Vol.2020-MPS-128 No.8
Vol.2020-BIO-62 No.8
2020/6/29

畳み込みニューラルネットワークによる画像認識

矢田真城 [1]　魚住龍史 [2]

概要：近年では，コンピュータビジョンの問題に対して機械学習の手法が適用されている．画像認識とは，画像に写る内容を理解することであり，数多くの画像認識に関する研究において，深層学習の手法が採用されている．画像認識に深層学習を適用した手法のうち，よく用いられている手法が畳み込みニューラルネットワークである．畳み込みニューラルネットワークは，異なる 2 種類の処理を行う層を多層化することでより高次の特徴量を抽出できる．畳み込みニューラルネットワークに関して，和文に限定しても非常に数多くの文献が存在するが，図説に加えて，数式も活用したネットワーク構造の可視化が行われた文献は十分でない．そこで本稿では，畳み込みニューラルネットワークについて図解し，畳み込みニューラルネットワークの応用例として，物体検出に用いられる R-CNN(Regions with CNN features)とその後継である Fast R-CNN, Faster R-CNN についてまとめ，Faster R-CNN の実装方法を提供する．

キーワード：コンピュータビジョン，物体認識，物体検出，畳み込みニューラルネットワーク, R-CNN, Fast R-CNN, Faster R-CNN

Image recognition using convolutional neutral networks

SHINJO YADA[†1]　RYUJI UOZUMI[†2]

Abstract: In recent years, machine learning methods have been applied to computer vision. Image recognition is to identify what appears in an image. Deep learning have been used in many researches on image recognition. A convolutional neutral network (CNN) is often used among the methods that apply deep learning to image recognition. The CNN extracts higher-order features by forming multiple layers that perform two different types of processing. Regarding the CNN, the documents which visualize the structure of the network using mathematical content and figure are not sufficient. In this paper, we illustrate the CNN, summarize R-CNN, Fast R-CNN, and Faster R-CNN for object detection, and provide the implementation of these methods.

Keywords: Computer vision, Object recognition, Object detection, Convolutional neutral network, R-CNN, Fast R-CNN, Faster R-CNN

1. はじめに

視覚することの難しさは，与えられた情報が解を得るためには十分でなく，何らかのモデルを用いて複数ある解の候補の中から最適と思われる 1 つを決定しなければならない点にある．コンピュータに視覚をもたせるコンピュータビジョンは時代とともに発展をとげ，いまでは 3D モデリング，画像内容の検索，対象物体の認識と検出など，与えられたデータから視覚情報を自動的に抽出する分野として確立され，今最も注目される話題のひとつになった．

コンピュータビジョンで取り扱われる研究分野のひとつとして画像認識がある．物体認識(object recognition)と物体検出(object detection)は，いずれも画像認識で取り扱うトピックであり，画像から物体を識別するという点で類似しているが，その目的が異なる．物体認識は，画像内の物体のクラスあるいは画像内の物体のインスタンスを予測するプロセスのことである．これに対して，物体検出は，画像内の物体のクラスあるいはインスタンスを予測するだけでなく，画像内の位置まで推定するプロセスのことである．物体検出の結果，ひとつの画像内に写る複数の物体を識別し，識別された各物体の位置が四角い領域で表示される．

近年，画像認識に深層学習を適用した手法のうち，よく用いられている手法が畳み込みニューラルネットワーク(convolutional neutral network; CNN)である．畳み込みニューラルネットワークは，ディープ・ニューラルネットワークから派生したものである．ニューラルネットワークは，生物の神経細胞が構成するネットワークをコンピュータ上で模倣したものであり，ディープ・ニューラルネットワークは，多数の層をもつ深いニューラルネットワークのことである．畳み込みニューラルネットワークでは，それぞれで処理が異なる「畳み込み層」と「プーリング層」とを積み重ねることで多層化する．畳み込み層では，ユニット間の結合を局所的に限定することで，モデルを表現するために必要なパラメータ数を減らしている．プーリング層は，画像をいくつかの領域に区切り各領域を代表する値を抽出して並べることで，対象となる位置の変化に対する頑健性

1 エイツーヘルスケア株式会社 データサイエンス本部 生物統計第 1 部.
Biostatistics Department I, Data Science Division, A2 Healthcare Corporation
2 京都大学大学院医学研究科 医学統計生物情報学.
Department of Biomedical Statistics and Bioinformatics, Kyoto University Graduate School of Medicine.

