卒業論文

事前学習が Vision Transformer に与える影響

18A1066　桝田 修慎

指導教員　山口裕 助教

2022 年 2 月

福岡工業大学情報工学部情報工学科

概要

画像認識分野では，畳み込み層を重ねることで良い認識結果をもたらした VGG をはじめ，残差機構を取り入れ，更に層を深くすることを可能にした ResNet が State-of-the-art を達成し，デファクトスタンダードとなっていた．しかし，今回のテーマである Vision Transformer は畳み込み層を使っておらず，Attention 層を用いて代用している．また，Vision Transformer での大規模なデータセットを用いた事前学習は，認識結果に影響し，ImageNet ReaL，CIFAR-100，VTAB などの画像認識のベンチマークで State-of-the-art を達成し，更新した．そこで本研究では，Vision Transformer を用いて，各自で用意したデータセットに対して事前学習をおこなった場合とそうでない場合，事前学習のデータセット規模を変えた場合で認識結果を向上させることができるのか検証した．

キーワード　Vision Transformer

目次

第 1 章

序論

1.1 背景

近年，画像認識分野では，機械翻訳で脚光を浴びることになった Transformer[8] をコンピュータビジョンに適応させた Vision Transformer（以下 ViT と称する）が登場した [9]．ViT は，層を深くし畳み込みを行う畳み込みニューラルネットワーク（以下 CNN と称する）とは違い，畳み込み演算を Attention 機構を用いて代用しており，特に大規模なデータで事前学習を行なったときの，小・中規模の画像認識ベンチマーク（ImageNet ReaL，CIFAR-100，VTAB，etc.）では，過去の State-of-the-art の CNN モデルと比べて少ない計算リソースで訓練することができ、更に素晴らしい結果も出している．

本研究では，Vision Transformer が提案された論文「An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale」を参考にし，事前学習やデータ拡張の有無が，学習及び推論に与える影響を検証した．

1.2 本研究の目的

本研究の目的を以下に示す．

1. 一定の条件下での振る舞いを従来のモデル（VGG，ResNet）と比較し，ViT の優れている点・そうではない点を明らかにする．
2. 事前学習やデータ拡張が各モデルに及ぼす影響を調べる．

1.3 深層学習

深層学習とは，脳の神経回路を模したニューラルネットワークをより深くしたものを指す．現在では画像のみならず，音声，自然言語など多くの分野に対して優れた性能を発揮している．

1.3.1　畳み込みニューラルネットワーク

畳み込みニューラルネットワーク（CNN）は，入力した画像に対してカーネルを移動させながらスカラ積を求めていく畳み込み層を複数重ねているネットワークのことである．カーネルは画像全体に同じものを適用するため，CNN は移動させたカーネルと画像のスカラ積による局所性と，画像全体に渡る同一カーネルの重みの使用による移動不変性を持つ．

1.3.2　VGG

VGG は CNN の一種で，3x3 という小さいカーネルを用いており，2014 年当時では珍しい 16 層及び 19 層の深いネットワークである [10]．畳み込みとプーリング，線形層のシンプルなアーキテクチャであるにも関わらず，2014 年の ImageNet チャレンジのローカリゼーション・クラシフィケーションタスクで 1 位と 2 位を収めている．

1.3.3　ResNet

一般的に，CNN では層を重ねることで，より高次元な特徴を抽出することができるが，層が深くなるにつれて勾配が発散・消失するという問題があった．しかし，ResNet は，畳み込み層を重ねるだけではなく，前の層，もしくはより浅い層の出力を次の層の入力とする残差機構（スキップ接続）を取り入れることで，より畳み込み層を多く積み重ねながらも，SoTA を達成した [11]．ResNet のブロックの一部を図 1.1 に示す．入力を x とし，2 層の weight layer(畳み込み層) を $f(x)$ とする．入力 x は weight layer と活性化関数 ReLU を経由し，$f(x)$ として出力される．そしてスキップ接続の x を加算し，最終的な出力は $f(x)+x$ である．

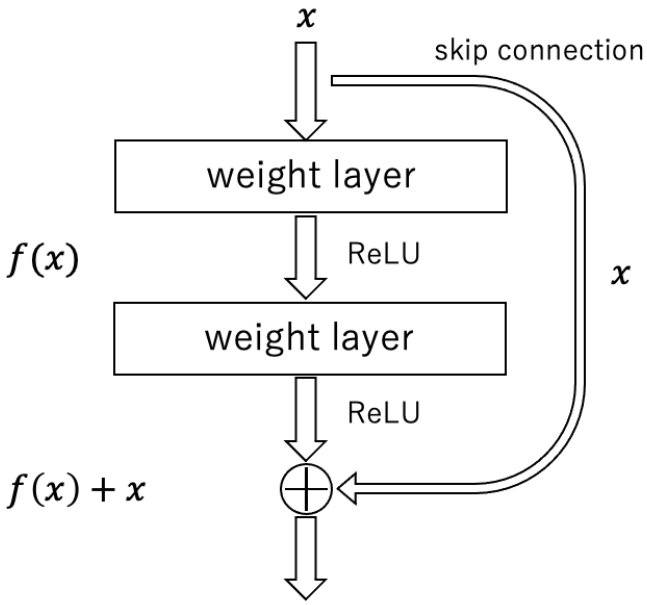


図 **1.1:** Residual learning[11]

1.3.4　Transformer

Transformer は，それまで機械翻訳モデルで多く使われてきた畳み込みニューラルネットワーク・再帰ニューラルネットワークのような複雑なアーキテクチャを持つネットワークとは違い Attention 機構のみを用いて構成されているエンコーダ・デコーダモデルである [8]. Transformer のアーキテクチャを図 1.2 に示す.

