イマドキノCV
三つ巴の学習戦略

■牛久祥孝　■井上中順　■片岡裕雄

イマドキノ CV では「三つ巴の学習戦略」と題して，近年のコンピュータビジョン分野において最重要と位置付けられる「データラベルの利活用」や「認識モデルの構築」を取り扱う。前者においては，人間によるアノテーションコスト問題や，より少量の教師による認識モデル構築，自動データセット構築などのノウハウやテクニックなどに触れる。後者においては，2021 年の最も話題の技術の一つである Transformer についての詳解を加えるとともに，画像向けの Vision Transformer やその後急速に提案され始めている MLP 型ネットワーク構造についても，できる限りわかりやすく説明する。

1　前置き ── 近年のコンピュータビジョン分野から

2010 年代のコンピュータビジョンシーンは，華々しいほどの輝きを放っていた。AlexNet [1] が深層学習（deep learning; ディープラーニング）というキャッチーなテクニカルワードとともに世界的な第 3 次 AI ブームを巻き起こしたかと思えば，画像識別（2015 年，Residual Network; ResNet [2]）や強化学習（2016 年，AlphaGo [3]）など，ルールが明確に定義されている分野では，あっという間に人間のレベルを置き去りにして，さらに先に進んでいる。物体検出，領域分割，人物行動認識といった，比較的昔から存在していて自動運転や監視映像解析に応用できそうなタスクでも，深層学習により劇的な性能向上が実現した。現在でも，世界の研究者のみならず開発者が競争を繰り広げ，日に日にその性能向上が見える形で共有されている。また，深層学習では画像や動画像の処理のみならず，言語や音声も含めたクロスモダリティ領域も，統合が進んだと言える。これは，入力と出力の対応関係さえ明示すれば，間の特徴表現の学習を適切に行ってくれる深層学習の最大の特徴が活かされており，個別のモダリティの作り込みが必要であった従来法と比較すると，格段に参入障壁が下がったことが影響している。かくして，画像と言語に関するモダリティのみでも，画像説明文，視覚的質問応答，視覚的対話応答，視覚的ナビゲーションなど代表的なタスクが登場しては，どのタスクも現実世界で使用可能なレベ

ルにまで技術が高められている。

さて，深層学習の起源はどこにあるだろうか？ 多層パーセプトロン（multi-layer perceptron; MLP），誤差逆伝播法に代表される第 1 次，第 2 次 AI ブームを経て，現在は第 3 次 AI ブームの真っ只中にいるわけであるが，その実は緻密な技術提案の連続であり，2010 年初頭に大きな飛躍があったわけではないことが明らかにされている。もちろん，多層パーセプトロン，畳み込みニューラルネットワーク（convolutional neural network; CNN），再帰ニューラルネットワーク（recurrent neural network; RNN）など，キーとなる技術が深層学習時代以前に存在し，深層学習ブーム後の競争の末に出てきて現在猛威を奮っている Transformer [4] などは，今後の研究開発競争の中心となりそうである。

ここで，第 3 次 AI ブームが始まるきっかけについて簡単に触れることにする。間違いなく ILSVRC（ImageNet Large-Scale Visual Recognition Challenge [5]）の 2012 年大会が，その起点であると言われている[1)]。この話題はもはや説明不要なくらい有名であるため本稿での詳解は避けるが，コンピュータビジョン分野のみならず世界的な社会現象となっていることを鑑みるに相当な衝撃であった。

ImageNet プロジェクトやそのコンペティションから得た学びとして，「**データラベルの利活用**」と「**認識モデル**」の重要性がある。データラベルについては，画像データセットを大規模に収集することは膨大なコストを要するし，データセットと組み合わせて使用する際の認識モデル側の改善は精度向上には重要，ということである。

