# Python でらくらく Deep Learning
## －Python の深層学習ライブラリを使う－

平田　隆幸*

# Learn Deep Learning with Python without Difficulty
## －Usage of Python Library of Deep Learning－

**Takayuki HIRATA***

(Received September 28, 2020)

Deep learning is one of the key concepts of neural networks in machine learning. Powerful libraries of Python are available for deep learning programming. In the field of pattern recognition, Python is a standard programming language. User-friendly tools are provided by many user groups. Here, I will show some useful software development tools for the beginners who are not familiar to both Python and deep learning. Furthermore, I will introduce the deep learning tool that enable users to construct AI applications without programming.

***Key Words*:** Deep Learning, Neural Network, Python, Keras, Artificial Intelligence.

## 1. はじめに

Deep Learning（深層学習）は，現在のキーワードにさえなってきた[1]．Google のチーム が開発した AlphaGo[2]が，囲碁の人間のヨーロッパチャンピオンに勝利し，さらに，元世界チャンピオンである李世ドル，世界ランキング 1 位の柯潔を打ち破ったことは衝撃を与えた．AlphaGo は，人間のプロに棋譜を学習（教師付き学習）して強くなった．さらに，人間の棋譜を使わず強化学習により，自己対戦のみで強くなった AlphaGo Zero が，AlphaGo を 100 対 0 で打ち負かした[3]．AI（人工知能）が，人間の手本無しに知的ゲームで人間を越えたことは，象徴的なできごとであった．

AlphaGo Zero を含めた AlphaGo においても，ニューラルネットワークの学習方法に Deep Learning が用いられている．Deep Learning の発展によって，AI が多くの分野で採用されるようになった．そして，さまざまな領域で AI が人間を超えてきている．

ライブラリを使うことで容易に Deep Learning を体験できるようになった．ハードウェアに関しても，一般的な PC でさまざまなことができるようになってきた．PC の CPU だけでなく，GPU を使って高速処理が可能となってきた．ここで，１）CPU のみでの実装，２）CPU＋GPU での実装，３）Google Colaboratory を使うことにより，Web 上で深層学習を手軽に体験できるのをみていく．

さらに，科学者のコミュニティを越えた社会の話題になっている量子コンピュータ [4]も深層学習の分野への応用が期待されている．

本論文では，Python をもちいた深層学習を工学部の学生が簡単に学べるようになるにはどうすれば良いのかについて考える．Programming が苦手であると感じる学生が深層学習を学ぶ入り口になることを願う．

## 2. ニューラルネットワーク

### 2.1 ニューロン

ニューラルネットワークは，ニューロン（神経細胞）によって構成されたネットワークである．最初に，情報処理素子としてのニューロンを考えよう．もっとも単純なニューロンのモデルは，マカロックとピッツの形式ニューロン[5]である．

* 大学院工学研究科知能システム工学専攻
* Human and Artificial Intelligent Systems Course, Graduate School of Engineering

マカロックとピッツの形式ニューロンは，多入力一出力の情報処理素子である．最も簡単な 2 入力 1 出力の形式ニューロンを考える(図 1 参照)．ここで，シナプス結合に対応する重みを $w_1$, $w_2$ とする，入力信号を $x_1$，$x_2$ とし，出力信号を y とする．形式ニューロンは，入力によって発火するかどうかの閾値 h をもっている．ニューロンへの入力に重みを掛けた総和 $z = w_1 \cdot x_1 + w_2 \cdot x_2 \geqq h$ の場合，出力 y=1 となり（発火），それ以外は y=0 の 2 値をとる．ニューロンの出力は，シナプス結合を介して，次のニューロンに入力される．

