広島県立総合技術研究所東部工業技術センター研究報告　No. 38（2025）
技術報文

深層学習による画像認識（第１報）

ビジョントランスフォーマの評価

廣川　勝久

Deep Learning for Image Recognition (I)

Performance Evaluation of Vision Transformer

HIROKAWA Katsuhisa

自然言語処理に対応した深層学習モデルの進歩は著しく、言語モデル用に開発された中核技術を画像認識へ適用した Vision Transformer が提案された[arXiv:2010.11929 (2020)]。本報告では、この Vision Transformer がパラメータ数の小規模なモデルでは少量の教師データからでも効果的に学習できることを示す。また、学習率の設定がモデルの収束速度や最終精度に大きく影響することを実験的に検証し、適切な学習率の設定の必要性を明らかにする。さらに、画像に付与されたラベルに応じて Attention の注目領域が異なることを可視化により確認し、ラベル属性ごとの学習難易度や収束性の違いについて考察する。

キーワード：Vision Transformer、Attention、Classification、Deep Learning

1. 緒　　言

近年、自然言語処理の分野においては、Attention 機構を基盤とする Transformer アーキテクチャが主流となっている [1)]。Transformer は ChatGPT や Gemini などの大規模言語モデルにも採用されており、その有効性が広く認識されている。Transformer は計算効率に優れており、膨大な情報を処理できると同時に、情報量やモデル規模の増加にも、コンピュータの計算能力が許す限り柔軟に対応可能であることが示されている [2)]。

しかしながら、Transformer は特定の帰納バイアスを持たない構造であるため、モデルの性能を十分に引き出すには、大規模な学習データが不可欠である。そのため、中小規模のデータセットに対して Transformer を適用する際は、事前に大規模データで学習されたモデルを用い、転移学習やファインチューニングを通じて再学習を行う手法が一般的に用いられている。

我々はこれまでに、Transformer アーキテクチャを画像認識に応用した **Vi**sion **T**ransformer: **ViT** [3)]の転移学習およびファインチューニングに関する検討を行ってきた。先行研究では、ViT は ResNet [4)]や DenseNet [5)]と比較してパラメータ数が少ないにもかかわらず、転移学習において最も高い性能を示した。一方、ファインチューニングにおいては、ViT のみが過学習に陥る傾向を示した [6)]。これは、最小構成の ViT モデルである vit_b_16 であっても、学習データに対して非常に高い学習能力を有していたためであると考えられる。

このような特性は、事前学習された ViT を他の画像認識に応用する際には、計算資源や学習データの量といった制約によって、応用範囲が限定される可能性がある。また、ViT のアーキテクチャを取り入れたモデルの構築や性能評価においても影響を及ぼす。そこで本報告では、ViT の高い学習能力を活かしつつ、より少ない計算資源とデータ量での学習を可能にすることを目的として、vit_b_16 より小規模なモデルを構築する。さらに、本モデルを完全初期化の状態から学習させ、適切な学習率の設定によって、少量のデータからでも効果的な学習が可能であることを示す。加えて、Attention スコアにより形成される注目領域が、学習の収束性や学習難易度に与える影響についても検討する。

2. Vision Transformer

2.1 Vision Transformer の概要

ViT は、自然言語処理において高い性能を示した Transformer アーキテクチャを、画像認識などの空間的タスクに適用することを目的として提案された手法である。Transformer は本来、入力系列を処理する Encoder と、処理結果から出力系列を生成する Decoder の 2 つの構成要素からなるが、ViT では主に Encoder 部分のみを用いて画像の特徴抽出および分類を行う。**図 1** に ViT の全体構造を示す。入力画像はまず、固定サイズの小さな矩形（パッチ）に分割され、1 次元のトークン列へと変換される。トークン列は Transformer の Encoder に入力

