深層学習を用いた
人工光型植物工場生産レタスの
チップバーン自動検出に関する研究

2020 年 8 月

千葉大学大学院融合理工学府

基幹工学専攻電気電子工学コース

嶋村茂治

（千葉大学審査学位論文）

深層学習を用いた
人工光型植物工場生産レタスの
チップバーン自動検出に関する研究

2020 年 8 月

千葉大学大学院融合理工学府

基幹工学専攻電気電子工学コース

嶋村茂治

概要

近年，植物を安定的に生産する技術として植物工場が注目を集めている。植物工場は太陽光利用型と人工光型に大別され，人工光型植物工場では成長が早い葉菜類が栽培されるケースが多く，特にレタス類は主要作物となっている。レタス類を栽培する人工光型植物工場の主要な課題の一つにチップバーンと呼ばれる植物生理障害がある。チップバーンが発生するとその箇所は茶褐色に変色して野菜としての商品価値を損なうため，目視によりその有無を確認し，人手により発生した葉，または個体全体を除去する作業が行われる。しかし，この作業には多大なる労力とコストが発生する。チップバーンを自動検出して除去できれば，自動化・省力化による商業的利益は大きく，人工光型植物工場の普及の原動力となる。

本研究は，深層学習の一種である畳み込みニューラルネットワーク（Convolutional Neural Network：CNN）による画像診断技術を，人工光型植物工場で栽培されるレタスのチップバーン発生・未発生の 2 クラス分類へ導入することを目的とする。深層学習用の多量の画像を得るために，チップバーンが発生する栽培条件を特定し，実際にチップバーンレタスを栽培する。画像診断用の CNN については，GoogLeNet，ResNet などのモデル，Adam，Nadam などの最適化手法を試行し，チップバーンの検出性能を比較することで，人工光型植物工場生産レタスの画像診断技術を確立する。

キーワード:
人工光型植物工場，植物生理障害，チップバーン，画像診断，深層学習，畳み込みニューラルネットワーク

Detection of tipburn of lettuce produced at plant factory with artificial light using deep learning

Shigeharu SHIMAMURA

Abstract:
Plant factory with artificial light (PFAL) is attracting worldwide attention as a technology for stably producing crops. One of the major problems of PFAL is tipburn which is a physiological disorder of crops. Lettuce cultivated in FPAL in particular has a high frequency of tipburn. When tipburn occurs, leaf tips discolor blackly and the commercial value as vegetables is damaged. Identification of tipburn is done by human eye observation, and tipburn leaves are trimmed by hand or that lettuce is removed from products. These operations require much labor and cost. If tipburn identification can automatically be done using deep learning, the economic effect will be great and it will be a driving force for spreading PFAL. In this thesis, we aim to perform binary discrimination of tipburn occurrence and its non-occurrence about lettuce cultivated in PFAL using deep learning with convolutional neural networks (CNNs). In order to obtain a large amount of images for deep learning, the cultivation conditions that cause tipburn are specified and tipburn lettuce is actually grown. For CNNs for image diagnosis, models such as GoogLeNet and ResNet and optimization methods such as Adam and Nadam are tried, and the tipburn detection performance is compared. The results of the image diagnosis experiments indicate that the recognition of tipburn can be performed with high accuracy, and thus we establish image diagnosis technology for lettuce cultivated in PFAL.

Keywords:
plant factory with artificial light, plant physiological disorder, tipburn, image diagnosis, deep learning, convolutional neural network.

目次

第１章
まえがき

1.1　背景

近年，ロボット技術，情報通信技術（Information and Communication Technology: ICT），および人工知能（Artificial Intelligence: AI）技術などの最先端技術を活用することで，労働力の削減と生産物の高品質化を推進するスマート農業と呼ばれる新しい農業技術が，農業従事者の高齢化および労働力不足化が，今後，急速に深刻化する我が国の農業分野において，重要性を増している[1]。スマート農業の導入により，農業従事者の農作業の削減もしくは軽減を推進することで，新規農業従事者の獲得と農業生産技術の後継者への継承の推進が期待できる。

スマート農業を実現するための要素技術の一つとして，植物工場が農業分野のみならず工業分野からも注目を集めている。植物工場は太陽光利用型と人工光型に大別される。太陽光利用型植物工場は，ビニルやガラス等の透過性のある被覆資材で覆われた施設で，太陽光線を利用して植物栽培を行う技術であり，人工光型植物工場は，人工光源のみで植物栽培を行う技術である。これらのうち人工光型植物工場は，室内の閉じられた空間内で，屋外の天候の影響を受けることなく，環境条件を高精度に制御可能であり，植物を安全かつ安定的に生産できる[2]。人工光型植物工場では，成長が早い葉菜類が栽培されるケースが多く，特にレタス類は主要作物となっており，現時点での作付け割合は，全作物のうち約 85%を占めるとされている。レタス類を栽培する人工光型植物工場の主要な課題の一つに，チップバーンと呼ばれる植物生理障害がある。チップバーンとは，植物の葉の成長点が壊死する現象である[3][4][5]。人工光型植物工場では，植物に最適な栽培条件を設定することで，植物の成長を促進し，播種から収穫までの期間を大幅に短縮化できる。しかし，人工光型植物工場では，栽培条件を調整することで植物の成長を過剰に促進させてしまい，この影響によりチップバーンを誘発しやすい傾向がある。特に人工光型植物工場で生産されるレタス類は，チップバーンの発生頻度が高い。ひとたびチップバーンが発生するとその箇所は茶褐色に変色して，野菜としての商品価値を損なうため，目視によりその有無を確認し，人手により発生した葉，または個体全体を除去する作業が行われる[6]。しかし，この作業には多大なる労力とコストが発生する。

一方で，近年，計算機の性能向上に伴い，深層学習が発展している。画像認識においては，畳み込みニューラルネットワーク（Convolutional Neural Network: CNN）を用いた深層学習による画像診断技術の有効性が確認されている[7]。また，CNN については，新たなモデル，新たな最適化手法が，多数，報告されており，画像診断の精度は

ますます高まっている。深層学習による画像診断技術を用いることで，チップバーンを自動検出して，チップバーンが発生したレタスを自動除去できれば，自動化・省力化による商業的利益は極めて大きく，人工光型植物工場の普及の原動力となり得る。

1.2 研究目的

本研究は，深層学習の一種である CNN による画像診断技術を，人工光型植物工場で栽培されるレタスのチップバーン発生･未発生の 2 クラス分類へ導入することを目的とする。レタスの一種であるフリルレタスのチップバーンは，植物体の中央部の葉先に発生するタイプ（A 型）と，植物体の周辺部の葉先に発生するタイプ（B 型）がある。さらに，チップバーン予兆，すなわち緑色の色素が失われてわずかに退色し，チップバーンが発生して葉先が茶褐色に変色して，野菜としての商品価値が損なわれる直前の状態がある。まず，深層学習で必須となる多量のカラー画像データを取得するために，A 型および B 型のチップバーンが発生する栽培条件を特定し，実際に A 型チップバーンレタス，B 型チップバーンレタスを栽培する。A 型チップバーン予兆レタスは，A 型チップバーンレタスの栽培の過程で作出できる。

画像診断用の CNN については，VGGNet[8]，GoogLeNet[9]，ResNet[10][11]，WideResNet[12]の各モデル，および Momentum SGD，Adam[13]，Adamax[14]，Nadam[14][15]の各最適化手法を試行し，チップバーンの検出性能の比較を通じて，チップバーン識別率の向上手法を検討する。また，CNN を用いた深層学習の有効性を明確に示すために，2 クラス分類の古典的かつ代表的な手法の一つであるサポートベクターマシン（Support Vector Machine: SVM）[16][17]を比較対象に加える。

CNN を用いた深層学習による画像診断実験の結果，チップバーン発生・未発生の 2 クラス分類において，B 型チップバーンでは正解率（test accuracy）0.929[18][19]，A 型チップバーンでは正解率 0.986[[20][21][22], A 型チップバーン予兆では正解率 0.990[23][24]を達成している。これらの正解率は，人工光型植物工場生産作物の経営的歩留り目標である 90%を上回っており，本研究の有用性が確認できる。また，古典的手法である SVM[16][17]と比べて，CNN はすべての場合でチップバーンの識別性能が上回ることを示し，CNN を用いた深層学習の有効性を確認する。以上により，人工光型植物工場生産レタスの深層学習による画像診断技術を確立する。

1.3　論文構成

以降，第 2 章では本研究に関する基礎的事項について述べ，第 3 章では提案手法について述べる。第 4 章では画像診断実験について述べ，第 5 章では本研究を総括し，今後の課題について述べる。

第 2 章
基礎的事項

2.1　人工光型植物工場生産レタスの課題

人工光型植物工場[2]でのレタス栽培における主要な課題の一つに，チップバーンと呼ばれる生理障害がある。葉菜類のチップバーンは，カルシウムの不足や乳管の損傷により，植物の葉先が茶褐色に変色する現象[3][4][5]であり，特にフリルレタスにおいてチップバーンの発生頻度が高い。チップバーンには，植物体の成長点付近，すなわち植物体の中央部分の葉先に発生するタイプ（A 型）と，植物体の外側部分の葉先に発生するタイプ（B 型）の 2 種類がある。A 型チップバーンは，成長量が旺盛のときに発生しやすい。一方，B 型チップバーンは，光強度の増大や空気の乾燥などのストレスによって発生しやすい。正常，A 型チップバーン，B 型チップバーンのフリルレタスを，図 2.1，図 2.2，図 2.3 にそれぞれ示す。チップバーンが発生した作物は，葉先が茶褐色に変色して，野菜としての商品価値を大きく損なう[6]ため，除去作業を行う必要がある。チップバーンの識別と除去は現状，目視と人手で行っており，これらには大きな労力とコストを要する。チップバーンを自動検出して除去できれば，自動化・省力化による商業的利益は大きく，人工光型植物工場の普及の原動力となり得る。

チップバーンにはその発症の前段階として予兆の状態が存在する。これはチップバーンの発生直前の段階，すなわち，葉先が茶褐色に変色する直前で，緑色の色素が失われてわずかに退色し，微かに葉先が縮れる症状が表れたものである。チップバーンの予兆を検出できれば，その個体の栽培条件に適切なフィードバックをかけることでチップバーンの発症を防止できる。また，チップバーンの予兆の段階であれば，葉先の茶褐色への変色がないため，野菜としての商品価値が保たれており，出荷が可能である。レタス類では，正常からチップバーンの予兆の段階を経て，チップバーン発生にいたる。正常とチップバーン予兆の間の遷移が可逆的であるのに対し，チップバーン予兆からチップバーン発生への遷移は不可逆的である。チップバーンの予兆を自動検出できれば，その後の栽培条件を調整することで，チップバーンの発生を抑制しながら栽培を継続し，出荷可能な状態になるまで成長させることができるため，商業的な価値は極めて大きい。A 型チップバーンの予兆の症状が表れたレタスを図 2.4 に示す。

図 2.1　正常レタス

図 2.2　A 型チップバーンレタス

図 2.3　B 型チップバーンレタス

図 2.4　A 型チップバーン予兆レタス

2.2　分類問題の評価指標

チップバーンの検出は，正常かチップバーンかの 2 クラス分類問題である。2 クラス分類における混同行列（confusion matrix）を表 2.1 に示す。ここで，正例を陽性（チップバーン），負例を陰性（正常）とする。

表 2.1　2 クラス分類における混同行列

	正例	負例
正であると予測	TP (true-positive)	FP (false-positive)
負であると予測	FN (false-negative)	TN (true-negative)

2 クラス分類問題の評価指標として，式(2.1)で表される正解率（accuracy）が良く用いられる。

$$\text{accuracy} = \frac{TP + TN}{TP + FP + FN + TN} \tag{2.1}$$

しかし，分類器の植物工場実装時には，通常は正常の個体が多く，チップバーンの個体は少ない。このように, 正例と負例のサンプル数の比率に著しく偏りがある場合には, accuracy は評価指標として適さない場合があり，式(2.2)で表される精度（precision）や，式(2.3)で表される再現率（recall） が用いられる。

$$\text{precision} = \frac{TP}{TP + FP} \tag{2.2}$$

$$\text{recall} = \frac{TP}{TP + FN} \tag{2.3}$$

precision は分類器が正であると予測したサンプルのうち，実際に正例である割合であり，recall は実際に正例であるサンプルのうち分類器が正であると予測した割合である。一般的に，precision と recall はトレードオフの関係にある。

2.3　ニューラルネットワーク

2.3.1　数理モデル

人間の脳には，ニューロンと呼ばれる神経細胞が多数存在し，これらが結合して，巨大なネットワークを構成する。ニューラルネットワークは，このような脳の仕組みを模倣した数理モデルである。

ニューロンは n 次元の入力 $x = (x_1, \ldots, x_N)$を受け取り，式(2.4)により y を出力する。

$$y = f\left(\sum_{i=1}^{N} w_i x_i - b\right) \tag{2.4}$$

ここで，w_iは重み，b は閾値と呼ばれる実数値のパラメータである。関数 $f(\cdot)$は一般に非線形の関数であり，活性化関数と呼ばれる。

ニューラルネットワークは，階層型（フィードフォワード型）と相互結合型（リカレント型）に大別される。階層型ニューラルネットワークは，ニューロンが複数の層に分かれており，ある層のニューロンは，その層より後ろの層のニューロンと結合している。図 2.5 は階層型ニューラルネットワークの一例であり，このように，前の層のニューロンと後ろの層のニューロンがすべて結合した層を全結合層という。

全結合層の問題点として，入出力が 1 次元ベクトルであるため，入力データの形状が無視されることがあげられる。画像を例にとると，画像は縦，横，RGB チャンネルという 3 次元形状を持ち，この形状には，空間的に近くのピクセルは似た値を持つなどの重要な空間情報が含まれている。画像などの多次元データの形状には汲み取るべき本質的なパターンがあるにもかかわらず，全結合層ではそれが無視される。