Attention 機構から求まる Attention スコアは，文章中の単語間の関連度のようなもので，ある単語を理解するのに，どの単語に注目すれば良いかを定量的に示すことができる．また，今回扱う一つの文章中の単語のみで求める Attention は Self-Attention と呼ばれる．以降，Attention は Self-Attention を示す.

Attention を求める例として，「I like an apple, it is tasty.」という文章を考える．初めに，文字列「I like an apple, it is tasty.」を入力とする．入力は単語ごとにベクトルに埋め込まれ $x_1, ..., x_7$ と表し，次元は d_{model} で，論文 [8] より 512 である．また，今回は簡単のためにカンマとピリオドは省略している．$x_1, ..., x_7$ はそれぞれ訓練過程で学習された重み行列 W_Q, W_K, W_V と掛け合わせ，それぞれ $q_1, ..., q_7, k_1, ..., k_7, v_1, ..., v_7$ とする．また，$d_k = d_q = d_v = 64, W_Q \in \mathbb{R}^{d_{model} \times d_k}, W_K \in \mathbb{R}^{d_{model} \times d_k}, W_V \in \mathbb{R}^{d_{model} \times d_v}$ である．今回は，「it」に対しての Attention を求める．「it」は先頭から 5 文字目なので，生成されたベクトル q_5, k_5, v_5 に対応する．次に，q_5 と文中全ての単語の $k_1, ..., k_7$ との要素積を求める．ここでのベクトルに対する値は目的の結果が得られるように調整してある．式 (1.1) に示す.

$$\begin{aligned} &q_5 \cdot k_1 = 20,\ q_5 \cdot k_2 = 30,\ q_5 \cdot k_3 = 10,\ q_5 \cdot k_4 = 140, \\ &q_5 \cdot k_5 = 100,\ q_5 \cdot k_6 = 30,\ q_5 \cdot k_7 = 40 \end{aligned} \tag{1.1}$$

式 (1.1) により求まった結果を $\sqrt{d_k} = \sqrt{64} = 8$ で割る．式 (1.2) に示す．これは，要素積が非常に大きくなった場合に対処するためである．

$$\begin{aligned} &\frac{q_5 \cdot k_1}{\sqrt{d_k}} = 2.5,\ \frac{q_5 \cdot k_2}{\sqrt{d_k}} = 3.75,\ \frac{q_5 \cdot k_3}{\sqrt{d_k}} = 1.25,\ \frac{q_5 \cdot k_4}{\sqrt{d_k}} = 17.5, \\ &\frac{q_5 \cdot k_5}{\sqrt{d_k}} = 12.5,\ \frac{q_5 \cdot k_6}{\sqrt{d_k}} = 3.75,\ \frac{q_5 \cdot k_7}{\sqrt{d_k}} = 5 \end{aligned} \tag{1.2}$$

式 (1.2) の結果を $softmax$ 関数を用いて 0 から 1 の値に収める．式 (1.3) に示す．

$$\begin{aligned} &softmax(\frac{q_5 \cdot k_1}{\sqrt{d_k}}) = 0.054,\ softmax(\frac{q_5 \cdot k_2}{\sqrt{d_k}}) = 0.081,\ softmax(\frac{q_5 \cdot k_3}{\sqrt{d_k}}) = 0.027, \\ &softmax(\frac{q_5 \cdot k_4}{\sqrt{d_k}}) = 0.378, softmax(\frac{q_5 \cdot k_5}{\sqrt{d_k}}) = 0.270,\ softmax(\frac{q_5 \cdot k_6}{\sqrt{d_k}}) = 0.081, \\ &softmax(\frac{q_5 \cdot k_7}{\sqrt{d_k}}) = 0.108 \end{aligned} \tag{1.3}$$

式 (1.3) で求まった値は $v_1, ..., v_7$ に対する重みであり，それぞれ掛け合わせることで，「it」という単語を理解するのに注目すべき単語には大きな重みがかかり，そうではない単語には小さな重みがかかることになる．今回の例では $q_5 \cdot k_4$ が 0.378 で最も大きな値であるので，k_4 を見ると，「apple」であることがわかる．つまり，「it」を理解するには「apple」が必要である．英文解釈的にも「it : それ」は「apple : リンゴ」を指すので正しい結果だと言える．

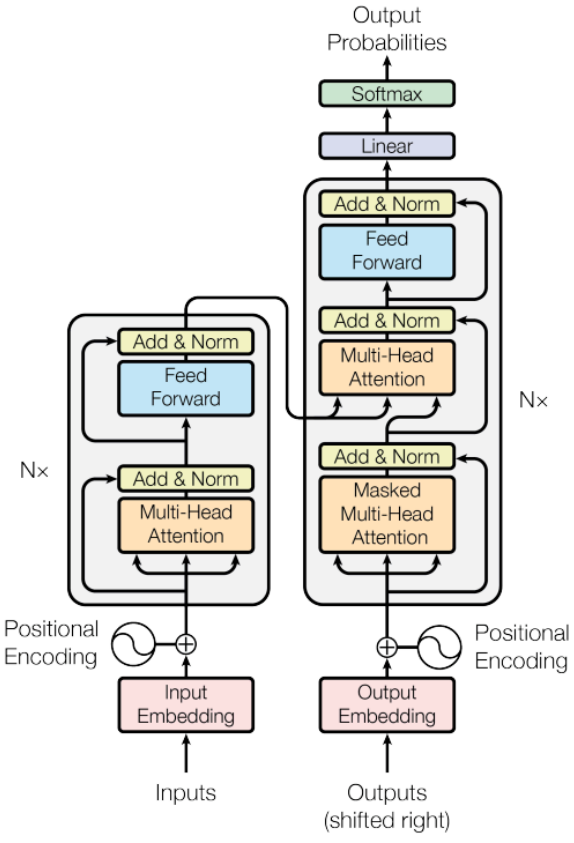


図 1.2: Transformer のアーキテクチャ [8]