Vol.2020-MPS-128 No.8
Vol.2020-BIO-62 No.8
2020/6/29

を与えることが可能となる[1].

本稿では，畳み込みニューラルネットワークについて図解し，物体検出の方法である R-CNN, Fast R-CNN, Faster R-CNN についてまとめ，Faster R-CNN の実装方法を提供する．

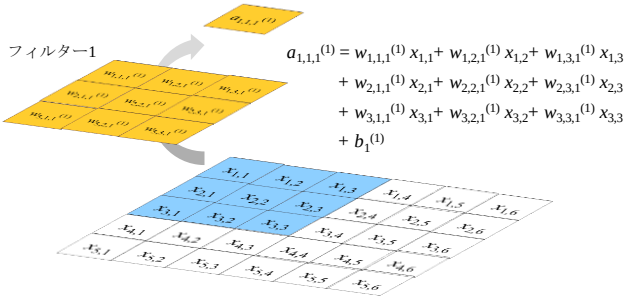
2. 畳み込みニューラルネットワークを用いた物体認識

畳み込みニューラルネットワーク（以下，CNN と表記する）は，入力層と出力層との間に中間層を設け，中間層を多層化させることでより高次の特徴量を取り出す．この点は，フィードフォワード・ニューラルネットワークと同様である．CNN の特徴は，中間層の多層化だけでなく，それぞれで処理が異なる「畳み込み層」と「プーリング層」とを積み重ねる点にある[2].

2.1 畳み込み層

以下，6×6 のピクセルからなるモノクロ画像に対し，3×3 の要素をもつフィルターを 3 種類用意し，畳み込みを行うことを考える．

図 1 は，ピクセル(1,1)の入力値を $x_{1,1}$，ピクセル(1,2)の入力値を $x_{1,2}$,…，ピクセル(6,6)の入力値を $x_{6,6}$ として，3 枚あるフィルターのうち 1 枚目のフィルター（フィルター1 と表記する）を用いて畳みこむ様子を図示したものである．図 1 では，入力層を 0 層目として，l 層目における k 枚目のフィルターでの位置(p, q)での重みを$w_{p,q,k}^{(l)}$, l 層目における k 種類目のフィルターでのバイアスを$b_k^{(l)}$, l 層目における k 種類目のフィルターを通して得られる位置(m, n)での重み付き入力を$a_{m,n,k}^{(l)}$と表している．対象とする画像から，左上の 9 個のピクセルに対してフィルター1 を用いてスキャンし，1 つ目のユニットの重み付き入力を求める（図 1 の 1 図目）．続けて，1 つだけずらした 9 個のピクセルに対してフィルター1 を用いてスキャンし，2 つ目のユニットの重み付き入力を求める（図 1 の 2 図目）．このずらしていく処理のことをストライドといい，ずらす幅をストライドサイズという．6×6 の画像に対し，ストライドサイズ 1 でストライドさせると，右下の 9 個のピクセルに対してスキャンする（図 1 の 3 図目）までに合計 16 個のユニットにおける重み付き入力が求められる．



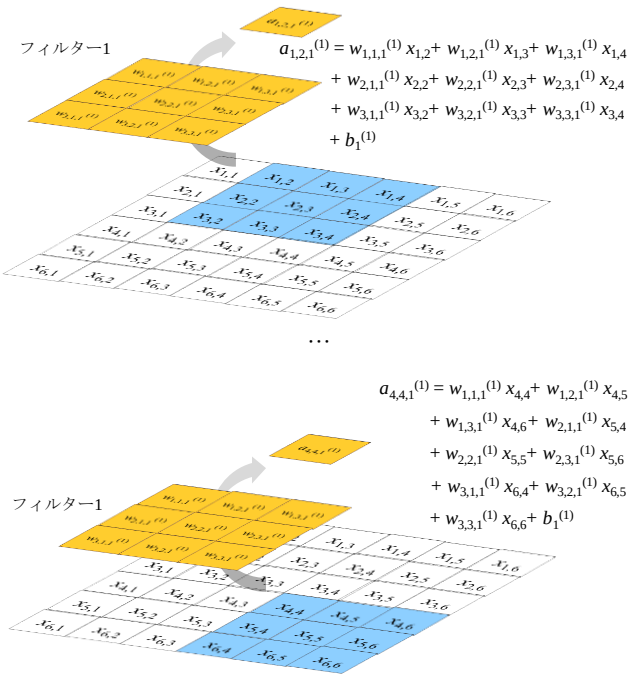


図 1　フィルターを用いた入力層からの畳み込み．6×6 のピクセルからなる画像に対して，3×3 の要素をもつフィルター1 を用いて左上から順にストライドサイズ 1 でスキャンし，畳み込み層のユニットでの重み付き入力を求める状況を示した．

