Scratch を用いた 初学者向け機械学習体験システムの構築

八木　　徹*・山口　敏和**

要　約

深層学習を含む機械学習が多くの人にとって身近なものとなり，各々の目的に活用するツールとなっていくことを想定した場合，機械学習を理解して活用できるようにするための教材が重要となる。本研究では，機械学習を体験し，学ぶためのシステムに求められる機能について考察した。さらに Scratch 3.0 の拡張機能を活用し，TensorFlow や ml5.js といったフレームワークを用いて Scratch で機械学習を行うための拡張機能ブロック（Blocks for Machine Learning）を開発した。具体的には，画像分類を行う BML IC（Image Classifier）のほか，分類器に k 近傍法を用いる BML KNN（k-Nearest Neighbor），転移学習を実行する BML TL（Transfer Learning）という 3 種類の機能を作成した。さらに，これらの機能を機械学習の体験に活用する方法についても検討をおこなった。

キーワード：深層学習，機械学習，画像分類，Scratch，TensorFlow，ml5.js

1. はじめに

現在，深層学習が大きく注目されている。そのきっかけとして，2012 年の画像認識コンテスト（ILSVRC）が挙げられる。この時，カナダのトロント大学のチームは，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）を用い，画像識別能力の大幅な改善に成功した[1]。

深層学習は，機械学習の一手法に位置付けられるが，特徴量表現の学習が可能という，それまでの機械学習にはない機能を備えている。機械学習においては入力データを表現する特徴量が重要であり，いかに良い特徴量を作り出せるかがその性能を左右する。深層学習以前の機械学習では，特徴量の設計は人の手に委ねられており，いわば職人技のような技術であった。しかし深層学習では，特徴量表現も学習の一部に組み込むため，学習の結果として最適な特徴量が得られるようになった。このため深層学習は，これまでの機械学習にはない力を発揮する手法となっている。現在，深層学習を利用した画像認識は，人と同等以上の認識精度を達成している[2, 3]。

深層学習は，学習能力と予測精度が高いことから，応用範囲が広く，今では社会にも影響を与える存在となっている[4]。このため，例えばコンピュータを活用する能力が，専門を問わず誰にも求められる基礎技術として認識されているのと同じように，機械学習とその応用が多くの人にとって身近なものとなり，自分の目的に活用するツールになっていく可能性も考えられる。そのような想定をした場合，深層学習を理解して使えるようにするための教材や，（人の）学習が重要なものとなる。

現在，深層学習を含めた機械学習の成果は，主にインターネットを通じて一般の人も体験するこ

2019 年 11 月 30 日受付
* 江戸川大学　情報文化学科教授　情報科学，物理化学
** 江戸川大学　情報文化学科講師　情報教育

とが可能となっている。例えば，AI関連技術を体験するWeb上のサービスとしては，Google社によるExperiments with Googleに，AI Experimentsがある[3]。このサイトでは機械学習そのものは実施しないが，画像や言語，音楽などの分野で機械学習の成果がどのように利用されるかを体験することができる。

また，業務上の具体的な問題解決をおこなうために，深層学習を活用するサービスも提供されている。Amazon SageMaker[4]やIBM Watson[5]，Sony Neural Network Console[6]など，多くの企業が参入し，多様な問題に対処するための様々な提案を行っている。

プログラミング環境としては，深層学習を実行できる機械学習フレームワークが多数提供されている（TensorFlow[7]やChainer[8]，Caffe[9]，theano[10]など）。これらのフレームワークを利用すれば，用途に応じた深層学習の環境を短期間で構築することができる。しかし，フレームワークを用いたソフトウェア開発には，フレームワーク自体の知識に加え，プログラミングの技術，深層学習の理論や実施方法までを含めた幅広い力が必要とされる。

ここまでに示したように，深層学習の成果を体験するだけであれば，誰もが利用できる環境がすでに存在する。しかし，データを集め学習を行い，何らかの問題を解決するために深層学習を活用するという一通りの流れを体験することは，初学者にとってはまだ敷居が高いといえる。

深層学習を理解し，自分の目的に利用できるようにすることを目標としたとき，初学者は何をするべきであろうか。また，なるべく多くの人が簡便に利用できる深層学習の体験環境としては，どのようなシステムが必要とされるであろうか。本論文では，これらの点について考察を行い，深層学習を含めた機械学習に関する体験環境として，Scratch 3.0の拡張機能を用いたシステムを構築した。

2. 深層学習の体験教材

本節では，深層学習を体験する上で重要となる点を検討し，本研究で開発するシステムの目標を設定する。次いで，その目標を実現する方法としてScratchによる体験環境の構築を検討する。

2-1. 深層学習の体験に必要なこと

深層学習を含めた機械学習では，事前に用意したデータから法則性を抽出し，得られた法則を利用して未知のデータに対する回帰や分類といった予測を行う。深層学習の理論を理解するためには，用いられている数学や計算のモデルを知る必要がある。しかし，このような理解をいきなり深めていくことは難易度が高く，万人に受け入れやすい手法とは言いにくい。