1.3.5　Vision Transformer

Vision Transformer は，機械翻訳で用いられていた Transformer をコンピュータビジョンに適応させたモデルであり，画像を複数のパッチに分割してそれぞれをベクトルとして埋め込み，平坦化して入力とする特徴がある [9]．Vision Transformer のアーキテクチャを図 1.3 に示す．

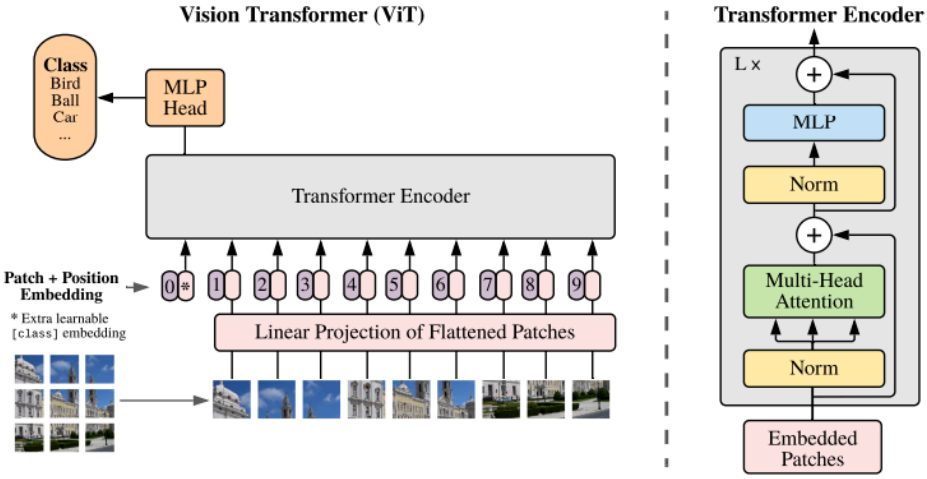


図 **1.3:** Vision Transformer のアーキテクチャ [9]

1.4　使用ツール・実験環境

1.4.1　Python[1]

Python は 1990 年代の始め，オランダにある Stichting Mathematisch Centrum で Guido van Rossum によって ABC と呼ばれる言語の後継言語として生み出された．Python はコードを簡潔に書くことができ，数値計算の NumPy，データ解析の Pandas など，専門的なライブラリが充実していることから，機械学習の研究開発をはじめとしたさまざまな分野で使用されている言語である．簡単に Python を始めるディストリビューションとして Anaconda がよく使われる．また，本実験で使用したバージョンは 3.7.10 である．

1.4.2　PyTorch[2]

PyTorch はディープラーニング・プロジェクトの構築を容易にする，Python のライブラリである．柔軟性を重視した設計であり，さらに，ディープラーニングモデルを Python の慣用的なクラスや関数の取り扱い方で実装できるようになっている．本実験で使用したバージョンは 1.9.1 である．

1.4.3　Pytorch Image models[3]

Pytorch Image Models（timm）は Ross Wightman によって作成されたディープラーニングライブラリであり，コンピュータビジョンの最先端のモデルが集められている．数行の記述でモデルを呼び出すことができ，必要に応じて書き換えることで，さまざまなタスクに適用できる．本実験で使用したバージョンは 0.5.2 である．

1.4.4　Albumentations[4]

Albumentations は Python のためのデータ拡張のライブラリである．モデルの汎化性能を上げるために行うデータ拡張のメソッドを多数揃えており，パイプラインを構成するとデータを効率的に拡張できる．本実験で使用したバージョンは 1.1.0 である．

1.4.5　ImageNet

ImageNet はディープラーニング研究のために無償で利用できる大規模なデータセットのことで，モデルの性能を測るためのベンチマークとして使われる．本実験では ImageNet の中でも「ILSVRC-2012 ImageNet」を使用する．このデータセットは，1000 のクラス，130 万枚の画像で構成されている．また，事前学習の規模によって認識精度が向上する ViT の性質を確かめるために，より大規模なデータセットとして，ImageNet-21k を使用した．このデータセットは 21,000 のクラス及び 1,400 万枚の画像で構成されている．

1.4.6　Kaggle

Kaggle は世界規模のデータサイエンスのプラットフォームであり，世界中のデータサイエンティストが技術を競う場である．本実験では Kaggle が提供している Notebook の GPU を利用して学習及び推論を行う．

1.4.7　Plant Pathology 2021 - FGVC8

本実験では Kaggle 上で提供されているコンペティションデータセットの Plant Pathology2021 を用いる．このコンペティションの目的はリンゴの葉の画像から，12 の病気を正確に診断することである．データセットの構成を表 1.1 に，各病名とユニークな値にエンコードしたペアを表 1.2 に示す．

表 **1.1:** データセットの構成

データセットの構成	説明
test_images	3 枚の画像
train_images	18,632 枚の画像
sample_submission.csv	提出用のサンプル csv ファイル
train.csv	image，labels の 2 カラムの csv ファイル

表 **1.2:** 病名とユニークな値のペア

ユニークな値	病名
0	complex
1	frog_eye_leaf_spot
2	frog_eye_leaf_spot complex
3	healthy
4	powdery_mildew
5	powdery_mildew complex
6	rust
7	rust complex
8	rust frog_eye_leaf_spot
9	scab
10	scab frog_eye_leaf_spot
11	scab frog_eye_leaf_spot complex