このようにして得られたユニットは，空間的な情報を活かすために，図 2 のように平面上に並べることができる．このユニット群のことを特徴マップとよぶ．入力層からの畳み込みにより得られる 1 層目の特徴マップ k の位置(m, n)にあるユニットの重み付き入力$a_{m,n,k}^{(1)}$は，1 層目における k 種類目のフィルターでのバイアスを$b_k^{(1)}$として

$$a_{m,n,k}^{(1)} = \sum_{p=1}^{3}\sum_{q=1}^{3}\sum_{k=1}^{3} w_{p,q,k}^{(1)} x_{m+p-1,n+q-1} + b_k^{(1)} \quad (1)$$

により求められる(m = 1, 2, 3, 4; n = 1, 2, 3, 4; k = 1, 2, 3).

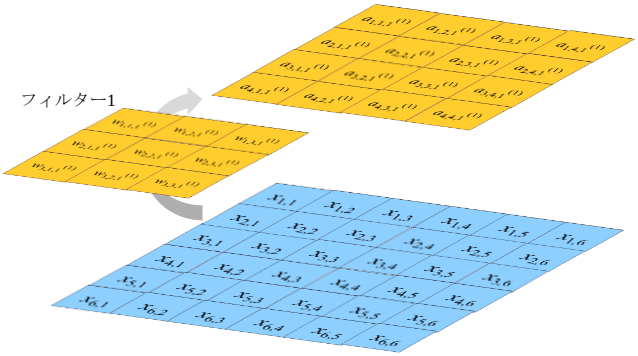


図 2　畳み込みにより得られる特徴マップ．6×6 のピクセルからなる入力層に対し，3×3 の要素をもつフィルター1 を用いて，ストライドサイズ 1 で畳み込みを行った結果得られる特徴マップを示した．

Vol.2020-MPS-128 No.8
Vol.2020-BIO-62 No.8
2020/6/29

同様に，3 種類あるフィルターの 2 種類目のフィルター（フィルター2 と表記する）を用いて入力層を畳み込み，最後となる 3 種類目のフィルター（フィルター3 と表記する）を用いて入力層を畳み込むことで，図 3 に示すような 3 つの特徴マップをもつ畳み込み層が得られる．図 3 の $z_{m,n,k}^{(1)}$は，入力層からの畳み込みにより得られる 1 層目の特徴マップ k の位置(m, n)にあるユニットからの出力であり，$h(a)$を活性化関数とすると

$$z_{m,n,k}^{(1)} = h\left(a_{m,n,k}^{(1)}\right) \tag{2}$$

である$(m = 1, 2, 3, 4; n = 1, 2, 3, 4; k = 1, 2, 3)$.

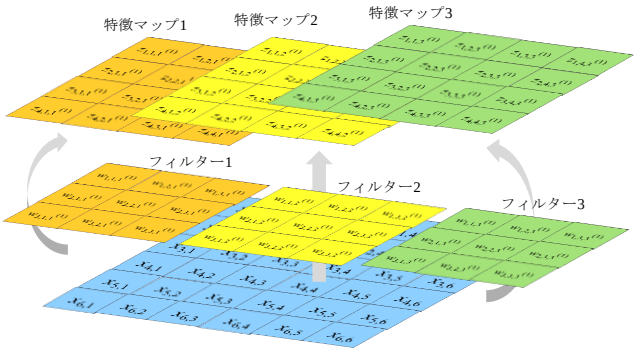


図 3　フィルターと畳み込み層．6×6 の要素からなる画像に対して，3×3 の要素をもつ 3 種類のフィルターを用いて得られる畳み込み層を示した．

2.2 プーリング層

図 4 に, サイズ 4×4 をもつ特徴マップ 1, 特徴マップ 2, 特徴マップ 3 からなる畳み込み層に対して，プーリングする領域のサイズを 2×2 としたときのプーリング結果を示した．