一方，理論の理解だけでなく，深層学習を実施する過程で何を行うのかや，何が必要とされるのかを知ることも重要である。すなわち，学習データの準備，学習の実施，学習結果を利用した予測，予測結果の活用という一連の流れを体験し，深層学習で行われていることの全体像を把握することである。

画像の分類を行う場合，深層学習に関連する作業のポイントを以下のように整理できる。

1. 学習用の画像データを用意する
2. 用意した画像データを用いた学習を実行する
3. 深層学習のハイパーパラメータを調整する
4. 学習結果を用いて画像の分類を実行する
5. 分類結果を利用するプログラムを作成する

また，深層学習による画像分類ネットワークを一からすべて構築するためには，大量の画像データが必要となり，簡易的な体験は難しい。比較的少ない画像データを用いて精度良い学習を行う方法として，転移学習という手法が広く用いられている。転移学習は，CNNの特徴量抽出部分のパラメータを固定し，分類器のみのパラメータを最

適化するという手法である。

そこで，深層学習を初めて学ぶ体験システムとして，転移学習による画像分類ネットワークを自分で構築でき，上記 1 から 5 のポイントを実現する機能を持つシステムを想定する。

2-2. Scratch による深層学習体験環境

前節で述べたポイントには，深層学習の結果を利用したプログラムの作成が含まれているが，前提知識をなるべく少なくし，多くの人が利用できるようにしたい。つまり，プログラミングの経験や知識が少ない人でもすぐに利用できるプログラム言語を利用することが望ましい。

Scratch はビジュアルプログラミング言語として修得が容易であり，幅広く利用されている[13]。その Scratch の 1 機能（=1 つのブロック）として，「機械学習を行う」や「画像を分類する」といったブロックを作り，他のブロックと同じように Scratch のプログラム作成に利用できれば，多くの人にとって敷居が低く使いやすい深層学習体験環境となりうる。

機械学習を実現する拡張版の Scratch として，すでに ML2Scratch（Machine Learning to Scratch）が公開されている[14]。ML2Scratch は，Scratch3.0 の拡張機能として作成されており，TensorFlow およびそのラッパーライブラリである ml5.js を利用して画像認識機能を実現している。

ML2Scratch では，Scratch のカメラ機能を利用して画像を撮影し，学習と分類に利用できる。分類器としては k 近傍法を用いているため高速に動作し，Scratch の起動後に撮影した画像ですぐに学習を実行できる。さらに，Scratch であるため，分類結果を利用したソフトウェア（例えばカメラで映した画像に反応して動作するゲームなど）を簡単に作成することができる。プログラミング・機械学習両方の初学者にとって最適なシステムといえる。このように，ML2Scratch は，テンポよい開発が可能で，学習者を飽きさせず，楽しく活用できる非常に優れたシステムとなっている。

一方，前節で挙げた深層学習体験のポイントを考えた場合，ML2Scratch には実装されていない項目も存在する。そのような点として，

(1) k 近傍法以外の機械学習モデルの利用
(2) ハイパーパラメータ調整
(3) 学習用の画像データの再利用

が挙げられる。そこで，これらの項目に対応するために必要となる点を検討する。

項目(1)に対しては，分類器にニューラルネットワークを用いた転移学習が行えるとよい。項目(2)については，転移学習で設定が必要となるハイパーパラメータの調整機能を導入する。例えば，ニューラルネットワークの隠れ層のユニット数，学習率，エポック数，バッチサイズなどの調整は，学習時の基本的な項目であり，これらの調整が学習結果にどのように影響を与えるのかを見ることも有意義である。項目(3)について，ML2Scratch の学習画像データは，k 近傍法のデータとして内部的に登録されるため，後から再利用することはできない。しかし，同じ画像データでモデルやパラメータを変えた学習を行い，比較するためには，画像データを再利用できるようにしたい。

これら 3 項目への対応は，体験システムをより複雑なものとしてしまう。転移学習を行う場合，必要となる画像データの枚数は k 近傍法よりも増え，ニューラルネットワークの学習にも時間がかかる。ハイパーパラメータの調整には試行錯誤が必要で，何度も学習を繰り返すこととなる。画像ファイルの管理も手間となりうる。このような煩雑なシステムは，ML2Scratch と比べて軽快さに欠けるものとなる恐れがある。しかし，少しじっくりと取り組むことを前提にすれば，この 3 項目を取り込むことで，現在広く行われている深層学習のポイントを体験できるシステムを作ることができる。

また，複雑化に伴う親しみにくさを少しでも緩和するために，なるべく単純な処理から始め，次第に複雑な作業に移行するような体験ができると

よい。このため，学習を伴わない画像分類から始め，転移学習の実行まで段階を踏めるようにする。

このような機械学習体験を Scratch で実施することに意義があると考え，本研究では Scratch 3.0 の拡張機能を用いて，ハイパーパラメータの調整を含めた転移学習を可能とするオリジナルブロックを開発することとした。また，転移学習機能に加え，k 近傍法や事前に最適化した学習済みネットワークを用いた画像分類など，複数のモデルの学習と予測を行うシステムとしている。このため，「深層学習」よりも幅広いカテゴリを表す「機械学習体験システム」という表現を用いることとした。