2025. 7. 1　加工技術研究部　受理

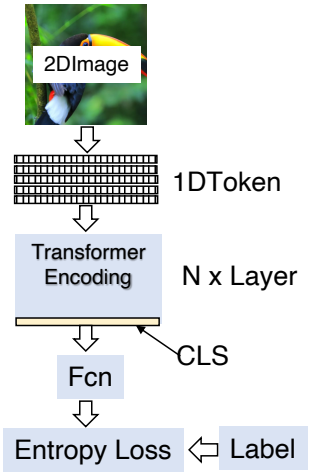


図 1　ViT の構造

され、Attention などの処理が行われ、最後にトークン(CLS: Classification)に含まれる情報のみを用いて、全結合ニューラルネットワークによりクラス分類が行われる。

2. 2 2 次元画像から 1 次元トークンへの変換

ViT では、入力画像を固定サイズの矩形（パッチ）に分割し、それぞれを 1 次元のトークンとして扱う。この処理は、パラメータの stride と kernel を同じ値：n に設定された 2 次元畳み込み層によって実現される（**図 2**）。この方法により 2 次元の矩形画像(n 画素×n 画素、c チャンネル)は、畳み込み層の出力チャネル数により所定の次元のベクトルからなる 1 画素の画像へと変換される（1 次元トークン）。得られた各トークンには位置埋め込みが加算され、さらにトークン(CLS)が先頭に付加された後、Transformer Encoder へと入力される。

2. 3 Transformer Encoder

Transformer Encoder は、各トークン間の関係性を学習するための Attention 機構と、その後段に配置される特徴抽出のための多層パーセプトロン（MLP）ブロックから構成される（**図 3**）。また、各ブロックをスキップする接続も用意されている。ViT では、これらの Encoder ブロックを複数層にわたり直列に適用し、入力トークンに対して繰り返し処理を行うことで、より高度な学習を可能としている。

2. 4 Attention 機構

Transformer における Attention 機構は、本手法の中核をなす技術であり、入力トークン間の依存関係を学習することで、局所的情報に限定されない広範な情報の統合を可能とする。特に ViT においては、画像空間上で物理的距離が離れていても意味的に関連する領域を、Attention により初期層から効果的に結びつけることができる。この点は、主に局所領域の処理に依存する畳み込みニューラルネットワーク（CNN）との本質的な相違点である。**図 4** に、その Attention 機構のフローチャー

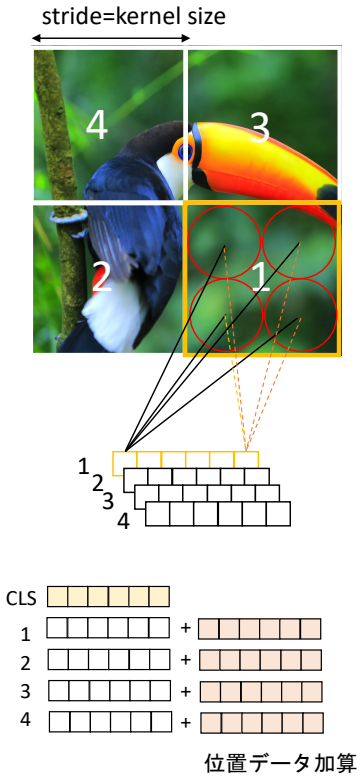


図 2　2 次元画像から 1 次元トークンへの変換

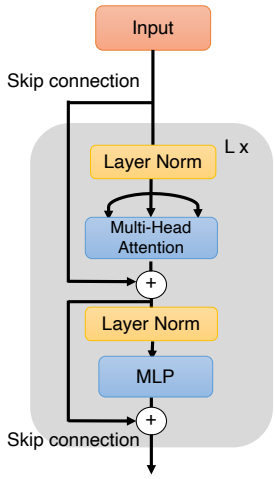


図 3　Transformer Encoder

トを示す。ViT では Self-Attention により、各トークンからそれぞれ Query（Q）、Key（K）、Value（V）の 3 つのベクトルを生成し、Query と Key の類似度に基づいて Attention 重みを計算し、その重みを Value に適用して出力を得る。Attention の計算を式で示すと

$$\mathrm{Attention}(Q,K,V) = \mathrm{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \qquad (1)$$