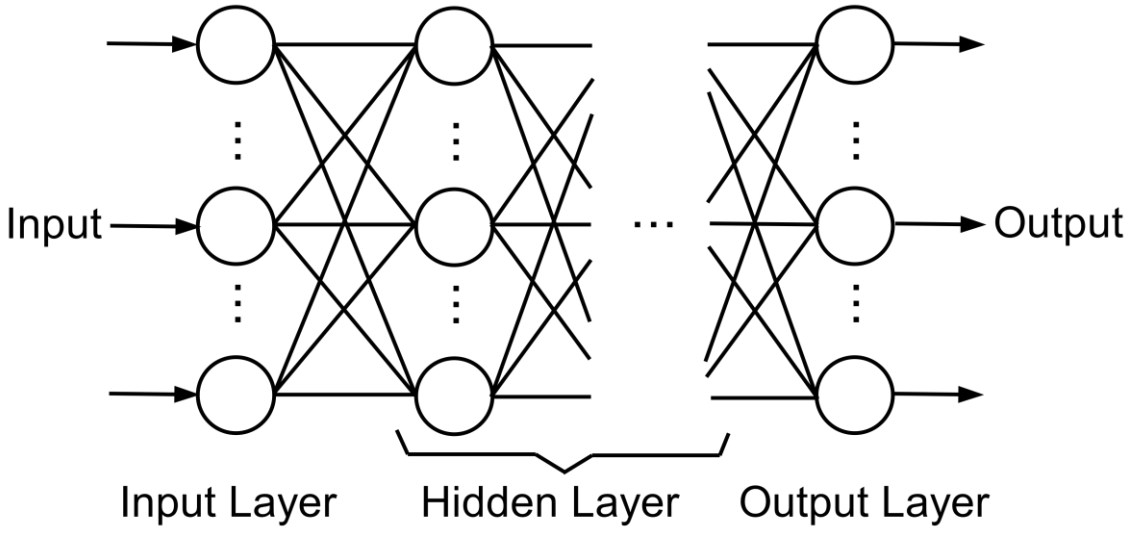


図 2.5　階層型ニューラルネットワーク

2.3.2　活性化関数

活性化関数は，通常，単調増加の非線形関数が採用される。シグモイド関数や双曲線正接関数（tanh）は古くからよく使われている活性化関数である。これらの関数は入力の絶対値が大きくなると次第に出力が飽和して一定値となるという特徴を持つ。近年では，入力が大きい値となっても出力が飽和しない正規化線形関数（ReLU）が活性化関数として採用されることが多い[25]。ReLU 関数は式(2.5)で表される。

$$f(x) = \max(x, 0) \tag{2.5}$$

また，分類問題の出力層においては通常，ソフトマックス関数が採用される。出力層のニューロン数を N としたとき，出力層の k 番目のニューロンについてのソフトマックス関数は式(2.6)で表される。

$$f(x_k) = \frac{\exp(x_k)}{\sum_{i=1}^{N} \exp(x_i)} \tag{2.6}$$

2.3.3　学習問題の定式化

ニューラルネットワークの各ニューロンの重みと閾値をすべてまとめて $\boldsymbol{\theta}$ とする。M 次元の入力 $\boldsymbol{x}$ と N 次元の目標出力 $\boldsymbol{d}$ の組 P 個からなる訓練用データセット$(\boldsymbol{x}^p, \boldsymbol{d}^p)$, $p = 1, \ldots, P$ を学習する場合について考える。入力をまとめて $X = [\boldsymbol{x}^1, \ldots, \boldsymbol{x}^P]$，目標出力をまとめて $D = [\boldsymbol{d}^1, \ldots, \boldsymbol{d}^P]$，出力層における n 番目のニューロンの実際の出力を y_n とすると，ニューラルネットワークの学習問題は，式(2.7)で与えられる。

$$\underset{\boldsymbol{\theta}}{\text{minimize}}\, L(\boldsymbol{\theta}; X, D) \tag{2.7a}$$

$$L(\boldsymbol{\theta}; X, D) = \frac{1}{P} \sum_{p=1}^{P} l(\boldsymbol{\theta}; \boldsymbol{x}^p, \boldsymbol{d}^p) \tag{2.7b}$$

ここで，関数 $l(\cdot)$は損失関数と呼ばれ，式(2.8)で表される二乗誤差関数，または式(2.9)で表されるクロスエントロピー誤差関数がよく用いられる。

$$l(\boldsymbol{\theta}; \boldsymbol{x}^p, \boldsymbol{d}^p) = \frac{1}{2} \sum_{n=1}^{N} (y_n(\boldsymbol{\theta}; \boldsymbol{x}^p, \boldsymbol{d}^p) - d_n)^2 \tag{2.8}$$

$$l(\boldsymbol{\theta}; \boldsymbol{x}^p, \boldsymbol{d}^p) = -\sum_{n=1}^{N} d_n \log y_n(\boldsymbol{\theta}; \boldsymbol{x}^p, \boldsymbol{d}^p) \tag{2.9}$$

2.3.4　出力の解釈

分類問題において，ニューラルネットワークの出力層では，例えば 3 クラス分類問題では[0.40, 0.10, 0.50]のように，入力データのクラス分類における予測は，クラス毎に実数値で出力される。ニューラルネットワークのクラス分類では，一般に，出力の最も大きいニューロンに相当するクラスだけを認識結果とする。先の例においては, 3 番目のクラスが最も確率が高いことから，入力データは 3 番目のクラスであると分類器は予測する。

2.3.5　勾配降下法

式(2.7b)は一般に凸関数ではなく, 大域的最適解を求めることが困難である。そこで, 代わりに局所的最適解を求めることを考える。局所的最適解を求める際によく用いられる手法の一つに勾配降下法がある。勾配降下法において，一つ前の時刻 $t-1$ におけるパラメータ $\theta(t-1)$に関する勾配は式(2.10)によって表され, 時刻 t におけるパラメータの更新は式(2.11)によって行われる。

$$\nabla L = \frac{\partial}{\partial \boldsymbol{\theta}} L(\boldsymbol{\theta}(t-1); \boldsymbol{X}, \boldsymbol{D}) \tag{2.10}$$

$$\boldsymbol{\theta}(t) \leftarrow \boldsymbol{\theta}(\mathrm{t}-1) - \alpha \nabla \boldsymbol{L} \tag{2.11}$$

ここで，$\alpha\,(0 < \alpha < 1)$ は，学習率と呼ばれる更新量の大きさを決める係数である。

2.3.6　バッチ学習とミニバッチ学習

ニューラルネットワークの学習は，一回の更新にすべての訓練用データセットを与えるバッチ学習と，一回の更新に一部の訓練用データセットを与えるミニバッチ学習に大別される。バッチ学習とミニバッチ学習の長所および短所を表 2.2 に示す。

表 2.2　バッチ学習とミニバッチ学習の長所および短所

	長所	短所
バッチ学習	一回の更新による 性能向上が大きい	一回の更新にかかる 計算コストが大きい
ミニバッチ学習	一回の更新にかかる 計算コストが小さい	一回の更新による 性能向上が小さい

なお，ミニバッチ学習において，一回の更新の際に与える訓練用データの個数 B をバッチサイズという。また，エポックと呼ばれる更新回数を示す単位がある。1 エポックは訓練用データをすべて使い切るのに要する更新回数である。すなわち，訓練用データの総数を P としたとき，P/B 回の更新が 1 エポックとなる。

2.3.7　ミニバッチ学習アルゴリズム

確率的勾配降下法

確率的勾配降下法（Stochastic gradient descent: SGD）は，ミニバッチとしてランダムに選択した訓練用データセットを用いて勾配降下法を行う手法である。SGD のアルゴリズムを Algorithm 1 に示す。ここで，T はパラメータ更新を行う回数，N は更新されるパラメータの総数であり，$|Q|$ は集合 Q の要素数を表す。

Algorithm 1 Stochastic gradient descent

Require: learning rate α

Require: training data indices set P

Require: training data set $(\boldsymbol{x}^p, \boldsymbol{d}^p)$, $p = 1, \ldots, P$

Require: loss function with respect to a training data $l(\boldsymbol{\theta}; \boldsymbol{x}, \boldsymbol{d})$

Require: initial decision variables $\boldsymbol{\theta}(0)$

for $t = 1 : T$ **do**

Choose indices from P and set them to $Q(t)$.

for n = 1 : N **do**

$$g_n(t) \leftarrow \frac{1}{|Q|} \sum_{\boldsymbol{q} \in \boldsymbol{Q}(\boldsymbol{t})} \frac{\partial}{\partial \theta_n} l(\boldsymbol{\theta}(\mathrm{t}-1); \boldsymbol{x}^q, \boldsymbol{d}^q)$$

$$\theta_n(t) \leftarrow \theta_n(\mathrm{t}-1) - \alpha \cdot g_n(t)$$

end for

end for

Momentum SGD

SGD では損失関数が変化することにより，勾配の降下方向への分散が大きくなる。Momentum SGD では，勾配の降下方向に一つ前の更新方向を加えることで，勾配の変化を抑える。Momentum SGD のアルゴリズムを Algorithm 2 に示す。

Algorithm 2 Stochastic gradient descent with momentum

Require: learning rate η

Require: momentum term μ

Require: training data indices set P

Require: training data set $(\boldsymbol{x}^p, \boldsymbol{d}^p)$, $p = 1, \ldots, P$

Require: loss function with respect to a training data $l(\boldsymbol{\theta}; \boldsymbol{x}, \boldsymbol{d})$

Require: initial decision variables $\boldsymbol{\theta}(0)$

$\boldsymbol{v}(0) = \boldsymbol{0}$

for $t = 1 : T$ **do**

Choose indices from P and set them to $Q(t)$.

for n = 1 : N **do**

$$g_n(t) \leftarrow \frac{1}{|Q|} \sum_{\boldsymbol{q} \in \boldsymbol{Q}(\boldsymbol{t})} \frac{\partial}{\partial \theta_n} l(\boldsymbol{\theta}(\mathrm{t}-1); \boldsymbol{x}^q, \boldsymbol{d}^q)$$

$$v_n(t) \leftarrow \mu \cdot v_n(\mathrm{t}-1) - \eta \cdot g_n(t)$$

$$\theta_n(t) \leftarrow \theta_n(\mathrm{t}-1) - \alpha \cdot v_n(t)$$

end for

end for

Adam

Adam[13]は，ミニバッチ学習アルゴリズムの一種である RMSprop に Momentum SGD を組み合わせた手法である。また，指数移動平均 $\boldsymbol{m}$, $\boldsymbol{v}$ の初期値が $\mathbf{0}$ から始まることを補正する項も導入されている。Adam のアルゴリズムを Algorithm 3 に示す。

Algorithm 3 Adam

Require: learning rate α

Require: exponential decay rates β_1, β_2

Require: fuzz factor ε

Require: training data indices set P

Require: training data set $(\boldsymbol{x}^p, \boldsymbol{d}^p)$, $p = 1, \ldots, P$

Require: loss function with respect to a training data $l(\boldsymbol{\theta}; \boldsymbol{x}, \boldsymbol{d})$

Require: initial decision variables $\boldsymbol{\theta}(0)$

$\boldsymbol{m}(0) = \boldsymbol{v}(0) = \mathbf{0}$

for $t = 1 : T$ **do**

Choose indices from P and set them to $Q(t)$.

for $n = 1 : N$ **do**

$$g_n(t) \leftarrow \frac{1}{|Q|} \sum_{\boldsymbol{q} \in \boldsymbol{Q}(\boldsymbol{t})} \frac{\partial}{\partial \theta_n} l(\boldsymbol{\theta}(\mathrm{t}-1); \boldsymbol{x}^q, \boldsymbol{d}^q)$$

$$m_n(t) \leftarrow \beta_1 \cdot m_n(\mathrm{t}-1) + (1-\beta_1) \cdot g_n(t)$$

$$\hat{m}_n(t) \leftarrow \frac{m_n(t)}{1-\beta_1^t}$$

$$v_n(t) \leftarrow \beta_2 \cdot v_n(\mathrm{t}-1) + (1-\beta_2) \cdot (g_n(t))^2$$

$$\hat{v}_n(t) \leftarrow \frac{v_n(t)}{1-\beta_2^t}$$

$$\theta_n(t) \leftarrow \theta_n(\mathrm{t}-1) - \alpha \cdot \frac{\hat{m}_n(t)}{\sqrt{\hat{v}_n(t)} + \epsilon}$$

end for

end for

Adamax

Adamax [13]は，Adam における L^2 ノルムを L^p ノルムに一般化し，$p \to \infty$の極限として定義した手法である。Adamax のアルゴリズムを Algorithm 4 に示す。

Algorithm 4 Adamax

Require: learning rate α

Require: exponential decay rates β_1, β_2

Require: fuzz factor ε

Require: training data indices set P

Require: training data set $(\boldsymbol{x}^p, \boldsymbol{d}^p)$, $p = 1, \ldots, P$

Require: loss function with respect to a training data $l(\boldsymbol{\theta}; \boldsymbol{x}, \boldsymbol{d})$

Require: initial decision variables $\boldsymbol{\theta}(0)$

$\boldsymbol{m}(0) = \boldsymbol{v}(0) = \boldsymbol{0}$

for $t = 1 : T$ **do**

Choose indices from P and set them to $Q(t)$.

for n = 1 : N **do**

$$g_n(t) \leftarrow \frac{1}{|Q|} \sum_{\boldsymbol{q} \in \boldsymbol{Q}(\boldsymbol{t})} \frac{\partial}{\partial \theta_n} l(\boldsymbol{\theta}(\mathrm{t}-1); \boldsymbol{x}^q, \boldsymbol{d}^q)$$

$$m_n(t) \leftarrow \beta_1 \cdot m_n(\mathrm{t}-1) + (1-\beta_1) \cdot g_n(t)$$

$$\widehat{m}_n(t) \leftarrow \frac{m_n(t)}{1-\beta_1^t}$$

$$v_n(t) \leftarrow \max(\beta_2 \cdot v_n(\mathrm{t}-1), |g_n(t)|)$$

$$\theta_n(t) \leftarrow \theta_n(\mathrm{t}-1) - \alpha \cdot \frac{\widehat{m}_n(t)}{v_n(t)+\epsilon}$$

end for

end for

Nadam

Nadam[14]は, Adam にネステロフの加速勾配法[15]を導入した手法である。Nadam のアルゴリズムを Algorithm 5 に示す。

Algorithm 5 Nadam

Require: learning rate α

Require: exponential decay rates β (0), . . . ,β (*t*)