「データラベルの利活用」について，近年の主要な大規模画像データセットは，クラウドソーシングにより収集されている。画像データセットでは，まず画像が必要であり，ペアとなる画像カテゴリをラベルとして人間が付与する必要がある。まず画像を収集するところから始めるわけであるが，最近では個人がカメラ片手に収集するわけではなく，すでにウェブ上に存在する画像を再利用する。例えば ImageNet の場合には，クラウドソーシングのサービスである Amazon Mechanical Turk（AMT）を用いて画像へのラベル付けやクロスチェックが行われている。約 5 万人の参加者が，数億のリンク先から厳選された約 1400 万枚の画像にラベルを付与したと言われている。画像カテゴリについても，WordNet と呼ばれる単語の区分けされた辞書空間を参考にして，日常レベルの物体が 2 万種類ほど選定されている。コンペティションで用いられたのは，そのうち 1,000 カテゴリのサブセットであり，この場合においても学習・検証画像を含めてだいたい 130 万枚程度のデータセットとなる。現在も用いられる代表的な大規模画像データセットの多くは，AMT により収集，ラベル付与，クロスチェックがなされている。

ただし，画像の使用においては，個人情報保護など権利関係も絡んでくる。大

1) そのほかに，2011 年に行われた電話会話音声認識のコンペティションで深層学習を用いた手法が大幅な精度改善を達成したことや，2014 年にシンプルな深層学習モデルが機械翻訳の State-of-the-art に比肩する性能を達成したことなども引き金となっている。なお，2012 年大会第 2 位のチームは，東京大学の (当時) 國吉・原田研究室のメンバーにより構成されており，当時博士課程学生として中心的な役割を果たしていた本稿の筆者の一人（牛久）は，まさにこのコンペティションにおいて深層学習を引き立てる役割を期せずして担うこととなった。

規模データセットは，使用権利について不透明な画像が含まれることがよくある。こうした画像は商用利用が許可されていないことも多く，利用については注意を要する。一方で，なるべく条件がクリアな画像を効率良く集めたいという理由から，Creative Commons（CC）によりライセンシングされている画像を用いる取り組みもある。Yahoo! が提供していた YFCC100M（Yahoo! Flickr Creative Commons 100 Million [6]）データセットは，CC ライセンスが付与された画像および動画を 1 億枚含んでいた[2)]。もっとも，データのラベル付けをいまだに人間が行っていることもあり，コンピュータの処理に追いつかないという意味で限界を迎えている。研究室の大学院生の時間を犠牲にして新しいデータセットを構築するという文化は，確実にどこかに生息している。新しい問題設定を世に送り出すことは非常に重要であるが，産業を速いペースで回していくためには，画像データを作成・収集する時間や，人間がラベル付けする時間は，なるべく短縮したいわけである。

2) しかし，2021 年現在ではどうやらアクセスを制限しているようである。

本稿の前半部分である 2 節では，「データラベルの利活用」をより効率化するためのテクニックを紹介する。現在，学習を効率化する戦略は世界的にも多数提案されている中，2021 年現在のコンピュータビジョン分野における主力と言われ，世界的にも最もホットなトピックの一つである「自己教師あり学習」（self-supervised learning）を紹介する。また，自己教師あり学習でも拭いきれないデータセット問題を解決するために派生した，自然画像を用いない事前学習技術である「数式ドリブン教師あり学習」（formula-driven supervised learning）についても紹介する。どちらも基本的には人間による教師を置き換える目的で提案されたが，前者が自然画像を用いるのに対して，後者は画像すらも任意の法則から生成し，同時に画像ラベルを付与することができる問題設定である。教師あり学習も含めた分類を表 1 に示す。

表 1　学習に関する分類

学習法	使用画像	人間教師
教師あり学習	自然画像	あり
自己教師あり学習	自然画像	なし
数式ドリブン教師あり学習	生成画像	なし

「認識モデル」について，ILSVRC 2012 年大会から現在に至る約 10 年は，より高精度に画像を分類できるネットワーク構造の開発競争が激化した期間である。2013 年大会では，畳み込みニューラルネットワークの中間層を可視化し，その構造の分析から精度向上のためのヒントを掴み認識モデルを微修正する技術に留まったものの，2014 年大会あたりでは Google から GoogLeNet [7] が，またオックスフォード大から VGGNet [8] が提案され，優勝を争った。GoogLeNet

