H-018

判別属性をパラメータで制御可能な物体判別器

Object discriminator with the parameter control of discrimination attribute

桑原 大輔† 中島 克人†
Daisuke Kuwahara Katsuto Nakajima

1. はじめに

深層学習手法は，2012 年に AlexNet[1]が火付け役となって以来，様々な分野で使用される技術となっている．この技術のメリットは，従来の人間が特徴量抽出などを設計していた手法とは異なり，モデルが自動で特徴量抽出方法を学習することで，判別問題などで従来手法よりも高精度を出すことを可能としていることである．課題としては，膨大な学習時間，学習対象が膨大なあるいは複雑なタスクの場合の学習困難性，学習したタスク以外を分類することの困難性，多大なモデルのデータ容量といったものがある．

本研究では「学習したタスク以外を分類することが困難」という深層学習の課題に着目する．例えば，2 枚の画像のそれぞれに写る椅子が同色の同一製品か否かの判別をタスクとして学習した判別器は，色の違いを無視して同一製品かどうかを判別できるとは限らないため，後者の目的のためには別途学習したモデルを作成しなくてはならない．

ある判別問題（メインタスク）において，それが判別の部分問題（サブタスク）に分解できる関係にあるとき，我々はメインタスクのために学習されたネットワークの途中を流れる判別のための特徴量の一部を制御することで，メインタスクが内包するサブタスクを実行できると考えた．

本研究の実験では，深層学習の 1 手法であり 2 つの入力の相違を判別する Siamese Network[2] (以降 SN と呼称)に対して，同色同形状の Lego Brick を陽性として学習させ，その後，我々が提案する Selective Element Network(以降 SEN と呼称)というサブネットワークに，異色同形状を陽性として学習させたものを SN のある層に追加し，精度や学習時間等を評価したので報告する．

2. 関連研究

2.1 物体検出のための深層学習手法

深層学習を用いる物体判別ネットワークには様々なものがあるが，その中でも高い精度を有するものとして GoogLeNet[3]，ResNet[4] 等が知られている．これらのネットワークは効率的な学習と精度向上の両立のために，Inception モジュールや ResBlock といった応用的かつ複雑な構造を持つ．一方，AlexNet は深層学習ブームの最初期に提案されたネットワークであるが，非常にシンプルな構造である事から，深層学習の基礎研究分野では今でも多く使用されている．

2.2 Siamese Network

SN は，2 つの入力の類似度を求める深層学習ネットワークであり，図 1 に示すように特徴量抽出ネットワーク層(2 つの入力が画像の場合は CNN 層)，差分計算層，類似度算出ネットワーク層(以降 FC 層と呼称)の 3 種の層から構成されている．図中の SEN は我々が挿入するサブネットワークであり，詳細は後述する．SN は 2 つの入力画像を，ネットワーク重みを共有する 2 つの CNN 層にそれぞれ入力し，その出力の差分を差分計算層で求め，FC 層で分類し，最終的に 2 入力の類似度を出力する．

† 東京電機大学 Tokyo Denki University

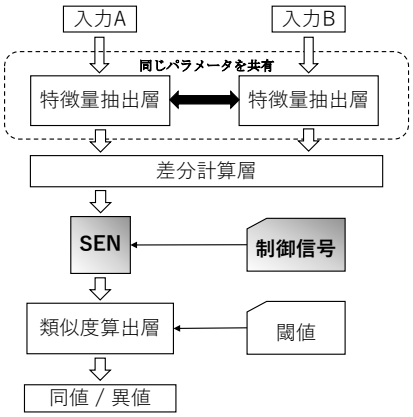


図 1 SEN を挿入した Siamese Network の構成図

3. 提案手法

3.1 Selective Element Network

判別のメインタスクを実行するための学習済ネットワークが特徴量抽出層とその特徴に基づく判別を行う層からなる場合，我々はこれらの層間で特徴量制御を実施し，このネットワークがサブタスクを実行できるようにするために，SEN と称するサブネットワークを挿入する．

SEN はサブタスクごとに学習が必要ではあるが，メインタスクのためのネットワーク本体よりも十分に小さいため，学習の手間と時間は少なく済む事が期待できる．SEN はネットワークの途中出力において，目的のサブタスク以外のデータが含まれている情報を阻害することによって，サブタスクの処理を行うというアプローチに基づいて挿入する層である．図 1 において，SEN は SN における差分計算後に挿入しているが，挿入箇所やその数に制約はない．

