AI画像分類プログラムの作成を容易にするPythonモジュールの開発

Development of a Python Module to Facilitate AI Image Classification Programming

根岸かほ＊　八木徹＊
Kaho Negishi*　Toru Yagi*

＊江戸川大学
*Edogawa University

現在，情報活用能力，及びAI人材の育成が重視されている．こうした背景の中で，プログラミング初学者でも，AIを利用したプログラムの作成ができるようになることは，AI学習の裾野を広げ，AI活用能力を持つ人材を育成するための基礎としても重要である．代表的なAIフレームワークとして，TensorFlowや学習済みモデルのTensorFlow Hubの他，独自の画像分類モデルを作成するWeb上のサービスとしてTeachable Machineなどがある．しかし，初学者にとっては，これらを活用したPythonプログラムを作成することは難易度が高い．そこで，AI学習を行うため，TensorFlowの機能や学習モデルを利用しやすくするPythonライブラリを作成することは，特に意義深いといえる．本研究では，TensorFlow Hubの学習済みモデルや，Teachable Machineで作成したモデルを用いて，画像分類を行うPythonモジュールを開発し，そのモジュールを用いたサンプルプログラムを作成した．

キーワード：TensorFlow Hub，Teachable Machine，画像分類，プログラミング学習，Stretch3

1. はじめに

1.1 背景

現在，情報技術は日々著しい発展を続けている．例えば，移動通信システムの最大通信速度は，1980年から2020年の40年間で，約100万倍にまで向上している［1］．このような通信システムの高速大容量化をベースに，クラウドやビッグデータ，IoT，人工知能（以下AI）といった，様々な情報技術が結びつくことで，新たな価値が生まれ，これまでにないサービスの登場につながっている．

こうした情報技術の発展の中でも，特にAIの進歩には目覚ましいものがある．画像分類や音声認識，自動翻訳などのほか，画像や文章を生成するいわゆる生成AIなどが注目されている．これらの技術はすでに家電製品や，Web経由で提供されるサービスに組み込まれ，身近なものとなっているものも多い．

このような急速な技術の発展は，社会や職業に大きな変化をもたらすと考えられるが，それが具体的にどのようなものとなるかを想定することは難しい．文部科学省は，この予測困難な社会において，必要な情報の収集・選択と得た情報を活用し，他者と協働することで，新たな価値の創造をしていくための情報活用能力が重要であると指摘している［2］．

この情報活用能力は，情報や情報手段を主体的に選択・活用して，問題解決をしたり，自分の考えを形成したりしていく力を指しており，新学習指導要領（小中学校（2017告示），高等学校（2018年告知））においては，言語能力と並び，学習の基盤となる資質・能力と位置づけられている［3,4］．この学習指導要領に基づき，2020年度からは情報活用能力の育成・ICT活用が実施されている．

学習指導要領に基づいた具体的な学習内容として，小学校では文字入力などの基本的な操作の習得と，新たにプログラミング的思考の育成，中学校では，技術・家庭科（技術分野）において，プログラミングと情報セキュリティに関する内容の充実が挙げられている．さらに高等学校では，情報科において情報Ⅰを新設し，すべての生徒がプログラミ

2024年1月29日受付　2024年2月23日受理

ングに加え，情報セキュリティを含むネットワークやデータベースの基礎についての学習が挙げられている．

また，AI戦略等を踏まえたAI人材の育成［5］について，文部科学省は，デジタル社会の基礎知識として「数理・データサイエンス・AI」を「読み・書き・そろばん」的な素養であり，その素養をすべての国民が育み，様々な分野で人材が活躍する環境を構築する必要がある，としている．この教育改革に向けた主な取り組みとして，新学習指導要領の下，全ての高等学校卒業生に対して「理数・データサイエンス・AI」の基礎的リテラシーを習得すること挙げている．さらに大学では文理を問わずAIリテラシー教育を推し進めることとしている．さまざまな技術とAIを掛け合わせて，新たな価値を創出するためにも，AI人材の育成がますます重視されている．