は畳み込みサイズを数種類用意して統合する Inception モジュールを積み上げることで層を増やし，VGGNet は比較的小さなサイズ（3×3）の畳み込みカーネルを用意して数段回組み合わせることで構造をより深くしている。2015 年大会では，今でも多くの研究者が用いている ResNet [2] が提案された。ResNet は，入力に対して畳み込み処理を施した特徴マップを残差（residual）として入力に加算する基本ユニットを積み上げることで，モデルを構成している。CV 分野において，残差はカメラ幾何や特徴抽出などの文脈でも古くから使用されてきたが，深層学習の構造に取り入れるというアイデアは，シンプルながら「層を深くすると性能が向上する」という定説に即して効果を最大限に発揮した。その後，CNN はある意味 ResNet の改善が繰り返される流れとなった。途中で構造自体の改善も進み，人間が探索するのではなく，学習のパラメータの一部に構造を組み替える仕組みを導入し，自動で認識モデルを探索する Neural Architecture Search（NAS）[9] や，入力や構造の深さ・幅をバランシングさせて少ないパラメータながらも効果的に学習を行い精度を向上させる EfficientNet [10] などが提案された。しかし，ドラスティックな変化が 2020 年の終わりごろに起こることとなる。自然言語処理に端を発する Transformer の仕組みが Vision Transformer（ViT）[11] として画像認識に使用され，2021 年には CNN を置き換えるのではないかと言われるように猛威を奮っている。Transformer は自己アテンション機構をそのネットワーク構造内に多数保有しており，直感的には画像内の CNN よりも広い領域から特徴を抽出できると言われている。事前学習データセットさえ整えれば CNN よりも高精度になるという報告もあり，近年注目を集めている。

本稿の後半部分の 3 節では，こうした近年の認識モデルについて説明する。CNN についてはすでに多くの書籍が取り扱っているので本節の説明に留め，3 節では画像認識向けの Transformer と，より最近の動向について触れることとする。

2　データラベルの利活用

画像認識に深層学習ブームが到来した一因は，大規模なデータを用いた「事前学習」と小規模なデータを用いた「転移学習」が成功を収めたことにある。2010 年代以前は，物体検出などのタスクごとにそれぞれ特徴量を設計し，サポートベクトルマシンなどの識別器を学習するという方式が主流であった。それが現在では，ImageNet や MS COCO など，大量の画像を有するデータセットで事前にニューラルネットワークを学習し，その結果をもとに別の小規模なデータセットで学習を行うという 2 段階の学習が主流となっている。この学習方式の

1 段階目が事前学習，2 段階目が転移学習であり，これによって独自の小規模なデータセットでも，事前学習のデータを適切に準備すれば，極めて高性能なニューラルネットワークを構築できる。

しかし，実社会の課題に対しては，ImageNet などのオープンデータで必ずしも十分な事前学習ができるとは限らない。写真や映像を入力とした物体やシーンの認識では ImageNet が有効であるが，イラストや医療画像などを入力とした新たな応用を考えると，大規模な事前学習用のデータが別途必要になる可能性がある。さらに，公平性の観点から，人種や性別などに関するバイアスが事前学習の際にできる限り乗らないように配慮することも望まれている。

これらの問題の解決策は現在世界中の研究者が模索しているところであるが，2021 年現在，正解ラベルを用いない，さらには実世界データすらも用いない事前学習法が発展しており，これらを利用すれば，ごく少量のデータで高精度な転移学習を実現することができる。以下では，イマドキの事前学習法として，自己教師あり学習と数式ドリブン教師あり学習を初学者にもわかりやすく解説する。

2.1 イマドキの事前学習

まず，従来の事前学習とイマドキの事前学習の違いを見てみよう。ImageNet などの正解ラベル付き学習データを用いた従来の事前学習では，多くの場合，分類問題に関する交差エントロピー損失を最小化する。すなわち，学習用の画像 x_i（$i = 1, 2, \ldots, N$）を大量に用意して，それぞれに対して，正解ラベル y_i をそれぞれ付与した上で，損失

$$L = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{\exp(z_{iy_i})}{\sum_j \exp(z_{ij})} \tag{1}$$

が最も小さくなるようなニューラルネットワークのパラメータを求めるのである。ここで，z_i はニューラルネットワークの出力から得られる画像のベクトル表現である。この損失 L は，ニューラルネットワークがすべての学習用画像について 100% 正確にラベルを予測できるようになったときかつそのときに限り，値が 0 になるように設計されている。実際にはそのようなネットワークを得ることは難しく，多くの場合，95〜99% のラベル予測精度で学習が終了する。また，式 (1) の N は画像枚数であり，高精度な事前学習を実現するには，N が 100 万以上であることが望ましい。しかし，100 万枚以上の画像一つ一つに，人手で正解ラベルを付与するには多大なコストがかかる。仮にクラウドワーカーの力を借りて，すべての画像にラベルを付与したとしても，それらのラベルがすべて適切であるかどうかを確かめることすら難しい。もし，ラベル付けに専門知識が必要となると，事前学習はあきらめざるを得ないかもしれない。