3. Scratch 機械学習用ブロックの開発

本研究で開発した機械学習体験システムは，Scratch 3.0 を基礎とする 3 種類の拡張機能（ブロック群）を備えている。共通の名称を Blocks for Machine Learning の略である BML とした。3 種の拡張機能は以下のとおりである。

(1) BML IC : Blocks for Machine Learning (Image Classifier with pre-trained model)
学習済みのネットワークを用いて画像分類を行う拡張機能

(2) BML KNN : Blocks for Machine Learning (k-Nearest Neighbor)
k 近傍法を用いた機械学習と画像分類を行う拡張機能

(3) BML TL : Blocks for Machine Learning (Transfer Learning)
転移学習の実施と画像分類を行う拡張機能

上記 3 種類の機能を備えることにより，複数の手法による画像分類の体験を，段階を追ってできるようにした。

BML IC では，ILSVRC2012 のデータセット (1000 種類のクラスに対応付けた 120 万枚の画像)[17] を用いて事前に学習したネットワークを使い，画像分類をすぐに実施できる。BML KNN は，k 近傍法による高速な機械学習とその結果を利用した画像分類を行う。BML TL では，転移学習の実行と各種ハイパーパラメータの調整をし，得られた結果を用いて画像分類を行うことができる。このように，学習に用いるデータや学習方法の違いによって画像分類がどのように動作するのかを比較し，機械学習への理解を深めるツールとする。

図 1 は，Scratch 3.0 で拡張機能を選択する際の画面である。通常は Scratch 3.0 に標準で搭載されている拡張機能のアイコンが並ぶが，それに加え今回開発した 3 種類の機能を選択するアイコンが表示されている。このアイコンを選択すると，対応する拡張機能のブロックが Scratch のブロックメニューに追加される。

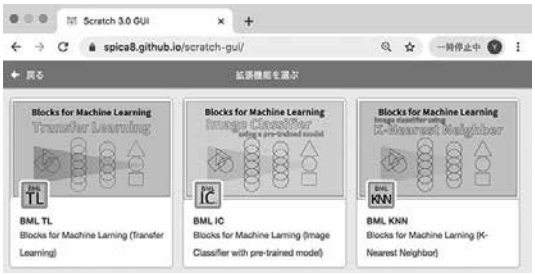

図 1 Scratch 3.0 拡張機能選択画面

開発の概要を以下に示す。本システムでは機械学習のフレームワークとして TensorFlow と ml5.js を利用している。拡張機能の BML IC には，内部処理として ml5.js の ImageClassifier クラスを利用し，学習済みモデルとしては，MobileNet V2 を用いている。MobileNet は，Google 社の Andrew G. Howard らによって開発された畳み込みニューラルネットワークの一種である[18,19]。学習に ISVRC2012 のデータを用いているため，分類されるクラス名は英語で出力される。

拡張機能 BML KNN では，画像の特徴量抽出に MobileNet V1 を用い，分類器として k 近傍法を用いた機械学習と画像分類が実施できる。この機能は ML2Scratch と同等のものである。ブロック内部では ml5.js の featureExtractor 及び KNN-

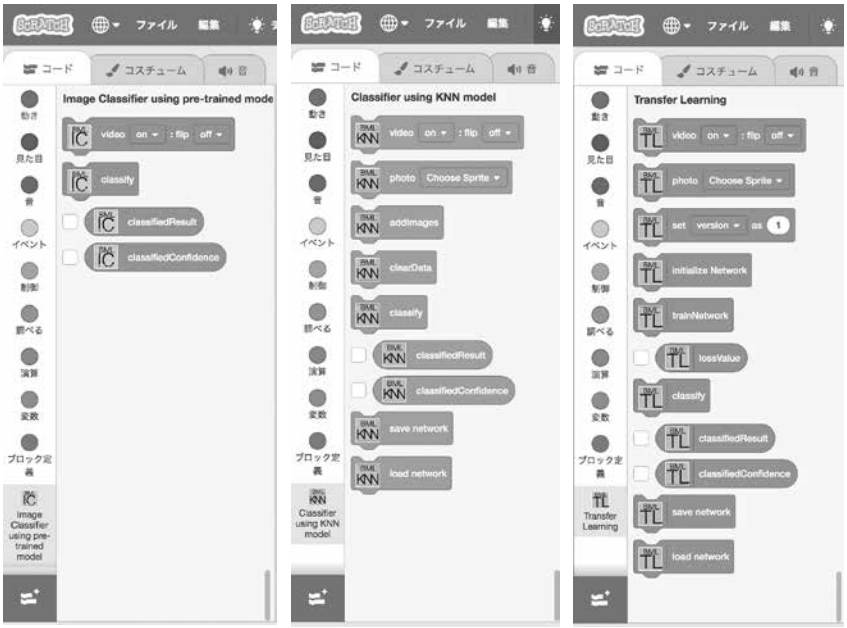

図 2　開発した拡張機能で追加されるブロック
左から順に BML IC，BML KNN，BML TL のブロックを示す。

Classifier を利用している。

拡張機能 BML TL の転移学習では，画像の特徴量抽出に KNN と同じ MobileNet V1 を利用し，分類器にはニューラルネットワークの全結合層を用いている。転移学習は，ml5.js の featureExtractor によって実施する。損失値の計算には交差エントロピー法を用いている。