このような背景の中，プログラミング教育とAIリテラシー教育の充実が求められている．

1.2 本研究の目的

本研究では，プログラミング初学者が，AIを利用したプログラム作成に取り組むための教材を開発することを目的とする．ここで，プログラミング初学者としては，Scratch［6］によるビジュアルプログラミングを経験した後に，Pythonの初歩を学ぶ段階を想定する．

ScratchでのAIプログラミング環境としてStretch3がある．Stretch3では，拡張ブロックとして機械学習の実行や画像分類を行う機能が用意されている［7］．また，Teachable Machineで作成したモデルを読み込んで利用することも可能となっている．

我々は，このStretch3に近い操作感でPythonでのAI利用プログラミングを可能にするライブラリの開発を目指す．このような方針のもと，これまでに姿勢推定を簡便に実施するPythonモジュールを開発してきた．今回は，TensorFlow Hub［8］の学習済みモデルを用いた画像分類や，Teachable Machineで作成したオリジナルモデルを用いた画像分類を簡単に実行できるPythonモジュールの開発を行った．

2. 設計と開発

2.1 使用環境

Scratchを用いてプログラミングを学んだ後に，次の学習ステップとして，コードを入力してプログラミングを行う場合，さまざまな選択肢がある．たとえば，p5.js［9］は，Web上でJavaScriptのコーディングと実行を可能にするシステムである．setup( )とdraw( )の2つの関数を作成することで，ブラウザ上で簡単にグラフィカルな出力を伴うプログラムを作成できる．その為，GUI作成のハードルが低く，初学者でもアニメーション処理を含んだプログラムをすぐに作成，実行できるため，視覚的な変化を楽しみながら学習できるという利点がある．

また，p5.jsはJavaScriptのライブラリであり，AIプログラミングを行う場合，TensorFlow.js［10］が利用できる．このTensorFlow［11］は，機械学習を行うための汎用的なライブラリであるが，機能が多岐に渡り複雑であるため，初学者にとっては利用のハードルが高い．しかし，JavaScriptでのAIプログラミングを支援するライブラリとしてml5.js［12］がある．ml5.jsは，TensorFlow.jsを利用しやすくするラッパーライブラリとなっている．このため，p5.jsとml5.jsを利用することで，初学者でもAIプログラミングをブラウザ上で行うことが可能である．

一方で，Pythonはデータサイエンスや機械学習で広く用いられているプログラミング言語であるため，AIリテラシー教育の中で扱う言語としても適していると言える．このPythonで，p5.jsのように簡単にGUIを含むプログラミング環境を提供するライブラリとしてPygame Zero［13］がある．

Pygame Zeroは，p5.jsと同様にdraw( )とupdate( )の2つの関数を記述するだけでアニメーション処理を含んだプログラムをすぐに作成できるという利点がある．また，Scratchの次のステップの学習も意識されており，公式チュートリアルに「Scratchからの移行」が解説されている［14］．そこで本研究においても，このPygame Zeroを活用してサンプルコードを作成している．

しかし，Pythonには，JavaScriptにおけるml5.jsのような，初学者に対してTensorFlowを扱いやすくするライブラリがない．TensorFlowには，学習済みのモデルとしてTensorFlow Hubが公開されているため，この既存モデルを用いるだけであれば，プログラミング経験者であれば比較的容易に利用できる．しかし，プログラミング初学者にとっては，TensorFlow Hubの利用はできても，TensorFlowで利用されるNumPyライブラリのデータ構造（ndarray）の操作に対する障壁がある．さらに，オリジナルの学習モデルを利用するにはさらに困難が予想される．

以上より，初学者に対するAIを利用したプログラム作成のハードルを下げるためには，TensorFlowの複雑さや，NumPyデータの操作を隠蔽し，Stretch3の拡張ブロックに近い感覚でPythonでのプログラム作成に移行できるようなモジュールを作成することが，重要であると考えた．