図 4 の 1 図目が特徴マップ 1 に対するプーリング結果である. サイズ 4×4 をもつ 1 枚の特徴マップに対して 2×2 の小領域を 1 つのユニットとしてまとめるので，1 枚の特徴マップからは4つのユニットをもつ1つの層が形成される．図中の$z_{i,j,k}^{(l)}$は l 層目の特徴マップ k の位置(i,j)にあるユニットからの入力を，$z_{m,n,k}^{(l+1)}$はプーリングの結果得られる$(l+1)$層目の特徴マップ k の位置(m, n)のユニットからの出力を，それぞれ表している．

図 4 の 2 図目は，特徴マップ 1，特徴マップ 2，特徴マップ 3 からなる畳み込み層からプーリング層が得られる全体像を示したものである．プーリング層では重みやバイアスはなく，ユニットの入力と同ユニットからの出力は同じ値をもつ．プーリングの方法としては，プーリング対象となる小領域に含まれるユニットの入力値の最大値を，プーリング層からの出力値とする方法や，プーリング対象となる小領域に含まれるユニットの入力値の平均値を，プーリング層からの出力値とする方法などがある．

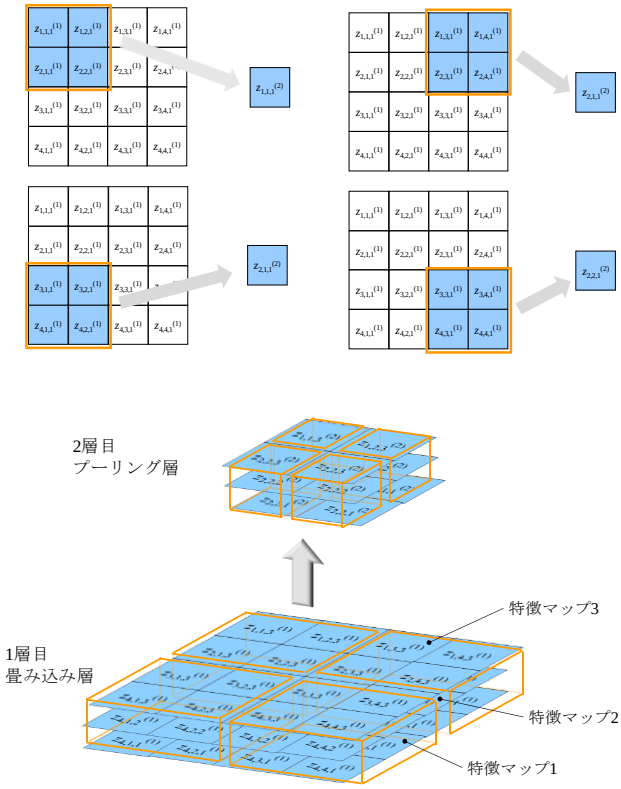


図 4　プーリング層の作成．1 図目に，サイズ 4×4 をもつ特徴マップに対して 2×2 のユニットをひとつのユニットにプーリングする様子を図示した．2 図目に，サイズ 4×4 をもつ 3 枚の特徴マップをもつ畳み込み層からプーリング対象となる領域を 2×2 としてプーリング層を作成する様子を図示した．

2.3 パラメータの決定

フィードフォワード・ニューラルネットワークのパラメータを，勾配降下法(gradient descent method)や確率的勾配降下法(stochastic gradient descent method)で推定する場合，パラメータの反復計算において各ユニットの出力と各ユニットの誤差が必要である．各ユニットの出力は入力層から出力層の順に計算でき，各ユニットの誤差は誤差逆伝播法[3]により出力層から入力層へと情報の流れと逆に計算できる．畳み込みニューラルネットワークにおいても，勾配降下法や確率的勾配降下法を用いてパラメータの反復計算を行う場合, 各ユニットの出力と各ユニットの誤差が必要であり，各ユニットの出力は入力層から出力層の順に，各ユニットの誤差は誤差逆伝播法により出力層から入力層の順に，それぞれ求めることができる．ただし，CNN の場合，中間層を構成する畳み込み層及びプーリング層が複数の層をもつため，計算がやや複雑になる．