のように表せる。ここで、d_kは Key ベクトルの次元数

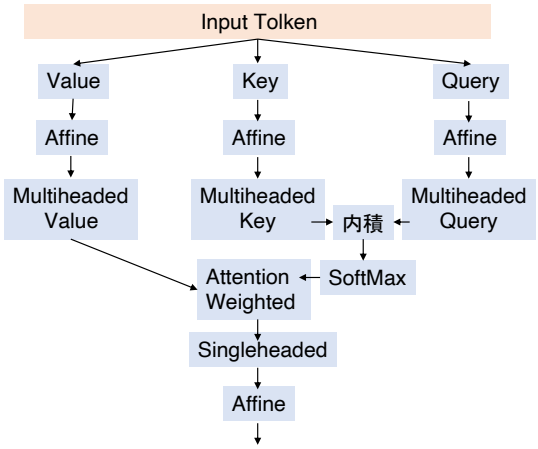


図 4　Attention 機構

である。なお Self-Attention 全てが同一の入力トークン列から導出されるため、そのままでは各トークンが自身に最も高い重みを与える。これを防ぐため、学習可能な affine 変換により 3 つのベクトルはそれぞれ異なる空間に射影される。これにより、各トークンは他のトークンとの関係性に基づいて各トークンがどの程度情報を受け取るかを決定する重みが定まり、それに Value を適用することで出力が得られる。

3. Vision Transformer の実装

3.1 学習データ

ViT の評価には、**MNIST** : **M**odified **N**ational **I**nstitute of **S**tandards and **T**echnology[7]と **CelebA**: Large-scale **Celeb**-Faces **A**ttributes[8]を用いた。MNIST では学習係数の違いよる ViT の収束速度や学習精度への影響を評価した。一方、CelebA においては、同一データセットを用いた学習におけるラベルの違いがモデル性能に与える影響について検証した。特に、本手法の中核をなす技術である Attention 機構とラベルとの関係を画像により確認した。

3.2 ViT の具体的実装

図 1 に示した ViT の全体の流れは、次のように簡単に実装できる。

```
def forward(self, x):
    x = self.input(x)
    x = self.dropout(x)
    for t in self.transformer:
        x = t(x)
    x = self.class(x)
    return x
```

Encoder を繰り返し実装するためには上記 self.transformer を下記のように定義する。

```
self.transformer=nn.ModuleList(
     [Encodeing(dropout = dropout)for _ in
     range(Layer_num)])
```

図 2 に示した 2 次元画像を 1 次元トークン列に変換する処理は、2 次元畳み込みモジュールを用いることで容易に実現できる。2 次元畳み込みの結果、出力はチャンネル数、画素の並びとなるため、これを 1 次元のトークン列として並び替える必要がある。以下に示すのは、その一連の処理を実装した例である。本報告では、CLS と位置埋め込みの初期値を 0 から 1 の一様分布からサンプルし、学習可能なパラメータとして定義した。

```
self.conv = nn.Conv2d(L1, L2,
       kernel_size=(Div_num, Div_num),
       stride=(Div_num, Div_num))
self.flat = nn.Flatten(start_dim=2,end_dim=3)
self.position = nn.Parameter(
       nn.init.uniform_(torch.empty(
       (1, Patch * Patch + 1, L2),
       device=device)))
self.cls = nn.Parameter(
       nn.init.uniform_(torch.empty(
       (1, 1, L2), device=device)))
def forward(self, img):
    batch = img.size(0)
    x = self.conv(img)
    x = self.flat(x)
    x = torch.permute(x, (0, 2, 1))
    cls = torch.repeat_interleave(
                self.cls, batch, dim=0)
    x = torch.cat((cls, x), dim=1)
    x += self.position
    return x
```

