 5 種のデータセットを全て順番に分割した、具体的な数は表 5.11 で示した．その後、実験 1 の CNN 構造 2 を用い、損失関数は 40 回前後で収束しているため、反復回数を 40 回に設定し、実験を行い、5-fold の平均値を求めた．

実験結果は図 5.3 と表 5.6 で示す、データ数を増やすことと共に、正答率も上がっている、正解率が一番高いのが 1 クラスのデータ数が 8,000 程度のデータセットを使用した CNN で、正答率は 98.12%である．

1 クラスのデータ数	訓練データ数	テストデータの数
500 程度	19,568	4,893
1,000 程度	38,188	9,548
2,000 程度	75,976	19,245
4,000 程度	153,346	38,337
8,000 程度	291,638	72,910

表 5.5: 並列手法　5 種のデータセットのデータ数一覧表

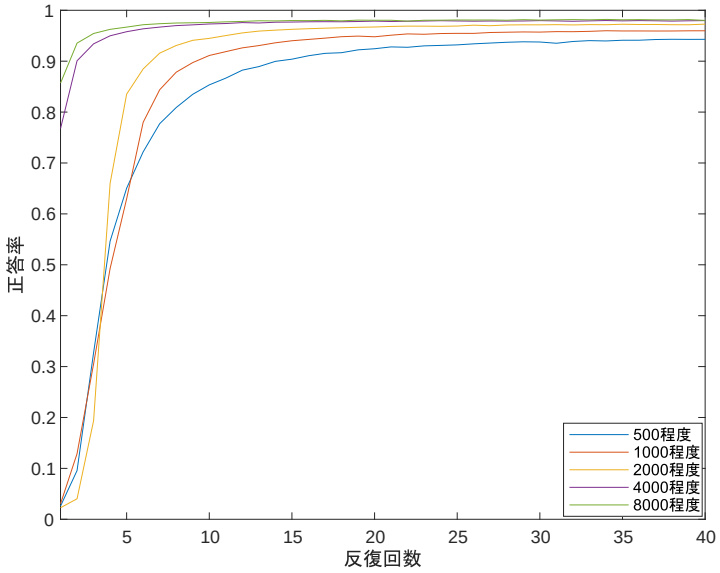

図 5.3: 並列手法　5 種のデータセットを用いた 48 種ひらがなのくずし字分類の平均正答率推移；x 軸：反復回数；y 軸：正答率；比較実験を行い、40 回の反復によって緑色曲線の 1 クラスのデータ数が 8,000 程度の正答率は最も高い

1 クラスのデータ数	平均正答率
500 程度	94.29%
1,000 程度	95.98%
2,000 程度	97.28%
4,000 程度	97.98%
8,000 程度	98.12%

表 5.6: 並列手法　5 種のデータセットを用いた 48 種ひらがなのくずし字分類の平均正答率

クラス名	濁点・半濁点両方なし	濁点あり	半濁点あり
ラベル	0	1	2
文字種数	48	20	5
1 文字種のデータ数	20 個程度	50 個程度	200 個程度
1 クラスのデータ数	1,032	1,014	989
訓練データ数	2,428		
テストデータ数	607		

表 5.7: 並列手法　濁点と半濁点両方なし、濁点あり、半濁点ありの 3 分類実験データ数一覧表

5.5.2　濁点と半濁点両方なし、濁点あり、半濁点ありの 3 分類

並列手法の濁点と半濁点両方なし、濁点あり、半濁点ありの 3 分類実験では、48 種の濁点と半濁点両方なしのひらがなのくずし字、20 種の濁点ありのひらがなのくずし字及び 5 種の半濁点ありのひらがなのくずし字を 3 分類する CNN に関する実験である．最初に、3 つのクラスのデータ数及びクラス内異なる文字種のデータ数のバランスを保つよう、各文字種を Under Sampling または Over Sampling を施し、1 クラスに 1,000 個程度を確保した．また、5-fold 交差検証によって順番に分割し、損失関数は 150 回前後で収束しているため、反復回数を 150 回に設定し、5-fold の平均値を求めた．実験に関する具体的なデータ数は表 5.7 で示した．

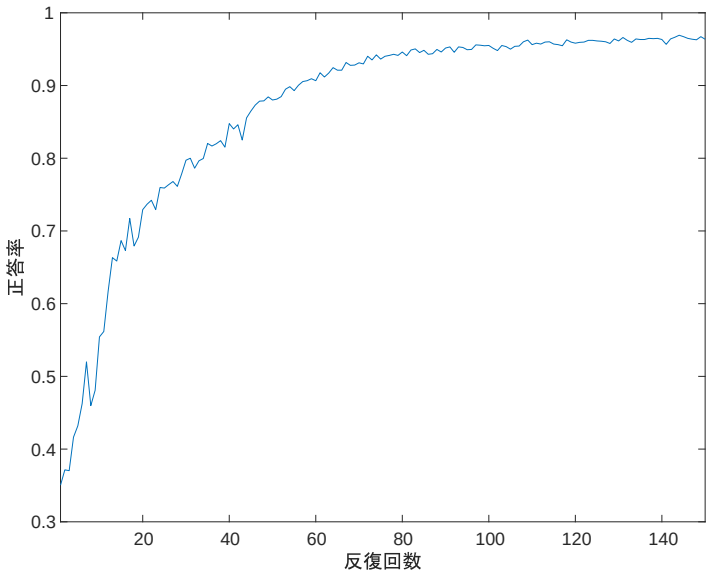

図 5.4: 並列手法　濁点と半濁点両方なし、濁点あり、半濁点ありの 3 分類の平均正解率；x 軸：反復回数；y 軸：正答率；150 回の反復によって、最終の正答率はおよそ 96.52%