1.5　論文の構成

2 章では実験に使用するモデル ViT のアーキテクチャを数式を用いて説明し，実験の手順や条件について示す．3 章では実験結果を図と表を用いて示す．4 章では 3 章の結果をもとに議論を行い，理解を深める．5 章では今回の実験の総括を行う．

第 2 章

実験モデル

2.1　ネットワークモデル

ViT の入力は画像である．画像サイズを H，W，チャネル数を C とすると，入力 x は式 (2.1) のように表せる．

$$x \in \mathbb{R}^{H \times W \times C} \tag{2.1}$$

また，二次元画像を扱うために，二次元の平坦化したパッチに整形する．式 (2.2) に示す．

$$x_p \in \mathbb{R}^{N \times (P^2 \times C)} \tag{2.2}$$

ここで，N はパッチ数であり，P はパッチサイズである．また，N は H，W，P を用いて式 (2.3) のように表せる．

$$N = \frac{HW}{P^2} \tag{2.3}$$

ViT のエンコーダに入力するには，x_p をさらに埋め込む必要がある．長さ $P^2 \times C$ を D 次元のベクトルとして，線形投影したものを埋め込みパッチ z_0 とする．また，入力データの先頭には [class] トークンを付与し，埋め込みパッチ $z_0^0 = x_{class}$ とする．式 (2.4) に示す．

$$z_0 = [x_{class};\, x_p^1 E;\, x_p^2 E;\, \cdots;\, x_p^N E] + E_{pos}, \qquad E \in \mathbb{R}^{(P^2 \times C) \times D},\ E_{pos} \in \mathbb{R}^{(N+1) \times D} \tag{2.4}$$

ここで，E はパッチを D 次元のベクトルへの埋め込みを示し，E_{pos} は各パッチの位置が一意に定まるように情報を付加する位置エンコーディングを表す．上式で得られた入力 z_0 は Multi-Head Attention に入力される．Multi-Head Attention では，Attention を求める計算を複数回行う．具体的な算出については 1.3.4 に示した．Attention の算出式を式 (2.5) に示す．

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{2.5}$$

初めに，入力ベクトルから Q，K，V ベクトルを生成する．このとき，Q，K の次元は d_k で，V の次元は d_k である．計算時は，Q，K の要素積を $\sqrt{d_k}$ で除算し，softmax 関数で 0 から 1 の値にする．最後に V をかけると Attention が算出できる．Attention の算出過程を図 2.1 に示す．

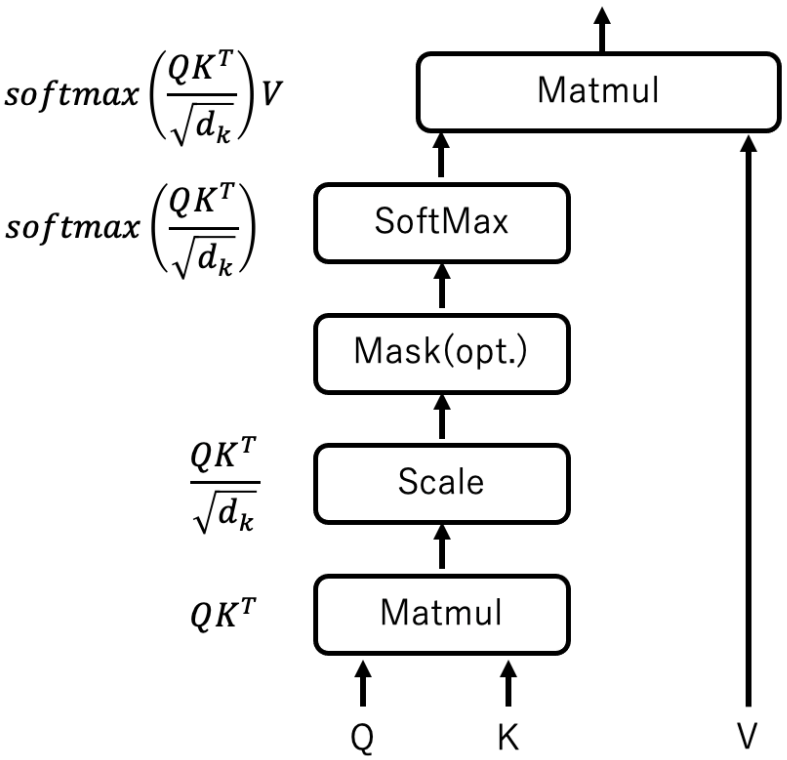


図 2.1: Attention の算出過程

2.2　評価指標

本実験で使用するモデルの評価指標とその説明を以下に示す.

2.2.1　Confusuion Matrix

Confusuion Matrix は二値分類の正解及び不正解の種類を示す表のことであり，以下の 4 つの要素で構成されている.

1. True Positive（TP）：真陽性．正しく正例と判断.
2. False Positive （FP）：偽陽性．誤って正例と判断.
3. True Negative （TN）：真陰性．正しく負例と判断.
4. False Negative （FN）：偽陰性．誤って負例と判断.