図 4 の Attention 機構のフローチャートに対応する実装例を次に示す。この例では、Multihead Attention を実装した。トークン列間の依存関係を確認するには、Attention スコアに対応する softmax 処理後の行列を観察することで可能である。

```
def forward(self, x):
    x=self.lynorm(x)

    query = self.fck(x)
    key   = self.fcq(x)
    Value = self.fcv(x)

    mh_dim = x.shape[2]//head_num
    query = query.reshape(
                  x.shape[0],x.shape[1],
                  head_num, mh_dim)
    key   = key.reshape  (
                   x.shape[0],x.shape[1],
                   head_num, mh_dim)
    Value = Value.reshape(
                   x.shape[0],x.shape[1],
                   head_num, mh_dim)

    query = torch.permute(query,(0, 2, 1, 3))
    key   = torch.permute(key,  (0, 2, 3, 1))
    Value = torch.permute(Value,(0, 2, 1, 3))

    tmp= (query @ key)/ mh_dim**0.5
    tmp= self.softmax(tmp)
    tmp= self.dropout(tmp)
    tmp = torch.matmul(tmp,Value)
    tmp = torch.permute(tmp,(0, 2, 1, 3))
    tmp = tmp.reshape(x.shape[0],
```

```
            x.shape[1],x.shape[2])
        tmp = self.fc(tmp)
        x   = self.dropout(tmp)
        return x
```

4. Vision Transformer の性能評価

4.1 学習係数による学習速度と学習精度評価

学習係数が ViT の学習速度および精度に与える影響について検証を行った。学習に用いた ViT モデルの主なパラメータを**表 1** に示す。本研究では、少量の学習データに対しても学習可能な、コンパクトな ViT モデルとした。MNIST データセットの 5 万枚の画像を用い、200 エポックにわたって学習を実施し、その際の学習曲線を**図 5** に示す。学習係数は 0.003、0.0003、0.00003 の 3 種類を設定し、最適化手法には RAdam を採用した。

一般に、学習係数が大きいほど学習は高速に進行する傾向があるが、本実験においては、学習係数 0.003 において最も学習速度が遅く、また損失も高いまま収束する結果となった。一方、0.0003 および 0.00003 の学習係数を比較したところ、学習データに対しては 0.00003 の方がより低い損失を達成したが、未学習データ（テストデータ）に対しては 0.0003 の方が優れた性能を示した。これらの結果は、適切な学習係数の選定が Vit モデルの学習に重要であることを示している。

表 1　ViT の主なパラメータ

	パラメータ
Datasets	MNIST
画素	32画素×32画素
ラベル数	10
学習データ数	5万画像
画像分割数	4×4
ヘッド数	4
Hidden size D	192
MLP size	768
レイヤー数	12
パラメータ数	5.3M

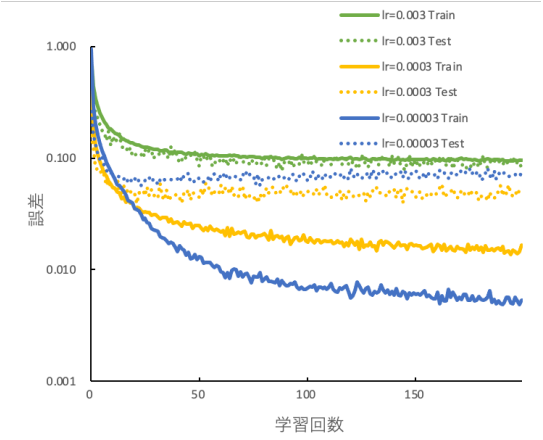


図 5　学習係数と学習速度

4.2 Attention 機構の動作検証

ViT が持つ Attention 機構が画像に対してどのように作用するかを検証するために、CelebA の画像に異なるラベルを学習させ、比較を行った。対象としたラベルは、メガネ、笑顔、くせ毛の 3 種類であり、それぞれ独立に学習を実施した。モデル構造は、入力チャンネル数をカラー画像の 3 に画像分割数を 8×8 に変更した。他のパラメータは MNIST と同じであり、学習には 20 万枚の画像を用いた。各ラベルに対する学習の進行を示す学習曲線を**図 6** に示す。実験結果からラベルに応じて損失の収束傾向に差が生じた。最も学習損失が小さかったのはメガネのラベル分類であり、次いで笑顔、くせ毛の順であった。これにより、ラベルごとに学習の難易度が異なることが明らかとなった。