結果は図 5.4 で示す．平均正答率は最終的に 96.52%に達した．

クラス名	濁点なし	濁点あり
ラベル	0	1
文字種数	20	20
1 文字種のデータ数	200 個程度	200 個程度
1 クラスのデータ数	4,327	4,181
訓練データ数	6,806	
テストデータ数	1,702	

表 5.8: 順次手法 1　濁点なし、濁点ありの 2 分類実験データ数一覧表

5.6　順次手法 1

順次手法 1 のひらがなのくずし字を 48 分類の手法は、並列手法のひらがなのくずし字を 48 分類する手法と同じであるため、実験はしない．故に、本実験では、濁点あり、濁点なしの 2 分類実験、及び濁点なし、濁点あり、半濁点ありを 3 分類する実験を行う．

5.6.1　濁点あり、濁点なしの 2 分類

順次手法 1 の濁点あり、濁点なしの 2 分類実験では、第 4 章が述べたように、「か」行、「さ」行、「た」行、「は」行のひらがなのくずし字と「が」行、「ざ」行、「だ」行、「ば」行のひらがなのくずし字を 2 分類する CNN に関する実験である．最初に、合計 40 種のひらがな文字種データから Under Sampling を施し、1 文字種から 200 個程度のデータを抽出する．その後、「か」行、「さ」行、「た」行、「は」行の 20 種のひらがなのくずし字を全部濁点無し、すなわちラベル 0 のクラスにまとめ、「が」行、「ざ」行、「だ」行、「ば」行の 20 種のひらがなのくずし字を全部濁点あり、すなわちラベル 1 のクラスにまとめる．これで 1 クラスのデータ数は約 4000 個ある．最後に 5-fold 交差検証によって順番に分割し、損失関数は 80 回前後で収束しているため、反復回数を 80 回に設定し、実験を行い、5-fold の平均値を求めた．実験に関する具体的なデータ数は表 5.8 で示した．

結果は図 5.5 で示す．平均正答率は最終的に 98.37%に達した．

5.6.2　濁点と半濁点両方なし、濁点あり、半濁点ありの 3 分類

順次手法 1 の濁点なし、濁点あり、半濁点ありの 3 分類実験では、「は」行、「ば」行及び「ぱ」行のひらがなのくずし字を 3 分類する CNN に関する実験である．最初に、3 つのクラスのデータ数及びクラス内の異なる文字種のデータ数のバランスを保つよう、各文字種を Under Sampling または Over Sampling を施し、1 クラスを 1000 個程度を確保した．また、5-fold 交差検証によって順番に分割し、損失関数は 150 回前後で収束しているため、反復回数を 150 回に設定し、実験を行い、5-fold の平均値を求めた．実験に関する具体的なデータ数は表 5.9 で示した．

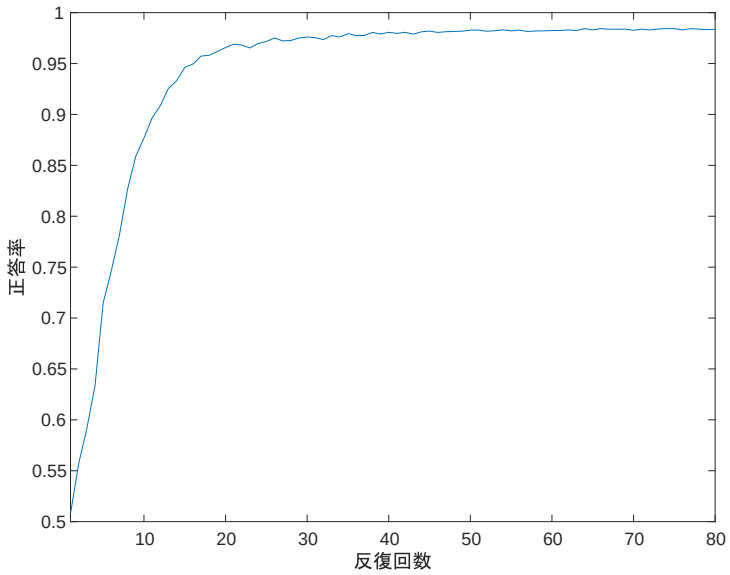

図 5.5: 順次手法 1　濁点なし、濁点ありの 2 分類の平均正答率推移；x 軸：反復回数；y 軸：正答率；80 回の反復によって、最終の正答率はおよそ 98.37%

クラス名	濁点・半濁点両方なし	濁点あり	半濁点あり
ラベル	0	1	2
文字種数	5	5	5
1 文字種のデータ数	200 個程度	200 個程度	200 個程度
1 クラスのデータ数	1,025	1,063	989
訓練データ数	2461		
テストデータ数	616		