SEN は，[0,1]の範囲の任意数の制御信号を入力とする．各制御信号にはサブタスクをそれぞれ 1 つ割り当てる．各入力は SEN が持つ重みに乗算され，サブタスク以外の要素を抑制するように機能する．

SEN の構成は式(1)の通りである．

$$SEN(x_{in}, a_0, \dots, a_n) = x_{in} \odot \left(Clip\left(\sum_{i=0}^{n} a_i Norm(\omega_i) \right) + f\left(\sum_{i=0}^{n} a_i \right) \right)$$
$$Clip(x) = min(1, max(-1, x)) \quad (1)$$
$$f(x) = \begin{cases} U, & (x = 0) \\ O, & (otherwise) \end{cases}$$

ここで，x_{in}はSENの前レイヤの出力を，a_iは制御信号を，ω_iは SEN が持つ重みパラメータを，Uは要素が全て 1 のテンソル，Oは要素が全て 0 のテンソルを示す．また，$Norm(x)$は入力を[-1,1]の範囲に正規化する関数である．式(1)での，a_0からa_nは[0,1]の入力のみを受け取るスカラ値であり，それ以外の変数は全て同じ形状のテンソルである．

式(1)では，制御信号に対して対応した重みを乗じ，それをアダマール積でネットワークの途中出力と合成することで，途中出力からサブタスク以外の要素を阻害している．つまり，ω_iには特徴量の阻害に関するパラメータが格納されていることになる．

3.2 Selective Element Network の学習

SEN を備えるネットワークは次のように構築される．

① 任意のネットワークにメインタスクを学習させる

② ①のネットワークに SEN を挿入する

③ ②のネットワークにサブタスクの学習を実施

①は図 1 の SN に対するメインタスクの学習に相当する．このネットワークを以降バックボーンと呼ぶことにする．

②においては，バックボーン中の任意の層間に SEN を挿入できるが，実現したいサブタスクによって適した挿入位置が変わり得る．今回はメインタスクの判別属性の一部の特徴を無視する事により，サブタスクを実行できると考えられるため，バックボーンの特徴抽出層の後に SEN を挿入することにする（図 1 参照）．

③では，バックボーンの重みは固定し，SEN の重みのみをサブタスク用の学習データを用いて学習する．

図 1 の SN がメインタスクとして 2 枚の入力画像の同種・異種の判定をすると仮定すると，サブタスク用学習データは，入力として画像 2 枚と対応する制御信号を 1 つ，そして入力画像がサブタスクにおいて同種か異種かを示す教師ラベルを 1 組とするデータセットとする．

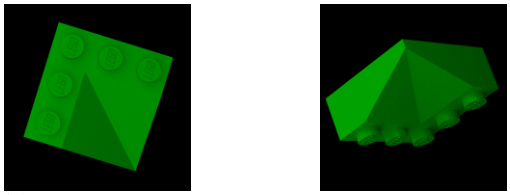
4. 実験

4.1 メインタスクとサブタスクのデータセット

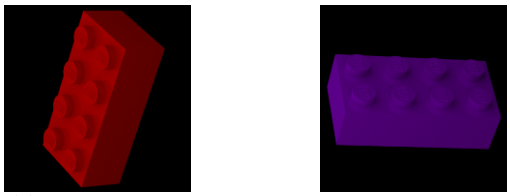
今回の実験では，Lego Brick Dataset[5]という 50 種類の Lego Brick に関して 800 視点からレンダリングされた計 40,000 枚のモノクロ CG 画像を持つデータセットの一部を加工して用いる．まず，Lego Brick 1 種類ごとに 20 視点の画像を選択し，それぞれを色相が等間隔で彩度を最高値にした 9 色に着色する．それで得られた 50 種×20×9＝9,000 枚の画像に対して，224px × 224px への画像サイズのリサイズと，画像の各画素値範囲を[0,1]へ正規化する処理を行う．これにより得られた画像を元に，メインタスク用の学習セットとサブタスク用の学習セットをそれぞれ作成する．

- メインタスク用のデータセット
  同色同形状の Brick の画像をペアリングしたものを陽性とするようにラベル付けして構成したもので，以後，同色同形状データセットと呼ぶ事にする．クラス数は 50 種×9 色＝450 であり，1 クラスの画像数は 20 枚としている．
- サブタスク用のデータセット
  色の違いを問わず，同形状の Brick 画像をペアリングしたものを陽性とするようにラベル付けしたもので，以後，異色同形状データセットと呼ぶことにする．クラス数は 50 であり，1 クラスの画像数は 180 枚としている．