2.4 実装するためのフレームワーク

畳み込みやプーリングといった，よく使う処理を行うプログラムをひとまとめにしたライブラリの集合体であるフ

Vol.2020-MPS-128 No.8
Vol.2020-BIO-62 No.8
2020/6/29

レームワークを用いることが考えられる．TensorFlow は，深層学習において最もよく用いられるフレームワークのひとつである．その理由としては，CPU でも GPU でもほぼ同じコードで実行できること，ネットワークでの処理を可視化するツールが整備されていること，低レベル API から高レベル API まで幅広くカバーしていることなどが挙げられる[4]．TensorFlow は，C++や Java などのプログラミング言語にも対応しているが，API の完成度の高さから，Python を用いることが一般的である．

低レベル API の場合，様々な機能を指定できる反面，処理の中身まで記載するためコードが長くなりやすい．このため，CNN のような複雑なネットワークを構築するためには，少ないコードで簡潔に記載できる高レベル API が用いられる．Keras は高レベル API のひとつであり Python のコードで記述される．Keras には TensorFlow と統合したバージョンと，TensorFlow, Theano, Microsoft Cognitive Toolkit (旧称 CNTK)のいずれかをバックエンドとするバージョンと，TensorFlow と統合したバージョンが存在するが，現在ではこれらバージョンの違いをほとんど意識せずに使用できる．CNN の実装例は，文献[4][5][6][7][8][9]など多数紹介されている．Keras では，畳み込み層やプーリング層での処理を行うレイヤーが用意されており，これらを add メソッドで順に追加していくことでネットワークを構築できる．構築するにあたっては、Sequential API を用いる方法と Functional API を用いる方法とがある．Sequential API を用いる場合，ネットワークの構造に対応させて簡潔にコード化できるが，層への入力や層からの出力が複数あるようなモデルを記述することができない．このため，複数の入出力をもつレイヤーや共通レイヤーをもつようなネットワークモデルを構築する際には，Functional API が用いられる．

3. 畳み込みニューラルネットワークを用いた物体検出

3.1 Regions with CNN features (R-CNN)

R-CNN[10]のパイプラインは，以下に示すように，大きく 3 つのステージから構成される．

・第 1 ステージ

入力画面から，物体が存在すると思われる領域の候補（region proposal；以下「領域候補」という）を複数抽出する．選択的探索法であれば，まず入力画面をいくつかの小さい領域に分割し，隣り合う領域同士の類似度を計算する．続いて最も類似度の高い領域のペアを選択し，そのペアを統合してひとつの領域とみなす．類似度の計算と最も類似度の高いペアの統合を，全ての領域候補が抽出されるまで繰り返すことによって領域候補を抽出する．

・第 2 ステージ

抽出された領域候補の画像を全て一定の大きさにリサイズして，既に学習済みの畳み込みニューラルネットワークに入力する．畳み込みニューラルネットワークの途中の畳み込み層から出力される特徴マップから，サポートベクトルマシンなどの分類器を用いて，抽出された領域候補ごとに，領域候補に含まれている物体に対してクラスを付与し，その物体らしさをスコア化する．

・第 3 ステージ

スコアがある一定以上となる領域候補を，対象となる物体が含まれるバウンディングボックスとして検出したとき，複数のバウンディングボックスが選択されることがあり得る．そこで，同一の物体に対して複数のバウンディングボックスが検出されないようにするために，バウンディングボックスごとに検出の信頼度を表すスコアを算出し，このスコアが非最大値の領域を外すことで，ある物体に対するただひとつのバウンディングボックスを決定する．この処理を NMS (non-maximal suppression) という．第 3 ステージでは, NMS を行った上でバウンディングボックスの形状を予測する．

3.2 Fast Region-based Convolutional Network method (Fast R-CNN)

R-CNN [10]は，優れた物体検出のためのネットワークであったが，3.1 に示したとおり学習が多段階（大きく分けて第 1 ステージから第 3 ステージまでの 3 段階）で行われるため，実行に時間がかかり物体検出が遅いという欠点があった．Girshich [11]は，これらの欠点を克服し，実行の速度をあげ物体の検出性能を高めたネットワークとして，Fast R-CNN を提案した．