4. 機械学習ブロックの詳細

本節では，今回開発した 3 種類の拡張機能に含まれるブロックの詳細を示す。初めにそれぞれの拡張機能に共通するブロックについて述べ，次いで各機能固有のブロックを解説する。

図 2 に，拡張機能で追加されるブロックの一覧を示す。BML IC では学習を行わないためブロック数が最も少なく，BML KNN は画像データの保存と学習を実行する分のブロックが増えている。BML TL は，ハイパーパラメータの設定やネットワーク初期化，損失率の表示と言った機能のため，最も多くのブロックを含んでいる。

4-1. 各 BML 共通のブロック

表 1 に，複数の拡張機能で共通する役割のブロックを示す。このうち，「photo（Choose sprite）」は BML KNN と BML TL だけに用意されているブロックである。それ以外は 3 つの拡張機能すべてに共通するブロックとなっている。

表 1　複数の拡張機能に共通するブロック

ブロック	動作内容
video（on/off）: flip（on/off）	ビデオ on/off と画像反転 on/off
photo（Choose sprite）	画像を撮影し，選択したスプライトのコスチュームに画像を追加する （BML KNN，BML TL のみ）
classify	画像分類を実行
classifiedResult	分類結果のクラス名を保持
classifiedConfidence	分類結果の確信度を保持

4.1.(a) 画像分類の実行

画像分類は「classify」ブロックにて実行する。このブロックは，3 種類すべての拡張機能で用意され，ブロックの意味は共通して「画像分類の実行」である。しかし，具体的な動作はすべて異なる。BML IC の「classify」ブロックは，学習済みネットワークにインターネット経由でアクセスして分類結果を取得する。BML KNN と BML TL の場合は，それぞれ k 近傍法か転移学習の学習結果に基づいて分類を行い，結果を取得する。このため，BML KNN と BML TL の「classify」ブロックを実行するためには，事前の学習が必要である。

「classify」ブロックを使う際には，「video（on/off）: flip（on/off）」ブロックでビデオを ON にする必要がある。「classify」ブロックを実行すると，その瞬間のビデオ画像を取得し，画像分類の入力データとする。

分類結果は「classifiedResult」ブロックに格納される。このブロックは Scratch の値ブロックであり，分類されたクラス名を文字列として格納する。また，画像分類は非同期処理であり，分類完了前に処理が次に移る。このため，画像分類処理の完了が判別できるようするため，「classify」ブロックを実行した直後に「classifiedResult」ブロックは “undefined” を返し，分類が終了した後に，クラス名を返すようにした。

もう一つの値ブロック「classifiedConfidence」には，入力画像がそのクラスに分類される確からしさを表す確信度（ソフトマックス関数の出力値）が格納される。

画像分類を実行し，「classifiedResults」ブロックから分類結果を取得するためのサンプルプログラムを図 3 に示す。「classify」の非同期処理中に何か別の処理を行う場合と，分類処理終了を待つ場合に応じて使い分ける。

4.1.(b) 画像の撮影と学習データの保存

学習を行う際には，学習データとしての「画像」と，その画像に写っているものの名前，つまり画像の「答え」を示す必要がある。このように

(a) 分類の完了を待つ場合

(b) 分類処理中に何らかの処理を実行する場合

図 3 classifiy ブロック実行後に，結果を取得するサンプルプログラム

(a)分類の完了を待つ場合と，(b)分類処理中に何らかの処理を実行する場合で使い分ける。

データとその答えを紐づけた学習を，教師あり学習と呼ぶ。BML KNN と BML TL の学習ではこの教師あり学習を実施する。

本システムでは，学習データの保存に Scratch のスプライトを利用することとした。スプライト名を画像データの名称（画像の答え）とし，そのスプライトのコスチュームとして画像データを保存する。これにより，画像（＝コスチューム）とその画像の名称（＝スプライト名）を対応付けることが可能となる。

スプライトのコスチュームに画像データを登録する方法は 2 通りある。一つはあらかじめ用意した画像をスプライトのコスチュームとしてアップロードする方法と，もう一つは BML の「photo（Choose sprite）」ブロックを利用する方法である。ここでは後者について示す。

「photo（Choose sprite）」ブロックを利用する前に，「video（on/off）: flip（on/off）」ブロックでビデオを ON にすることと，画像のクラスを示

(a)　空のスプライト

(b)　画像が保存されたスプライト

図 4　スプライトの準備と画像保存
(a)は，空のスプライトを用意し，それぞれ対応する名称を設定したもの。(b)は「photo（スプライト名)」ブロックで画像を登録した後のもの。

すスプライトを用意しておくことの 2 点が必要となる。図 4-(a)は，空のスプライトを用意し，それぞれに「正面」「右」「左」と対応するゼスチャの名称を付けたものである。