Require: exponential decay rates γ

Require: fuzz factor ε

Require: training data indices set *P*

Require: training data set ($\boldsymbol{x}^p$, $\boldsymbol{d}^p$) , *p* = 1, . . . , *P*

Require: loss function with respect to a training data *l* ($\boldsymbol{\theta}$; $\boldsymbol{x}$, $\boldsymbol{d}$)

Require: initial decision variables $\boldsymbol{\theta}$ (0)

$\boldsymbol{m}$ (0) = $\boldsymbol{v}$ (0) = **0**

for *t* = 1 : *T* **do**

Choose indices from *P* and set them to *Q* (*t*).

for n = 1 : *N* **do**

$$g_n(t) \leftarrow \frac{1}{|Q|} \sum_{\boldsymbol{q} \in \boldsymbol{Q}(\boldsymbol{t})} \frac{\partial}{\partial \theta_n} l(\boldsymbol{\theta}(\mathrm{t}-1); \boldsymbol{x}^q, \boldsymbol{d}^q)$$

$$m_n(t) \leftarrow \beta(t) \cdot m_n(\mathrm{t}-1) + (1-\beta(t)) \cdot g_n(t)$$

$$\hat{m}_n(t) \leftarrow \frac{\beta(t+1)}{1-\prod_{i=1}^{t+1}\beta(i)} \cdot m_n(t) + \frac{1-\beta(t)}{1-\prod_{i=1}^{t}\beta(i)} \cdot g_n(t)$$

$$v_n(t) \leftarrow \gamma \cdot v_n(\mathrm{t}-1) + (1-\gamma) \cdot (g_n(t))^2$$

$$\hat{v}_n(t) \leftarrow \frac{\gamma}{1-\gamma^t} \cdot v_n(t)$$

$$\theta_n(t) \leftarrow \theta_n(\mathrm{t}-1) - \alpha \cdot \frac{\hat{m}_n(t)}{\sqrt{\hat{v}_n(t)} + \epsilon}$$

end for

end for

2.4　畳み込みニューラルネットワーク

2.4.1　ネットワーク構造

畳み込みニューラルネットワーク[25]（Convolutional Neural Network: CNN）はニューラルネットワークの一種であり，画像認識や音声認識などによく用いられる。CNNは，図 2.6 に示すように，画像の局所的な特徴抽出を担う畳み込み層と，局所ごとに特徴をまとめあげるプーリング層（サブサンプリング層）を繰り返した構造を持つ。畳み込み層では，入力データと畳み込みフィルタによる積和演算が行われる。一方，プーリング層では，入力データの小領域に対して最大値や平均値をとって一つの要素に集約する処理が行われる。

畳み込み層のフィルタのパラメータは画像中のすべての場所で共有されるため，単純な全結合層と比較してパラメータ数が大幅に減少する。また，プーリング層を交えることで，さらにパラメータ数を削減すると同時に，入力データの平行移動に対するロバスト性を段階的に付加できる。

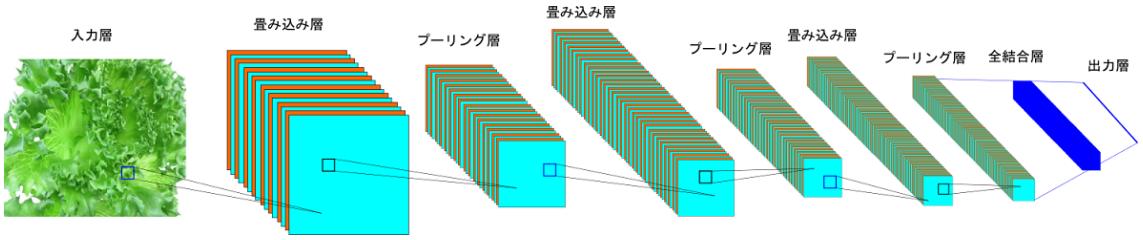
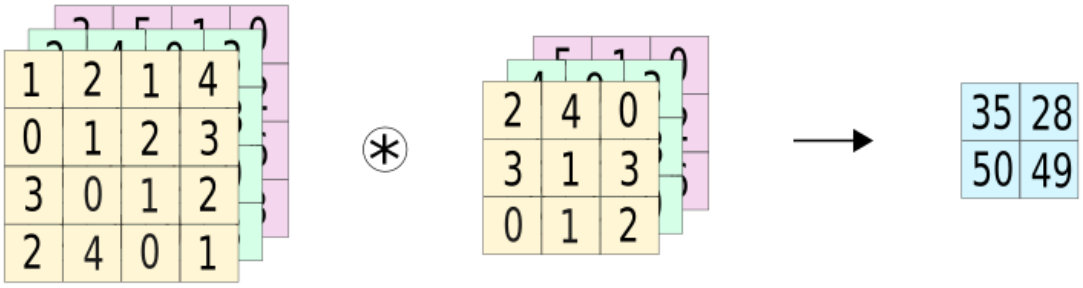


図 2.6　畳み込みニューラルネットワークの構造

2.4.2　構成要素

畳み込み層

畳み込み層では入力データとフィルタの畳み込み演算を行う。畳み込み層においてはフィルタのパラメータが全結合層における重みに相当し，フィルタのパラメータの更新によって最適化を行う。畳み込み層は，全結合層とは異なり，入力側の層と出力側の層の結合が局所的で同一チャネル内では重みが共有される。よって，入力データの局所的な特徴抽出とパラメータ数の大幅な削減が可能となる[25]。畳み込み層を図 2.7 に示す。



入力データ　　フィルタ　　出力データ

図 2.7　畳み込み層

プーリング層

プーリング層では，入力データを複数の領域に分割して，各領域を一つの値に集約することで，縦および横方向に圧縮するプーリングと呼ばれる処理が行われる。プーリングには，対象領域の最大値を出力とする Max プーリングと，対象領域の平均値を出力とする Average プーリングがある。プーリング層は，入力データの微小な位置変化に対して出力が不変という特徴を持つ[25]。プーリング層（Max プーリング）を図 2.8 に示す。

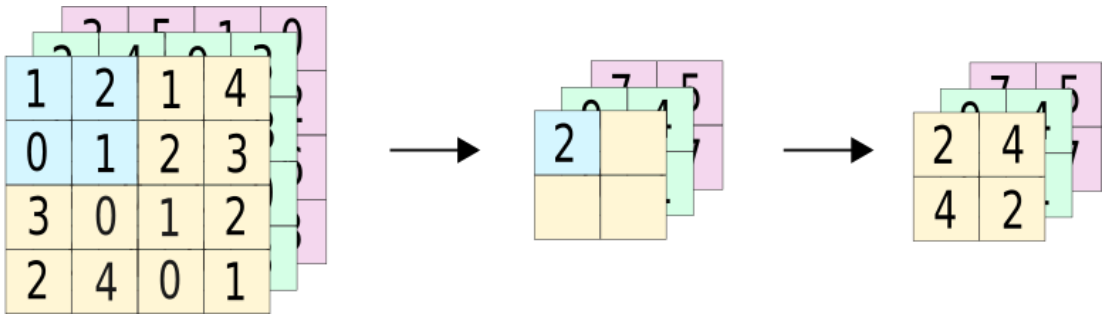


図 2.8　プーリング層（Max プーリング）

Batch Normalization

Batch Normalization [26]は入力 $\boldsymbol{x} = (x_1, \ldots, x_n)$を式(2.12)によって正規化した後，式(2.13)による変換を行い，$\boldsymbol{y} = (y_1, \cdots, y_n)$を出力する。

$$\hat{x}_i = \frac{x_i - \mu_i}{\sqrt{\boldsymbol{\sigma_i^2} + \boldsymbol{\epsilon}}}, i = 1, \ldots, n \tag{2.12}$$

$$y_i = \beta \hat{x}_i + \gamma, i - 1, \ldots, n \tag{2.13}$$

ここで，μ_i, σ_i^2 はそれぞれ x_i のバッチ単位での平均，分散であり，ϵ はゼロ除算防止のための微小な実数値である。また，β, γ はチャネルごとに異なる実数値であり，学習時に更新されるパラメータである。

推論時には，バッチサイズが 1 であることから，$\mu_i = x_i$ となり，$y_i = 0$ となるため，代わりに学習時における μ_i, σ_i^2 の移動平均を μ_i, σ_i^2 とする。

Batch Normalization を活性化関数の直前に適用することで，入力データの分布の偏りを削減し，より速く学習を進めることができる。ResNet[10]以降の CNN モデルでは，Batch Normalization が標準的に採用されている。

Dropout

Dropout[27]は学習時にニューロンを確率 α でランダムに除去する手法である。除去されたニューロンの出力は 0 となり，信号を伝達しなくなる。推論時にはニューロンの除去は行わない。

複数のモデルで個別に学習を行い，推論時にこれらのモデルの出力の平均値をとる手法をアンサンブル学習という。Dropout は毎回異なるモデルで学習を行い，推論時にこれらのモデルの平均をとっていると解釈できるため，一つのネットワークでアンサンブル学習を行っているとみなすことができる。

2.4.3　ネットワークモデル

VGGNet

VGGNet[8]は，畳み込み層とプーリング層で構成されたシンプルな CNN モデルである。このモデルは，入力層およびプーリング層の間に小規模な畳み込み層を積層しており，これによってパラメータ数を削減している。この構造は，後の CNN モデルでも取り入れられている。

GoogLeNet

GoogLeNet[9]は，図 2.9 に示す inception モジュールを用いた CNN モデルである。GoogLeNet の主な特徴は，複数の畳み込み層とプーリング層から構成される inception モジュールと呼ばれる小型のネットワークを，通常の畳み込み層と同様に積み重ねることで一つの大きな CNN を構成する点にある。inception モジュールの目的は，畳み込み層の重みをスパース化することで，CNN の表現能力を損なうことなく，パラメータ数を削減することである。

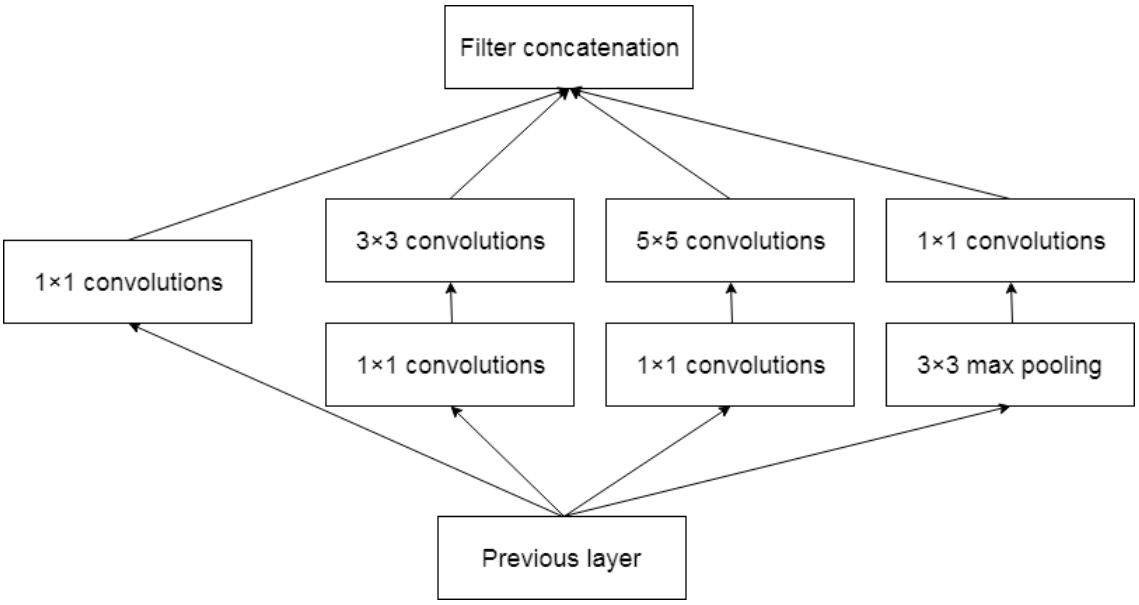


図 2.9　inception モジュール

ResNet

ResNet[10][11]（Residual Network）は，residual ブロックを用いた CNN モデルである。residual ブロックは，あるネットワークの出力 $F(\boldsymbol{x})$に入力 $\boldsymbol{x}$ を加算するショートカット接続を付加した構造となっている。このショートカットにより，勾配消失や勾配爆発が抑制され，層を深くしても学習性能の低下を抑制できる。また，residual ブロックはネットワーク $F(\boldsymbol{x})$とショートカット $\boldsymbol{x}$ の二つに分岐しており，これら二つのネットワークのアンサンブル学習を行っているとみなすことができる。図 2.10 は residual ブロックの一例であり，$F(\boldsymbol{x})$はフィルタサイズ 3 × 3，出力チャネル数 64 の畳み込み層を二つ重ねたネットワークである。このような residual ブロックを次のように表す。