表 5.9: 順次手法 1　濁点と半濁点両方なし、濁点あり、半濁点ありの 3 分類実験データ数一覧表

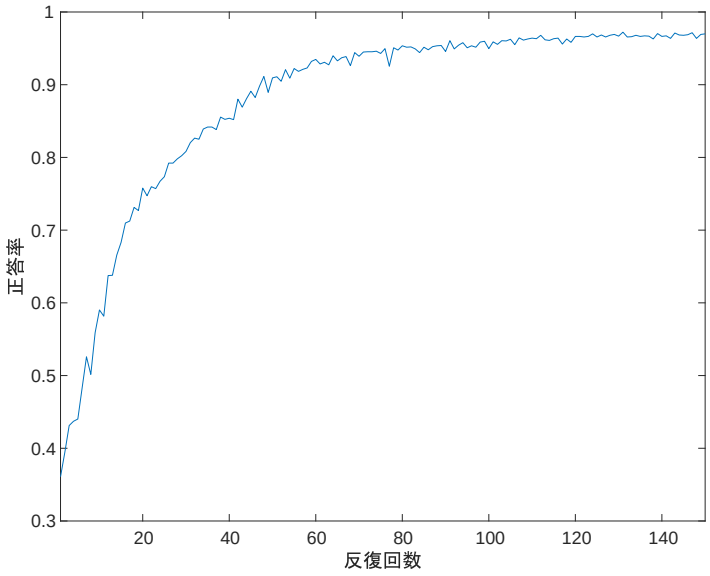

図 5.6: 順次手法 1　濁点と半濁点両方なし、濁点あり、半濁点ありの 3 分類の平均正答率推移；x 軸：反復回数；y 軸：正答率；150 回の反復によって、最終の正答率はおよそ 96.99%

結果は図 5.6 で示す．平均正答率は最終的に 96.99%に達した．

5.7　順次手法 2

本実験は第 4 章で提出した順次手法 2 に関する実験である．本実験では、ひらがなのくずし字を 68 分類する実験と、濁点あり、半濁点ありを 2 分類する実験を行う．

5.7.1　ひらがな 68 分類

ひらがなのくずし字を 68 分類する実験では、5 種のデータセットを用いられた．

まず第 4 章で述べたように、68 分類できるよう、半濁点ありのひらがなを濁点ありの対応のひらがなにまとめる．その後、各クラスのデータ数（表 5.10）により、Under Sampling または Over Sampling を施し、1 クラスをそれぞれ 500、1,000、2,000、4,000、8,000 個程度のデータ数に確保し、この 5 種のデータセットを作成した．また、5-fold 交差検証によって 5 種のデータセットを全て順番に分割した、具体的な数は表 5.11 で示した．その後、実験 1 の CNN2 を用い、損失関数は 40 回前後で収束しているため、反復回数を 40 回に設定し、実験を行い、5-fold の平均値を求めた．

クラス	データ数	クラス	データ数	クラス	データ数	クラス	データ数
あ	3,697	し	13,386	ど	1,622	み	1,586
い	7,022	じ	1,060	な	7,892	む	967
う	3,662	す	3,946	に	15,982	め	1,886
え	440	ず	1,220	ぬ	1,165	も	7,491
お	2,734	せ	2,263	ね	1,100	や	3,292
か	8,439	ぜ	185	の	14,337	ゆ	980
が	3,204	そ	1,909	は	8,725	よ	3,047
き	5,078	ぞ	507	ば、ぱ	2,182	ら	5,581
ぎ	556	た	5,684	ひ	2,713	り	8,656
く	5,447	だ	1,356	び、ぴ	790	る	6,797
ぐ	406	ち	1,274	ふ	4,742	れ	3,897
け	2,519	ぢ	227	ぶ、ぷ	602	ろ	1,257
げ	525	つ	3,618	へ	4,121	わ	1,383
こ	3,322	づ	646	べ、ぺ	1,199	ゐ	145
ご	840	て	12,976	ほ	1,122	ゑ	204
さ	3,927	で	1,349	ぼ、ぽ	477	を	9,208
ざ	639	と	11,068	ま	4,610	ん	3,386

表 5.10: 68 分類の各クラス文字種及びデータ数

1 クラスのデータ数	訓練データ数	テストデータの数
500 程度	29,236	7,309
1,000 程度	55,570	13,893
2,000 程度	111,288	27,822
4,000 程度	217,977	54,495
8,000 程度	432,875	108,219

表 5.11: 順次手法 2　 5 種のデータセットのデータ数一覧表

1 クラスのデータ数	平均正解率
500 程度	94.55%
1,000 程度	96.47%
2,000 程度	97.53%
4,000 程度	97.95%
8,000 程度	98.16%