「photo（Choose sprite)」ブロックでは，Choose sprite のプルダウンメニューからスプライト名を選択する。Choose sprite のメニューは，スプライトを作成すると自動的に更新される。スプライト名を選択後，「photo（スプライト名)」となったブロックを実行すると，現在映っているビデオ画面のスナップショットを撮り，対応するスプライトのコスチュームに画像を保存する。

図 5 に，撮影を行うプログラムのサンプルを示す。このプログラムでは，押すキーを「a」「b」「c」と変えると，それぞれ「正面」「右」「左」のスプライトにキーを押した瞬間の画像を保存する。プログラムを記述する場所は任意であるが，特定のスプライトに依存する処理ではないので，Scratch のステージのプログラムとするのが妥当と言える。

撮影後に画像がコスチュームとして保存されたものが図 4-(b)である。スプライトの「表示する」オプションを「非表示」にすると，ステージ画面には画像が表示されない。また，スプライトのコスチュームタブを選択すると，保存した画像の一覧を確認できる。

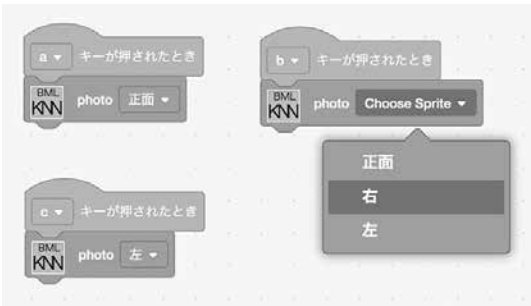

図 5　画像撮影を行うプログラム例

4-2. BML IC のブロック

BML IC のブロック一覧を表 2 に示す。BML IC は，これまでに示した共通ブロックだけで構成されており，「画像撮影」→「分類実行」→「分類結果取得」という最も単純な処理を行うようになっている。データの学習は不要で，すぐに画像分類を行うことができる。

表 2　BML IC に含まれるブロック一覧

ブロック	動作内容
video（on/off)： flip（on/off)	ビデオ on/off と画像反転 on/off
classify	画像分類を実行
classifiedResult	分類結果を保持
classifiedConfidence	分類結果の確信度を保持

4-3. BML KNN のブロック

BML KNN に含まれるブロックの一覧を，表 3 に示す。BML KNN に特有のブロックは「addImages」と「clearData」である。「addImages」ブロックは，Scratch のスプライトに登録したコスチュームを読み込み，1 スプライトを 1 つの分類クラスとして k 近傍法の学習を行う。「clearData」ブロックは，k 近傍法の学習に追加した画像情報を消去し，学習を初期化する。学習をやり直す際や，同じ画像情報の重複登録を避ける場合に用いる。

表 3　BML KNN に含まれるブロック一覧

ブロック	動作内容
video（on/off）: flip（on/off）	ビデオ on/off と画像反転 on/off
photo（Choose sprite）	画像を撮影し，選択したスプライトのコスチュームに画像を追加する
addImages	スプライト名を分類クラスとし，対応する画像（コスチューム）を用いて k 近傍法の学習を行う
clearData	学習で登録した画像情報を初期化する。
classify	画像分類を実行する
classifiedResult	分類結果を保持する値ブロック
classifiedConfidence	分類結果の確信度を保持する値ブロック
save network	学習結果をファイルに保存する
load network	save network で保存したファイルを読み込む

BML KNN では，分類するクラスの数を指定するパラメータは無く，学習時に存在するスプライトを全て分類クラスとして利用する。このため，学習時に余分なスプライトが存在してはならない点に注意が必要である。分類実行時の k 近傍法のパラメータ（k 値）は ml5.js のデフォルトである k=3 を用いている。

学習結果はファイルに保存し，いつでも再利用できる。「save network」ブロックを用いると，学習結果を json 形式のファイル（myKNN.json）として保存する。次に Scratch を起動した際に，「load network」ブロックを用いてこのファイルを読み込むことで，保存した学習結果を利用した画像分類を行うことができる。

4-4. BML TL のブロック

MBL TL に含まれるブロックの一覧を，表 4 に示す。BML TL に特有のブロックとして，ハイパーパラメータの設定を行う「set（params）as（val）」ブロックと，ネットワークを初期化する「initialize Network」，及び学習を実行する「train Network」ブロックがある。

表 4　BML TL に含まれるブロック一覧

ブロック	動作内容
video（on/off）: flip（on/off）	ビデオ on/off と画像反転 on/off
photo（Choose sprite）	画像を撮影し，選択したスプライトのコスチュームに画像を追加する
set（params）as（val）	ネットワークと機械学習のパラメータを設定する
initialize Network	パラメータ設定後にネットワークを初期化する
train Network	ネットワークの学習を行う
lossValue	損失値を保持する値ブロック
classify	画像分類を実行する
classifiedResult	分類結果を保持する値ブロック
classifiedConfidence	分類結果の確信度を保持する値ブロック
save network	学習結果をファイルに保存する
load network	save network で保存したファイルを読み込む

「set（params）as（val）」ブロックで設定可能なハイパーパラメータの一覧を表 5 に示す。（params）の選択肢として，learningRate（学習率），hiddenUnit（隠れ層ユニット数），epoches