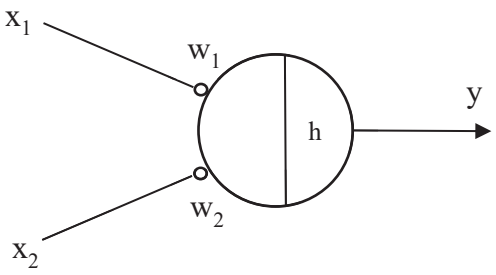



図 1　マカロックとピッツの形式ニューロン．ここで，$x_1$，$x_2$ は入力信号，y は出力信号，$w_1$，$w_2$ はシナプス結合の重みを表す．

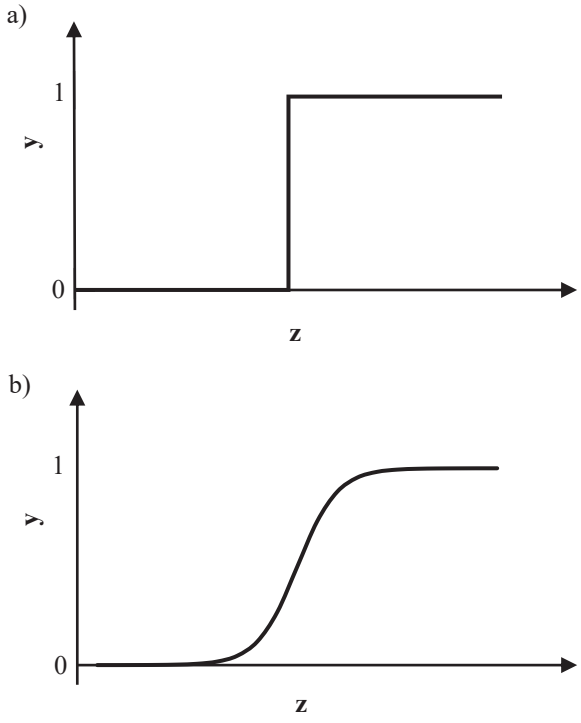



図 2　活性化関数 u(z)の例．a)ステップ関数出力，b)シグモイド関数．

入力が閾値を超えると，ニューロンは興奮状態になり，発火する．それゆえ，ニューロンが発火するかどうかを決定する閾値は，重要なパラメータである．マカロックとピッツの形式ニューロンでは，閾値 h という 1 パラメータによって，出力が決まるステップ関数がもちいられている(図 2 -a 参照)．より一般的には，ニューロンが発火するかどうか決める関数として，活性化関数が使用されている．ニューラルネットワークの学習において，活性化関数は重要な役割を果たす．典型的な活性化関数，ステップ関数とシグモイド関数の例を図 2 に示す．なお，ステップ関数とシグモイド関数以外にも，ソフトサイン，ReLU 関数などが活性化関数としてもちいられている．

### 2.2 ニューラルネットワーク

ニューラルネットワークとは，人間の脳を模して情報処理させようと研究が始まった(参考書として，先駆的日本人研究者も甘利氏の神経回路網[6])．ニューラルネットワークは，ニューロン（神経細胞）の挙動をモデル化した人工ニューロンを複数個の結合させたものである．人間の脳をモデルに並列処理により，高度な情報処理の実現を目的としている．並列処理，例えば画像処理など，が得意なのが特徴である．

結合させたニューロンで構成されたニューラルネットワークに目的とする情報処理をさせるためには，学習が重要である．ニューラルネットワークの学習によく使われるが機械学習である．深層学習は，ニューラルネットワークの機械学習の一つである(初歩的かつ網羅的な参考書として[7])．図 3 に，機械学習，深層学習の包含関係図を示す．次に，ニューラルネットワーク，機械学習，深層学習と順にみていく．

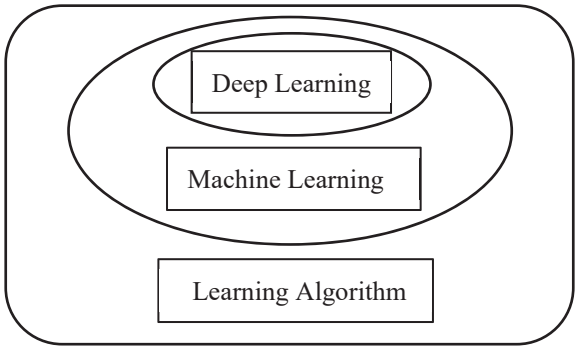



図 3　深層学習とは（概念図）．

機械学習は，データを学習して，分類や予測などをおこなえるアルゴリズムやモデルを自動的にコンピュータが構築できるようにするものである．機械学習は，学習方法のより，答えを教えずに学習させる「教師なし学習」，正解を与えた状態で学習させる「教師あり学習」，「強化学習」の３つに分けることができる．ライブラリを使った Python による機械学習の本も多く出版されている[8][9].

さて，深層学習とは，機械学習の一つである．深層学習は，画像認識，音声認識，完全情報ゲームのアルゴリズムなどの分野で使われている．ニューラルネットワークという観点からも現在注目を集めている，畳み込みニューラルネットワーク，リカレントニューラルネットワークの学習に，深層学習が用いられる．

ニューラルネットワークには，ニューロンの結合様式によっての様々なものがある．ここでは，畳み込みニューラルネットワーク，リカレントニューラルネットワーク，そして GAN など代表的なニューラルネットワークを順に見ていく．

### 2.3 さまざまな Neural Networks

Convolutional Neural Networks（畳み込みニューラルネットワーク）や Recurrent Neural Networks（回帰型ニューラルネットワーク）以外にも，全結合ニューラルネットワークモデルなど多くの Neural Networks モデルがある．

最初に，Convolutional Neural Networks（以降 CNN と略す）をみていこう．ニューラルネットワークの歴史は古い．1950 年代には，パーセプトロンによる第 1 次のブームが, 1980 年代には, アルゴリズム「バックプロパゲーション」，多層パーセプトロン，畳み込みニューラルネットワーク(CNN)などによる第 2 次のブームが起こっている．CNN の歴史は，ブームとともにあった. ニューラルネットワークとしては，決して新しくはない．しかし，コンピュータの計算能力が限られていたため，初期の CNN はニューロン層が多くなかった．