イマドキの事前学習はこれらの問題の解決を目指したもの，すなわち，正解ラベル y_i が付与されていない状態，さらには，実世界の画像 x_i すらも与えられていない状態で事前学習を実現するものである。

自己教師あり学習

自己教師あり学習とは，ある規則に従いデータから自動的に生成されたラベルを用いた学習の総称である。学習データとしては，ラベルが付与されていないデータ，すなわち画像 x_i のみを用いるので，自己教師あり学習は教師なし学習の一種と考えることができるが，それぞれの画像から明示的に算出可能なラベルを用いた教師あり学習とも見なせるため，「自己教師あり」という名前がついている。次の例を見れば，その意味が理解できるだろう。

ジグソーパズルと自己教師

初学者にとって，自己教師あり学習のイメージを最も把握しやすいのは，ジグソーパズルである。ジグソーパズルは，1 枚の絵や写真を細かいピースに分けてバラバラにした上で，それを元の画像に戻す問題である。ニューラルネットワークにこのパズルを解かせて，それを事前学習としようというのが基本アイデアとなる。

ここでは，理解のしやすさを優先して，問題を極端に簡略化した例を考えよう。いま図 1 のように，画像 x が 1 枚与えられたとして，それを 3×3 に 9 分

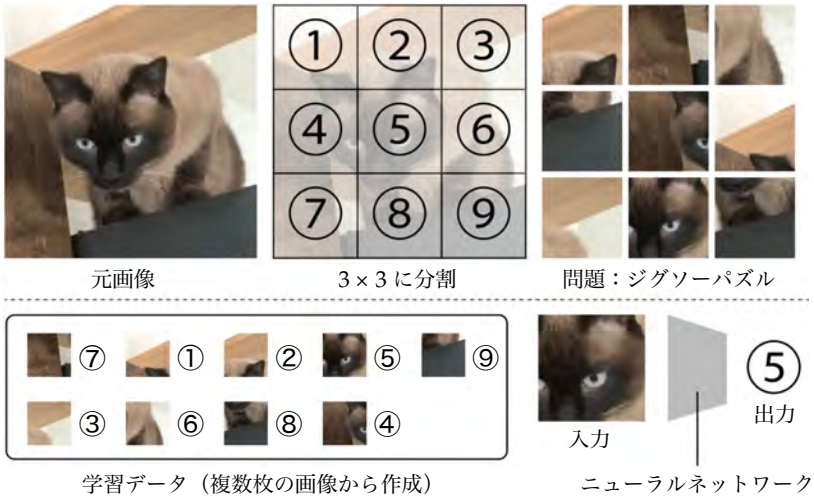

図 1　ジグソーパズルを例とした自己教師あり学習の簡略版。元画像を 9 分割してパズルを作成し，元の位置を当てる 9 クラス分類問題を設計している。これにより，正解ラベルが付与されていない大量の画像を用いて学習ができる。

割する。ここで得られた 9 個のピースを $x_1, x_2, \ldots, x_9$ とおくと，ジグソーパズルは，ある一つのピース x_i が与えられたとき，それが九つの位置（①〜⑨）のうちどこにあるべきものかを解く問題となる。例えば，x_3 は ③が正解となる。これは，$y_3 =$ ③ のように，位置が正解ラベルだと考えると，ジグソーパズルが位置に関する 9 クラス分類問題として定式化されることを意味している。すなわち，図 1 のように番号を付与しておけば，式 (1) の交差エントロピー損失をそのまま用いて，ニューラルネットワークを学習できる。

ここで注目すべきは，人手による正解ラベル（y_i）の付与が行われていない点である。つまり，ラベルが付与されていない画像から，学習に必要なラベルを得ることができている。前述のとおり，この学習方式は，ラベルなしデータのみを用いて実現できているという意味では教師なし学習であるが，ある規則に従い自動的にラベルを生成したあとは，任意の教師あり学習向けの損失関数を用いることができるという点で，極めて教師あり学習に近い性質を持っている。この例を見れば，なぜ「自己教師あり」という名前が定着したのかも理解できるだろう。これが「自己教師あり学習」の最も簡単な例である。