$$\begin{bmatrix} 3\times 3, & 64 \\ 3\times 3, & 64 \end{bmatrix}$$

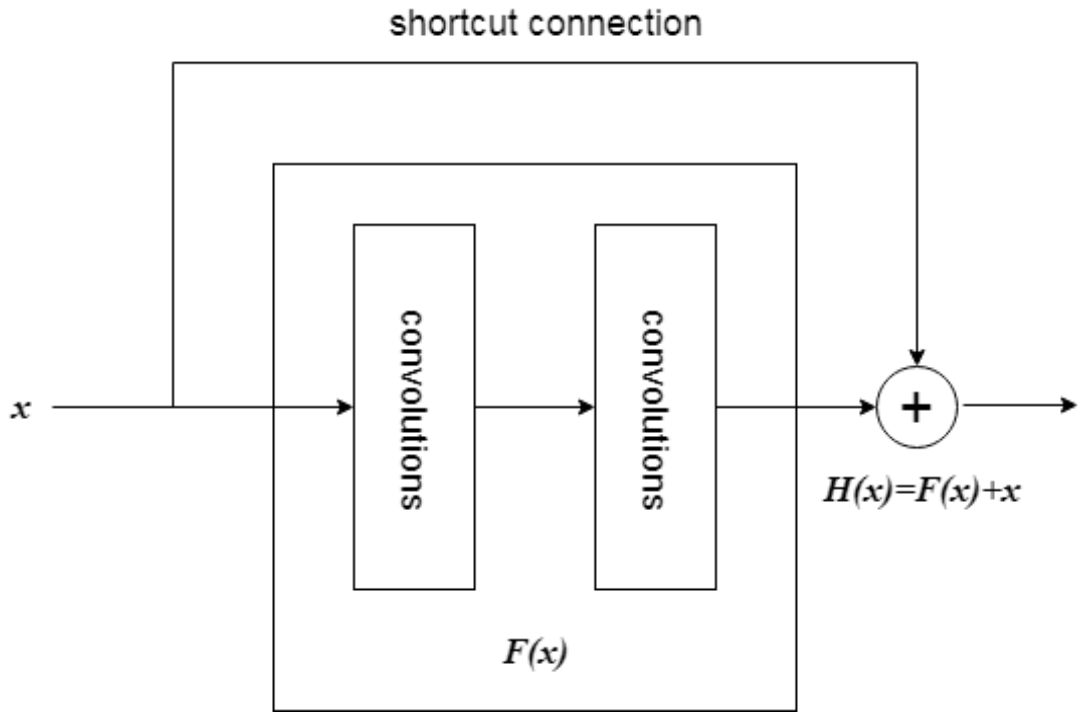


図 2.10　residual ブロックの一例

WideResNet

WideResNet [12]は, ResNet においてチャネル数を増加させた CNN モデルである。層を深くするより，チャネル数を増やす方が，精度および学習速度の点で優れた結果となる。

2.5 サポートベクターマシン

2.5.1 各種モデル

2 クラス分類の古典的かつ代表的手法の一つにサポートベクターマシン[16]（Support Vector Machine: SVM）がある。SVM は機械学習において，分類や回帰によく用いられる分類器である。SVM は，ベクトル空間 R^N を二つに分離する超平面を用いて，未知サンプル点 $\boldsymbol{x} \in R^N$ を二つの集合に分類する。分離超平面は識別関数 $f(\boldsymbol{x})$: $R^N \rightarrow R$ で表され，$f(\boldsymbol{x})$の値の正負により，入力 $\boldsymbol{x}$ を 2 クラスに分類する。与えられた学習サンプル点集合（訓練用データセット）に対して，識別関数 $f(\boldsymbol{x})$を求めることを SVM の学習という。本節では，まず，SVM の基盤である線形ハードマージン SVM について述べる。つぎに，線形ハードマージン SVM の制約を緩和することで誤分類を許容し，学習サンプル点の分布に重なりがある場合でも学習を可能とする線形ソフトマージン SVM について述べる。最後に，カーネル関数を用いて非線形識別を可能とする非線形 SVM について述べる。

なお，SVM への入力の生成には，特徴抽出器が用いられる。代表的な特徴抽出器の一つに Histogram of Oriented Gradients[17]（HOG）がある。HOG は局所領域でエッジ方向ごとにその強度をヒストグラム化した特徴量で，コンピュータビジョンや画像処理において，物体検出によく用いられる。

2.5.2　線形ハードマージンサポートベクターマシン

あるベクトル空間 R^N に，P 個の学習サンプル点 $\boldsymbol{x}^p, p = 1, ..., P$ とそれに対応する 2 値ラベル $y^p \in \{1, -1\}, p = 1, ..., P$ が分布していることを考える。この空間上に，正サンプル点集合 $X^+ = \{\boldsymbol{x}^p | y^p = 1, p = 1, ..., P\}$ と負サンプル点集合 $X^- = \{\boldsymbol{x}^p | y^p = 1, p = 1, ..., P\}$ をもれなく分離する超平面 $f(\boldsymbol{x}) = 0$ が存在すると仮定する。$f(\boldsymbol{x})$は線形識別関数と呼ばれ，一般に，

$$f(\boldsymbol{x}) = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + b \tag{2.14}$$

と表される。一般に，このような超平面は無数に存在する。そこで，図 2.11 に示すような，超平面からもっとも近い正サンプル点 $\boldsymbol{x}^+$（図中の丸：○）と負サンプル点 $\boldsymbol{x}^-$（図中の四角：□）までの距離（最小幾何マージン）を考える。無数にある超平面の中で，この最小幾何マージンが大きいほど，学習サンプル点から少し外れた未知サンプル点が別ラベルの集合に分類される可能性が低くなる。したがって，最小幾何マージンを最大化する超平面を求めることで，未知サンプル点に対する汎化能力の高いパターン識別器を構築できる。

最小幾何マージンを最大化する超平面を求める問題を定式化することを考える。まず，超平面 $f(\boldsymbol{x}) = 0$ のパラメータ $\boldsymbol{w}, b$ が唯一となるように，すべての学習サンプル点が，超平面によってもれなく分離され，かつ超平面からもっとも近い正サンプル点 X^+と負サンプル点 X^-に対して $f(\boldsymbol{x}^+) = 1$，$f(\boldsymbol{x}^-) = -1$ となる条件

$$y^p f(\boldsymbol{x}^p) - 1 \geq 0, p = 1, \dots, P \tag{2.15}$$

を定める。超平面 $f(\boldsymbol{x}) = 0$ と学習サンプル点 $\boldsymbol{x}^p$ とのユークリッド距離は$|f(\boldsymbol{x}^p)| / \|\boldsymbol{w}\|$ で与えられるので，最小幾何マージンは $1 / \|\boldsymbol{w}\|$ となる。したがって，最小幾何マージンを最大化する超平面を求める問題は，

$$\underset{\boldsymbol{w},\mathrm{b}}{\text{minimize}} \frac{1}{2}\boldsymbol{w}^{\mathrm{T}}\boldsymbol{w} \tag{2.16a}$$

$$\text{subject to } y^p f(\boldsymbol{x}^p) - 1 \geq 0, p = 1, \dots, P \tag{2.16b}$$

と定式化できる。ここで，最適化問題(2.16)の最適解において，制約条件(2.16b)の等号が成立する $\boldsymbol{x}^p$ をサポートベクトルと呼ぶ。最小幾何マージンを最大化する超平面は，サポートベクトルによって定まるので，サポートベクトル以外の学習サンプル点を除いたとしても，最適解は変化しない。

2.5.3　線形ソフトマージンサポートベクターマシン

学習サンプル点を超平面によってもれなく分離する線形ハードマージン SVM に対して，線形ソフトマージン SVM は，学習サンプル点の分離もれにペナルティを与えて誤分類を許容する。たとえば，図 2.12 に示すような，超平面によってもれなく分離することができない学習サンプル点の分布を考える。ここで，正サンプル点 $\boldsymbol{x}^+$であれば超平面 $f(\boldsymbol{x}^+) = 1$ から，負サンプル点 $\boldsymbol{x}^-$であれば超平面 $f(\boldsymbol{x}^-) = -1$ から，超平面 $f(\boldsymbol{x}) = 0$ の方向へ，それぞれスラック変数 $\zeta^p, p = 1, ..., P$ の距離だけ，学習サンプル点のマージン内侵入を許容する条件

$$y^p f(\boldsymbol{x}^p) - 1 + \zeta^p \geq 0, p = 1, \dots, P \tag{2.17}$$

を定める。また，誤分類に対してペナルティを与えるため，スラック変数 ζ^p の総和にコストパラメータ C を乗じたペナルティ項を目的関数(2.16a)に加える。このようにして，スラック変数とペナルティ項の導入により誤分類の程度を調整できる線形ソフトマージン SVM の学習問題は

$$\underset{\boldsymbol{w},b,\boldsymbol{\zeta}}{\text{minimize}} \frac{1}{2}\boldsymbol{w}^{\mathrm{T}}\boldsymbol{w} + C\sum_{p=1}^{P}\zeta^p \tag{2.18a}$$

$$\text{subject to } y^p f(\boldsymbol{x}^p) - 1 + \zeta^p \geq 0, p = 1, \dots, P \tag{2.18b}$$

$$\zeta^p \geq 0, p = 1, \dots, P \tag{2.18c}$$

と定式化される。この問題で $C \to \infty$とすると，スラック変数の最適解は $\zeta^* = 0$ となるので，線形ハードマージン SVM の学習問題(2.16)と一致する。線形ソフトマージン SVM でも，最適化問題 (2.18)の最適解において，制約条件(2.18b)の等号が成り立つ $\boldsymbol{x}^p$ をサポートベクトルと呼ぶ。

最適化問題(2.18)の双対問題を考える。まず，非負のラグランジュ乗数 $\lambda^p, \mu^p > 0, p = 1, ..., P$ を用いたラグランジュ関数

$$L(\boldsymbol{w}, b, \boldsymbol{\zeta}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = \frac{1}{2}\boldsymbol{w}^{\mathrm{T}}\boldsymbol{w} + C\sum_{p=1}^{P}\zeta^p - \sum_{p=1}^{P}\lambda^p\ (y^p f(x^p) - 1 + \zeta^p) - \sum_{p=1}^{P}\mu^p\zeta^p \tag{2.19}$$

を導入する。ラグランジュ関数(2.19)を用いると，最適化問題(2.18)の最適解は，双対問題

$$\underset{\boldsymbol{\lambda},\mu}{\text{maximize}}\ \min_{\boldsymbol{w},b,\boldsymbol{\zeta}} L(\boldsymbol{w}, b, \boldsymbol{\zeta}, \boldsymbol{\lambda}, \boldsymbol{\mu}) \tag{2.20a}$$

$$\text{subject to } \lambda^p, \mu^p \geq 0, p = 1, \dots, P \tag{2.20b}$$

の最適解と一致する。式(2.20a)の最小値関数ついて考える。ある $\boldsymbol{\lambda}, \boldsymbol{\mu}$ が与えられた下で，$L(\boldsymbol{w}, b, \boldsymbol{\zeta},, \boldsymbol{\lambda}, \boldsymbol{\mu})$が最小となるのは，$\boldsymbol{w}$ に関しては凸関数であり，$b, \boldsymbol{\zeta}$ に関しては，線形関数であることから，

$$\nabla_w L(\boldsymbol{w}, b, \boldsymbol{\zeta}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = \boldsymbol{w} - \sum_{p=1}^{P} \lambda^p y^p \boldsymbol{x}^p = \boldsymbol{0} \tag{2.21a}$$

$$\frac{\partial L(\boldsymbol{w}, b, \boldsymbol{\zeta}, \boldsymbol{\lambda}, \boldsymbol{\mu})}{\partial b} = -\sum_{p=1}^{P} \lambda^p y^p = 0 \tag{2.21b}$$

$$\nabla_\zeta L(\boldsymbol{w}, b, \boldsymbol{\zeta}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = C\boldsymbol{1} - \boldsymbol{\lambda} - \boldsymbol{\mu} = \boldsymbol{0} \tag{2.21c}$$

を満たすときである。式(2.21)を用いて，最適化問題(2.20)の目的関数(2.20a)の $\boldsymbol{w}, b, \boldsymbol{\zeta}, \boldsymbol{\mu}$ を消去すると，最適化問題(2.18)の双対問題は,

$$\underset{\boldsymbol{w},b,\boldsymbol{\zeta}}{\text{maximize}} -\frac{1}{2}\boldsymbol{\lambda}^{\mathrm{T}} H \boldsymbol{\lambda} + \boldsymbol{1}^{\mathrm{T}} \boldsymbol{\lambda} \tag{2.22a}$$

$$\text{subject to } \boldsymbol{y}^{\mathrm{T}} \boldsymbol{\lambda} = 0 \tag{2.22b}$$

$$0 \leq \lambda^p \leq \mathrm{C}, p = 1, \ldots, P \tag{2.22c}$$

$$\text{where } h_{pq} = y^p y^q \boldsymbol{x}^{p\mathrm{T}} \boldsymbol{x}^q, p = 1, \ldots, P, q = 1, \ldots, P \tag{2.22d}$$

と変形できる。この問題は，凹最適化問題であり，局所的最適解は唯一である。また，最適化問題(2.22) の最適解($\boldsymbol{w}^*, b^*, \boldsymbol{\zeta}^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*$)は，相補性条件

$$\lambda^{p*}(y^p f^*(\boldsymbol{x}^p) - 1 + \zeta^p) = 0, p = 1, \ldots, P \tag{2.23a}$$

$$\mu^{p*}\zeta^{p*} = 0, p = 1, \ldots, P \tag{2.23b}$$

を満たす。ただし，$f^*(\boldsymbol{x}^p) = \boldsymbol{w}^{*\mathrm{T}}\boldsymbol{x}^p + b^*$である。式 (2.23a) から，サポートベクトル制約条件（式(2.18b) の等号）を満たす $\boldsymbol{x}^s$ に対応する λ^{s*} は 0 以上となる。また，式 (2.21c), (2.23b)より，マージン内に侵入するサポートベクトル，すなわち $\zeta^{s*} > 0$ となる $\boldsymbol{x}^s$ については $\lambda^{s*} = C$ となる。最適化問題(2.22)の最適解 $\boldsymbol{\lambda}^*$を用いた線形識別関数(2.14)を考える。式(2.21)を用いて $\boldsymbol{w}$ を消去すると，線形識別関数は,

$$f(\boldsymbol{x}) = \sum_{s \in S} \lambda^{s*} y^{s*} \boldsymbol{x}^{s\mathrm{T}} \boldsymbol{x} + b^* \tag{2.24a}$$

$$b^* = \frac{1}{|S'|} \sum_{s' \in S'} \left(y^{s'} - \sum_{s \in S} \lambda^{s*} y^s x^{s\mathrm{T}} x^{s'} \right) \tag{2.24b}$$