2.2.2　Accuracy

Accuracy は正解率ともいい，Confusion Matrix を用いて式 (2.6) と表せる．

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{2.6}$$

予測した値のうち，正しく正例及び負例を予測できた割合を表し，得られる値は 0 から 1 をとり，1 に近いほど良いスコアである．

2.2.3　F1-Score[5]

F1-Score は，Precision と Recall の調和平均で計算される指標である，Presision と Recall は Confusion Matrix を用いて式 (2.7) で表せる．

$$Presision = \frac{TP}{TP + FP}\ Recall = \frac{TP}{TP + FN} \tag{2.7}$$

Presision は正例と予測したもののうち真の値も正例の割合，Recall は真の値が正例のもののうちどの程度を正例の予測として含めることができているかの割合である．また，これらはトレードオフの関係になっており，どちらかの値を高くしようとするともう一方の値は低くなる．そして Presision と Recall のバランスをとった指標が F1-Score である．式 (2.8) に示す．

$$F1 - Score = \frac{2}{\frac{1}{Recall} + \frac{1}{Presision}} = \frac{2 \cdot Recall \cdot Presicion}{Recall \cdot Presicion} = \frac{2TP}{2TP + FP + FN} \tag{2.8}$$

また，F1-Score は二値分類における評価指標であり，他クラス分類に拡張したものに，mean-F1，macro-F1，micro-F1 がある．本実験では macro-F1 Score を用いる．macro-F1 Score はラベルごとに F1-Score を求め，それらの平均値を評価のスコアとする．

2.2.4　Cross Entropy Loss[6]

Cross Entropy Loss の算出式は式 (2.9) で示される．

$$Loss(x, class) = -\log\left(\frac{\exp(x[class])}{\sum_{j=0}^{n-1}\exp(x[j])}\right) = -x[class] + \log\left(\sum_{j=0}^{n-1}\exp(x[j])\right) \tag{2.9}$$

ここで，x はモデルからの出力（予測），$class$ は正解ラベル，n はクラス数を示す．例えば，$x = [-1.6059, -0.1475, -0.3950, -0.8846]$，$class = 1$，$n = 4$ とする．この場合の Cross Entropy Loss は式 (2.10) のように算出される．

$$\begin{aligned} Loss &= -x[class] + \log\left(\sum_{j=0}^{n-1}\exp(x[j])\right) \\ &= -x[1] + \log\Big(\exp(\mathrm{x}[0]) + \exp(\mathrm{x}[1]) + \exp(\mathrm{x}[2]) + \exp(\mathrm{x}[3])\Big) \\ &= 0.1475 + \log(2.1501) \\ &= 0.9130 \end{aligned} \tag{2.10}$$

2.3　手順

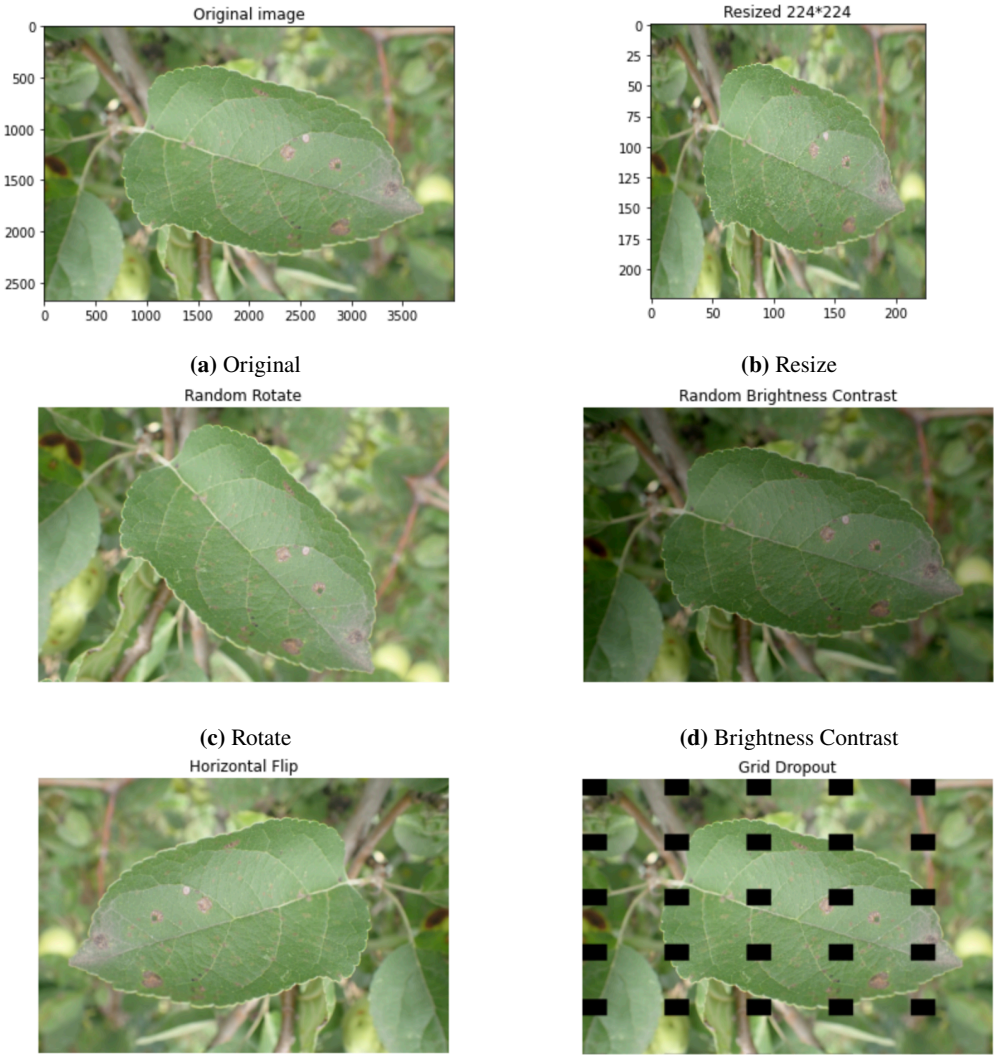
実験の条件を表 2.1 に示す．

表 2.1: 実験の条件

オプション / 条件	事前学習	データ拡張	ILSVRC-2012 ImageNet	ImageNet-21k
条件 1	なし	なし	○	×
条件 2	あり	なし	○	×
条件 3	あり	あり	○	×
条件 4（ViT のみ）	あり	あり	○	○