現在のブームは，コンピュータの計算能力の向上により，多層のニューラルネットワークモデルが可能となって，第 3 次のブームになったといえる．例えば，AlphaGo は多層のニューラルネットワークを学習させることつまり Deep Learning によって, 棋力の向上を実現している．Alpha GO は，強化学習(A reinforcement learning (RL))と 13 層の policy network と value network の２つの neural network を用いて，プロ棋士越えの棋力を実現している．

次に，得意な応用分野という観点から Recurrent Neural Networks（以降 RNN と略す）と比べてみる．RNN は，音声認識など時系列解析によく使われる．一方，CNN は，画像認識などある時刻でのパターン認識を得意とする．それぞれが得意とする分野で使い分けられている．

Deep Learning をニューラルネットワークの応用という観点から見てみよう．医療分野における画像診断（CT, NMR）で，病巣部を見つけるのに Deep Learning の技術は利用されている[参照，review paper [10][11]]．AI を使った医療画像診断は，AI を補助的に利用するに留まらず Deep Learning によって飛躍的に向上した．乳癌のスクリーニング検査において，AI システムは，一般の医師のみならず熟練した専門医より良い成績を達成した[12].

## 3. Deep Learning とは

Deep Learning とは, 多層化したニューラルネットワーク(最低 4 層以上)を機械学習させることである(例えば，review 論文[13]). ここで，ニューラルネットワークの学習について考えてみよう．ニューラルネットワークにおける学習とは，シナプス結合の重みを変えることである．学習によりシナプス結合の重みを最適化し，システムに実現したい task を達成させることである．（AlphaGo の例でいうと，実現したい task とはもっとも勝利の確率が高い手（最善手）を選ぶことである．）

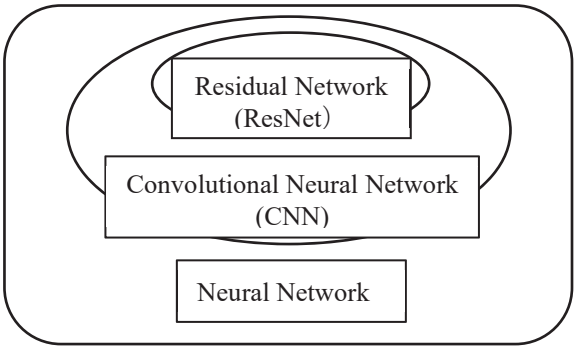



図４　畳み込みニューラルネットワーク．

多層化という視点からニューラルネットワークの発展の歴史を振り返ってみよう．ニューラルネットワークは，層を増やすことにより複雑な処理を実現してきた【簡単に言えば，多層化つまり Deep Learning(深層学習)である】. しかし, 層を多くすることには，メリットばかりではなく，デメリットも存在する．それは決定しなければならないパラメータ（シナプス結合の重み w）の数が増大することであ

る．それゆえ，多層のネットワークおけるシナプス結合の重みを更新する方法が重要になってくる．この問題を解決するために，現れたのが逆誤差伝搬（back propagation）と畳み込みニューラルネットワークモデルであり，ニューラルネットワークの研究を大きく前進させた．

さらに，畳み込みニューラルネットワークにおいて，より多層化することでニューラルネットワークの性能を向上させることが試みられた．特徴抽出を考えた場合，層を重ねることにより，一般的には，より複雑な特徴の抽出ができることが期待されるからである．Convolution 層はフィルタを持ち，Pooling 層と組み合わせて特徴を検出する役割を持たせるのである．低・中・高レベルの特徴を多層形式で自然に統合し，認識レベルを強化することができた．このようなネットワークの代表的な例として ImageNET[14]，AlexNET[15]などがある．

2014 年の画像認識の分野でトップを争う ImageNet コンペティションにおいて，複数の部門で優秀な成績をおさめたモデル（VGG モデル[16]）は，多層の CNN であった．層の数を 16 層や 19 層まで深くしたのだ．それで Very Deep と名付けられていた．また，2014 年の the ImageNet Large-Scale Visual Recognition Challenge 2014 (ILSVRC14)で勝利を飾った GoogLeNet[17]も 22 層だった．

コンピュータの計算能力の向上により，CNN の層を増やせるようになってきた．しかし，CNN の層を増やすだけでは限界が来た．層を更に深くする試みはあったものの，思い通りに学習が進まなかったのである．さらに，単純に層を増やすと，性能が悪化することが報告された．

効率よく多層化するために考えられたのが，現在もよく使われている ResNet[18]である．ResNet は，Microsoft Research( 現 Facebook AI Research) の Kaiming He 氏が 2015 年に考案したニューラルネットワークのモデルである．ResNet は「ある層で求める最適な出力を学習するのではなく，層の入力を参照した残差関数を学習する」ことで最適化しやすくしている．これを，ResNet では，残差ブロックと Shortcut Connection を導入することで実現している．残差ブロックでは，畳込み層と Skip Connection の組み合わせになっている．ResNet は，残差ブロックを導入することで，層の深度の限界を押し上げ，精度向上を果たした．2015 年において ResNet は 152 層まで深くすることが出来た．Very Deep の 16，19 層や GoogLeNet の 22 層と比較すると，層が深くなったことが分かる．