対照学習

ジグソーパズルのアイデアをもとにした自己教師あり学習が提案されたのは 2016 年であり，そのときは認識精度の意味で，通常の教師あり学習とは大きな差があった。しかし，そこからの発展が目覚ましく，現在ではラベルなし，あるいは 1.0% 程度のラベル付与で，教師あり学習に匹敵するほどの精度が得られるところまで進化を遂げている。その進化の鍵となったのがデータ拡張に基づいた「対照学習」（contrastive learning）である。具体的な手法名としては，SimCLR [12]，MoCo [13]，SimSiam [14] がよく知られており，ここではその基礎の理解を目的として，SimCLR の原理を少し簡略化して解説する。

SimCLR とは，Simple Framework for Contrastive Learning の略であり，その仕組みを理解する上で，「照らし合わせ」が重要なキーワードとなる。理論的には，Noise Contrastive Estimation（NCE）[15] 損失と呼ばれる損失関数の発展を説明し，また距離学習との関連を見た上で，対照学習に辿り着くのがよいが，初学者には敷居が高い。細かい数式やその定義に至る背景は原論文に譲るとして，ここでは，ジグソーパズルの例になぞらえて，対照学習がどのような「パズル（問題）」を作って解いているかを考えてみよう。

SimCLR の対照学習におけるパズルを一言で表すと，「二つのデータ拡張[1] された画像を照らし合わせて，それらの元画像が同じかどうかを当てる問題（パズル）」である。例えば，図 2 のように猫と犬の画像があったとして，それぞれにさまざまなデータ拡張を施した画像の集合を考えよう。このデータ拡張され

[1] 幾何変換やフィルタリングなどの加工を施した上で画像を複製すること。

図 2　対照学習における「パズル（問題）」の構成例

た画像の中から，ランダムに 2 枚の画像を選んでくると，元の画像が一緒かどうか？という問題を考えることができる。このように 2 枚の画像の「照らし合わせ」から定義されるパズルを解くものとして対照学習を解釈すれば，その方針と本質を理解しやすい。

では，どうやってこのパズルを解くのか？ ここで注意すべきは，目的はこのパズルを解くこと自体ではなく，あくまで物体検出など実際のタスクで有効な画像特徴量を得ることであり，それゆえパズルへの過学習を避けなくてはならない，ということである。そのため，SimCLR では図 3 のような構造で，画像特徴量を得るネットワークの上に，パズルを解くヘッダを準備する。ヘッダ部分はパズルを解くことに特化してよいが，画像特徴量は汎用的なものが残ってほしいというのが，この構造の狙いである。

ここからは学習方法の詳細を解説する。まず，画像 x から画像特徴量 h を抽出するニューラルネットワーク f を考える。これは，ResNet や Vision Transformer

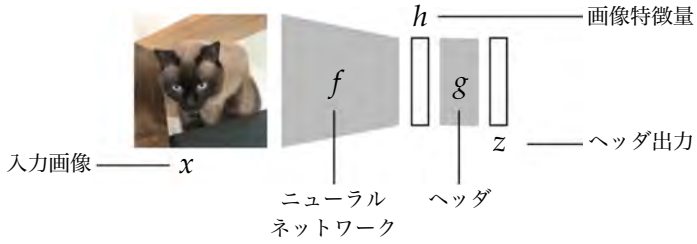

図 3　SimCLR のネットワーク構造。画像特徴量を計算するためのニューラルネットワーク f に，問題を解くためのヘッダ g が付与されている。事前学習終了後に g は切り捨てて，f から得られた特徴量を使うか，別の問題に関するヘッダを付けて転移学習（fine-tuning）を行う。

を想定すればよい。すなわち，$h = f(x)$ が画像特徴量（256 次元程度のベクトル）となる。さらにヘッダとして，別のニューラルネットワーク g を考える。このネットワークはパズルを解くための簡単なものであって，1〜3 層のものでよい。そして，ヘッダの出力を $z = g(h)$ と書く。この z も 128〜256 次元程度のベクトルとする。