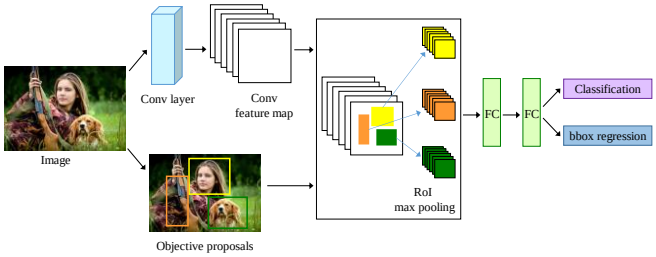


図 5　Fast R-CNN の構造．Girshich [11], 原田[12], 山下[13]を参考に作成した．画像を CNN に入力し途中の畳み込み層から畳み込み特徴マップを得る．これと並行して選択的探索法などを用いて領域候補を抽出する．関心領域プーリング層では，抽出された領域候補に対応する特徴マップを切り出し，一定のサイズになるよう変換した上で最大プーリングを行う．全結合では，入力データに対するクラスの事後確率を算出し，バウンティングボックスへの回帰を行いクラスに依存した位置と大きさを出力する．

Fast R-CNN の改善点のひとつが，関心領域((region of interest; RoI)プーリング層(RoI pooling layer)の導入にある．関心領域プーリング層では，領域候補に対応する特徴マップに対して最大プーリングを行い，固定サイズの特徴マップを得ることにある．

Fast R-CNN のもうひとつの改善点として，学習時にクラス分類とバウンディングボックスへの回帰を同時に行う点が挙げられる．K 個の異なるクラスを検出する場合，Fast R-CNN の出力層から，関心領域ごとに当該クラスの事後確率 $\mathbf{p} = (p_0, p_1, p_2, \ldots, p_K)^{\mathrm{T}}$ と，クラス u でラベル付けされたバウンティングボックスのパラメータ $\mathbf{t^u} = (t_x^k, t_y^u, t_h^u, t_w^u)$ が出力される．学習のための訓練データに対して，真のクラス u でラベル付けされたバウンティングボックスのパラメータ $\mathbf{v} = (v_x, v_y, v_w, v_h)^{\mathrm{T}}$ が所与のもとで，マルチタスク損失 L を

$$L(\mathbf{p}, u, \mathbf{t^u}, \mathbf{v}) = L_{cls}(\mathbf{p}, u) + \lambda[u \geq 1]L_{loc}(\mathbf{t^u}, \mathbf{v})$$
$$= -\ln(p_u) + \lambda[u \geq 1]\sum_{i\in\{x,y,w,h\}} smooth_{L_1}(t_i^u - v_i) \quad (3)$$

と定義し，L を最小とするようネットワークモデルのパラメータを推定する．(3)において，[ ]はアイバーソンの記法(Iverson bracket)であり，[ ]の中が真であれば 1 で偽であれば 0 をとる．また，(3)右辺第 2 項の$smooth_{L_1}()$は，$smooth_{L_1}$ 誤差関数

$$smooth_{L_1}(x) = \begin{cases} 0.5x^2 & (|x| \leq 1 \text{ のとき}) \\ |x| - 0.5 & (\text{それ以外}) \end{cases} \quad (4)$$

である．(3)の λ は，クラス認識の損失$L_{cls}(\mathbf{p}, u)$とバウンティングボックスへの回帰による損失$L_{loc}(\mathbf{t^u}, \mathbf{v})$とのバランスをとるためのハイパーパラメータである．

3.3 Faster R-CNN

Faster R-CNN [14]では，領域提案ネットワーク(region proposal network; RPN)を導入し，物体の領域候補を抽出するモジュールと，物体のクラス予測及びバウンティングボックスへの回帰を行うモジュールとを単一のネットワークで行うことにより，物体検出システムの低コスト化を図っている．

領域提案層では，アンカーと呼ばれる四角い箱を用意し，畳み込み特徴マップから選択された局所領域ごとに，物体の有無を予測し，物体があると予測されるならその位置情報を出力する．アンカーは，基準となるアンカーサイズ，アンカーの大きさ(scale)，アンカーの縦横比(aspect ratio)を用いて生成される．

領域提案層に続く関心領域プーリング層では，領域提案層から出力される，アンカーごとの，物体かどうかの予測確率と物体の位置情報に関する予測結果とを用いて，関心領域に対する特徴マップを抽出しプーリングを行う．全結合層(full connected layer)により 1 次元に集約した後，クラス分類とバウンディングボックスへの回帰を行う．