で与えられる。ここで，S はサポートベクトルの番号集合，S'は $\lambda^{s*} = C$ に対応するサポートベクトルを除いたサポートベクトルの番号集合，$|S'|$は集合 S'の要素数である。b^* の算出式については，数値計算の誤差の影響を減らすために，S' の要素すべてについて平均をとる形になっている。

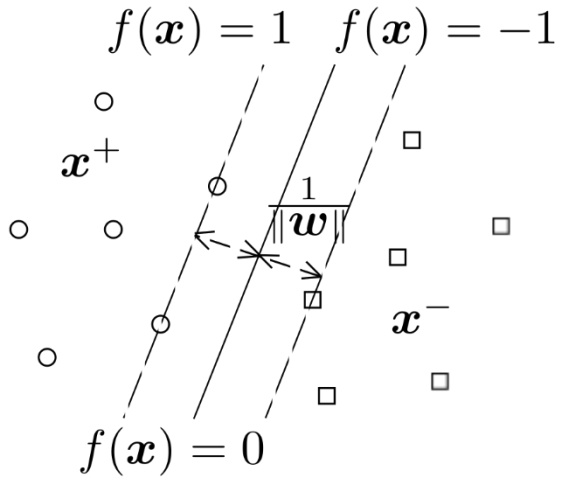


図 2.11　線形ハードマージン SVM の超平面

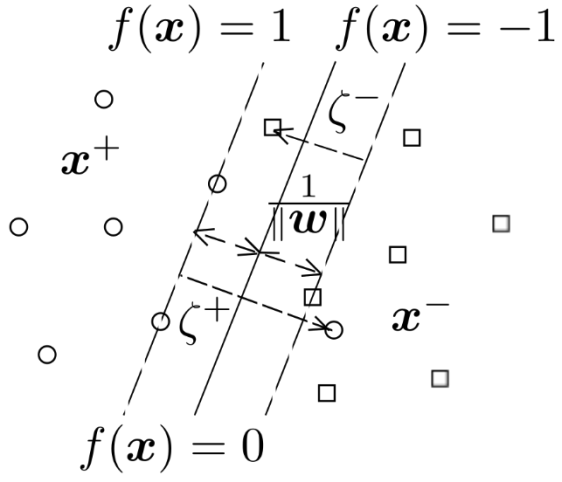


図 2.12　線形ソフトマージン SVM の超平面

2.5.4　非線形サポートベクターマシン

線形ソフトマージン SVM は，誤分類を許容することで，学習サンプル点の分布に重なりのある場合でも超平面の学習が可能である。しかし，学習サンプル点の分布によっては，超平面による分離が，かならずしも有効であるとは限らない。そこで，ベクトルの非線形写像

$$\boldsymbol{x} \to \Phi(\boldsymbol{x}) \tag{2.25}$$

によって，学習サンプル点を超平面による分離が有効な高次元空間に写像した上で，超平面 $f(\boldsymbol{\Phi}(\boldsymbol{x})) = 0$ を学習することを考える。写像空間上で学習した超平面は，元の空間上では超曲面となり，未知サンプル点の非線形識別が可能となる。線形ソフトマージン SVM の学習問題(2.22)と識別関数(2.24)に注目すると，この変換によって置き換わるのは，$\boldsymbol{x}^{p\mathrm{T}}\boldsymbol{x}^{q} \to \boldsymbol{\Phi}(\boldsymbol{x}^{p})^{\mathrm{T}}\boldsymbol{\Phi}(\boldsymbol{x}^{q})$の箇所のみである。したがって，$\boldsymbol{\Phi}(\boldsymbol{x}^{p})^{\mathrm{T}}\boldsymbol{\Phi}(\boldsymbol{x}^{q})$さえ知ることができれば，$\boldsymbol{\Phi}(\boldsymbol{x})$を直接計算する必要はない。そこで，ある写像空間上での二つのサンプル点の内積を表すカーネル関数

$$k(\boldsymbol{x}^{p}, \boldsymbol{x}^{q}) = \Phi(\boldsymbol{x}^{p})^{\mathrm{T}}\Phi(\boldsymbol{x}^{q}) \tag{2.26}$$

を導入する。このカーネル関数 $k(\boldsymbol{x}^{q}, \boldsymbol{x}^{p})$を用いると，非線形ソフトマージン SVM の学習問題は,

$$\underset{\boldsymbol{w},b,\boldsymbol{\zeta}}{\text{maximize}} -\frac{1}{2}\boldsymbol{\lambda}^{\mathrm{T}}H\boldsymbol{\lambda} + \mathbf{1}^{\mathrm{T}}\boldsymbol{\lambda} \tag{2.27a}$$

$$\text{subject to } \boldsymbol{y}^{\mathrm{T}}\boldsymbol{\lambda} = 0 \tag{2.27b}$$

$$0 \le \lambda^{p} \le C, p = 1, \dots, P \tag{2.27c}$$

$$\text{where } h_{pq} = y^{p}y^{q}k(\boldsymbol{x}^{p}, \boldsymbol{x}^{q}), p = 1, \dots, P, q = 1, \dots, P \tag{2.27d}$$

と定式化できる。学習サンプル点 $\boldsymbol{x}^{1}, \dots, \boldsymbol{x}^{P}$ に対するカーネル関数 $k(\boldsymbol{x}^{q}, \boldsymbol{x}^{p})$のグラム行列

$$K(\boldsymbol{x}) = \begin{bmatrix} k(\boldsymbol{x}^{1}, \boldsymbol{x}^{1}) & \cdots & k(\boldsymbol{x}^{1}, \boldsymbol{x}^{P}) \\ \vdots & \ddots & \vdots \\ k(\boldsymbol{x}^{P}, \boldsymbol{x}^{1}) & \cdots & k(\boldsymbol{x}^{P}, \boldsymbol{x}^{P}) \end{bmatrix} \tag{2.28}$$

が半正定値行列であれば，最適化問題(2.27)も凹最適化問題になることが知られている。最適化問題(2.27)の最適解 $\boldsymbol{\lambda}^{*}$を用いると，非線形ソフトマージン SVM の識別関数は,

$$f(\boldsymbol{x}) = \sum_{s \in S} \lambda^{s*} y^{s*} k(\boldsymbol{x}^{s}, \boldsymbol{x}) + b^{*} \tag{2.29a}$$

$$b^{*} = \frac{1}{|S'|} \sum_{s' \in S'} \left(y^{s'} - \sum_{s \in S} \lambda^{s*} y^{s} k(x^{s}, x^{s'}) \right) \tag{2.29b}$$

と表すことができる。

第 3 章
提案手法

3.1　概要

本研究は，深層学習の一種である CNN による画像診断技術を，人工光型植物工場で栽培されるレタスのチップバーン発生・未発生の 2 クラス分類へ導入することを目的とする。まず，深層学習用の多量のカラー画像データを取得するために，チップバーンが発生する栽培条件を特定し，実際にチップバーンレタスを栽培する。画像診断用の CNN については，VGGNet[8]，GoogLeNet[9]，ResNet[10][11]，WideResNet[12]の各モデル，および Momentum SGD，Adam[13]，Adamax[13]，Nadam[14][15]の各最適化手法を試行し，チップバーンの検出性能を比較することで，チップバーン識別率の向上手法を検討する。

3.2　深層学習用画像データの取得

3.2.1　要求事項

深層学習に用いる訓練用データセットは，チップバーンが発生していない「正常レタス」と，チップバーンが発生した「チップバーンレタス」のカラー画像の集合となる。これらのカラー画像を大量に取得するために，チップバーンが発生する栽培条件を特定し，その条件下で実際にレタスを栽培して，意図的に正常レタスとチップバーンレタスを作出する。このようにして取得したデータラベルの精度は，極めて高いと考えられる。

3.2.2　栽培のための材料および方法

フリルレタスは，多くの人工光型植物工場で栽培される作物の一つであり，経済的効果は大きい。また，チップバーンが発生しやすい品種として知られている。そこで，供試品種にはフリルレタスの一品種であるフリルアイス（雪印種苗株式会社）を採用する。

人工光型植物工場として，ショーケース型とハウス型の二つのタイプの植物工場を採用する。どちらのタイプの植物工場ともに，千葉大学柏の葉キャンパス環境健康フィールド科学センターに設置されている。正常レタスと B 型チップバーンレタスに関しては，ショーケース型植物工場（株式会社ハンモ製 HM-PF-DC-AL01）で栽培を行う。ショーケース型植物工場は，室内設置のため室外の気候変動の影響を受けず，小型で栽培条件の制御が容易である。ショーケース型植物工場の外観を図 3.1 に，スペックを表 3.1 に示す。一方，A 型チップバーンレタスに関しては，よりチップバーンが発生しやすい環境を創出するために，温度や CO_2 濃度を容易に制御できるハウス型植物工場で栽培を行う。温度と CO_2 濃度は，チップバーンの発生を促進する栽培条件を設定するためには，重要な要因である。ハウス型植物工場の外観を図 3.2 に，スペックを表 3.2 に示す。

栽培方式は固形培地を用いない水耕栽培とする。照明は，ショーケース型植物工場では白色 LED 光源，ハウス型植物工場では RGB LED 光源を用いる。どちらの照明とも光合成に適する波長を含み，発熱が小さく，人工光型植物工場に適する。照明熱は，ショーケース型植物工場では換気ファンによって，栽培装置内の空気を排出することで熱を除去し，一方，ハウス型植物工場ではエアコンによって，温度調節を行う。さらに，ハウス型植物工場では，レタスの生長を促進させるために CO_2 を施用する。

表 3.1　ショーケース型植物工場のスペック

Size	W 1,330mm × D 400mm × H 1,570mm
number of cultural shelves	3 (1 for seed bed, 2 for cultivation)
Controller	lighting cycle, fan control
Lighting	white LED (18W) ×6
Ventilator	8 fans
nutrient solution dispenser	circulator, tank, water level sensor

表 3.2　ハウス型植物工場のスペック

Size	W 2,950mm × D 1,950mm × H 2,390mm
number of cultural shelves	4
Controller	lighting cycle, air conditioner, CO_2 level
Lighting	RGB LED (14.5W) ×10
nutrient solution dispenser	deep float technique, aeration

図 3.1　ショーケース型植物工場

図 3.2　ハウス型植物工場

3.2.3　レタスの栽培条件

チップバーンは，（1）温度，（2）湿度，（3）光強度，（4）日長，（5）培養液の肥料濃度，等の要因によって発生することが知られている。また，チップバーンは相対的に，（1）温度においては高温，（2）湿度においては乾燥，（3）光強度においては強い，（4）日長においては長い，（5）培養液の肥料濃度においては高濃度，となったときに発生しやすい。これらの諸条件を考慮し，栽培環境を設定する。

栽培に用いる培養液は園試処方とする。園試処方は，水耕栽培における最も一般的な処方の一つであり，葉菜類のみならず，ほとんどの作物の栽培が可能な処方として知られている。

ショーケース型植物工場（正常および B 型チップバーンレタス作出）では，培養液は循環させ，24h/day の連続運転とする。流量は各栽培棚に対し，10L/min に設定する。換気ファンは 18h/day 稼働させる。その他の項目の設定値を表 3.3 に示す。

ハウス型植物工場（A 型チップバーンレタス作出）では，培養液を循環させず，たん液型水耕栽培において，エアレーションを 24h/day 連続で行う。また，植物の光合成を促進させ，成長を促す目的で，CO_2 を施用する。植物工場内に液化炭酸ガスボンベを設置し，CO_2 センサによって，植物工場内の CO_2 濃度が 2,000ppm（通常の大気の CO_2 濃度は約 400ppm）に保たれるように設定する。その他の項目の設定値を表 3.3 に示す。

なお，通常，水耕栽培では培養液濃度を EC（electrical conductivity [dS/m]，電気伝導度）によってコントロールする。EC は，イオン濃度が高いほど高い数値となる。したがって，数値が高いほど培養液の肥料濃度が高くなる。また，光強度については PPFD（photosynthetic photon flux density [μmol/m²s]，光合成有効光量子束密度）により測定する。

表 3.3　栽培条件

	normal lettuce	A/B tipburn lettuce
cultivation system	showcase-type	house-type
temperature [℃]	18	24
humidity [%]	65～75	55～70
day length [h]	L15-D9 (15h)	L20-D4 (20h)
light intensity [μmol/m²s]	PPFD200	PPFD200～250
nutrient concentration [dS/m]	EC1.5	EC2.4

3.2.4　レタスの栽培手順

レタスの栽培手順は，以下の通りとする。(1) 育苗用専用容器に水道水を含ませたウレタンを設置し，播種を行う。(2) 発芽 3 日後，発根を確認した個体を，順次，栽培棚に移植する。(3) 2 週間育苗した後，生育棚に定植し，栽培を継続する。

3.2.5　画像データ取得

画像データの取得は，播種後 35 日目から，毎日連続的に実施する。フリルレタスを一個体ずつ取り出し，目視にて全数検査し，A 型チップバーン，A 型チップバーン予兆および B 型チップバーン発生の有無を確認した後，白色背景，白色照明（蛍光灯）のもとで上面からデジタルカメラ（F 値 4.0）を用いてカラー画像の撮影を行う。カラー画像データの例を図 3.3～3.6 に示す。正常レタスの作出において，すべての個体にチップバーンの発生は認められず，A 型および B 型チップバーンレタスの作出においては，すべての個体に各型のチップバーンの発生が認められている。

3.3　画像診断用畳み込みニューラルネットワーク

画像診断用の CNN については，VGGNet[8]，GoogLeNet[9]，ResNet[10][11]，WideResNet[12] の各モデル，Momentum SGD，Adam[13]，Adamax[13]，Nadam[14][15]の各最適化手法を試行し，チップバーンの検出性能を比較することで，チップバーンの識別率を向上させる手法を検討する。