図 2 では，猫と犬の画像それぞれにさまざまなデータ拡張を施していたが，ここでは簡単のため，図 4 のように，2 枚の異なる学習用画像 x_i と x_j を考え，それぞれに 2 通りのデータ拡張を施す。すると，4 枚の画像 $x_i^{(1)}$, $x_i^{(2)}$, $x_j^{(1)}$, $x_j^{(2)}$ が得られる。ここで，右上の添え字の (1), (2) はデータ拡張の一つ目，二つ目を表しているが，添え字が多くなるため，これら 4 枚の画像を順に $\tilde{x}_1$, $\tilde{x}_2$, $\tilde{x}_3$, $\tilde{x}_4$ と書き直すことにする。次に，4 枚の画像それぞれに対して，ヘッダの出力 $z_k = g(f(\tilde{x}_k))$ を計算する。ここで得られた，四つの出力の間のコサイン類似度 $s_{k,k'} = \cos(z_k, z_{k'})$ を考えると，全部で 16 個の値が得られることになる。それらを並べて，

$$S = \begin{bmatrix} s_{1,1} & s_{1,2} & s_{1,3} & s_{1,4} \\ s_{2,1} & s_{2,2} & s_{2,3} & s_{2,4} \\ s_{3,1} & s_{3,2} & s_{3,3} & s_{3,4} \\ s_{4,1} & s_{4,2} & s_{4,3} & s_{4,4} \end{bmatrix} \tag{2}$$

と書いておく。これでパズルを解くための準備は完了である。

いま，この行列 S と単純な閾値処理で，先に述べた「二つのデータ拡張された画像を照らし合わせて，それらの元画像が同じかどうかを当てる問題（パズル）」を解くことを考えてみよう。つまり，$s_{k,k'}$ の値が閾値以上であれば元画像が同じ，そうでなければ元画像が違う，と判断する。これで正解を導くには，S

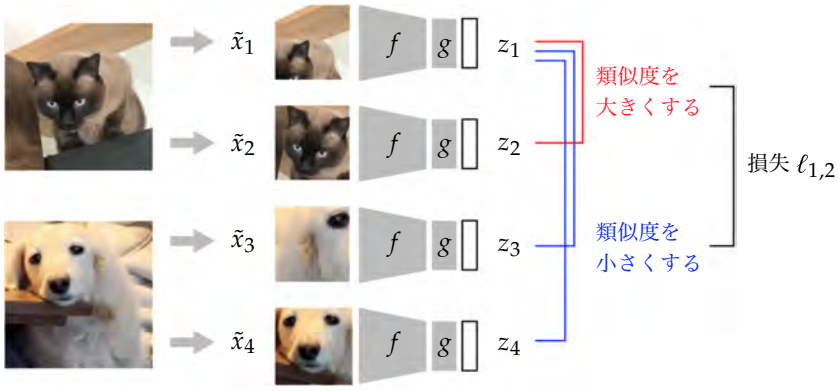

図 4　対照学習における損失の計算順序。ここでは簡単のため，2 枚の画像に対して 2 種類のデータ拡張を施した例を示している。

がどのような行列になっていればいいだろうか？ 例えば，$s_{1,2}$ を見ると，これは一つの元画像 x_1 から 2 種類のデータ拡張を経て計算されたものである。よって，$s_{1,2}$ の値は大きいほうが都合が良い。一方，$s_{1,3}$ を見ると，これは画像 x_1 の 1 番目のデータ拡張と，画像 x_2 の 1 番目のデータ拡張から計算されたものであるから，元画像が異なる。よって，$s_{1,3}$ は相対的に小さいほうが都合が良い。同様に，$s_{1,4}$ も小さいほうがよい。また，$s_{1,1}$ はそもそも同じベクトルから算出されており，1.0 をとるはずなので無視をする。以上をまとめると，パズルを正しく解くためには，$s_{1,2}$ が大きく，$s_{1,3}$ と $s_{1,4}$ が相対的に小さければよいことがわかる。よって，この大小関係が実現するように，例えば以下のように損失関数を設計する。

$$\ell_{1,2} = -\log \frac{\exp(s_{1,2}/\tau)}{\exp(s_{1,2}/\tau) + \exp(s_{1,3}/\tau) + \exp(s_{1,4}/\tau)} \tag{3}$$