以上，CNN と多層化について概観した．

## 4. Deep Learning で画像処理

Python を使った Deep Learning による画像認識をみていく．Keras[19]は，TensorFlow または CNTK，Theano 上で実行可能な Python の高水準のニューラルネットワークライブラリであり，深層学習ライブラリとしてよく使われている．ここでは，Python のライブラリを使うことにより，簡単に画像認識ができることを紹介する．具体的には，2 種類の開発環境のもとで Python のプログラムを実行する：1）IDLE を使って PC の terminal 上で走らせる，2）Google の Colaboratory を使って Web 上で走らせる．データセットとして CIFAR-10 を使い，Keras を用いて画像の分類を実行してみよう．

### 4.1 画像データセット(CIFAR-10)

CIFAR-10 データセット[20]とは，機械学習の研究にもっともよく用いられる画像データセットである．CIFAR-10 データセットは，10 のクラスに分類された分解能が 32×32 の 60000 枚のカラーイメージから構成される．それぞれ，1 クラスに付き 6000 枚のカラーイメージを含んでいる．テスト用のものは，それぞれのクラスからランダムに選ばれた 1000 枚のイメージから構成されている．訓練用のものは，残りからランダムな順に選んでつくられている．訓練用の batch は 5000 枚のイメージから成り立っている．

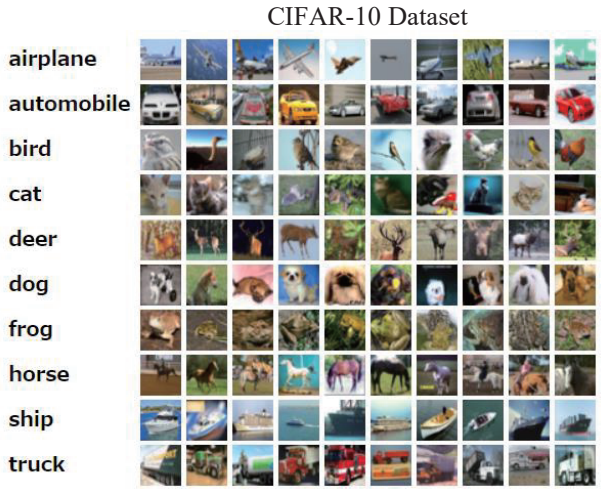



図５　CIFAR-10 の画像データ例（文献[20]より引用）．

図５に CIFAR-10 データセットのサンプル例を示す．飛行機，自動車，鳥，猫，鹿，犬，カエル，馬，船，トラックの 10 のクラスがある．Python version，Matlab version，binary version が公開されている．画像分解能が 32×32 で非常に軽いのも特徴であり，例

えば Python version は 163MB である．

### 4.2 IDLE で Python を走らせる

Deep Learning を体験するため，開発環境として IDLE を使って PC 上で Python を走らせた．2020 年 9 月 20 日時点での，最新の安定な Python の version は 3.8 である．しかし，version 3.8 の Python では，Keras 2.3.1 は動かない場合がある．それゆえ，Keras2.3.1 をスタンドアローンで走らすため，PC の Python version を Python 3.7 で動作させた．なお, 2020 年 6 月 18 日に，Keras 2.4.0 がリリースされている．しかし，バックエンドとしては TensorFlow のみの対応となっており，Keras は TensorFlow 2.0 とのパッケージ，tensorflow.keras として配布されている[19]．

さまざまなライブラリを使うため，Python の version を変えてプログラムを走らせるように，Anaconda の仮想環境を作成した．以下に，Anacoda Prompt にて以下のコマンドを実行する手順を示す．

① Anaconda 仮想環境の作成

conda create --name=py37 python=3.7

② 仮想環境「py37」の有効化

activate py37

環境が切り替わると Anacoda Prompt に(py37)と表示される．

Python 3.7 が走る環境で，cifar10_cnn.py を走らせた．

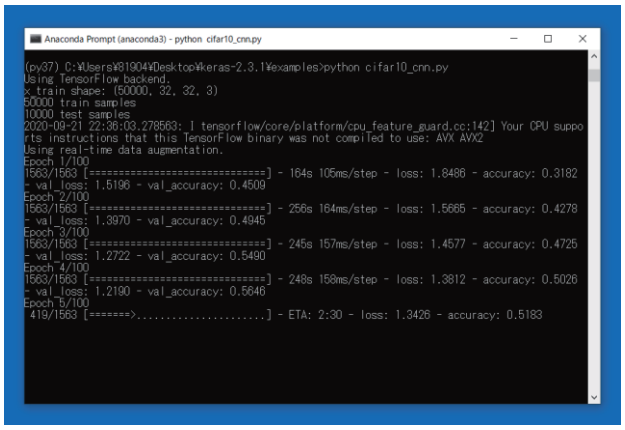



図 6　Anaconda Prompt で Keras の example の cifar10_cnn.py を走らせている様子．

最終的に得られた結果を以下に示す．

Saved trained model at C:¥Users¥81904¥Desktop¥keras-2.3.1¥examples¥saved_models¥keras_cifar10_trained_model.h5