図 2 にそれぞれのデータセットにおいて陽性とする画像の例を示す．



(a)メインタスク用データセットの同一クラス例



(b) サブタスク用データセットの同一クラス例

図 2　データセットの陽性画像例

4.2 使用ネットワーク

本研究で使用するネットワークは，CNN 層に AlexNet におけるそれの Kernel サイズを変更したものを，差分計算層に 2 ベクトルの減算を，FC 層に独自のネットワークを使用して構成した．表 1 に具体的な構造を示す．この表において SEN 層を除いた SN をバックボーンと称する．なお，SEN 層の Kernel サイズの 1 とは，本実験における制御信号(=サブタスク)の数を示し，Subtract 層の 2 は，CNN1 から Flatten までの 11 レイヤをそれぞれ経由してきた 2 つの入力を示す（図 1 参照）．

表 1　ネットワーク構造

layer	Input	Kernel	Stride	Actiation
CNN1	224x224x3	12x12x96	2	ReLU
LRN1	107x107x96	-	-	-
CNN2	107x107x96	5x5x256	1	ReLU
LRN2	103x103x256	-	-	-
MaxPool1	103x103x256	3x3	2	-
CNN3	51x51x256	3x3x384	1	ReLU
MaxPool2	49x49x384	3x3	2	-
CNN4	24x24x384	3x3x384	1	ReLU
CNN5	22x22x384	3x3x256	1	ReLU
MaxPool3	20x20x256	3x3	1	-
Flatten	9x9x256	-	-	
SEN	**20,736**	**20,736x1**	**-**	**ReLU**
FC1	20,736	20,736x4096	-	ReLU
FC2	4,096	4,096x4096	-	ReLU
FC3	4,096	4,096x1	-	Sigmoid

4.3 学習

後述のそれぞれの実験では表 1 に示すネットワークに対して，サブタスクデータセットを用いて次の 4 つの学習を行い，それぞれ評価した．

- 手法 A：スクラッチ学習
  - 未学習のバックボーンに対する学習
- 手法 B：ファインチューニング
  - メインタスクに関して学習済のバックボーンに対する学習

- 手法 C：FC 層の置換による転移学習
  - メインタスクに関して学習済のバックボーンの FC 層に対しのみの学習
- 手法 D：SEN の追加（提案手法）
  - メインタスクに関して学習済のバックボーンの途中に図 1 のように SEN を挿入し，SEN に対してのみ学習

表 2 に学習に使用したハイパーパラメータを示す．

表 2　学習に使用したハイパーパラメータ

項目	設定		
オプティマイザ	Adam[5]		
損失関数	Cross Entropy		
最大 Epoch 数	100		
バッチサイズ	16		
学習率	Epoch	手法 A, B, C	手法 D
	5 以下	10^{-4}	10^{-2}
	15 以下	10^{-5}	10^{-3}
	25 以下	10^{-6}	10^{-4}
	50 以下	10^{-7}	10^{-5}

4.4 実験内容

今回は以下の 5 つの実験を行った．

(1) サブタスクの精度評価実験

サブタスクデータセットの検証データに対する手法 A，B，C，D の学習結果の精度比較を行う．

(2) 学習速度比較実験

425 枚の画像を使用して手法 A，B，C，D に関してそれぞれ学習を 11 回実施し，そのうち初回を除いた 10 回の学習時間を測定する．初回を除くのは，ライブラリの読み込みや，OS のキャッシュの影響を排除するためである．

(3) 推論速度比較実験

(2)の実験結果として得た 4 手法の各ネットワークを使用して 280,128 ペアの推論をそれぞれ 11 回行い，初回を除く 10 回の平均推論速度を比較する．

(4) SEN の最適な数式検証実験

式(1)が適切なものであるかを確認するために SEN の別の構成として

$$SEN(x_{in}, a_0, \dots, a_n) = ReLU\left(x_{in} \odot \sum_{i=0}^{n} (a_i \omega_i + U)\right) \quad (2)$$

$$SEN(x_{in}, a_0, \dots, a_n) = x_{in} \odot \sum_{i=0}^{n} (a_i \omega_i + U) \quad (3)$$

の 2 種を用意し，実験(1)と同様にサブタスクの検証データに対する判別精度を比較する．

(5) SEN の入出力評価実験

提案手法である手法 D において，SEN の制御信号を 1 として入力した際の，SEN への入力と出力を取り出し，ヒートマップ表示して，SEN がメインタスクからサブタスクへの切り替えのために特徴マップの一部を変更する様子を確認する．