さらに、メガネラベル分類を学習させたモデルに未学習の画像を入力し、その Attention スコアを可視化した結果を**図 7** に示す。**図 7(a)** は ViT 学習後に入力した未学習画像を、**図 7(b)** には入力画像に対する Attention マップを示す。Attention スコアは Transformer Encoder の最終層から取り出した。**図 7(c)** は入力画像に Attention マップを重ね、画像の注視領域を分かりやすくした。これらの図から、ViT は主に目の領域を注目していることが確認できた。笑顔の場合は口元に加えて、図中の白丸が示すように目尻にも注意が向けられており、表情に基づく特徴抽出が行われていることが分かる（**図 8**）。一方、くせ毛のラベルに対応する画像（**図 9**）では、画像全体に広く Attention が分散しており、白丸が示すように、一部の画像では髪の領域に対するスコアが低くなる例も確認された。これらの結果から、くせ毛という属性

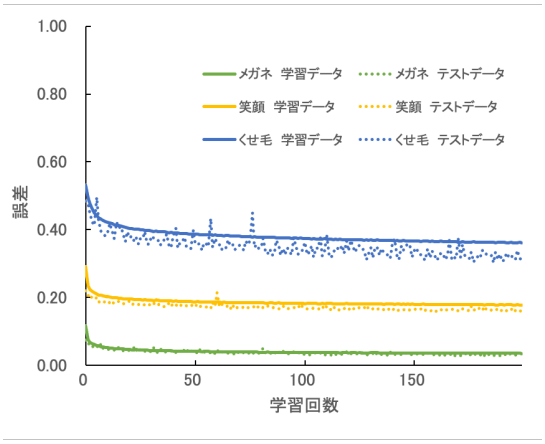
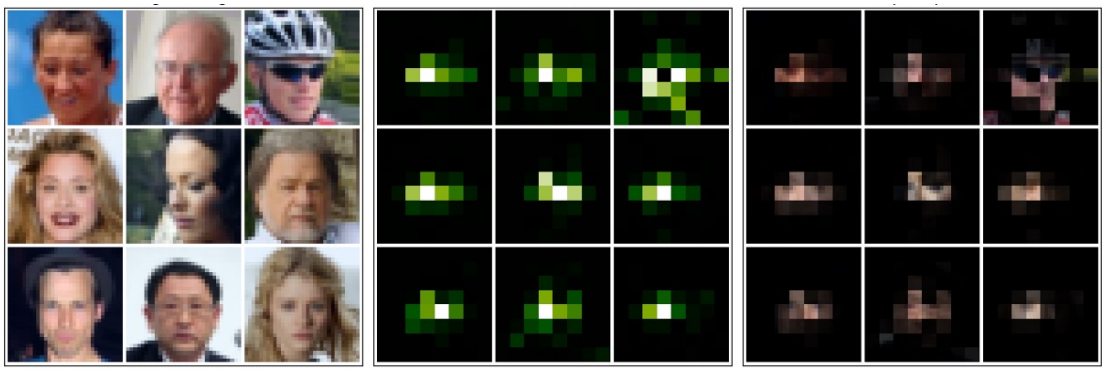
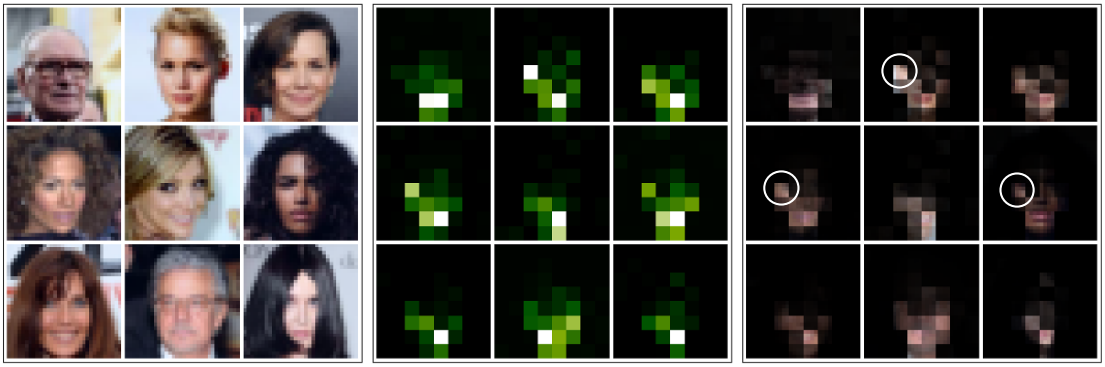