表 5.12: 順次手法 2　 5 種のデータセットを用いた 68 種のひらがなのくずし字分類の平均正解率

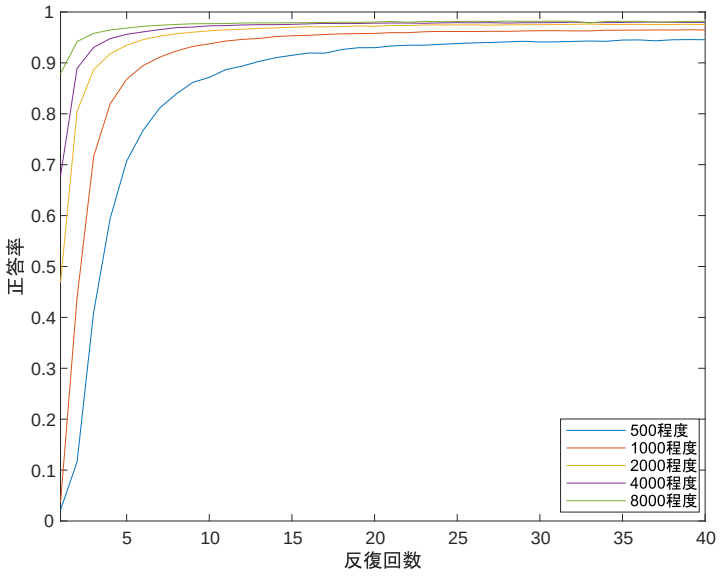

図 5.7: 順次手法 2　 5 種のデータセットを用いた 68 種のひらがなのくずし字分類の平均正解率推移；x 軸：反復回数；y 軸：正答率；比較実験を行い、40 回の反復によって緑色曲線の 1 クラスのデータ数が 8,000 程度の正答率は最も高い

結果を、図 5.7 と表 5.12 に示す、データ数を増やすことと共に、正答率も上がっている、正解率が一番高いのが 1 クラスのデータ数が 8,000 程度のデータセットを使用した CNN で、正解率は 98.16%である．

5.7.2　濁点あり、半濁点ありの 2 分類

順次手法 2 の濁点あり、半濁点ありの 2 分類実験では、「ば」行と「ぱ」行のひらがなのくずし字を 2 分類する CNN に関する実験である．最初に、2 つのクラスのデータ数及びクラス内の異なる文字種のデータ数のバランスを保つよう、各文字種を Under Sampling または Over Sampling を施し、1 クラスを 1000 個程度を確保した．また、5-fold 交差検証によって順番に分割し、損失関数は 200 回前後で収束しているため、反復回数を 200 回に設定し、実験を行い、5-fold の平均値を求めた．実験に関する具体的なデータ数は表 5.13 で示した．

結果は図 5.8 で示す．平均正答率は最終的に 98.34%に達した．

クラス名	濁点あり	半濁点あり
ラベル	0	1
文字種数	5	5
1 文字種のデータ数	200 個程度	200 個程度
1 クラスのデータ数	1063	989
訓練データ数	1641	
テストデータ数	411	

表 5.13: 順次手法 2　濁点あり、半濁点ありの 2 分類実験データ数一覧表

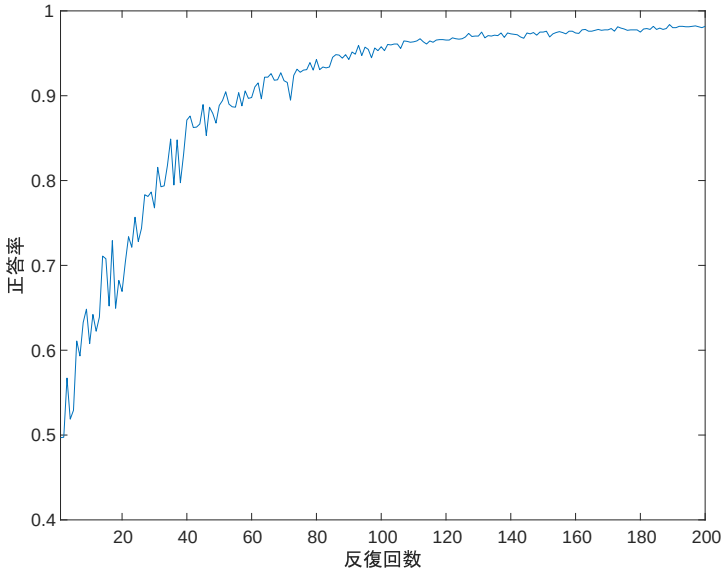

図 5.8: 順次手法 2　濁点あり、半濁点ありの 2 分類の平均正答率推移 ; x 軸 : 反復回数 ; y 軸 : 正答率 ; 200 回の反復によって、最終の正答率はおよそ 98.34%

5.8 考察

5.8.1 分類タスクに対する考察

実験 5.4、実験 5.5.1 と実験 5.7.1 では、それぞれ全ひらがなのくずし字 73 種を分類する CNN の実験、ひらがな 48 種を分類する CNN の実験とひらがな 68 種を分類する CNN の実験を、1 クラス 500 程度のデータセット、同様の CNN モデル、及び 40 回の反復回数の条件を元に実験を行った．そして、最終的にそれぞれ 94.55%、94.29%及び 94.55%という正答率を得た．この 3 つの結果はあまり大きな変化は生じなかった．以上のことから、2 つの結論に至った．1 つ目は、同じひらがなのくずし字に対して、そのひらがなの濁点や半濁点の有無を判別することは、CNN にとって難しいことではない．2 つ目は、確かに分類タスクにとって、クラスが多いほど、タスクも難しくなるが、このひらがなのくずし字を分類する研究において、48 種から 73 種までの分類タスクに生じた変化は、同じ CNN に対して基本的に影響はなかった．