4.5 実験結果

各実験の結果は以下の通りである．

(1) サブタスクの精度評価実験

表 3 に各手法で学習したモデルの精度を示す．これらの数値は 100 Epoch までの学習途上において，検証データに対する判別精度が最大であった時のものを示す．なお，表 3 における「メインタスク」とは，バックボーンにサブタスクに関する学習を行わずにサブタスクの検証データを判別させた結果である．

表 3 より，平均精度が最高の手法は A であり，提案手法の手法 D はこれに 15 ポイント劣る．しかし，サブタスク未学習の「メインタスク」モデルより約 20 ポイント精度向上しており，サブネットである SEN の部分だけを学習するだけで，バックボーンをメインタスク判別からサブタスク判別に大きく役割変更させられる事が分かる．

表 3 における精度差は，学習で更新したパラメータ量と比例関係にあるとも考えられるため，手法 D の精度向上を行うためにはバックボーンへ挿入する SEN の層数増や，SEN 単体のパラメータ量の増加が有効であると考えられる．

(2) 学習速度比較実験

表 4 に各手法で 425 枚の画像を使用して学習時間の計測を行った結果を示す．表 4 より経過時間，ユーザ CPU 時間共に，手法 D が最短であることが分かる．

手法 C と D では，パラメータ変更をしない CNN 層部分の推論を学習時に毎回行うのは無駄と言える．そこで，学習前に各画像を CNN 層に推論させておき，推論結果を SEN や差分計算層の入力に用いるようにして学習できる．その結果を表 5 に示す．CNN 層までの推論を省略できるため，手法 A，B に比べて圧倒的に学習時間の短縮が図れることが分かる．

(3) 推論速度比較実験

表 6 に手法 A から D の各手法で 280,128 画像ペアの推論を行った結果を示す．

経過時間やシステム CPU 時間は推論処理以外の要素が大きく影響する．そのため，純粋な推論時間を確認する指標として，ここではユーザ CPU 時間を比較する．

手法 A と手法 B では推論時間が約 1,970 秒とほぼ同じであり，手法 C が 10 秒程度短く，SEN を挿入した手法 D は 10 秒程度長い．これは 280,128 画像ペア当たりの差であり，手法 C と手法 D の秒数差約 21 秒は，1 画像ペア当たり約

表 3　精度比較

手法 A	手法 B	手法 C	手法 D	メインタスク
99.49%	99.47%	91.70%	84.41%	55.04%

表 4　平均学習時間

手法	Elapsed	User CPU	System CPU
A	22,828 sec	18,708 sec	1,227 sec
B	23,079 sec	18,780 sec	1,424 sec
C	19,795 sec	15,489 sec	1,533 sec
D	18,829 sec	15,185 sec	886 sec

表 5　CNN 層出力を入力とした際の学習時間

手法	Elapsed	User CPU	System CPU
C	6,252 sec	6,250 sec	533 sec
D	5,942 sec	5,977 sec	532 sec

表 6　平均推論時間

手法	Elapsed	User CPU	System CPU
A	2,522 sec	1,969 sec	166 sec
B	2,553 sec	1,966 sec	200 sec
C	2,552 sec	1,959 sec	205 sec
D	2,452 sec	1,980 sec	107 sec

75μ 秒でしかない．このことから，SEN を挿入する事による推論速度の低下はわずかであると言える．

(4) SEN の最適な数式検証実験

表 7 に式(1)に代えて，式(2)，(3)を SEN の構成として学習した際のサブタスク判別精度比較を示す．学習のハイパーパラメータは，最大 Epoch 数を 10 とした事以外は表 2 と同じである．

なお，表 7 におけるメインタスクとは，表 3 におけるそれと同様，バックボーンにサブタスクに関する学習を行わずにサブタスクの検証データを判別させた結果である．

表 7 より，最も精度が高いのが正規化を利用した式(1)であり，式(1)のアダマール積の右辺を単純化した式(3)でも精度低下はそれ程大きくない事が分かった．一方，活性化関数として ReLU を使用した式(2)では精度低下が大きかった．これは，SEN が前層の出力に対して ReLU によってマイナスを通さないという大きな制限を設けたことによると考えられる．

(5) SEN の入出力評価実験

SEN のネットワークが真陽性，真陰性を推論した際の SEN の入出力値をヒートマップに変換したものを図 3 に示す．これは全チャネルの値に対して[1,0]の範囲で正規化を行った後，各チャネルに対して疑似カラーを設定し，全チャンネルを左上から右下に 2 次元に並べたものである．つまり，3 行 2 列目の画像は 16x(3-1)+2-1=33 個目のチャネルの出力結果を示す．