2.2 作成クラス

本研究で開発するモジュールのクラス図を**図1**に示す．今回のシステムは，EtoolClassifyとEtoolModel，及びEtoolCameraの3つのクラスで構成されている．EtoolCameraはWebカメラを利用して画像を取得するクラスで，以前開発したものを利用している［15］．

EtoolClassifyは，TensorFlow Hubによる画像分類モデルを利用して，画像データに写るものが何かを推測するクラスである．EtoolClassifyでは，オブジェクトを生成する際に，TensorFlow Hubから学習済みモデルをダウンロー

ドし，セットアップを行っている．初期化時に，利用画像のソース（EtoolCamera経由かファイル経由か）及び，取得する推測結果の数，モデルの種類を指定できるようにした．EtoolClssifyクラスのclassify( )メソッドは，EtoolCameraで取得した画像データ，もしくは画像ファイルのパスを指定すると，対応する画像分類を実行し，推測結果を取得できる．

EtoolModelは，Teachable Machine［16］の学習データを利用できるクラスである．Teachable Machineは，オリジナルの学習モデルをプログラミング初学者でも容易に作成できるサイトである．Teachable Machineでオリジナルの学習モデルを自ら作成し，これを利用することで，AI学習の幅を広げることができる．EtoolModelでは，オブジェクト生成時に，Teachable Machineの学習で得られるモデルと対応するラベルのパスを指定する．classify( )メソッドでは，EtoolClassifyと同様に，EtoolCameraオブジェクトから取得した画像データ，もしくは画像ファイルのパスを指定して画像分類を実行できる．

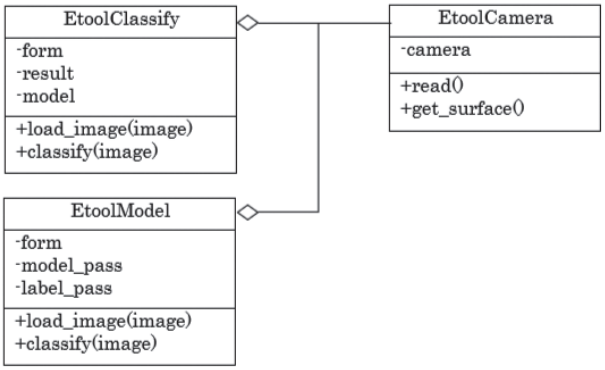

図1　作成したモジュールのクラス図

3. 結果

本研究で開発したPythonモジュールEtoolClassifyクラスとEtoolModelクラス，及び作成済みのクラスであるEtoolCameraを利用したサンプルプログラムを作成した．サンプルプログラムを**図2**と**3**に示す．いずれのサンプルも，画像表示などのGUIにPygame Zeroを利用している．

図2のサンプルプログラム1は，EtoolCameraのWebカメラを入力画像として，画像分類を行うプログラムである．1から6行目までで必要なモジュールのインポートと画面サイズの設定を行っている．本研究で作成したモジュールは2行目のetoolclassifyである．

8・9行目ではそれぞれEtoolClassifyとEtoolCameraのオブジェクトを生成している．10行目では，Webカメラの画像を取得している．EtoolCameraの初期化では，特に指定がなければ，EtoolCameraから取得した画像を処理するように設定している．さらに，EtoolClassifyの初期化では，TensorFlow Hubから学習済みモデルをダウンロードしている．特に指定がなければ，efficientnetv2-sモデルを利用するが，キーワードmodel_nameでTensorFlow Hubに用意されている画像分類モデルを指定可能である．しかし，ここで特に重要な点は，オブジェクトを用意するだけで画像分類モデルの利用準備が完了するという簡易さである．

13行目では，10行目で取得したWebカメラの画像データを，classify( )メソッドに引数として渡している．この画像データは，OpenCVのVideoCaptureから取得されたものであるが，classify( )メソッドの中身で適切にデータ変換して画像分類を実行している．classify( )メソッドの戻り値である画像分類の推測結果とWebカメラの画像データが，それぞれresultとimageに代入される．resultは，分類結果（デフォルトでは上位5つのクラス名と確度）を格納したリストである．