（学習のエポック数），numLabels（分類するクラス数），batchSize（バッチサイズ）が選択可能である。

1つのブロックでは1つのパラメータの値が設定でき，複数のブロックを組み合わせることにより，ハイパーパラメータを個別に設定できる。分類するクラスの数は numLables で指定するが，学習時には，numLabels の値とスプライトの数を一致させる必要がある。

表5 「set（params）as（val）」ブロックで設定可能なハイパーパラメータの一覧

params の選択肢	内容
learningRate	転移学習時の学習率を設定する。初期値は 0.0001
hiddenUnits	隠れ層のユニット数を指定する。初期値は 100
epoches	学習のエポック数を指定する。初期値は 20
numLabels	分類するクラス（ラベル）の数を指定する。初期値は 2
batchSize	学習時のバッチサイズを指定する。学習データに対する割合を，0 から 1 の数値で示す。初期値は 0.4

ハイパーパラメータ設定後は，「initializeNetwork」ブロックを実行し，ネットワークの初期化を行う。この処理により，プログラム内部では，指定されたユニット数の中間層を持つニューラルネットワークの全結合層が作られる。

ネットワークを初期化したのちに，「train Network」ブロックを用いて転移学習が実行できる。学習中の損失値は「lossValue」ブロックに格納される。また，学習が完了する前に「classify」ブロックを実行すると，「classifiedResults」ブロックに “notLearned” が設定される。この値を監視することで，学習ステップが完了したか否かを判定できる。

学習結果の保存は「save network」，データの読み込みは「load netowork」ブロックにて行う。保存されるファイルは，model.weights.bin と model.json ファイルの2つになる。「load network」ではこれら2つのファイルを指定する必要がある。

5. BML 拡張機能の利用例

BML の3つの拡張機能それぞれについて，簡単なサンプルプログラムとともに利用例を示す。

5-1. BML IC の利用

BML IC を利用したプログラムのサンプルとその実行の様子を，それぞれ図6と7に示す。画像の分類プログラムは任意の場所に記述できるが，この例では分類結果をセリフとして表示するスプライトに記述することを想定している。「classify」ブロックが実行された時，画像分類の結果が得られるまでの待ち時間で，スプライトは「うーん」と考えるポーズをとる。分類結果が得られると，分類されたクラスの名称を吹き出しに表示する。

また，classifiedConfidence の値に応じてメッセージを変えている。確度が高いと自信をもって「ですね！」と言い，確度が低いと「かなぁ」と迷

図6　BML IC を用いたサンプルプログラム

図 7　BML IC サンプルプログラム実行の様子

いながら答えるようにしている。このように，画像分類の結果を格納する値ブロック（classifiedResults と classifiedConfidence）の値に応じて動作を変えるプログラムが容易に作成可能である。

5-2. BML KNN の利用

BML KNN で k 近傍法の機械学習を実行するためには，事前に学習用の画像データを用意する必要がある。ここでは，学習用画像データの準備は前述の図 5 プログラムにて実行し，図 4 のスプライトに対応する画像が保存されているものとする。

学習実行のプログラムを図 8 に示す。BML KNN の学習は，「addImages」ブロックを一つ実行するだけである。「addImages」実行後は，存在するスプライトの画像データを自動的に読み込んでいく。このサンプルでは，手を下ろした画像と指の向きの画像を撮影して，正面と左右の方向を認識するモデルを作成している。プログラムの記述は，図 5 の画像撮影を行うプログラムと同じ場所でよい。

図 8　k 近傍法の学習を実行するプログラム
スペースキーを押すとスプライトの画像を読み込み，学習する。

学習結果を利用するサンプルプログラムとその実行の様子を図 9 及び図 10 に示す。このプログラムは，ゼスチャによりスプライトを動かすもので，対応するスプライトに記述する。指の左右で対応する方向に向きを変え，手を下ろした正面画像では向きを変えずに直進のみを行う。このサンプルは目標物（例えば動き回るネズミなど）を追いかけるゲームなどに応用することができる。

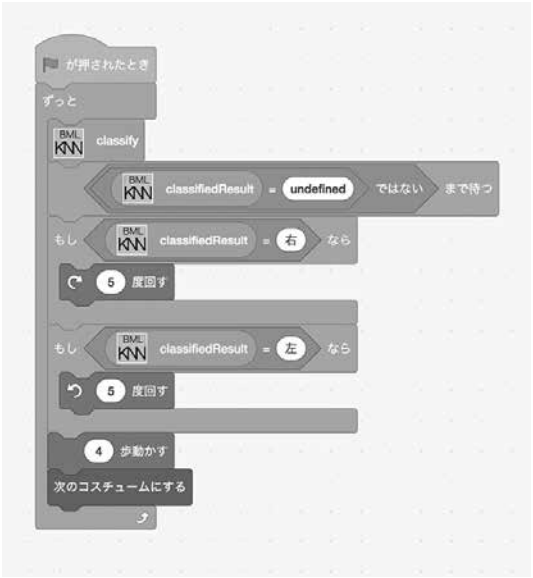

図 9　BML KNN ブロックを用いたサンプルプログラム

図 10　BML KNN サンプルプログラム実行の様子

5-3. BML TL の利用

BML TL で転移学習を行う場合，学習データの準備方法は BML KNN のそれと同じである。ただし，画像の枚数は BML KNN の場合よりも多くする必要がある。