10000/10000 [==============================]

- 16s 2ms/step

Test loss: 0.7182457206726074

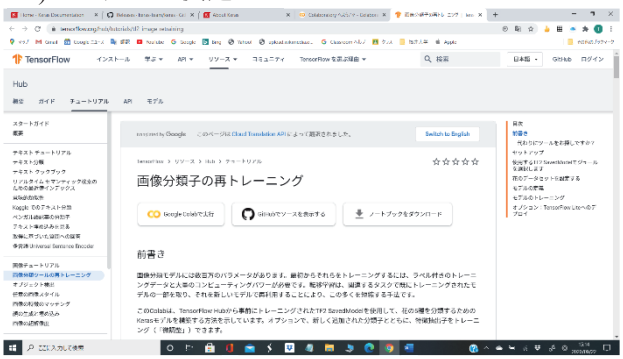
Test accuracy: 0.7627999782562256

訓練されたモデルが keras_cifar10_trained_model.h5 として保存され，約 76%の精度で，分類に成功していることが分かる．

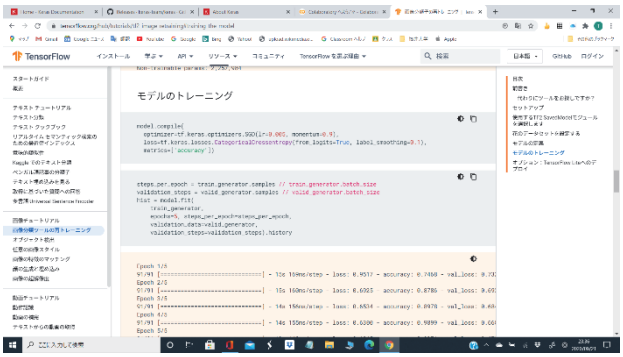
### 4.3 Google Colaboratory

Colaboratory は，Web 上で Python を走らせることができるアプリケーションである[21]．

a) モデルの決定




b) トレーニング




c) 分類の結果

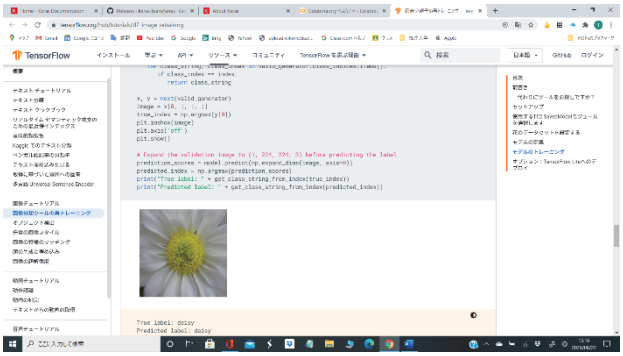



図 7　Google Colaboratry の機械学習のチュートリアルのスナップショット．機械学習の例，画像分類の再トレーニングで, Keras モデルで花の分類をおこなっている様子を a)-c)で示している．

Google のアカウントを持っていると，Web 上で Colaboratory にアクセスできる. ノートブックを使って, Python のコードを記述し実行できる. Colaboratry の ノートブックは，Colaboratory がホストする Jupyter ノートブックである[22]．なお，Anaconda ディストリビューションで PC に Python をインストールすると，Jupyter ノートブックを使用できるようになるので, 多くの Python ユーザーにはなじみ深いものかもしれない.

Colaboratory にアクセスすると, 左側の目次の欄に機械学習および機械学習の例という項目が出てくる. 機械学習の例では，機械学習のチュートリアルが準備されている．機械学習のチュートリアルとして，画像分類の再トレーニングで, Keras モデルで花の分類を Python を使って体験できる（参照，図７）.

さて，Web 上の Colaboratory で Deep Learning のモデル cifar10_cnn.py を走らせてみよう. なお，ノートブックの設定で，ハードウェアアクセラレータで GPU を選択しておこう. ニューロン数が少ない小さなニューラルネットワークモデルなどの場合を除き, 計算に要する時間がハードウェアアクセラレータ NONE の時と比較して大幅に短縮される.