5.8.2 データ数が分類精度に与える影響

実験 5.5.1 と実験 5.7.1 では、ひらがな 48 種を分類する実験とひらがな 68 種を分類する実験を、1 クラス 500 程度から、1,000 程度、2,000 程度、4,000 程度、8,000 程度までそれぞれデータセットを作成し、実験を行った．実験の結果は表 5.6 と表 5.11 に示した．結果は、データ数が多いほど、平均正解率も高くなる．そしてこの二つの実験の推移グラフ及び正答率表を対比結果から見ても、前の考察と同じく、分類タスクの変化は、結果にあまり変化をもたらしていない．また、両方の実験は同じく、データ数の増加と逆に、正答率が上がる速度は下がっている．これは、この実験で用いた CNN が限界に近づいているからだと考えられる．また、これ以上の正答率を得るには、Data Augmentation ではなく、確実に少数クラスのデータを増やすか、より優れた CNN に改良するなどの方法が必要である．

5.8.3 濁点と半濁点両方なし、濁点あり、半濁点ありの精度

実験 5.5.2、全ひらがなに対する濁点・半濁点なし、濁点あり、半濁点ありの 3 分類実験の平均正答率は 96.52%である．一方、実験 5.6.2、「は」行の濁点・半濁点なし、濁点あり、半濁点ありの 3 分類実験の平均正答率は 96.99%である．2 つの実験は、同じく 1 クラス 1000 個程度、同様の CNN モデル、及び 150 回の反復回数の条件を元に実験を行った．結果から見れば、確かに濁点なし、濁点あり、半濁点ありの判別を、全ひらがなの範囲で識別するより、「は」行だけの範囲に絞り、それから識別するほうが、0.47%の正答率が上がった．しかしこの結果もそれほど大きな差ではないと考えられる．

5.8.4 濁点ありと濁点なし、濁点ありと半濁点ありの精度

実験 5.6.1 では、「か」「さ」「た」「は」行の範囲に絞って、濁点あり、濁点なしの 2 分類実験を行った．実験は 98.37%の正答率が得た．実験 5.7.2 では、「は」行の範囲に絞って、濁点あ

り、半濁点ありの 2 分類実験を行った．実験は 98.34%の正答率が得た．2 分類に関する CNN の実験では、いずれも 98%以上の正答率が出た．また、実験 5.7.2 で使用したデータセットは、実験 5.6.2 で作成した 1 クラス 1,000 個程度のデータセットから「ば」行のクラスと「ぱ」行のクラスを抽出し、そのまま 2 分類の実験を行ったものである．両方の結果を比べると、「ば」行と「ぱ」行の 2 分類の精度は、「は」行、「ば」行、「ぱ」行の 3 分類の精度より 1.35%上がった．結果的に、2 分類の正答率は全体的に 3 分類の正答率より高いという点から推測すると、濁点や半濁点の有無の判別には、精度と分類タスクが関係している．

5.8.5 全実験を踏まえて

最後に、提案した 3 種の手法を全体的に比べ、手法一並列手法の 1 つの CNN の正答率はそれぞれ 98.12%と 96.52%である．手法二順次手法 1 の 3 つの CNN の正答率はそれぞれ 98.12%、98.37%と 96.99%である．手法三順次手法 2 の 2 つの CNN の正答率はそれぞれ 98.16%と 98.34%である．この研究は、判別の段階を分けて、実験を行ったが、一体化したシステムを作ってないため、提案した 3 種の手法の最終的の正答率が得られなかった．本研究では、全てのひらがなのくずし字の出現頻度を同じであると仮定し、確率から提案した 3 種の手法の最終的正答率を算出する．

手法一の並列手法では、両方の CNN が同時に正答を出力しなければならない．数式は以下に示す

$$98.12\% \times 96.52\% = 94.71\%$$

最終的な正答率が 94.71%である．

手法二の順次手法 1 では、順次に判別するため、濁点と半濁点両方ないのひらがな 28 種が 48 分類の CNN によって判別した後、最終の出力文字コードとする．濁点のあるひらがな 30 種（「か」「が」「さ」「ざ」「た」「だ」行）はさらに濁点の有無を判別する 2 分類 CNN に入れ、判別する．半濁点のあるひらがな 15 種（「は」「ば」「ぱ」行）はさらに濁点・半濁点両方なし、濁点あり、半濁点ありを判別する 3 分類 CNN に入れ、判別する．数式は以下に示す

$$\frac{28}{73} \times 98.12\% + \frac{30}{73} \times 98.12\% \times 98.37\% + \frac{15}{73} \times 98.12\% \times 96.99\% = 96.86\%$$

最終の正答率が 96.86%である．

手法三の順次手法 2 では、順次に判別するため、半濁点のないひらがなが 63 種は 68 分類の CNN によって判別した後、最終の出力文字コードとする．半濁点のあるひらがな 10 種（「ば」「ぱ」行）はさらに濁点あり、半濁点ありを判別する 2 分類 CNN に入れ、判別する．数式は以下に示す

$$\frac{63}{73} \times 98.16\% + \frac{10}{73} \times 98.16\% \times 98.34\% = 97.93\%$$