図 3 より，真陽性の場合，SEN は全体的に SEN へ入力された特徴的な値を中央値に近い値に変換していることが分かる．また，一部の値に関しては，中央値よりも小さな値に変換していることが分かる．真陰性の場合には，全体的に中央値寄りに値を変換しているのは同じだが，一部の値は中央値より大きな値に変換している．これらのことから，本ネットワークでは，中央値以下の値を FC 層に入力することで陽性が，中央値以上の値を入力することで陰性が出力されるのだと判断できる．

表 7　SEN を式(1)，(2)，(3)として学習した結果

式(1)	式(2)	式(3)	メインタスク
82.99%	75.21%	81.26%	54.58%

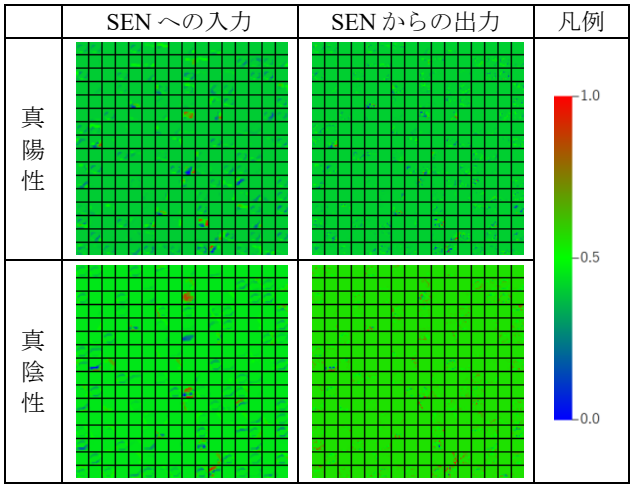


図 3　SEN の入出力ヒートマップ

5. 考察

5 章の各実験結果より，SEN の特徴として

- 少量パラメータでタスクの切り替えが可能
- 転移学習よりも高速な学習速度
- 推論速度にほとんど遅延をもたらさない
- 学習精度が他手法より 10 ポイントほど低い

ということが分かる．

SEN がタスクの切り替えに関して有意な結果をもたらすことができる理由としては，3.1 節で示した SEN のアプローチが意図通りに効果を発揮しているためだと考えられる．これは，図 3 でも推測することが可能である．

SEN の弱点としては，他手法と比べた際の精度の低さが挙げられる．これに対しては，SEN の利点を弱めることになるが，SEN のパラメータの増量や，SEN をバックボーンネットワークに複数個追加することで改善が可能だと考えられる．

6. まとめと今後の課題

本研究では，判別タスクを行うようなネットワークの途中層に SEN と称するサブネットワークを挿入することで，元々のタスクに内包されるサブタスクの実行に特化したネットワークを作成する手法を提案した．

実験では，同色同形状物体を陽性とするメインタスクデータセットを学習した AlexNet ベースの Siamese Network（SN）をバックボーンとし，色に関わらず同状形状物体を陽性とするサブタスクデータセットを学習させた SEN を挿入し評価を行った．その結果，メインタスクのみを学習した元々の SN にサブタスクを判別させた場合の精度に対して約 20 ポイントの精度向上を確認した．

SEN は，非常に軽量なネットワークであり，バックボーンに追加してサブタスク用のデータセットで学習することも短時間で済み，推論時のオーバヘッドも非常にわずかである．

サブタスク用データセットでバックボーンをスクラッチで学習する場合に比べ，判別精度は 10 数ポイント低下するが，メモリ容量が限られるエッジ端末や携帯機器などで，複数のタスクを切り替えて実行したい場合などの使い方が考えられる．

今後の課題としては，GoogLeNet や ResNet などをはじめとした様々なアーキテクチャをバックボーンとして SEN を追加した際の精度実験や，2 つ以上のサブタスクを切り替える場合の動作や精度の確認などを考えている．

参考文献

[1] A. Krizhevsky, et al., “ImageNet Classification with Deep Convolutional Neural Networks,” NIPS'12, 2012.
[2] G. Koch, et al., “Siamese Neural Networks for One-shot Image Recogntion,” ICML deep learning workshop 2015.
[3] C. Szegedy, et al., “Going Deeper with Convolutions,” CVPR, 2014.
[4] K. He, et al., “Deep Residual Learning for Image Recognition,” CVPR, 2016.
[5] J. Hazelzet, “Images of LEGO Bricks,” https://www.kaggle.com/joosthazelzet/lego-brick-images, 参照 2021/4.
[6] D. P. Kingma, J. Ba, “Adam: A Method for Sto-chastic Optimization,” arXiv:1412.6980, 2014.