Keras の cifar10_cnn.py をノートブックに貼り付けて実行した.

```
Using TensorFlow backend.
Downloading data from https://www.cs.toronto.edu/~kriz/cifar-10-python.tar.gz
170500096/170498071 [==============================] - 6s 0us/step
x_train shape: (50000, 32, 32, 3)
50000 train samples
10000 test samples
Using real-time data augmentation.
Epoch 1/100
1563/1563 [==============================] - 54s 35ms/step - loss: 1.8768 - accuracy: 0.3105 - val_loss: 1.5757 - val_accuracy: 0.4338
Epoch 2/100
1563/1563 [==============================] - 48s 31ms/step - loss: 1.5857 - accuracy: 0.4207 - val_loss: 1.3747 - val_accuracy: 0.5005
Epoch 3/100
……
中略
……
Epoch 99/100
1563/1563 [==============================] - 47s 30ms/step - loss: 0.8121 - accuracy: 0.7338 - val_loss: 0.7903 - val_accuracy: 0.7485
Epoch 100/100
1563/1563 [==============================] - 47s 30ms/step - loss: 0.8069 - accuracy: 0.7348 - val_loss: 0.6688 - val_accuracy: 0.7797
Saved trained model at /content/saved_models/keras_cifar10_trained_model.h5
10000/10000 [==============================] - 2s 191us/step
Test loss: 0.6688448459625244
Test accuracy: 0.779699981212616
```

以上のような結果が得られた．訓練されたモデルが keras_cifar10_trained_model.h5 として保存され，約 78%の精度で，分類に成功していることが分かる.

## 5. プログラミング無しで Deep Learning

Python を使って, 簡単に Deep Learning を用いた画像処理が可能であることを見てきた. しかし, Python のプログラミングは必要であった．Deep Learning を使った画像処理には，プログラミングが必要なのだろうか？Sony が提供している Neural Network Console を使うことによって, プログラミングを必要とせずに Deep Learning を体験できる．次に，一歩進んで, プログラミングなしで Deep Learning を体験できるのを見ていこう.

### 5.1 Neural Network Console とは

Sony の Neural Network Console[23]は，ドラッグ＆ドロップで簡単にニューラルネットワークの設計・実行ができるアプリケーションである．Python を知らなくても Deep Learning を体験でき, プログラミングなしで Deep Learning をもちいた AI アプリケーションの開発を可能にしてくれる.

Neural Network Console は，サインインすることで Web から実行することができる. ID が割り当てられ, Neural Network Console Cloud (dl.sony.com)で設計・開発が実行できる．長時間のシステム利用を除いて，無料で, 本格的な AI アプリケーション開発がおこなえる．なお，Google Colaboratory と同様に，長時間使用する場合など有料のサービスも提供している．同時に，Sony は，ローカル PC で動作する Windows 版の Neural Network Console も無償で公開している. Windows 版を download することにより PC のローカル環境でも実行できる. なお, 現在の最新のものは, 2020 年 7 月 30 日リリースの Version 1.8 である.

### 5.2 Neural Network Console を使う

ローカル PC で Windows 版の Neural Network Console を動作させた例を図８に示す. Sample project の 01_logistic_regression.sdcproj が示されている．これは，プロジェクトを選択すると最初に現れるエディット画面である．Main という表示が出ている中央のコラムにモデルがグラフィックに示されている．ここで，ドラッグ＆ドロップでモデルを構築していくことになる．ブロックを重ねたり，ラインで結ぶことによって，モデルの動作順を決めることができる．プログラミングにおいて，フローチャートを描くのに対応している．

次に，このプロジェクトを例に動作を見ていこう（図９参照）．最初に，a) グラフィックを使って，ドラッグ＆ドロップによってモデルを構築する，次に，b) run ボタンを押してモデルを学習させる，最後に，c) 学習したモデルで文字の判別をおこなう．

ここでは，手書き入力された２種類の数字 4 と 9 の識別がなされている．入力文字が 9 と認識されると 1 と出力し，そうでない場合は 0 と出力されるモデルになっている:2 値の出力. 図 9 の c) を見ると，上からの順に左の入力画像に対し，0，0，1，1 という出力，つまり 4，4，9，9 という判定が得られたことが分かる．これは，図 9 の a) モデル化で，BinaryCrossEntropy を使いと 2 値の出力が得られるように設計していることによる．また，学習の過程が図 9 の b) のようにグラフで見ることができるようになっているのも親切である．

手書き入力の判別モデル以外にも，CIFAR-10 の画像データの分類を ResNet を使っておこなっているプロジェクトのサンプルが用意されている（resnet-110.sdcproj）．ResNet 以外にも，Alexnet などを使い Deep Learning が体験できる sample が示されている．