14行目は，13行目で取得した画像データimageをスクリーンに張り付けている．imageに代入されているデータは，PygameのSurfaceオブジェクトとなっており，Pygame Zeroの画面にblit( )関数を用いて直接貼り付けることができる．15行目は，同じく13行目で取得した画像分類の推測結果resultをスクリーンに表示している．

以上，EtoolClassifyクラスを利用した画像分類実行のサンプルプログラムを示した．ここでのポイントは，EtoolClassifyのオブジェクト生成のみで画像分類の準備が完了することと，学習モデルで利用する画像データ（NumPyのndarrayデータ）に対する前処理を気にせず利用できる点であり，見た目のコード量を減らすことができる．

次に，EtoolModelを利用した画像分類のサンプルを示す．**図3**のサンプルプログラム2は，一定インターバル（2秒ごと）にWebカメラの画像を取得し，その画像の分類結果をスクリーンに表示する．分類にはTeachable Machineの学習モデルを利用している．このため，事前の準備として，Teachable Machineの画像プロジェクトを用いて，オリジナルの学習モデルを作成しておく必要がある．さらに，作成したモデルをkeras.h5モデルに変換し，ダウンロードする．ダウンロードファイルは，学習モデルのデータを表すkeras_model.h5ファイルと，クラスのラベルを表すlabels.txtファイルとなる．これらを適切なフォルダに保存しておく．

図3の8，9行目では，Teachable Machineで作成した学習モデルとクラス名ファイルのパスを指定している．続いて，11，12行目では，EtoolModelオブジェクトとEtoolCameraオブジェクトを生成している．ここでもEtoolClassifyと同様に，オブジェクト生成時の初期化で，学習モデルのセットアップが完了している．

```
1  import pgzrun                    #Pygame Zero のインポート
2  import etoolclassify as ei       #画像分類推測ツール
3  import etoolcam as ec            #Web カメラ読み込みツール
4
5  WIDTH = 860    #生成する画面の幅(pixcel)
6  HEIGHT = 540   #生成する画面の高さ(pixcel)
7
8  classifyEstimate = ei.EtoolClassify()      #EtoolClassify オブジェクト生成
9  webcam = ec.EtoolCamera(0, WIDTH, HEIGHT)  #EtoolCamera オブジェクト生成
10 webcam = webcam.read()                     #画像を取得
11
12 def draw():
13     result, image = classifyEstimate.classify(webcam)      #推測結果と画像を取得
14     screen.blit(image, (0, 0))                              #画像を貼り付ける
15     screen.draw.text(str(result), (50,400), color=(255,255,255))  #推測結果を表示
16
17 pgzrun.go()    #Pygame Zero の実行
```

図2　サンプルプログラム1

```
1  import pgzrun
2  import etoolcam as ec
3  import etoolclassify as ei
4
5  WIDTH = 860
6  HEIGHT = 540
7
8  model = R"C:¥Users…¥keras_model.h5"   #学習モデルの読み込み
9  label = R"C:¥Users… ¥labels.txt"      #クラス名ファイルの読み込み
10
11 modelEstimate = ei.EtoolModel(1, model, label)     #EtoolModel オブジェクト生成
12 webcamEstimate = ec.EtoolCamera(0, WIDTH, HEIGHT)  #EtoolCamera オブジェクト生成
13
14 def classifier():
15     webcam = webcamEstimate.read()                  #Web カメラの画像を取得
16     result, image = modelEstimate.classify(webcam)  #推測結果と画像を取得
17     screen.clear()                                  #前の画像を消す
18     screen.blit(image, (0,0))                       #画像を貼り付ける
19     screen.draw.text(str(result), (50,400), color=(255,255,255)) #推測結果を表示
20
21 def draw():
22     clock.schedule_unique(classifier, 2) #2 秒ごとに classifier を呼び出す
23
24 pgzrun.go()#Pygame Zero の実行
```