実験手順を以下に示す．

1. 条件 1 では，事前学習・データ拡張を行わなずに認識精度を検証する．論文より [9]，ViT は大規模なデータセットで事前学習を行なった場合に良い認識精度を発揮することがわかっているので，以降，このスコアを一つの基準として比較を進める．
2. 条件 2 では，事前学習を行い，データ拡張は行わない．予想される結果として，訓練データにオーバーフィッティングすることが考えられる．
3. 条件 3 では，オーバーフィッティングを防ぎ，検証データに対して認識精度を向上させるために，データ拡張を行う．これにより，訓練時と検証時の各指標の差異が小さくなると考えられる．また，今回の実験で使用したデータ拡張は Resize，Rotate，Horizontal Flip，Grid Dropout，Brightness Contrast である．それぞれ図 2.2 に示す．



(a) Original　**(b)** Resize

(c) Rotate　**(d)** Brightness Contrast

(e) Horizontal Flip　**(f)** Grid Dropout

図 2.2: 条件 1 の実験結果

4. 条件 4 では，ViT のみで実験を行う．先に述べた通り，ViT は事前学習の規模が大きいほど各画像認識のベンチマークで良い認識結果が得られている．そこで，条件 3 までで事前学習に使用してきたデータセットの ILSVRC-2012 ImageNet に加え，より規模の大きい ImageNet-21k を使用する．これにより各指標がこれまでよりいい結果を示すと予想される．

2.3.1　モデルのパラメータ

本実験で使用した各モデルのパラメータ及び設定を表 2.2 に示す．

表 2.2: 各モデルのパラメータ及び設定

設定＼モデル	vit_base_patch_224	resnet18d	vgg11
image size	224x224	224x224	224x224
batch size	32	32	32
debug size	0.2	0.2	0.2
epoch	10	10	10
learning rate	1e-4	1e-4	1e-4
optimizer	Adam（条件 4 のみ SGD）	Adam	Adam
criterion	Cross Entropy Loss	Cross Entropy Loss	Cross Entropy Loss

2.3.2　Optimizer SGD[7]

SGD（Stochastic Gradient Decent : 確率的勾配降下法）代表的な最適化手法の一つで，求めた勾配方向にパラメータを更新する手法である．パラメータの更新式を式 (2.11) に示す．

$$W \leftarrow W - \eta \frac{\partial L}{\partial W} \tag{2.11}$$

ここで，W は更新する重みパラメータを表し，$\frac{\partial L}{\partial W}$ は W に関する損失関数 L の勾配である．また，η は学習率であり，本実験では 1e-4 とした．

2.3.3　Optimizer Adam

Adam は，パラメータの要素ごとに適応的に学習率を調整しながら学習を行う AdaGrad[7] と，過去の勾配を徐々に忘れて新しく求めた勾配を大きく反映させる RMSProp を組み合わせた手法である [12].

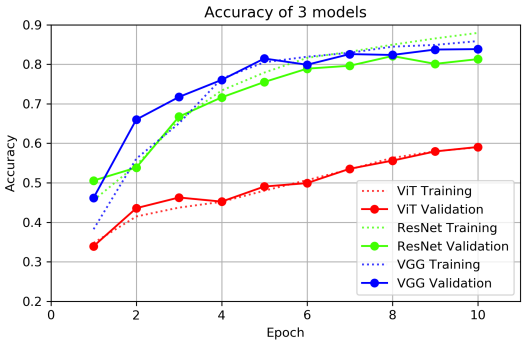
第 3 章

実験結果

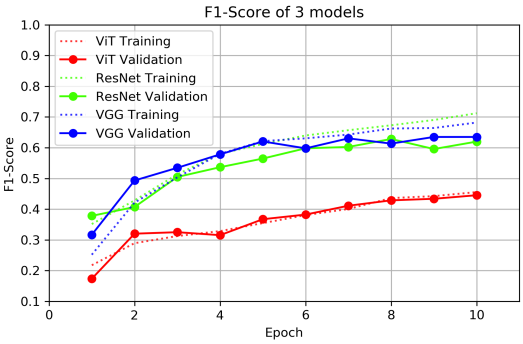
条件ごとの実験結果を表 3.1，表 3.2，表 3.3，表 3.4 に，また，それに対応するグラフを図 3.1，図 3.2，図 3.3，図 3.4 示す．

表 **3.1:** 条件 1 の実験結果

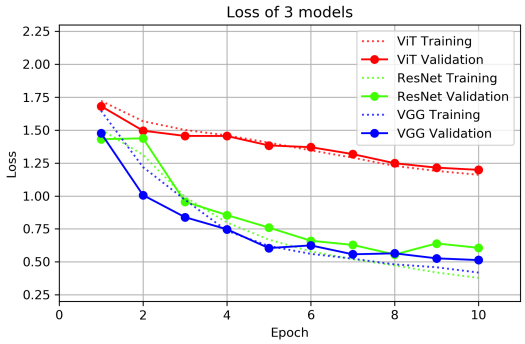
指標 \ モデル	vit_base_patch_224	resnet18d	vgg11
best valid accuracy	0.5906	0.8209	0.8384
best valid f1-Score	0.4453	0.6283	0.6349
best valid loss	1.1983	0.5554	0.5132