最終の正答率が 97.93%である．

結果的に手法 3、順次手法 2 は最も高い正答率が得られた．また、この結果から推測すると、データ数が充分かつ各クラスのデータがバランスが取れた状況ならば、全 73 種のひらが

なのくずし字を直接分類するのは一番良い方法と考えられる．しかしデータセットの状況、すなわち半濁点のあるひらがなのくずし字のデータ数が足りない状況から見ると、やはり提案した手法 3、順次手法 2 が最も現実的な方法だと考えられる．

第6章　おわりに

6.1　まとめ

本研究では、畳み込みニューラルネットワークを用いて、日本語古典籍の中のひらがなのくずし字を認識することである．

本研究には、人文学オープンデータ共同利用センターが公開した日本古典籍字形データセットの中の字形画像データを実験のデータとして使用する．しかし、字形画像データからひらがなのくずし字をまとめた結果、各文字種のデータ数が極めて不均衡であることが分かった．データ数の多い文字種は一万以上のデータが持っていると比べ、データ数の少ない文字種は数個、数十個のデータしか持っていない．また全体的なひらがなのくずし字にとって、濁点や半濁点のあるひらがなのデータ数が少ないという特徴があった．

そのため、第三章ではまず、データの不均衡問題をある程度改善できる三つの手法、Under Sampling、Over Sampling 及び Hybird of Methods を紹介した．Under Sampling とは、全体のデータが相対的に均等になるまで、データの多いクラスからランダムにデータを取り出すことである．Over Sampling とは、データの少ないクラスから、全体のデータが相対的に均等になるまで、何らかの手法を用いて、データを増やすことである．そして Hybird of Methods とは、Under Sampling と Over Sampling を混合した手法で、多数派クラスのデータを Under Sampling でデータを減少し、少数派クラスのデータを Over Sampling でデータを増加することである．本研究は基本的に Hybrid of Methods を使用する．

第四章では、一般的なひらがなのくずし字を認識する基本手法を紹介した上で．このデータセットにおいて、ひらがなのくずし字と濁点・半濁点の有無の判別を分けて判別する手法を 3 つ提案した．基本手法は、全 73 種のひらがなのくずし字を直接分類する手法である．そのため、73 種のひらがなのくずし字を分類する CNN を訓練する必要がある．手法一の並列手法では、濁点と半濁点有無の判別と、ひらがなのくずし字の判別を分けて、かつ並列に判別する手法を提案した．48 種のひらがなのくずし字を判別する CNN を訓練する以外．濁点・半濁点両方なし、濁点あり、半濁点ありを 3 分類する CNN も訓練する．そして両方の結果を合わせて、最終の文字コードを出力する．手法二の順次手法 1 では同じく濁点と半濁点有無の判別と、ひらがなのくずし字の判別を分けて判別する手法であるが．並列手法の異なる点は、まず 48 種のひらがなのくずし字を判別し、その後、もし「か」、「さ」、「た」行の文字コードが出力したら、濁点なしと濁点ありの 2 分類 CNN に入れて判別し、もし「は」行の文字コードが出力したら、濁点・半濁点なし、濁点あり、半濁点ありの 3 分類 CNN に入れ、判別する．手法三の順次手法 2 では、濁点なしのひらがなと濁点ありのひらがなを直接分類し、その後「は」行の文字コードを出力するだけに、濁点ありと半濁点ありの 2 分類 CNN に入れ

て判別する．

第五章では、実験環境及び実験プロトコルを述べた．実験プロトコルでは、具体の Data Augmentation の方法、K-fold という実験の評価方法及び各実験で用いたパラメータを紹介した．その後、Baseline を再現し、CNN モデルを決まった上、第四章で述べた基本手法及び提案した 3 種の手法の実験を行った．最後に実験の結果について考察をした．考察では、ひらがなのくずし字を分類する研究において、48 種から 73 種までの分類タスクに生じた変化は、同じ CNN に対して基本的に影響はなかったが、濁点と半濁点両方なし、濁点あり、半濁点ありの 3 分類から濁点なしと濁点ありや、濁点ありと半濁点ありの 2 分類に減少することによって、判別精度が上がった．また全体的に踏まえ、データ数が充分かつ各クラスのデータがバランスが取れた状況ならば、全 73 種のひらがなのくずし字を直接分類するのは一番良い方法と考えられるが、データセットの状況、すなわち半濁点のあるひらがなのくずし字のデータ数が足りない状況から見ると、やはり提案した手法 3、順次手法 2 が最も現実的な方法だと考えられる．

6.2 今後の課題

本研究では、畳み込みニューラルネットワークを用いて、ひらがなのくずし字を判別する手法を提案し、実験を行った．しかし実験の結果を、全ひらがなの出現頻度を同じであると仮定し、提案した 3 種の手法の最終的な正答率を確率に基づいて算出しただけであって、一体化としたシステムを作っていないことが、本研究での不足点と考えている．今後の課題としては、まず一体化したシステムを作り、総合的な正答率を得るのが最も重要なことであると考えている．