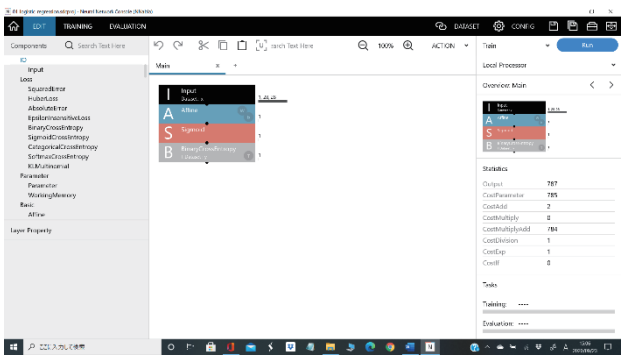
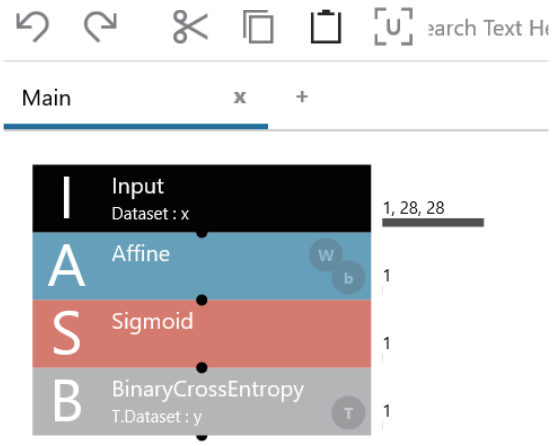
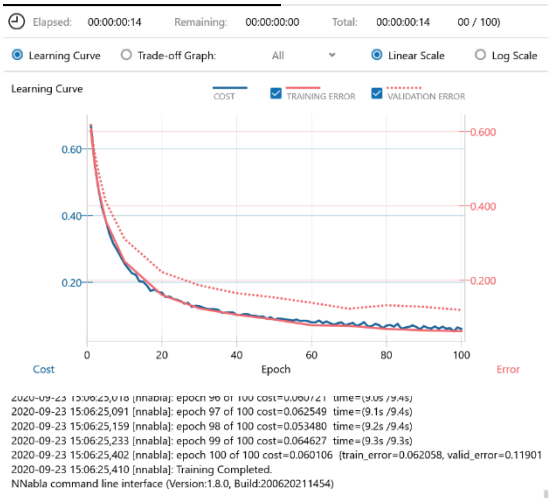


図８　Neural Network Console の sample project の例.

a) モデル化




b) 学習




c) 評価

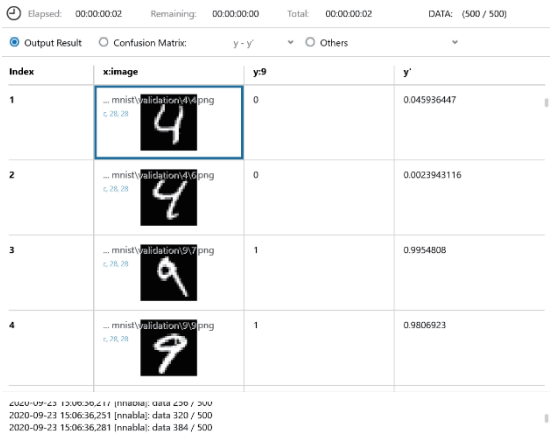



図９　Neural Network Console の動作手順．a)モデル化，b)学習，c)評価のステージが示されている．

## 6. おわりに

コンピュータプログラミングが苦手でも Deep Learning を使ったさまざまな研究ができる可能性を

示した．Keras などのライブラリを使うことにより，簡単に Deep Learning を使ったアプリケーションをつくれるようになってきた．

ここでは，Keras を取り上げたが，Python の Deep Learning に関連したライブラリには Chainer, PyTorch …など多くのものがある．これらのライブラリは常に進化している．例えば，日本発(株式会社 Preferred Networks)の Chainer は，2019 年に開発終了し，PyTorch へ移行した．また，ここで取り上げた Keras もスタンドアローンの Keras から TensorFlow 同梱版の「tf.keras」へと移行している．Keras は，3 つのバックエンド(TensorFlow, Theano, CNTK) が利用可能でしたが，Keras 2 で開発はストップし，以降は Tensorflow のみに対応し tf.keras へ移行した（[19]参照）．

AlphaGo で，Deep Learning が注目されるようになったことを述べたが，Deep Learning はますます身近な技術なってきた．AlphaZero[24]では，囲碁・チェスだけでなく，日本の将棋についても応用されている．さらに，Python を使って将棋 AI を簡単につくる本[25]も出版されるようになってきた．このように，Deep Learning の壁はどんどん低くなり，学習しやすくなってきている．さらに，プログラミングを必要とせず Deep Learning を使ったアプリケーションを設計・開発できる Neural Network Console を紹介した．本論文により，Deep Learning を学習してみようという壁がより低くなってくれる一助になればと思う．