図 6 に入力画像から領域提案層を経て物体の候補領域を生成するまでの処理の流れを示した．画像を既に学習済みの CNN に入力し特徴マップを抽出する機能，抽出された特徴マップから物体の領域候補を提案する機能，提案された領域候補に対して物体のクラス分類と位置を予測する機能から構成されている．CNN により抽出された特徴マップから物体の領域候補を提案する領域提案層の学習，領域提案層からの出力を用いて入力画像に対する物体のクラス分類とバウンティングボックスへの回帰を行う Fast R-CNN 層の学習には，マルチタスク損失が用いられる．

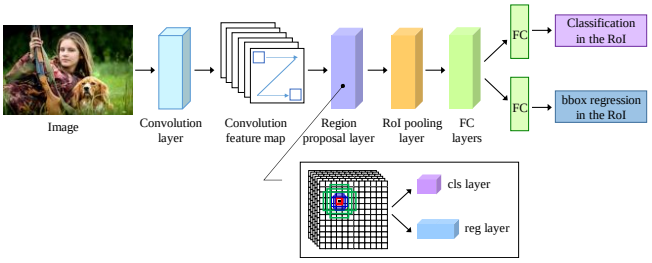


図 6　入力画像から物体の候補領域を生成するまでの処理の流れ．画像を CNN に入力し，畳み込み層から特徴マップを得る．領域提案層では，この特徴マップに対してアンカーボックスを用いて物体かどうかを予測し，物体の位置情報を推定する．関心領域プーリング層にて，物体かどうかの予測結果と物体の位置情報から，関心領域に対する特徴マップを抽出しプーリングを行う．全結合層を経て全結合された特徴量は，クラス分類層とバウンディングボックスへの回帰を行うバウンディングボックス回帰層へと入力される．

領域提案層におけるマルチタスク損失は，以下で定義される．

$$L(\{p_i\}, \{t_i\}) = \frac{1}{N_{cls}}\sum_i L_{cls}(p_i, p_i^*)$$
$$+\lambda\frac{1}{N_{reg}}\sum_{i\in\{x,y,w,h\}} p_i^* L_{reg}(t_i, t_i^*)$$
$$= \frac{1}{N_{cls}}\sum_i\{-p_i^*\ln p_i - (1-p_i^*)\ln(1-p_i)\}$$
$$+\lambda\frac{1}{N_{reg}}\sum_{i\in\{x,y,w,h\}} p_i^* smooth_{L_1}(t_i - t_i^*) \quad (5)$$

ここに，右辺第 2 項の$smooth_{L_1}()$は$smooth_{L_1}$誤差関数であり，λ はクラス認識の損失とボックスへの回帰による損失とのバランスをとるためのハイパーパラメータである．また，p_i は領域提案層で生成される i 番目のアンカー（以下「i 番目のアンカー」と表記する）が物体かどうかを予測した確率，t_i は i 番目のアンカーにおける物体の位置情報である．p_i^*は i 番目のアンカーに対する真のラベルであり，当該アンカーの中身が物体であれば $p_i^* = 1$，物体でなければ（つまり背景として扱うのであれば）$p_i^* = 0$ となる．t_i^*は i 番目のアンカーの真の位置情報である．

Fast R-CNN 層におけるマルチタスク損失は，領域提案

層におけるマルチタスク損失と同様である．ただし，以下の点が異なる．

・領域提案層のマルチタスク損失では，領域提案層で生成されるアンカーを要素とする（定義式の添え字 i は領域提案層で生成される i 番目のアンカーである）．これに対して Fast R-CNN 層のマルチタスク損失の要素は，領域提案層から出力される関心領域である（定義式の添え字 i は領域提案層から出力される i 番目の関心領域である）．

・領域提案層のマルチタスク損失では，アンカーと真のボックスとの IoU (intersection over union) により物体かどうかを予測する．これに対して Fast R-CNN 層のマルチタスク損失では，予測された物体のバウンティングボックスと真のバウンティングボックスとの IoU により物体のクラスを分類する．

・領域提案層では，クラス認識の損失の要素は $L_{cls}(p_i, p_i^*) = -p_i^* \ln p_i - (1 - p_i^*)\ln(1 - p_i)$ である．これに対して Fast R-CNN 層では，クラス認識の損失の要素は $L_{cls}(p_i, p_i^*) = -\ln(p_i^c)$ である．ここに，p_i^c はクラス c に分類される領域提案の予測確率である．