図 6　各種ラベルにおける学習速度



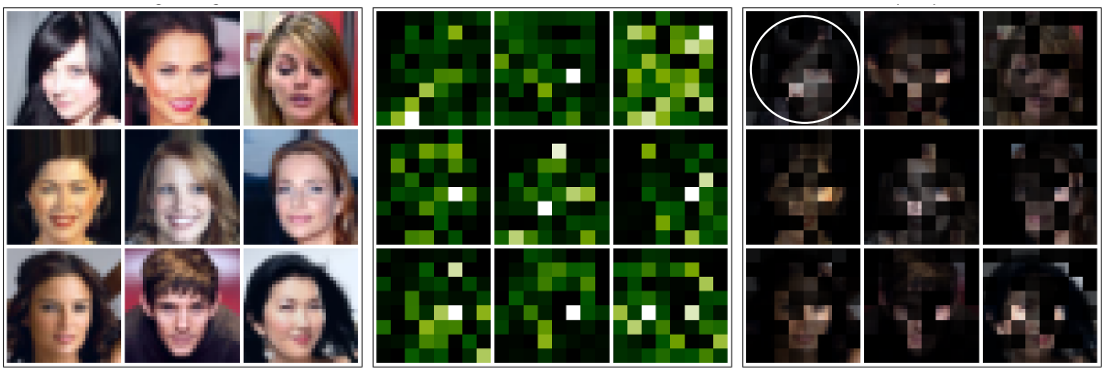
(a) 入力画像　　(b) Attension マップ　　(c) 注視画像

図 7　メガネラベル付け画像の Attention 領域



(a) 入力画像　　(b) Attension マップ　　(c) 注視画像

図 8　笑顔ラベル付け画像の Attention 領域



(a) 入力画像　　(b) Attension マップ　　(c) 注視画像

図 9　くせ毛ラベル付け画像の Attention 領域

が、ViT にとって識別が困難であったことが伺える。これら学習曲線と可視化画像の実験より、Attention の集中する領域が広くなるにつれて学習損失が高くなる傾向があることが分かった。

5. 結　　言

本研究では、パラメータ数を抑えたコンパクトな ViT モデルが、初期状態から少数の教師画像を用いて効果的に学習可能であることを示した。加えて、学習係数の違いが ViT の学習収束速度および最終的な学習精度に与える影響を評価し、適切な学習係数の選定がモデルの性能向上に不可欠であることを明らかにした。さらに、属性ごとに Attention の集中する領域が異なることを定量的かつ視覚的に示し、特にくせ毛のように画像全体に広く情報が分散する属性では学習損失が高くなる傾向を

明らかにした。これは、ViT が局所的特徴に依存する属性に対しては優れた識別性能を発揮する一方で、大域的・非局在的な特徴が必要な属性分類では性能低下が起こる可能性を示した。

文　献

1） Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser and Illia Polosukhin Attention is all you need, arXiv:**1706.03762** (2017).

2） Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu and Dario Amodei: Scaling Laws for Neural Language Models, arXiv:**2001.08361** (2020).

3） Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit and Neil Houlsby: An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale, arXiv:**2010.11929** (2020).

4） Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun: Deep Residual Learning for Image Recognition, arXiv:**1512.03385** (2015).

5） Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger: Densely Connected Convolutional Networks, arXiv:**1608.06993** (2016).

6） 廣川 勝久：広島県立総合技術研究所東部工業技術センター研究報告，事前深層学習モデルの転移学習による能力比較（第１報），**37**（2024）.

7） THE MNIST DATABASE of handwritten digits : http://yann.lecun.com/exdb/mnist/

8） Large-Scale CelebFaces Attributes (CelebA) Dataset: http://mmlab.ie.cuhk.edu.hk/projects/CelebA.html