また、本研究の着目点は、オフライン型手書き文字認識の第二段階、切り出した文字を正確に識別することを研究の対象として研究を行っていたが、第一段階である、対象文字を画像の中から正確に切り出すこともまた大きな課題であると考えられる．本研究は、与えられたくずし字原本の画像から、自動的対象文字であるひらがなのくずし字を検出し、正確に切り出すことを次の課題として研究を進めたいと考えている．

謝辞

本研究を進めるにあたり、的確なご指導、ご助言を頂いた長谷川秀彦教授に心より感謝致します．また、貴重なご指摘を多数頂きました手塚太郎准教授に御礼申し上げます．研究や研究以外のところもいろいろ助けてくれた太田凌さんと鈴木健太さんにありがとナス．研究に関するアドバイスをいただいた Pengfei Xu(University of Helsinki) と Hao Liu(Beijing Normal University) に感謝を申し上げます．最後にいつも私の支えになってくれた霞ヶ丘詩羽先輩、アルトリア・ペンドラゴンさん、朽木冬子さん、松岡禎丞さん、茅野愛衣さん、種田梨沙さん、佐倉綾音さんに心から感謝致します．

参考文献

[1] Sargur N Srihari, Xuanshen Yang, and Gregory R Ball. Offline chinese handwriting recognition: an assessment of current technology. *Frontiers of Computer Science in China*, Vol. 1, No. 2, pp. 137–155, 2007.

[2] Xiang-Dong Zhou, Da-Han Wang, Feng Tian, Cheng-Lin Liu, and Masaki Nakagawa. Handwritten chinese/japanese text recognition using semi-markov conditional random fields. *IEEE transactions on pattern analysis and machine intelligence*, Vol. 35, No. 10, pp. 2413–2426, 2013.

[3] Tong-Hua Su, Tian-Wen Zhang, De-Jun Guan, and Hu-Jie Huang. Off-line recognition of realistic chinese handwriting using segmentation-free strategy. *Pattern Recognition*, Vol. 42, No. 1, pp. 167–182, 2009.

[4] Yong Ge, Qiang Huo, and Zhi-Dan Feng. Offline recognition of handwritten chinese characters using gabor features, cdhmm modeling and mce training. In *Acoustics, Speech, and Signal Processing (ICASSP), 2002 IEEE International Conference*, Vol. 1, pp. I–1053. IEEE, 2002.

[5] Cheng-Lin Liu. Normalization-cooperated gradient feature extraction for handwritten character recognition. *IEEE transactions on Pattern Analysis and machine intelligence*, Vol. 29, No. 8, pp. 1465–1469, 2007.

[6] Nello Cristianini and John Shawe-Taylor. *An introduction to support vector machines and other kernel-based learning methods*. Cambridge university press, 2000.

[7] Fumitaka Kimura, Kenji Takashina, Shinji Tsuruoka, and Yasuji Miyake. Modified quadratic discriminant functions and the application to chinese character recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, No. 1, pp. 149–153, 1987.

[8] Cheng-Lin Liu, Hiroshi Sako, and Hiromichi Fujisawa. Discriminative learning quadratic discriminant function for handwriting recognition. *IEEE Transactions on Neural Networks*, Vol. 15, No. 2, pp. 430–444, 2004.

[9] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, Vol. 86, No. 11, pp. 2278–2324, 1998.

[10] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pp. 1097–1105, 2012.

[11] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[12] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pp. 1440–1448, 2015.

[13] Fei Yin, Qiu-Feng Wang, Xu-Yao Zhang, and Cheng-Lin Liu. Icdar 2013 chinese handwriting recognition competition. In *Document Analysis and Recognition (ICDAR), 2013 12th International Conference on*, pp. 1464–1470. IEEE, 2013.

[14] シュテファンカイザー. 世界の文字・中国の文字・日本の文字 : 漢字の位置付け再考. 世界の日本語教育. 日本語教育論集, Vol. 5, pp. 155–167, April 1995.

[15] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pp. 807–814, 2010.

[16] Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of machine learning research*, Vol. 15, No. 1, pp. 1929–1958, 2014.

[17] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[18] 人文学オープンデータ共同利用センター（center for open data in the humanities / codh）. *http://codh.rois.ac.jp/*.

[19] 日本古典籍字形データセット. *http://codh.rois.ac.jp/char-shape/*.

[20] Mateusz Buda, Atsuto Maki, and Maciej A Mazurowski. A systematic study of the class imbalance problem in convolutional neural networks. *arXiv preprint arXiv:1710.05381*, 2017.

[21] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9, 2015.

[22] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[23] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, Vol. 16, pp. 321–357, 2002.

[24] Kung-Jeng Wang, Bunjira Makond, Kun-Huang Chen, and Kung-Min Wang. A hybrid classifier combining smote with pso to estimate 5-year survivability of breast cancer patients. *Applied Soft Computing*, Vol. 20, pp. 15–24, 2014.

[25] tensorflow. *https://www.tensorflow.org/*.

[26] keras. *https://keras.io/*.

[27] 朝展北本. 日本古典籍字形データセットの公開と活用への期待. 第 2 回 CODH セミナー くずし字チャレンジ ～機械の認識と人間の翻刻の未来～.