図 3.3　正常レタス

図 3.4　A 型チップバーンレタス

図 3.5　B 型チップバーンレタス

図 3.6　A 型チップバーン予兆レタス

第 4 章
画像診断実験

4.1　実験目的

人工光型植物工場生産レタスのカラー画像データを用いて，CNN を用いた深層学習によるチップバーンの画像診断実験を実施する。CNN のコンセプトは，脳の視覚野（visual cortex）についての知見を数理モデル化したものであり，2 次元画像の処理に有効であるため，本画像診断実験においては適した手法と判断できる。画像診断用の CNN については，VGGNet[8]，GoogLeNet[9]，ResNet[10][11]，WideResNet[12]の各モデル，Momentum SGD，Adam[13]，Adamax[13]，Nadam[14][15]の各最適化手法を試行し，チップバーンの検出性能を比較する。また，CNN を用いた深層学習の有効性を明確に示すために，2 クラス分類の古典的かつ代表的手法の一つである SVM[16][17]を比較対象に加える。実験対象とする CNN モデルおよび最適化手法を，表 4.1 に示す。

表 4.1　CNN モデルと最適化手法

CNN model	VGGNet17	GoogLeNet22	ResNet50	WideResNet10
Optimizer	MomentumSGD	Adam	Adamax	Nadam

4.2 実験条件

レタスのカラー画像は上面から白色背景，白色照明（蛍光灯）のもとで，F 値 4.0 で撮影し，画像サイズは約 100KB とする。各型のチップバーンレタスの画像データ数を表 4.2 に示す。栽培するレタスの個体数は各型とも 24 個とする。CNN の学習において，バッチサイズ 50, エポック（Epoch）数 50 とし, 各画像データ中の 90%を訓練（train）用，残り 10%を検証（test）用とする。各型のチップバーン発生・未発生の 2 クラス分類を学習するため，学習時に用いる訓練用画像データ数は，各クラスでほぼ同数となるように調整する。CNN の学習に用いる計算機環境を表 4.3 に示す。

表 4.2　レタスの画像データ数

正常	2,245
A 型チップバーン	2,314
B 型チップバーン	2,132
A 型チップバーン予兆	1,034

表 4.3　計算機環境

OS	Ubuntu 14.04
CPU (クロック周波数)	Intel Core i7 6700K 4.0GHz
主メモリ	32.0 GB
GPU	TITAN X 12GB
プログラミング言語	Python3.6.6
ライブラリ	chainer5.0.0

画像診断用の CNN については，VGGNet[8]，GoogLeNet[9]，ResNet[10][11]，WideResNet[12] の各モデル，Momentum SGD，Adam[13]，Adamax[13]，Nadam[14][15]の各最適化手法を試行する。実験は，各 CNN モデルおよび各最適化手法を用いて，（1）正常と A 型チップバーン，（2）正常と B 型チップバーン，（3）正常と A 型チップバーン予兆，（4）正常とチップバーン（A 型または B 型）の各 2 クラス分類について行う。また，CNN の有効性を確認するために，最適化手法に Adam[13] を用いる場合の各 CNN モデルと SVM[16][17]の比較を行う。実験に用いる各 CNN モデルのネットワーク構造を表 4.4～4.7 に，各 CNN モデルに共通なパラメータを表 4.8 に，各最適化手法のパラメータを表 4.9 にそれぞれ示す。

表 4.4　VGGNet17 のネットワーク構造

Layer	Filter size	Stride	Output size
input			(224, 224, 3)
convolution × 2	(3, 3)	(1, 1)	(224, 224, 64)
max pooling	(2, 2)	(2, 2)	(112, 112, 64)
convolution × 2	(3, 3)	(1, 1)	(112, 112, 128)
max pooling	(2, 2)	(2, 2)	(56, 56, 256)
convolution × 3	(3, 3)	(1, 1)	(56, 56, 512)
max pooling	(2, 2)	(2, 2)	(28, 28, 512)
convolution × 3	(3, 3)	(1, 1)	(28, 28, 512)
max pooling	(2, 2)	(2, 2)	(14, 14, 512)
convolution × 3	(3, 3)	(1, 1)	(14, 14, 512)
max pooling	(2, 2)	(2, 2)	(7, 7, 512)
full connected			4096
full connected			4096
full connected			1000
full connected			2

表 4.5　GoogLeNet22 のネットワーク構造

Layer	Filter size	Stride	Output size
input			(244, 244, 3)
convolution	(7, 7)	(2, 2)	(112, 112, 64)
max pooling	(3, 3)	(2, 2)	(56, 56, 64)
convolution	(3, 3)	(1, 1)	(56, 56, 192)
max pooling	(3, 3)	(2, 2)	(28, 28, 192)
inception × 2			(28, 28, 480)
max pooling	(3, 3)	(2, 2)	(14, 14, 480)
inception × 5			(14, 14, 832)
max pooling	(3, 3)	(2, 2)	(7, 7, 832)
inception × 2			(7, 7, 1024)
average pooling	(7, 7)	(1, 1)	(1, 1, 1024)
full connected			256
full connected			2

表 4.6　ResNet50 のネットワーク構造

Layer	Filter size	Stride	Output size
input			(224, 224, 3)
convolution	(3, 3)	(1, 1)	(224, 224, 17)
residual	$\begin{bmatrix} 1\times1, & 17 \\ 3\times3, & 4 \\ 1\times1, & 17 \end{bmatrix}\times3$	(1, 1)	(224, 224, 17)
residual	$\begin{bmatrix} 1\times1, & 17 \\ 3\times3, & 8 \\ 1\times1, & 32 \end{bmatrix}\times4$	(2, 2)	(112, 112, 32)
residual	$\begin{bmatrix} 1\times1, & 32 \\ 3\times3, & 17 \\ 1\times1, & 64 \end{bmatrix}\times6$	(2, 2)	(56, 56, 64)
residual	$\begin{bmatrix} 1\times1, & 64 \\ 3\times3, & 32 \\ 1\times1, & 128 \end{bmatrix}\times3$	(2, 2)	(28, 28, 128)
average pooling	(28, 28)	(1, 1)	(1, 1, 128)
full connected			2

表 4.7　WideResNet10 のネットワーク構造

Layer	Filter size	Stride	Output size
input			(224, 224, 3)
convolution	(3, 3)	(1, 1)	(224, 224, 17)
residual	$\begin{bmatrix} 3\times3, & 17 \\ 3\times3, & 48 \end{bmatrix}$	(1, 1)	(224, 224, 48)
residual	$\begin{bmatrix} 3\times3, & 17 \\ 3\times3, & 96 \end{bmatrix}$	(2, 2)	(112, 112, 96)
residual	$\begin{bmatrix} 3\times3, & 96 \\ 3\times3, & 192 \end{bmatrix}$	(2, 2)	(56, 56, 192)
residual	$\begin{bmatrix} 3\times3, & 192 \\ 3\times3, & 384 \end{bmatrix}$	(2, 2)	(28, 28, 384)
average pooling	(28, 28)	(1, 1)	(1, 1, 384)
full connected			2

表 4.8　CNN モデルのパラメータ

中間層活性化関数	ReLU 関数
出力層活性化関数	softmax 関数
損失関数	クロスエントロピー関数
重み初期値	He の初期値[28]

表 4.9　最適化手法のパラメータ

Momentum SGD	η	10^{-2}
	μ	0.9
Adam	α	10^{-2}
	β_1	0.9
	β_2	0.999
	ε	10^{-8}
Adamax	α	0.002
	β_1	0.9
	β_2	0.999
	ε	10^{-8}
Nadam	α	0.002
	β_0	0.99
	γ	0.999
	ε	10^{-8}

4.3　実験結果

（1）正常と A 型チップバーン，（2）正常と B 型チップバーン，（3）正常と A 型チップバーン予兆，（4）正常とチップバーン（A 型または B 型）の各 2 クラス分類の実験結果（学習曲線）を，図 4.1～4.8 に示す。同図には，各 CNN モデルにおいて上位二つの最適化手法の結果を示す。*x* 軸はエポック数，*y* 軸は訓練用データセットに対する正解率（train accuracy）または検証用データセットに対する正解率（test accuracy）である。

CNN モデルに GoogLeNet[8], 最適化手法に Adam[13]を用いた場合に誤認識された検証用データ（画像）の例を，図 4.9～4.12 に示す。同図中の FN （false-negative, 見逃し）は「チップバーンを正常と誤認識」， FP （false-positive，空振り）は「正常をチップバーンと誤認識」を，それぞれ表す。

また，CNN の有効性を確認するために，最適化手法に Adam[13]を用いる場合の各 CNN モデルと SVM[16][17]の比較結果を表 4.10 に示す。同表中の数値は検証用データセットに対する正解率（test accuracy）である。

表 4.10　各 CNN モデルと SVM の比較（test accuracy）

	WideResNet	ResNet	GoogLeNet	VGGNet	SVM
正常-A 型	1.000	0.993	0.986	0.977	0.734
正常-A 型予兆	0.990	0.981	0.977	0.979	0.536
正常-B 型	0.961	0.929	0.914	0.849	0.551