図 11 に BML TL の転移学習を行うサンプルプログラムを示す。学習のステップであるため，ステージのプログラムとして記述することを想定している。図中の(a)の領域がハイパーパラメータを設定している箇所である。パラメータ名をプルダウンメニューにて選択し，適切な値を記述する。ここでは 4 つの値を個別に設定している。

ハイパーパラメータ設定後は「initializeNetwork」ブロックで初期化を行い，「trainNetwork」ブロックで学習を実行する（図中(b)）。学習時間は BML KNN よりも長くかかるため，このサンプルでは学習の完了を知らせるプロセスを記述している（図中(c)）。学習中に「classify」ブロックを実行すると，「classifiedResult」ブロックの値が "notLearned" となることを利用し，学習の終了を待っている。背景画像として「学習中」と「学習完了」の 2 種類の画像を用意し，切り替えることによって，学習の終了を判別することができるようにしている。

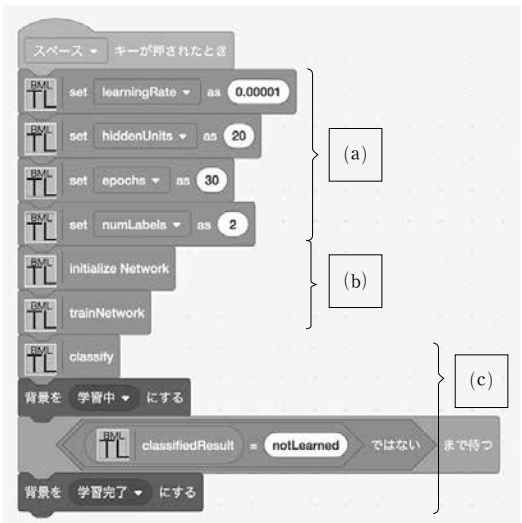

図 11　BML TL による学習ステップのサンプルプログラム

値ブロック（BML TL の場合，「lossValue」，「classifiedResult」 及 び「classifiedConfidence」の 3 つ）は，ブロックのリストをチェックすることにより，ステージ上に値を表示することが可能である（図 12)。「lossValue」ブロックを選択しておくと，学習中の値の推移を見ることができる。

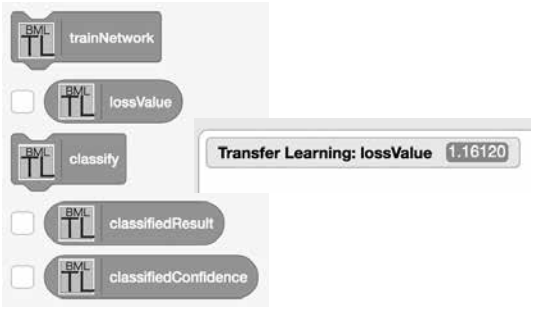

図 12　値ブロックの選択とステージ上の表示

一般的な深層学習では，学習データと検証データの両方について損失率の推移を求め，学習が正常に行われていることと，過学習となる手前で学習を止める確認に用いる。また，ハイパーパラメータを調整した学習を繰り返して最適な値の組み合わせを探索する。

今回開発した BML TL では，用意するデータは学習データのみであり，損失率はこの学習データに対して計算している。検証データの確認や，学習過程での損失率のグラフ化までは行っていない。このため，ハイパーパラメータを調節して，過学習におちいることの無いネットワークの最適解を見つけるという作業を体験することは困難である。現状では，より単純に，ハイパーパラメータを調整することで損失率や分類結果にどのように影響が出るのかを見る，という教材として位置づけるべきである。

本体験システムで，転移学習機能を充実させていくためには，ハイパーパラメータの最適化をサポートするブロックのような，機能の充実を図る方法が考えらえる。この点は今後の目標として，継続的な開発をおこなっていく。

BML TL にて転移学習を行った後には，分類

結果を活用するプログラムを作成できる。ブロックの利用法は BML KNN などと同等であるため，ここではサンプルプログラムは割愛する。

5-4. BML 拡張機能の活用

ここまで，Scratch のブロックとして開発した機械学習体験システムの詳細について示してきた。学習の体験としては，(1) BML IC での画像分類，(2) BML KNN での簡便な機械学習と画像分類体験，(3) BML TL での転移学習体験，の順で次第に複雑なモデルを扱うことを想定している。

BML IC 画像分類の体験では，分類結果について ILSVRC2012 のデータセット[17] と比較するとよい。分類クラスのリストを確認し，どのような画像で学習しているのかを見て，分類した画像と学習画像を比較することも機械学習の性質を知る上で有意義である。

BML KNN では，素早い機械学習が可能であるため，様々な画像での学習を試し，どのような画像を何枚程度用意すれば画像が分類できるのか，あるいは背景が変わるなどの条件の変化にどの程度対応できるのかなどを体験することができる。スプライトに画像を保存し，繰り返し利用できるため，学習データを調整して何度も機械学習を繰り返し，結果を比較検討することができる。