(a) Accuracy



(b) F1-Score

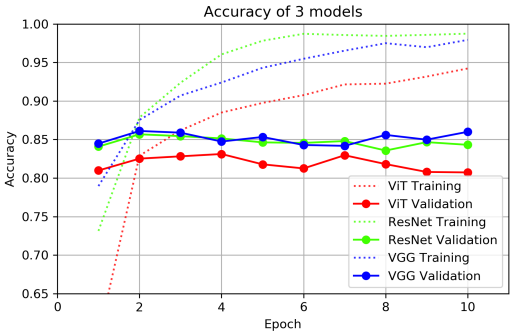


(c) Loss

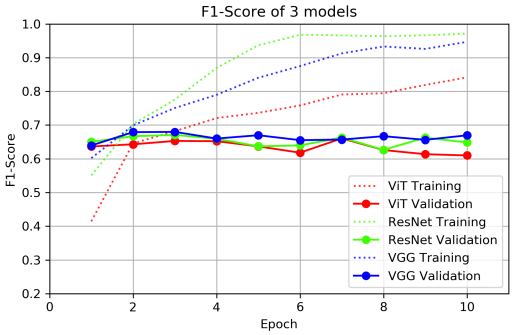
図 **3.1:** 条件 1 の実験結果

表 3.2: 条件 2 の実験結果

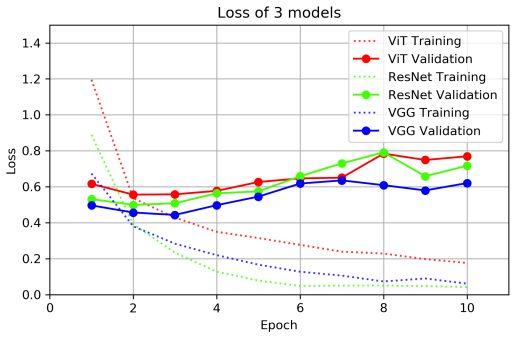
指標 \ モデル	vit_base_patch_224	resnet18d	vgg11
best valid accuracy	0.8312	0.8568	0.8611
best valid f1-Score	0.6605	0.6708	0.6798
best valid loss	0.5562	0.4989	0.4433



(a) Accuracy



(b) F1-Score

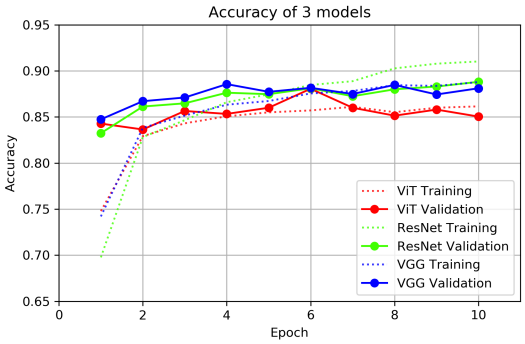


(c) Loss

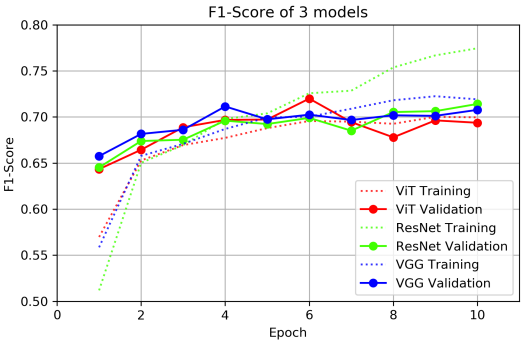
図 3.2: 条件 2 の実験結果

表 **3.3:** 条件 3 の実験結果

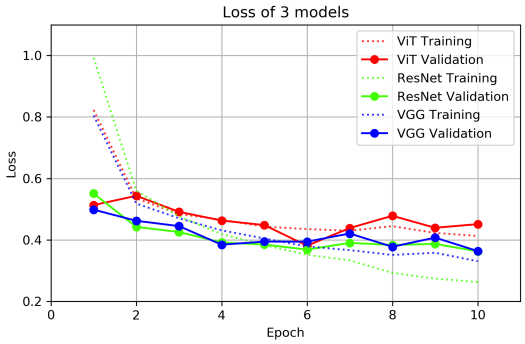
指標 \ モデル	vit_base_patch_224	resnet18d	vgg11
best valid accuracy	0.8806	0.8881	0.8856
best valid f1-Score	0.7199	0.7141	0.7114
best valid loss	0.3809	0.3626	0.3635



(a) Accuracy



(b) F1-Score

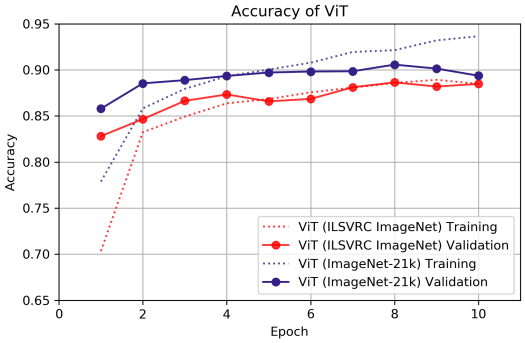


(c) Loss

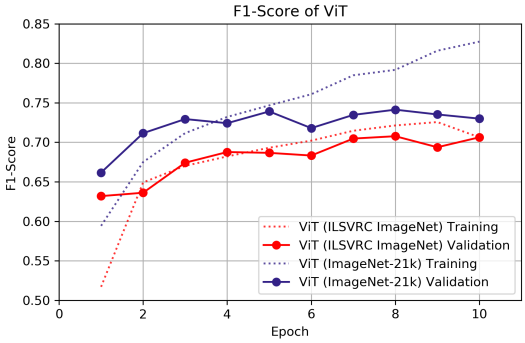
図 **3.3:** 条件 3 の実験結果

表 **3.4:** 条件 4 の実験結果

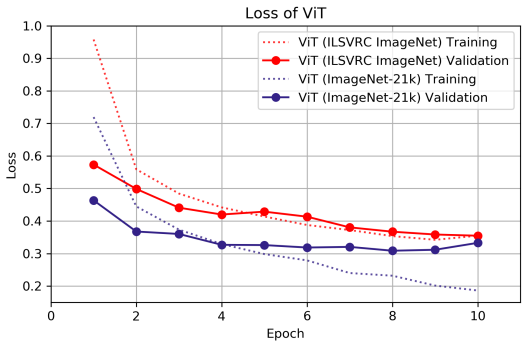
指標 \ モデル	vit_base_patch_224	
データセット	ILSVRC-2012 ImageNet	ImageNet-21k
best valid accuracy	0.8997	0.9057
best valid f1-Score	0.7284	0.7413
best valid loss	0.3301	0.3086