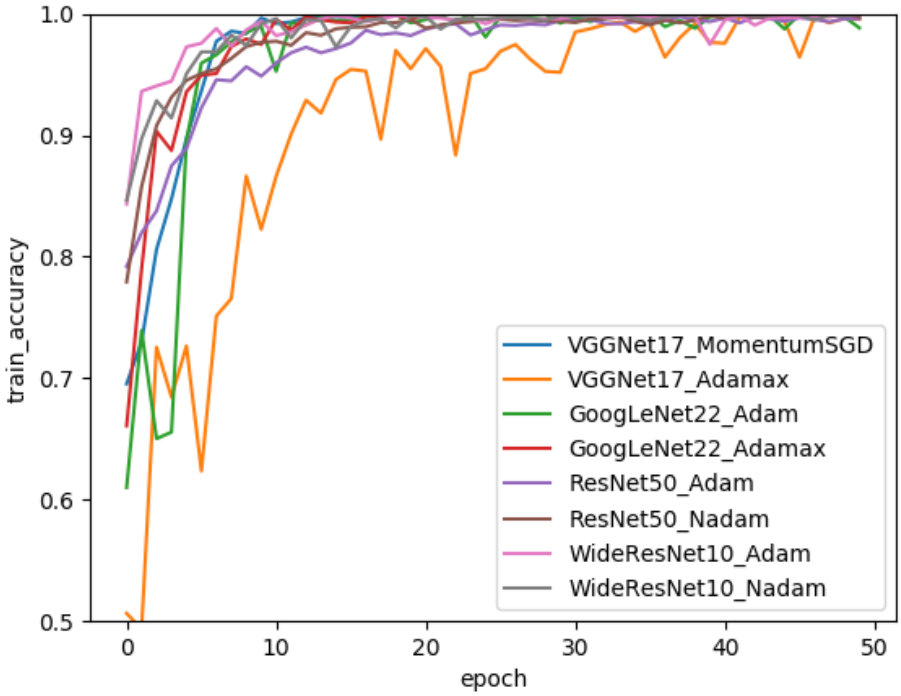


図 4.1　train accuracy（A 型チップバーン）

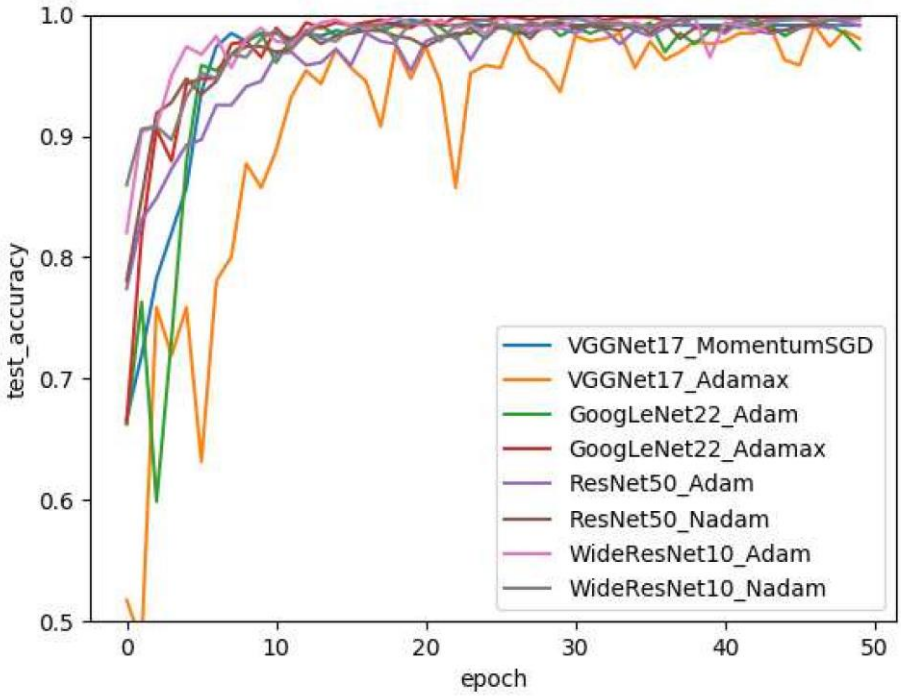


図 4.2　test accuracy（A 型チップバーン）

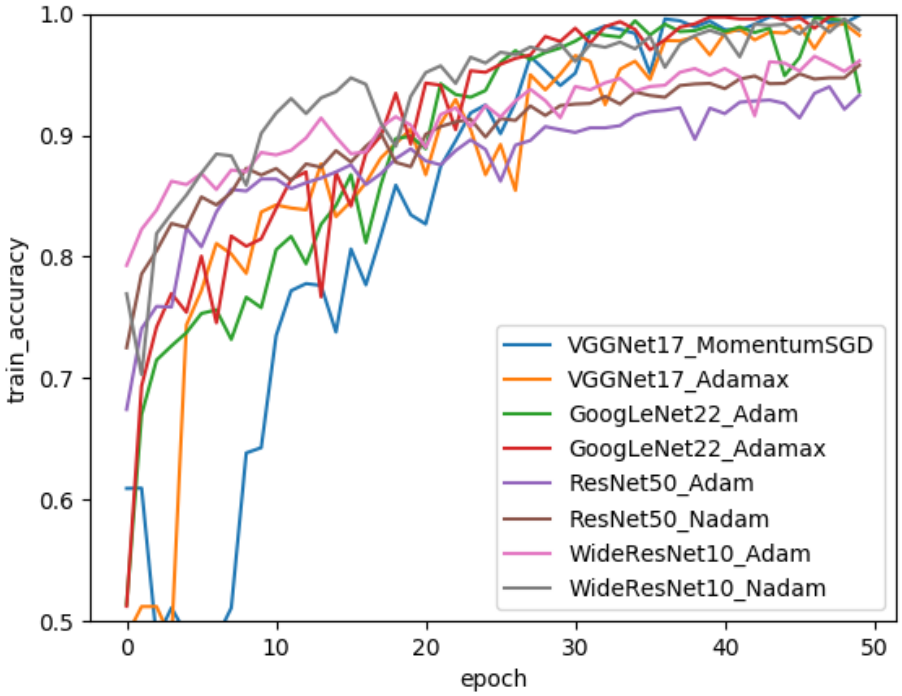


図 4.3　train accuracy（B 型チップバーン）

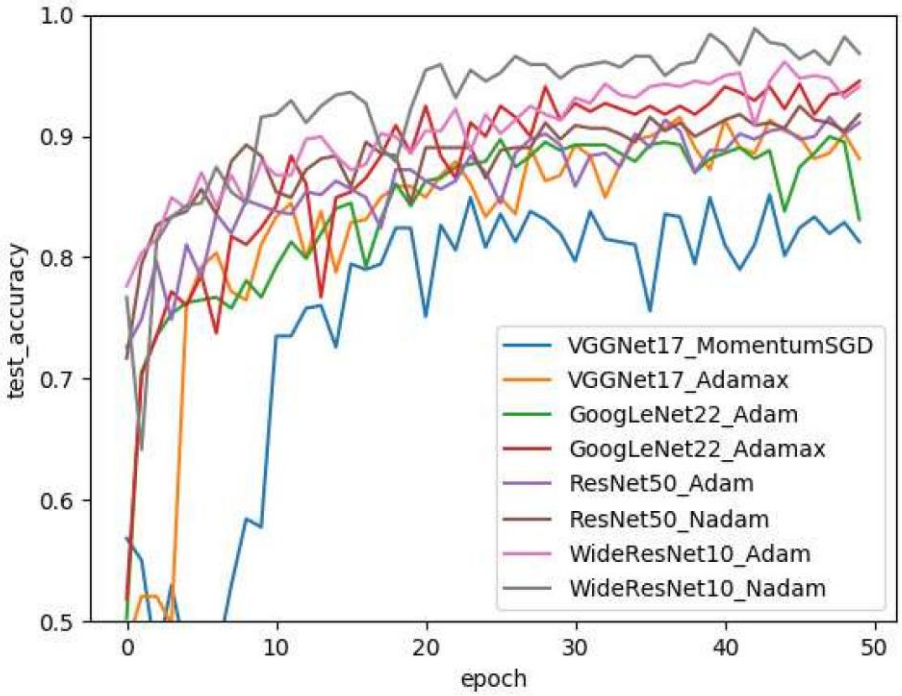


図 4.4　test accuracy（B 型チップバーン）

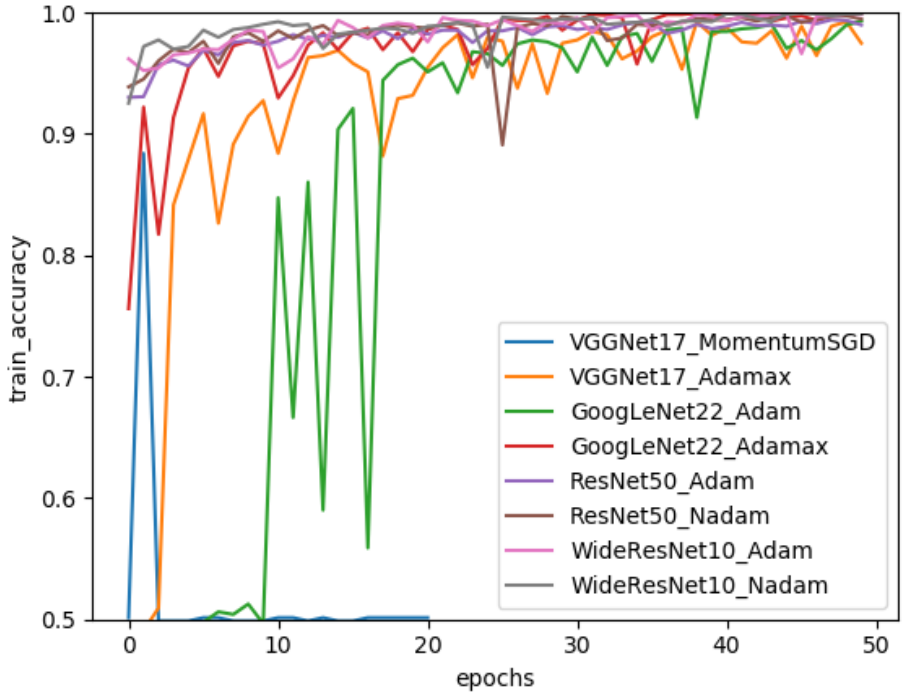


図 4.5　train accuracy（A 型チップバーン予兆）

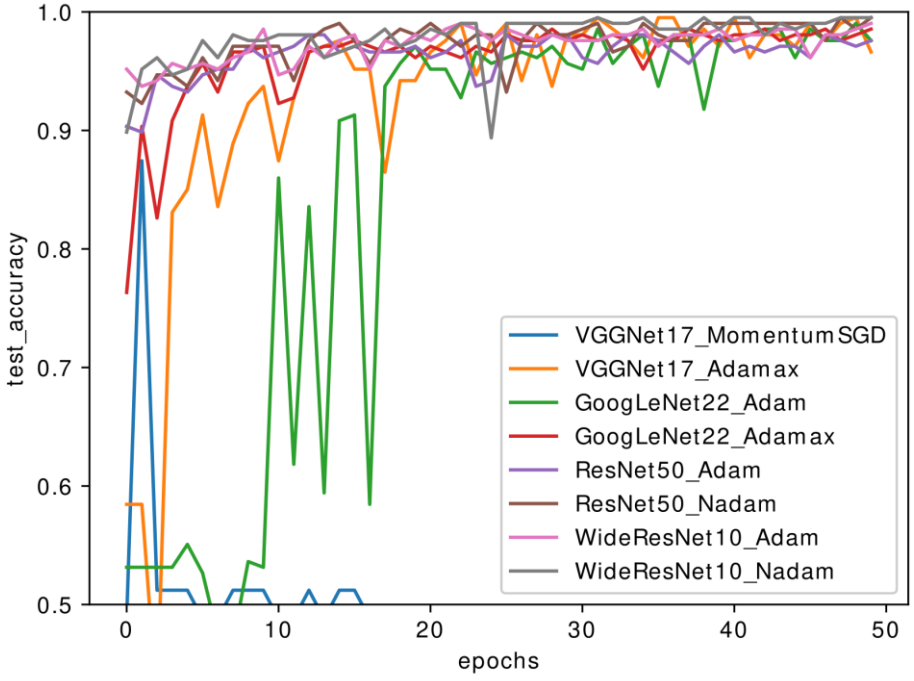


図 4.6　test accuracy（A 型チップバーン予兆）

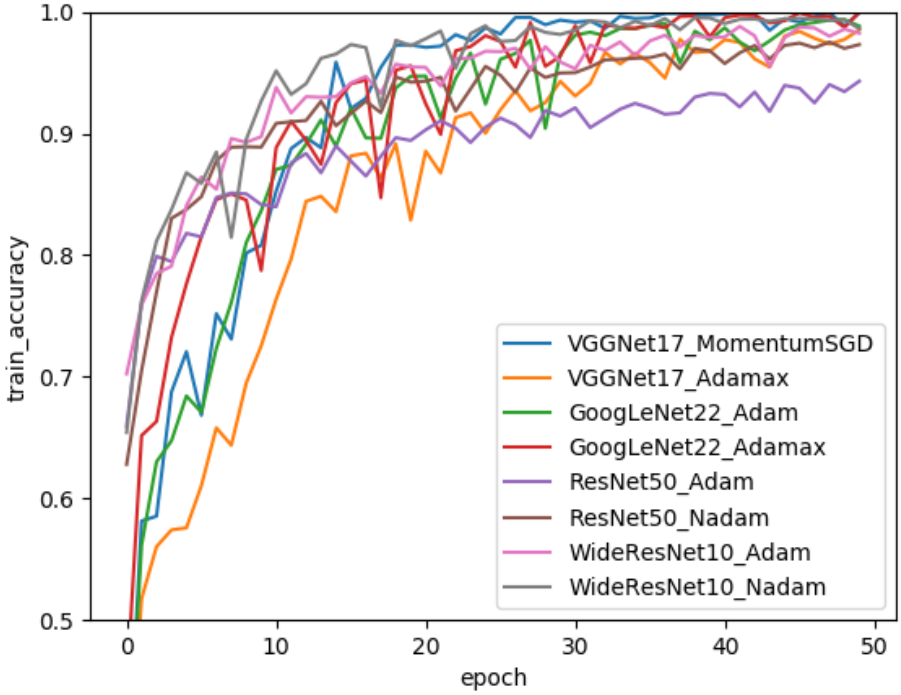


図 4.7　train accuracy（A または B 型チップバーン）

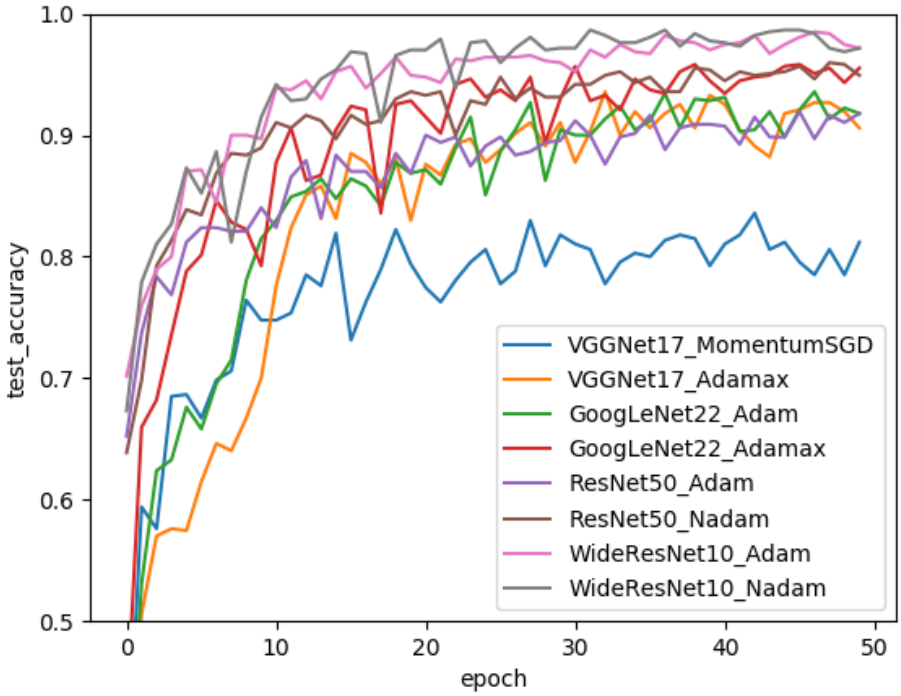


図 4.8　test accuracy（A または B 型チップバーン）

図 4.9　A 型チップバーン FN 画像例（A 型チップバーンを正常と誤認識）

図 4.10　A 型チップバーン FP 画像例（正常を A 型チップバーンと誤認識）

図 4.11　B 型チップバーン FN 画像例（B 型チップバーンを正常と誤認識）

図 4.12　B 型チップバーン FP 画像例（正常を B 型チップバーンと誤認識）

（1）正常と A 型チップバーンの 2 クラス分類の結果（図 4.2）を見ると，どの CNN モデルでも最終的にはほぼ 100%の正解率を出すことがわかる。VGGNet17[8]では Momentum SGD と Adamax[13]が，GoogLeNet22[9]では Adam[13]と Adamax[13]が，ResNet50[10][11]と WideResNet10[12]では Adam[13]と Nadam[14][15]が他の最適化手法と比べて優れた結果を出すことがわかる。この傾向は，以降の実験においても見られる。

（2）正常と B 型チップバーンの 2 クラス分類の結果（図 4.4）を見ると，（1）正常と A 型チップバーンの 2 クラス分類の結果と比べて全体的に正解率が低いことがわかる。WideResNet10[12]と Nadam[14][15]の組合せが 95～97%と最も高い正解率を出すことがわかる。

（3）正常と A 型チップバーン予兆の 2 クラス分類の結果（図 4.6）を見ると，（1）正常と A 型チップバーンの 2 クラス分類の結果と比較しわずかに正解率が低下するものの，ほとんどのモデルで 95%以上の正解率が確認される。

（4）正常とチップバーン（A 型または B 型）の 2 クラス分類の結果（図 4.8）を見ると，WideResNet10[12]および ResNet50[10][11]の最終的な結果にあまり違いは見られない。WideResNet10[12]と Nadam[14][15]の組合せが 96～98%と最も高い正解率を示すことがわかる。（4）正常とチップバーン（A 型または B 型）の 2 クラス分類の結果は, CNN の訓練時に A 型チップバーンと B 型チップバーンの画像データを同数で使用するため，（1）正常と A 型チップバーンの 2 クラス分類の結果と（2）正常と B 型チップバーンの 2 クラス分類の結果の中間的な結果になるといえる。