図3　サンプルプログラム2

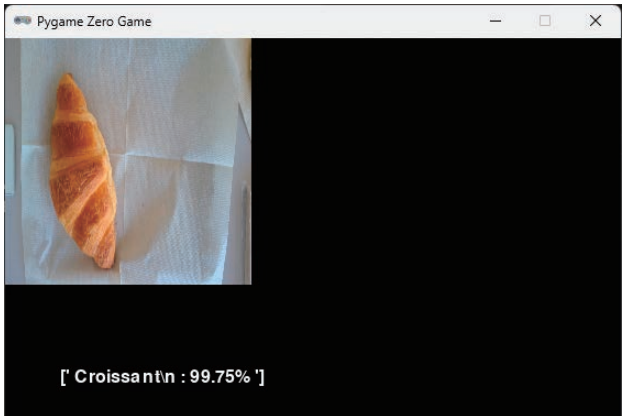

図4　サンプルプログラム2の実行結果画面

図6　Stretch3を用いたプログラム例

```
1   from keras.models import load_model
2   from PIL import Image, ImageTk
3   import tkinter as tk
4   import numpy as np
5
6   # 画像の準備
7   image = Image.open(R"C:¥Users…¥image.jpg").convert("RGB")
8   tk_img = ImageTk.PhotoImage(image)
9
10  # Window の準備
11  root = tk.Tk()
12  canvas = tk.Canvas(root,width=300, height=300)
13  canvas.pack()
14  canvas.create_image(0,0, anchor=tk.NW, image=tk_img)
15
16  # 学習モデルの読み込み
17  np.set_printoptions(suppress=True)
18  model = load_model(R"C:¥Users…¥keras_model.h5", compile=False)
19  class_names = open(R"C:¥Users…¥labels.txt", "r", encoding="utf-8_sig").readlines()
20
21  # 分類のための前処理
22  data = np.ndarray(shape=(1,224,224,3), dtype=np.float32)
23  size = (224, 224)
24  image = image.resize((size, resample=Image.BICUBIC)
25  image_array = np.asarray(image)
26  nomalized_image_array = (image_array.astype(np.float32) / 127.5) -1
27  data[0] = normalized_image_array
28
29  # 画像分類の実行
30  prediction = model.predict(date)
31  index = np.argmax(prediction)
32  class_name = class_names[index]
33  confidence_score = prediction[0][index]
34
45  # 分類結果の表示
46  confidence_100 = '{:.2%}'.format(confidence_score)
47  result = str(class_name[2:]) + str(confidence_score)
48  label = tk.Label(root, text=result, font=("Helvetica, 15))
49  label.pack()
40
41  root.mainloop()
```

図5　Teachable Machineのモデルを利用するサンプルプログラム

15行目は，Webカメラの画像を取得し，16行目では，その画像データを，classify( )メソッドに引数として渡している．画像分類の結果は，classify( )メソッドの戻り値である画像分類の推測結果とPygameのSurfaceデータに変換された画像データが，それぞれresultとimageに代入される．

16から18行目では，スクリーンを黒にリセットした後に，11行目で受け取った画像データをスクリーンに貼り付け，resultに代入されている画像分類の推測結果をスクリーンに表示している．

21行目のdraw( )関数はPygame Zeroで定期的に画像を表示する関数である．この中で，Pygame Zeroに組み込まれているclockオブジェクトを利用し，9行目のclassfier関数を2秒ごとに呼び出し続けている．このようにして定期的に画像を更新しながら，Webカメラに映された画像を分類し続けるアプリを作ることができる．

サンプルとして，Teachable Machineでパンの種類を見分ける学習モデルを作成した．この学習モデルはクロワッサン，メロンパン，カレーパン，サンドイッチ，明太フランスの5つと，Webカメラに何も映っていない6つのクラスを見分ける．この学習モデルを利用して，サンプルプログラム2を実行した画面が**図4**である．クロワッサンをWebカメラに写したところ，クロワッサンである確率が99.75%という推測結果が得られている．