BML TL を用いた体験では，KNN と同じ学習データを用いてモデルの違いによる結果を比較できる。体験教材として利用するためには，精度良い分類が可能な学習データやハイパーパラメータのセットをあらかじめ用意しておくことが望ましい。そのうえでデータの内容を変えたり，ハイパーパラメータを調整することが，学習結果や分類結果にどのよう影響するかを体験する教材とする。

機械学習体験システムとしては，学習結果の活用も重要である。この点については Scratch であることの強みが生かせる。画像分類がブロック一つで実施できるため，Scratch のプログラムに機械学習の成果を取り込むことができ，アイデア次第で様々なプログラムを作成するツールとなる。

6. まとめ

特徴量表現の学習が可能な深層学習は，予測精度が飛躍的に向上し，様々な分野で応用されている。深層学習を含めた機械学習が多くの人にとって身近なものとなり，自分の目的に活用するツールとなるという可能性も考えられる。このような状況において，機械学習を理解するための教材は重要であり，学習に伴う一連の作業を体験し，学習結果を活用するプログラム作成ができる機械学習体験システムは，今後さらに重要性を増していく。

本研究では，機械学習体験システムに求められる機能について考察した。さらに Scratch3.0 の拡張機能と TensorFlow，ml5.js といった機械学習フレームワークを活用し，Scratch で機械学習を行うためのブロックを開発した。画像分類のみを行う BML IC のほか，分類器に k 近傍法を用いる BML KNN，転移学習を実行する BML TL の 3 種類の機能を作成し，サンプルプログラムとその活用方法を示した。

今後は，転移学習の機能を充実させるよう本システムを改良するとともに，具体的な体験教材や，学習で得られる画像分類機能を用いたプログラム開発を行い，機械学習体験システムの充実を図っていく。

Appendix

本研究で開発した Scratch による機械学習体験システムは，ソースコードを GitHub にて公開している[20]。また，BML IC，BML KNN，BML TL それぞれの拡張機能を利用可能な Scratch を GitHub Pages で公開している[21]。

参考文献

［1］ A. Krizhevsky, I. Sutskever, and G.E. Hinton, “ImageNet classification with deep convolutional neural networks”, *Proc. NIPS*, pp. 1097-1105, (2012)

［2］ O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, , S. Ma , Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, “ImageNet Large Scale Visual Recognition Challenge”, *arXiv:1409.0575*, (2014)

［3］ K. He, X. Zhang, S. Ren and J. Sun, “Delving deep into rectifiers: Surpassing human-level performance on imagenet classification”, *Proc. ICCV*, pp. 1016–1034, (2015)
［4］ 統合イノベーション戦略会議, “AI 戦略 2019”, (2019)
この資料では,「数理・データサイエンス・AI」に関する知識・技能を, デジタル社会の基礎知識と位置づけている。また,『文理を問わず, 全ての大学・高専生が, 課程にて初級レベルの数理・データサイエンス・AI を修得』や『文理を問わず, 一定規模の大学・高専生が, 自らの専門分野への数理・データサイエンス・AI の応用基礎力を修得』という目標を掲げている。
［5］ Google, “AI Experiments | Experiments with Google”, https://experiments.withgoogle.com/collection/ai, (2019/11/13)
［6］ Amazon.com Inc., “Amazon SageMaker”, https://aws.amazon.com/jp/sagemaker/, (2019/11/13)
［7］ IBM, “IBM Watson”, https://www.ibm.com/watson/, (2019/11/13)
［8］ Sony Corporation, “Neural Network Console”, https://dl.sony.com/, (2019/11/13)
［9］ Tensorflow, https://www.tensorflow.org, (2019/11/13)
[10] Chainer, https://chainer.org, (2019/11/13)
[11] Caffe, https://caffe.berkeleyvision.org, (2019/11/13)
[12] Theano, http://www.deeplearning.net/software/theano/, (2019/11/13)
[13] Scratch, https://scratch.mit.edu/about/, (2019/11/13)
[14] ML2Scratch, https://champierre.github.io/ml2scratch/, (2019/11/13)
[15] TensorFlow, https://www.tensorflow.org, (2019/11/13)
[16] ml5.js, https://www.tensorflow.org, (2019/11/13)
[17] ImageNet, “Large Scale Visual Recognition Challenge 2012（ILSVRC2012）”, http://www.image-net.org/challenges/LSVRC/2012/, (2019/11/13)
ILSVRC2012 データセットのカテゴリ一覧は, 以下より参照可能
http://image-net.org/challenges/LSVRC/2012/browse-synsets/, (2019/11/13)
[18] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, H. Adam, “MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications”, *arXiv:1409.0575*, (2017)
[19] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, L. Chen, “MobileNetV2: Inverted Residuals and Linear Bottlenecks”, *arXiv:1801.04381*, (2018)
[20] T. Yagi, https://github.com/spica8/, (2019.11.13)
[21] T. Yagi, https://spica8.github.io/scratch-gui/, (2019.11.13)