3.4 実装するためのソフトウェア

Faster R-CNN を提案した Ren *et al.*[14]は，MATLAB で利用可能なコードと Caffe (Pycaffe)で利用可能なコードを GitHub 上で公開した[15]．MATLAB では，深層学習におけるネットワークの構築と実装を行うためのフレームワークとして Deep Learning Toolbox が用意されており，Faster R-CNN による物体検出に加え，YOLO v2 を利用した物体検出についても詳細に解説されている．フレームワークを用いることにより，大量のデータに対してもバリデートされた処理を行える．その反面，予め全ての処理が埋め込まれているため，学習の効率化やモデルを構成するパラメータを任意に設定したい場合には，それらを反映させるためにプログラムを書く必要が生じる．Python は，プログラミングコードの利便性と可読性，専門的ライブラリの豊饒さから，機械学習の開発者が広く利用しているプログラミング言語となっている．MATLAB においても Python のライブラリを呼び出すことができる．

4. おわりに

本稿では，ディープ・ニューラルネットワークのモデルとして CNN をとりあげ，簡単に解説した．更に CNN の応用例として，物体検出の方法である R-CNN とその後継にあたる Fast R-CNN, Faster R-CNN についてまとめた．ネットワークの表現力は，層の幅に対して多項的であるのに対し，層の深さに対して指数的であるといわれている[12][16]．よって，ネットワークの表現力を高めるためには，単純にパラメータ数を増やすよりも多層化するほうが効率的である．また多層化することで，誤差関数の局所的極小値が大局的極小値へ降下すると予想されている[16]．これらのことが，ディープ・ニューラルネットワークの利用価値を高めていると思われる．

コンピュータに視覚を与えるというのはどういうことか，コンピュータはヒトと同等以上の能力を有することができるのかなど，コンピュータビジョンという世界に興味を持ち，数理背景を理解しコンピュータ上で実装する際に，本稿がその一助になれば幸いである．生物はカンブリア紀に大きな進化を遂げたが，そのきっかけとなった事象が眼の獲得にあるといわれている[17]．コンピュータも視覚をもつことで爆発的な発展を遂げるのだろうか．

参考文献

[1] 我妻幸長. はじめてのディープラーニング. SB クリエイティブ社, 2018, 339p.
[2] 関根崇之. 60 分でわかる！ディープラーニング最前線. クルート社, 2018, 159p.
[3] Rumelhart, D., E., Hinton, G., E., and Williams, R., J. Learning representations by back-propagating errors. Nature, 1986, 323, 533–536.
[4] 太田満久, 須藤広大, 黒澤匠雅, 小田大輔. 現場で使える! TensorFlow 開発入門 Keras による深層学習モデル構築手法. 翔泳社, 2018, 383p.
[5] 牧野浩二, 西崎博史. Python による深層強化学習. オーム社, 2018, 248p.
[6] チーム・カルポ. TensorFlow&Keras プログラミング実装ハンドブック. 秀和システム, 2018, 368p.
[7] チーム・カルポ. 物体・画像認識と時系列データ処理入門 TensorFlow/Keras/TFLearn による実装ディープラーニング, 秀和システム, 2019, 530p.
[8] 木村優志. 現場で使える！Python の深層学習入門. 翔泳社, 2019, 277p.
[9] 中井悦司. TensorFlow と Keras で動かしながら学ぶディープラーニングの仕組み 畳み込みニューラルネットワーク徹底解説. マイナビ出版, 2019, 272p.
[10] Girshick, R., Donahue, J., Darrell, T., and Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In Conference on Computer Vision and Pattern Recognition, 2014.
[11] Girshick, R. Fast R-CNN. In International Conference on Computer Vision, 2015.
[12] 原田達也. 機械学習プロフェッショナルシリーズ 画像認識. 講談社, 2017, 277p.
[13] 山下隆義. イラストで学ぶ ディープラーニング 改訂第 2 版. 講談社, 2018, 288p.
[14] Ren, S., He,K., Girshick, R., and Sun, J. Faster R-CNN: Towards real-time object detection with region proposal networks. In Neural Information Processing Systems, 2015.
[15] Ren, S., He, K., Girshick, R., and Sun, J. Faster R-CNN: Towards real-time object detection with region proposal networks. arXiv:1506.01497v3, 2016.
[16] 瀧雅人.これならわかる深層学習入門. 講談社サイエンティフィック, 2017, 259p.
[17] アンドリュー・パーカー, 渡辺政隆・今西康子訳. 眼の誕生—カンブリア紀大進化の謎を解く. 草思社, 2003, 384p.