(a) Accuracy



(b) F1-Score



(c) Loss

図 **3.4:** 条件 4 の実験結果

第 4 章

議論

条件 1 の実験結果より，ViT は事前学習なしの場合では良い認識結果は得られていないことがわかる．各指標はそれぞれ良い方向に向かっているものの，ResNet，VGG には遥かに及んでいない．これは，ViT が CNN とは違い，画像に対する帰納バイアスがないことが原因だと考えられる．帰納バイアスとは，モデルが学習する時にもっている何らかの仮定のことである．例えば CNN であれば，「畳み込み演算のように同一カーネルによる局所的な情報処理を画像全体に渡って繰り返す 2 次元構造」が，帰納バイアスとして働いていて，画像認識タスクにおいて優れた性能を発揮するに至っている．このバイアスがないことが，大規模なデータセットでの事前学習が ViT に対して有効で，画像認識で良い結果を出す理由の一つである．
条件 2 の実験結果からは，事前学習をおこなった場合の ViT は他 2 モデルより，劣っていることがわかる．また，3 モデルとも Accuracy・F1-Score は training の値のみが上がり続け，Loss は限りなく 0 に近づいている．そして，validation の値は横ばいになっていることがわかる．これは，モデルが訓練データに対してオーバーフィッティングし，検証データに対する予測が上手くいっていないことが原因である．このことから次の検証では，検証データに対して汎化性能を上げるために，入力画像を加工するデータ拡張を行った．
条件 3 の実験結果では，データ拡張をおこなった場合の ViT の各指標は，training・validation 共に近い値をとっており，条件 2 のオーバーフィッティングを解消できている．その上，各指標の値もそれぞれ向上しており，特に ViT は大きく伸びている．
最後に ViT のみで実験を行った条件 4 での結果を確認する．これまでの条件と違う点は，事前学習に用いたデータセットの規模で，今までのクラス 1,000，画像枚数 130 万枚の ILSVRC-2012 ImageNet に加え，クラス 21,000，画像枚数 1,400 万枚の ImageNet-21k を使用した点である．図 3.4 からわかるように，ViT は大規模なデータセットで学習した場合に良い認識結果を示している．これは画像に対する帰納バイアスを持たない ViT に，大量の画像による事前学習を適用したことで，ハンデを克服したと言えるだろう．しかし，ImageNet-21k で事前学習した ViT は 5Epoch を超えたあたりから，validation との差が広がり，オーバーフィッティングしていることもわかる．原因は，大規模なデータセットにる事前学習の重みを適用したものの，本タスクの識別器としてはそれほど上手く機能していないことであろう．そして，この問題を解決するためには，より適切なデータ拡張やパラメータチューニングが必要だと考えられ

る．また，全ての実験の結果から，今回のタスクである「リンゴの葉の病気の診断」は，画像の大域的な構造よりも診断の対照である葉の局所的なテクスチャが特徴として使われることから，CNN が得意とするタスクであり，ViT 向きではなかったことも CNN とあまり差が出なかった原因の一つであるだろう．

第 5 章

結論

今回の実験で，Vision Transformer に事前学習を適用した時，自分で選択したデータセットに対して良い認識結果が得られ，事前学習時のデータセットの規模が認識結果に関係していることもわかった．また，データ拡張はモデルの過学習を防ぐのに貢献していた．従来のモデルとスコアがあまり変わらないのは，事前学習の画像枚数が論文の 1/230，1/20 で あったからだと考えられる．そして，最近の研究では，畳み込み層と Attention 層を組み合わせたモデルが，事前学習のデータセットの大小に関わらず，良い認識結果をもたらすこともわかっている [13]．このことから，一般的に利用できる規模のデータセットで高い認識精度を得るには，画像認識に特化した局所性を持つ機構である畳み込み層と，画像全体を参照しながら学習していく Attention 層を適切に組み合わせる必要があると考えられる．

謝辞

本研究を進めるにあたり，ご指導くださった山口裕助教に感謝の意を表します.

参考文献

[1] Python Software Foundation. 歴史とライセンス, 2001.

[2] Eli Stevens, Luca Antiga, and Thomas Viehmann. PyTorch 実践入門. マイナビ出版, 単行本（ソフトカバー）, 1 2021.

[3] fastai. timmdocs, 2021.

[4] Alexander Buslaev, Vladimir I. Iglovikov, Eugene Khvedchenya, Alex Parinov, Mikhail Druzhinin, and Alexandr A. Kalinin. Albumentations: Fast and flexible image augmentations. *Information*, Vol. 11, No. 2, 2020.

[5] 大輔門脇, 隆司阪田, 桂佑保坂, 雄司平松. Kaggle で勝つデータ分析の技術. 技術評論社, Kindle 版, 10 2019.

[6] PyTorch. Pytorch documentation cross entropy loss, 2021.

[7] 康毅斎藤. ゼロから作る Deep Learning ー 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, 単行本（ソフトカバー）, 9 2016.

[8] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.

[9] 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, 2021.

[10] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2015.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[12] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[13] Zihang Dai, Hanxiao Liu, Quoc V. Le, and Mingxing Tan. Coatnet: Marrying convolution and attention for all data sizes, 2021.

付録 A

実験結果の図

付録があればここに書く.