CNN モデルに GoogLeNet[9], 最適化手法に Adam[13]を用いた場合の誤認識について，（1）正常と A 型チップバーンの 2 クラス分類では，FN（見逃し：チップバーンレタスを正常レタスと誤認識）は 3 画像，FP（空振り：正常レタスをチップバーンレタスと誤認識）は 5 画像である。また，（2）正常と B 型チップバーンの 2 クラス分類では，FN は 5 画像，FP は 16 画像である。

最適化手法に Adam[13]を用いた場合の CNN と SVM[16][17]の比較結果（表 4.10）を見ると, すべての場合で CNN が SVM[16][17]より検証用データセットに対する正解率（test accuracy）において優れていることがわかる。

4.4 考察

CNN を用いた深層学習により，(1) 正常と A 型チップバーン，(2) 正常と B 型チップバーン，(3) 正常と A 型チップバーン予兆，(4) 正常とチップバーン（A 型または B 型）の各 2 クラス分類で，90%以上の高い正解率（test accuracy）が得られたのは，訓練用データセット（画像）のチップバーンの特徴が明確であったこと，また，チップバーンの有無のラベル精度が高かったことが要因と考えられる。

A 型チップバーンと B 型チップバーンとの比較において，A 型の方が B 型よりも正解率が高くなるのは，中心葉の先端に生じる A 型の方が，周辺葉の先端に生じる B 型よりも，チップバーンの特徴がより明確であったため，と考えられる。

A 型と B 型の両方のチップバーンともに，誤認識データは，FP（空振り）よりも FN（見逃し）の方が少ないという結果を得ている。本研究の背景から考えると，これら誤認識データの種別と数量は，重要な要素と考えられる。FN のように，チップバーンの見逃しが発生した場合，その個体は商品としてのクレーム対象となるため，経営面においては大きな問題となる。一方，FP は，歩留りに影響するのみで，クレームのような風評ダメージはない。もし，経営的な視点に立って FN の優先順位を高く考えるならば，現状の深層学習においては，FP の増加が歩留りの許容範囲に収まるように，FN を下げるチューニングが有効と考えられる。

誤認識の原因として，FP（空振り），すなわち正常レタスをチップバーンレタスとして誤認識したケースでは，(1) 葉の影部分をチップバーンと誤認識，(2) チップバーン予兆段階をチップバーンと誤認識，と推定される。また，FN（見逃し），すなわちチップバーンレタスを正常レタスと誤認識したケースでは，チップバーンの発生初期のものを正常レタスと誤認識，と推定される。

CNN と SVM[16][17]の比較については，CNN はほぼすべてのケースで検証用データセットに対する正解率（test accuracy）が 0.9 以上となり，植物工場生産作物の経営的歩留り目標の 90%を超えており，十分に高精度でチップバーンレタスを識別できることがわかる。SVM[16][17]では検証用データセットに対する正解率（test accuracy）が 0.5～0.7 で，CNN に比べて劣った結果となっている。これは，チップバーンの画像上の特徴が，緑色の退色および茶褐色への変色という色に強く現れ，葉先の縮れという形状の変化が軽微であることから，CNN が形状と色の両者の特徴をよく抽出できるのに対し，SVM[16][17]で用いた特徴抽出器の HOG[17]は形状の特徴を抽出できるのみで，色の特徴を十分に抽出できていないためであると考えられる。

第 5 章
むすび

5.1 まとめ

本研究は，深層学習の一種である CNN による画像診断技術を，人工光型植物工場生産レタスのチップバーン発生・未発生の 2 クラス分類へ導入することを目的とした。レタスの一種であるフリルレタスのチップバーンは，植物体の中央部の葉先に発生するタイプ（A 型）と，植物体の周辺部の葉先に発生するタイプ（B 型）がある。さらに，A 型チップバーン予兆，すなわち緑色の色素が失われてわずかに退色し，チップバーンが発生して葉先が茶褐色に変色して，野菜としての商品価値が損なわれる直前の状態がある。まず，深層学習で必須となる多量のカラー画像データ取得のために，A 型および B 型のチップバーンが発生する栽培条件を特定し，実際に A 型チップバーンレタス，B 型チップバーンレタスを栽培した。A 型チップバーン予兆レタスは，A 型チップバーンレタスを栽培する過程で作出した。

画像診断用の CNN については，VGGNet[8]，GoogLeNet[9]，ResNet[10][11]，WideResNet[12]の各モデル，および Momentum SGD，Adam[13]，Adamax[13]，Nadam[14][15]の各最適化手法を試行し，チップバーンの検出性能の比較を通じて，チップバーン識別率の向上手法を検討した。また，CNN を用いた深層学習の有効性を明確に示すために，2 クラス分類の古典的かつ代表的手法の一つである SVM[16][17]を比較対象に加えた。

CNN を用いた深層学習による画像診断実験の結果，チップバーン発生・未発生の 2 クラス分類において，B 型チップバーン，A 型チップバーン，A 型チップバーン予兆のいずれでも，人工光型植物工場生産作物の経営的歩留り目標である 90%を上回る正解率（test accuracy）を達成し，本研究の有用性を確認した。また，古典的手法である SVM[16][17]と比べて，CNN はすべての場合でチップバーンの識別性能が上回ることを確認し，CNN を用いた深層学習の有効性を確認した。以上により，人工光型植物工場生産レタスの深層学習による画像診断技術を確立した。

5.2　今後の課題

今後の課題として，誤認識された画像データのうち，FN（見逃し）の発生率をさらに下げる方法の確立，撮影条件を変えた他の訓練用画像データセットでの検証，CNNの他のモデルおよび他の最適化手法の検討，並びに実際の人工光型植物工場での撮影を想定して，レタス個体ではなく，レタス群落のカラー画像を用いた実験などがあげられる。また，実際の人工光型植物工場に実装可能な小型機器（Raspberry Pi 等を想定）を用いたチップバーン自動検出法の検証も，今後の課題である。

謝辞

本研究を進めるにあたり，直接御指導いただきました千葉大学大学院融合理工学府基幹工学専攻電気電子工学コース 小圷成一教授，中間公啓技術職員に深く御礼申し上げます。博士論文をご審査いただきました同コース 伊藤智義教授，下馬場朋禄教授，地球環境科学専攻都市環境システムコース 須貝康雄教授に御礼申し上げます。また，日頃から大変お世話になりましたシステム数理教育研究分野の皆様に篤く御礼申し上げます。

文献

[1] 平栗健史, 清水博幸, 進藤卓也, 木許雅則, 大田健紘: 解説 スマート農業に向けた取組み, 電子情報通信学会誌, Vol. 103, No. 6, pp. 951-599 (2020).
[2] 古在豊樹: 人工光型植物工場：世界に広がる日本の農業革命, pp. 193-212, オーム社 (2012).
[3] 丸尾 達: サラダナの tipburn に関する研究（第 1 報）葉位別 Ca 栄養について, 園学要旨, Vol. 58 秋, pp. 294-295 (1986).
[4] T. Maruo et.al: Which stage the tip burn occurrence is determined for individual lettuce leaf?, Hort. Res.(Japan), Vol. 18 (Sppl.1), p. 152, 2019 (in Japanese).
[5] E. Goto and T. Takakura: Reduction of lettuce tipburn by shorting day/night cycle, J. Agric. Meteorol., Vol. 59, No. 3, pp. 219-225 (2003).
[6] 下山真人, 溝田陽子, 末田香恵: 人工光型植物工場における葉物野菜の栽培技術, 大林組技術研究所報, No. 78, Vol. 34, pp. 1-6 (2014).
[7] 内田祐介, 山下隆義: 畳み込みニューラルネットワークの研究動向, 電子情報通信学会技術研究報告(パターン認識・メディア理解), Vol. 117, No. 362, pp. 25-38 (2017).
[8] K. Simonyan and A. Zisserman: Very deep convolutional networks for large-scale image recognition, Proc. Int. Conf. Learning Representations, pp. 1-14 (2015).
[9] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke and A. Rabinovich: Going deeper with convolutions, Proc. IEEE Conf. Computer Vision and Pattern Recognition, pp. 1-9 (2015).
[10] K. He, X. Zhang, S. Ren and J. Sun: Deep residual learning for image recognition, Proc. IEEE Conf. Computer Vision and Pattern Recognition, pp. 770-778, 2016.
[11] A. Veit, M. Wilber, and S. Belongie: Residual networks behave like ensembles of relatively shallow networks, Proc. Advances in Neural Information Processing Systems, pp. 550-558 (2016).
[12] S. Zagoruyko and N. Komodakis: Wide residual networks, British Machine Vision Conference 2016, BMVC 2016, Vol. 2016-September, pp. 87.1-87.12 (2016).
[13] D. P. Kingma and J. L. Ba: Adam: A method for stochastic optimization, Proc. Int. Conf. Learning Representations, pp. 1-15 (2015).
[14] T. Dozat: Incorporating nesterov momentum into adam, ICLR Workshop, No. 1, pp. 2013–2016 (2016).
[15] Y. E. Nesterov: A method for solving the convex programming problem with convergence rate $O(1/k^2)$, Dokl. Akad. Nauk SSSR 269, pp. 543-547 (1983).

[16] I. Tsochantaridis, T. Joachims, T. Hofmann and Y. Altun: Large margin methods for structured and interdependent output variables, J. of Machine Learning Research, No. 6, pp. 1453-1484 (2005).
[17] N. Dalal and B. Triggs: Histograms of oriented gradients for human detection, Proc. IEEE Conf. Computer Vision and Pattern Recognition, pp. 886-893 (2005).
[18] 嶋村茂治, 上原賢太, 小圷成一: 植物工場生産作物における植物生理障害の自動識別, 電気学会電子・情報・システム部門大会講演論文集, pp.1207-1210 (2018).
[19] 嶋村茂治, 上原賢太, 小圷成一: 植物工場生産作物における植物生理障害の自動識別, 電気学会論文誌 C, Vol. 139-C, No. 7, pp. 818-819 (2019).
[20] 嶋村茂治, 上原賢太, 小圷成一: 植物工場生産レタスにおけるチップバーンの自動識別, インテリジェント・システム・シンポジウム(FAN2018)講演論文集, pp. 101-104 (2018).
[21] 嶋村茂治, 上原賢太, 小圷成一: 人工光型植物工場生産レタスにおけるチップバーンの自動識別, 計測自動制御学会システム・情報部門学術講演会(SSI2019)講演論文集, pp. 672–677 (2019).
[22] K. Uehara, Y. Hatakenaka, S. Shimamura, S. Koakutsu: Automatic identification of lettuce tipburn in plant factory using convolutional neural networks, 電気学会電子・情報・システム部門大会講演論文集, pp. 1580-1581 (2019).
[23] S. Shimamura, K. Uehara, and S. Koakutsu: Automatic identification of plant physiological disorders in plant factory using convolutional neural networks, The Fifth Int. Conf. Electronics and Software Science (ICESS2019), pp. 7-11 (2019).
[24] S. Shimamura, K. Uehara, and S. Koakutsu: Automatic identification of plant physiological disorders in plant factories with artificial light using convolutional neural networks, Int. J. New Computer Architectures and their Applications, Vol. 9, No. 1, pp. 25-30 (2019).
[25] 小圷成一: ディープ・ラーニングについて - ディープ・ラーニングと IoT の現状と期待, 電気学会誌, Vol. 138, No. 5, pp. 270-271 (2018).
[26] S. Ioffe and C. Szegedy: Batch normalization: Accelerating deep network training by reducing internal covariate shift, Int. Conf. Machine Learning, Vol. 1, pp. 448-456 (2015).
[27] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov: Dropout: A simple way to prevent neural networks from overfitting, J. Machine Learning Research, Vol. 15, pp. 1929-1958 (2014).
[28] K. He, X. Zhang, S. Ren, and J. Sun: Delving deep into rectifiers: Surpassing humanlevel performance on imagenet classification, Proc. IEEE Int. Conf. Computer Vision, Vol. 2015 Inter, pp. 1026-1034 (2015).

業績リスト

査読付論文

[1] 嶋村茂治, 上原賢太, 小圷成一: 植物工場生産作物における植物生理障害の自動識別, 電気学会論文誌 C, Vol. 139-C, No. 7, pp. 818-819 (2019).

[2] S. Shimamura, K. Uehara, and S. Koakutsu: Automatic identification of plant physiological disorders in plant factories with artificial light using convolutional neural networks, Int. J. New Computer Architectures and their Applications, Vol. 9, No. 1, pp. 25-30 (2019).

国際会議

[1] S. Shimamura, K. Uehara, and S. Koakutsu: Automatic identification of plant physiological disorders in plant factory using convolutional neural networks, The Fifth Int. Conf. Electronics and Software Science (ICESS2019), pp. 7-11, Takamatsu Sunport Hall Building (Takamatsu), August 2, 2019.

学会発表

[1] 嶋村茂治, 上原賢太, 小圷成一: 植物工場生産作物における植物生理障害の自動識別, 電気学会電子・情報・システム部門大会講演論文集, pp.1207-1210, 北海道大学工学部(札幌), 2018 年 9 月 6 日.

[2] 嶋村茂治, 上原賢太, 小圷成一: 植物工場生産レタスにおけるチップバーンの自動識別, インテリジェント・システム・シンポジウム(FAN2018)講演論文集, pp. 101-104, 横浜国立大学理工学部(横浜), 2018 年 9 月 27 日.

[3] K. Uehara, Y. Hatakenaka, S. Shimamura, S. Koakutsu: Automatic identification of lettuce tipburn in plant factory using convolutional neural networks, 電気学会電子・情報・システム部門大会講演論文集, pp. 1580-1581, 琉球大学工学部(沖縄), 2019 年 9 月 4 日.

[4] 嶋村茂治, 上原賢太, 小圷成一: 人工光型植物工場生産レタスにおけるチップバーンの自動識別, 計測自動制御学会システム・情報部門学術講演会(SSI2019)講演論文集, pp. 672–677, 千葉大学西千葉キャンパス(千葉), 2019 年 11 月 25 日.