最後に，プログラミング技能は必要かということについて考えてみる．少年ジャンプに連載されまたアニメ化されたので，目にした機会が多いだろう Dr.STONE という漫画がある．Dr.STONE という漫画は，「ある病気により」文明が失われて，原始時代から文明を再構築するというストーリーである．現在の科学技術に基づいて物語は構成されている．物語のなかでは，真空管などをゼロからつくっていっている．ここでは，Python のプログラムの中身を知らなくても，Deep Learning による画像分類ができることを見てきた．同じように，トランジスタの動作をしらなくてもコンピュータやスマートフォンを使える．これと同じであると思えば，プログラミングができなくても Deep Learning を使いこなせれば良いという考え方でも問題ではない．しかし，漫画 Dr.STONE の状況を思えば，0 から Deep Learning のプログラミングができればもっと楽しく AI などを学習できるのではないだろうか．最後に，さらに勉強したい方のために Excel や Python を使った Deep Learning の初歩的な参考書([26]-[28])をあげておく．

**謝　辞**

論文を執筆するにあたり，議論および有益なコメントをしてくださった髙田宗樹教授，および研究室のメンバーに感謝いたします．

**参考文献**

[1] Y. LeCun, Y. Bengio & G. Hinton: Deep learning. Nature, 521, 436–444 (2015).

[2] D. Silver et al.: Mastering the game of Go with deep neural networks and tree search, Nature, 529, 484–489 (2016).

[3] D. Silver et al.: Mastering the game of Go without human knowledge, Nature, 550, 354–359 (2017).

[4] F. Arute et al.: Quantum supremacy using a programmable superconducting processor, Nature, 574, 505-511 (2019).

[5] W. S. McCulloch &W. Pitts: A logical calculus of the ideas immanent in nervous activity. The bulletin of mathematical biophysics, 5, 115-133 (1943).

[6] 甘利俊一：神経回路網の数理 - 脳の情報処理様式 - , 産業図書, pp. 328 (1978).

[7] 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, 技術評論社, pp. 277 (2018).

[8] クジラ飛行机，杉山洋一，遠藤俊輔: Python による AI・機械学習・深層学習アプリのつくり方，ソシム, pp. 399 (2018).

[9] A. Gèron 著 下田倫大 監訳 長尾高弘 訳: scikit-learn と TensorFlow による実践機械，オライリー・ジャパン, pp. 539 (2018).

[10] M.I. Razzak, S. Naz, A. Zaib: Deep Learning for Medical Image Processing: Overview, Challenges and the Future, In: Dey N., Ashour A., Borra S. (eds) Classification in BioApps. Lecture Notes in Computational Vision and Biomechanics, vol.26, Springer, (2018).

[11] R. Miotto, F. Wang, S. Wang, X. Jiang, J. T. Dudley: Deep learning for healthcare: review, opportunities and challenges, Briefings in Bioinformatics, Volume 19, Issue 6, November 2018, Pages 1236–1246, (2018).

[12] S. M. McKinney et al.: International evaluation of an AI system for breast cancer screening, Nature, 577, 89–94 (2020).

[13] J. Schmidhuber: Deep learning in neural networks: An overview, Neural Networks, Vol. 61, January, Pages 85-117 (2015).

[14] J. Deng, W. Dong, R. Socher, L. Li, Kai Li and Li Fei-Fei: ImageNet: A large-scale hierarchical image

database, 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, pp. 248-255, (2009).

[15] A. Krizhevsky, I. Sutskever, E. Hinton: ImageNet classification with deep convolutional neural networks, Proceedings of the 25th International Conference on Neural Information Processing Systems (NIPS'12) - Volume 1, pp 1097–1105 (2012).

[16] K. Simonyan, A. Zisserman: Very Deep Convolutional Networks for Large-Scale Image Recognition, In ICLR, arXiv:1409.1556 (2015).

[17] C. Szegedy et al.: Going deeper with convolutions, Proceeding of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1-9 (2015).

[18] K. He, X. Zhang, S. Ren and J. Sun: Deep Residual Learning for Image Recognition, 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, pp. 770-778 (2016).

[19] Keras: https://keras.io/ja/#keras

[20] CIFAR-10: https://www.cs.toronto.edu/~kriz/cifar.html

[21] Google Colaboratory: https://colab.research.google.com/notebooks/intro.ipynb#

[22] Jupyter: https://jupyter.org/

[23] Sony Neural Network Console: https://dl.sony.com/console/#/dashboard

[24] D. Silver et al.: A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play, Science, Vol. 362, 1140-1144 (2018).

[25] 山岡忠夫: 将棋 AI で学ぶディープラーニング, マイナビ出版, pp. 277 (2018).

[26] 涌井良幸・涌井貞美: ディープラーニングがわかる数学入門, 技術評論社, pp. 239 (2017).

[27] 斎藤康毅:ゼロから作る Deep Learning －Python で学ぶディープラーニングの理論と実装－, オライリー・ジャパン, pp.298, (2016).

[28] Antonio Gulli (アントニオ・グッリ) & Sujit Pal (サジット・パル)著, 大串正矢, 久保隆宏, 中山光樹訳：直感 Deep Learning －Python×Keras でアイデアを形にするレシピ-, オライリー・ジャパン, pp.309, (2018).