図5に，比較のため，EtoolModelクラスを用いずに，Teachable Machineの学習モデルを利用して画像分類を実施する場合のサンプルプログラムを示す．このプログラムは，Teachable Machineで学習モデルをkeras.h5モデルに変換する際に，モデルを使用するコードスニペットとして表示されるサンプルをベースに，tkinterでGUIを作成して分類結果を表示するようアレンジしたものである．また，分類する画像はファイルから読み込むようにしている．

画像の準備から，分類結果の利用まで，**図3**のサンプルプログラム2に比べるとコードの量が多くなっている．特に，NumPyデータの前処理作業は，初学者にとってハード

ルが高いものと言える．

さらに，Stretch3との比較を行うため，**図6**にTeachable Machineの学習結果を利用するStretch3プログラムのサンプルを示す．Stretch3では，TM2Scratch拡張機能を追加することで，Teachable Machineのモデル利用できる．学習済みモデルを「画像分類モデルURL」ブロックでURLで指定して読み込み，「画像を分類する」ブロックを指定するだけで画像分類を実行できる．今回作成したEtoolModelでは，オブジェクトの生成が「画像分類モデルURL」ブロックに，classifyメソッドが「画像を分類する」ブロックに対応すると言える．Windowや画像表示の扱いに差はあるが，Scratchのブロックに近い感覚でプログラミングを行うことができる．

4. まとめ

本研究では，TensorFlow Hubの学習済みモデルや，Teachable Machineで作成したモデルを用いた画像分類を簡便に実行するためのPythonモジュールを作成した．特に，Scratchを経験した学習者が，Pythonに移行しやすいようにPygame Zeroでの利用が可能なモジュールとした．

作成したPythonモジュールを用いることで，TensorFlow Hubを直接利用する場合や，Teachable MachineをKeras経由で利用する場合に比べ，モデルの準備をシンプルにし，画像の処理においてNumPyのndarrayデータを直接利用しなくて済むようになるため，初学者にとってAIを利用したプログラムを容易に作成できることを確認した．

今後は，画像分類以外のモデルについて類似する操作性で利用できるライブラリを開発していくこととする．

参考文献

[1] 総務省,「令和5年版情報通信白書」,(2023)

[2] 文部科学省,「教育の情報化に関する手引-追補版- 第2章 情報活用能力の育成」,(2020/6)

[3] 文部科学省,「平成29・30・31年改訂学習指導要領」, https://www.mext.go.jp/a_menu/shotou/new-cs/1384661.htm,(2024/2/7)

[4] 文部科学省,「新学習指導要領について」, https://www.mext.go.jp/b_menu/shingi/chousa/shisetu/044/shiryo/__icsFiles/afieldfile/2018/07/09/1405957_003.pdf,(2024/2/7)

[5] 文部科学省,「AI戦略等を踏まえたAI人材の育成について」,(2019)

[6] Scratch, https://scratch.mit.edu/,(2024/2/7)

[7] Junya Ishihara(2020), "Stretch3", https://github.com/stretch3/stretch3.github.io, (2024/2/7)

[8] TensorFlow Hub,「TensorFlow Hub」, https://www.tensorflow.org/hub,(2024/2/7)

[9] p5.js, https://p5js.org/,(2024/2/7)

[10] TensorFlow,「TensorFlow.js」, https://www.tensorflow.org/js,(2024/2/7)

[11] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Rafal Jozefowicz, Yangqing Jia, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Mike Schuster, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng., "TensorFlow: Large-scale machine learning on heterogeneous systems", Software available from tensorflow.org, (2015)

[12] ml5.js, https://ml5js.org/,(2024/2/7)

[13] Daniel Pope and Richard Jones(2015), Pygame Zero, https://github.com/lordmauve/pgzero,(2024/2/7)

[14] Pygame Zero,「Scratchからの移行」, https://pygame-zero.readthedocs.io/ja/latest/from-scratch.html, (2024/2/7)

[15] 八木 徹, 山口 敏和, "MoveNetによる姿勢推定を簡便に実行するためのPythonモジュールの開発", Invormatio Vol. 20, p69-74,(2023)

[16] Teachable Machine, https://teachablemachine.withgoogle.com/,(2024/2/7)