付録

- 本研究が使用した各実験のプログラムはネットにアップロードしている．

  URL：https://github.com/tgosros/research

- 本研究が訓練した各実験のモデルはネットにアップロードしている．

  URL：https://drive.google.com/open?id=1nNhYZ3deFRhMpmZvRl3l0Os-5Ilf8_Xo

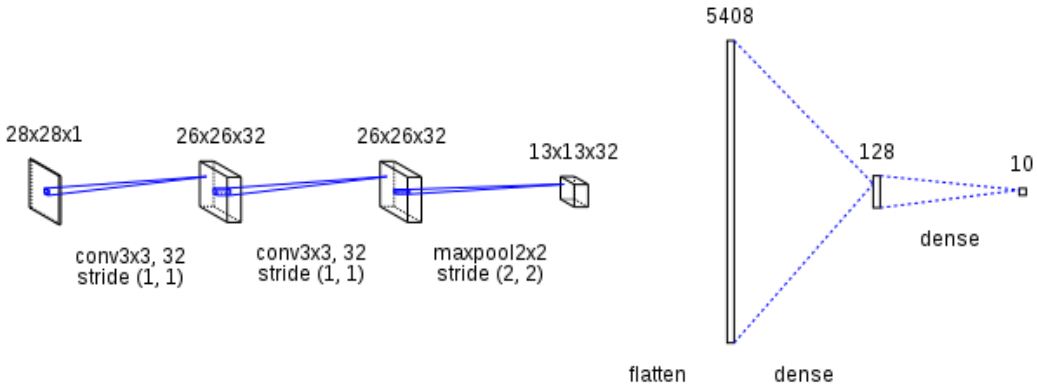

図 6.1: サンプルプログラムの CNN 構造

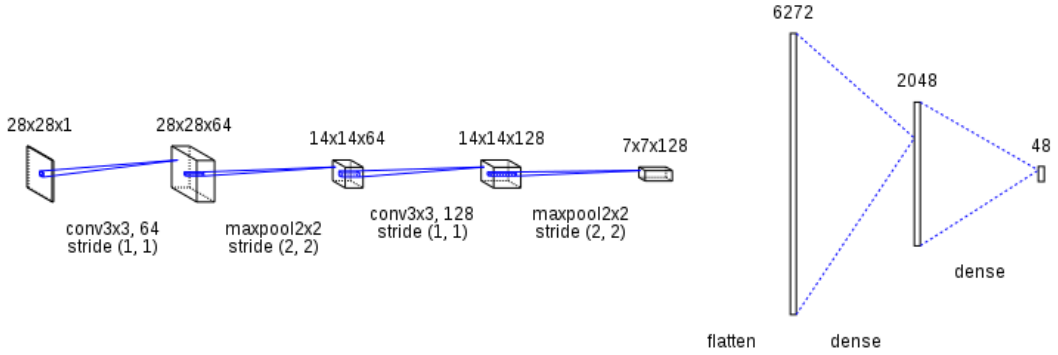

図 6.2: CNN1 の構造

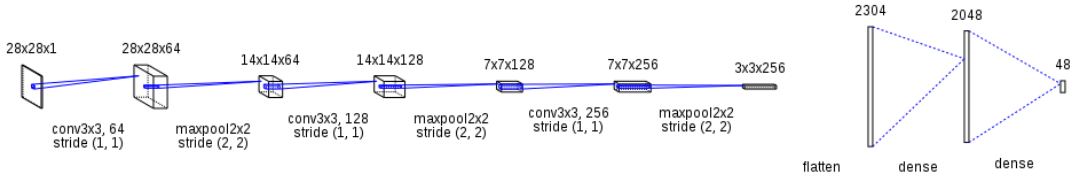

図 6.3: CNN2 の構造

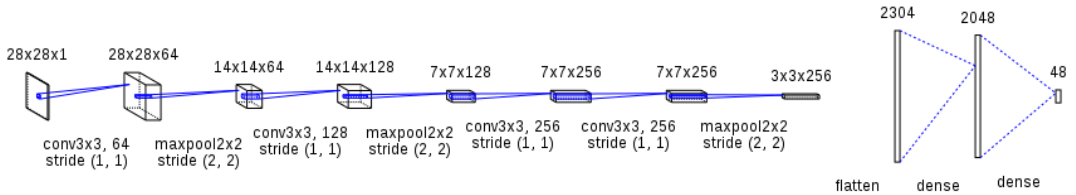

図 6.4: CNN3 の構造

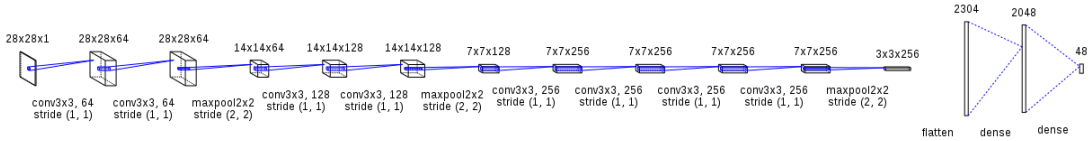

図 6.5: CNN4 の構造