ここで，τ はスケーリングのハイパーパラメータである。コサイン類似度をそのまま損失に入れてもよいが，式 (1) の交差エントロピー損失になぞらえて，exp 関数を通した上で比率を計算したほうが安定的になる。また，$\ell_{1,2}$ は行列 S の 1 行目に関するものであり，同様に 2 行目から 4 行目も考えると $\ell_{2,1}$，$\ell_{3,4}$，$\ell_{4,3}$ の損失が得られる。最後にこれらの平均

$$L = \frac{1}{4}(\ell_{1,2} + \ell_{2,1} + \ell_{3,4} + \ell_{4,3}) \tag{4}$$

を計算して，この値が小さくなるようにニューラルネットワークのパラメータを学習する。この例では画像を 2 枚を用いて説明をしたが，一般には一つのミニバッチに関する n 枚の画像に対して，

$$L = \frac{1}{2n}\sum_{k=1}^{n}(\ell_{2k-1,2k} + \ell_{2k,2k-1}) \tag{5}$$

を計算する。

このようにして定義された損失関数は，人手で付与される正解ラベルなしで計算できる。これは，式 (1) の交差エントロピー損失との決定的な違いである。このほかにも損失関数の定義方法にはいくつかあるので，興味のある読者は MoCo や SwAV などを調べるとよい。また，実装には PyTorch の VISSL（https://vissl.ai/）などのライブラリを活用するとよい。対照学習に基づいた自己教師あり学習は，画像認識に留まらず，2020 年頃から音や言語の学習にも幅広く応用されている。ここではコンピュータビジョンでのイマドキの事前学習として，自己教師あり学習の紹介をしたが，自然言語処理の分野では，BERT に代表される，マスキング機構を用いた学習手法が（多くは対照学習ではない）自己教師あり学習として発展を遂げており，それらとの関連も現在のホットトピックである。

数式ドリブン教師あり学習

人間のアノテーションによるラベルや，カメラにより撮影された自然画像をいっさい用いずに事前学習モデルを構築する「数式ドリブン教師あり学習」の枠組みも提案されている [16]。ある法則により画像パターンとそのペアになる画像カテゴリを自動生成することで大規模画像データセットを構築し，深層学習モデルを事前学習する枠組みである。

ImageNet をはじめとする教師あり学習が，転移学習の文脈で精度が出せるという観点で最も信頼できる手法として長らく扱われてきたことが，この枠組みの背景にある。画像識別のみならず，関連タスクである物体検出やセマンティックセグメンテーション，人物姿勢推定や行動認識など，あらゆるタスクに応用できたこともあり，深層学習はこの約 10 年間で隆盛への道をたどった。しかしながら，劇的な性能向上や事前学習・転移学習の成功の裏側で，大規模画像データセットに由来する問題も各所で指摘されている。ImageNet をはじめ多くのデータセットは，クラウドソーシングにより不特定多数のウェブ画像を収集していることなどから，その権利関係が不透明である。データの提供元のプロジェクトページなどでは，教育および学術研究に使用用途を限定し，商用利用を禁止している。さらに，2019 年 9 月には，結果的に学習済みモデルの出力に不公平性を与えうる，特に人間関連のラベルに関する偏りが ImageNet に含まれる問題が指摘された [17]。例えば，プログラマという画像カテゴリでは男性，看護師という画像カテゴリでは女性のほうがメジャーなサンプルとなり，性別を見て判断してしまうということも実際に起こっている。ユーザーに不都合を与える可能性があると考えたデータセット提供元は，その時点で人間関連のラベルのうち，偏りが生じてしまう可能性のあるカテゴリについては公開を停止している[2)]。同時に，今後のデータセットは，常に人目に触れつつ，コミュニティ全体でそのクオリティを改善していく努力が必要であろうとコメントしている。事態はこれで収束しない。2020 年 6 月には，80 Million Tiny Images（80M Tiny Images）が，大規模画像データセットを巡る技術レポート [18] の調査結果の影響で，公開停止に追い込まれている。技術レポートによると，このデータセット内には攻撃的なラベルや不適切なラベルが含まれていたとのことである。これに対して公開側は，そのデータセットの規模が巨大（8 千万枚弱の画像を含む）であること，各画像のサイズが極小であることを理由に，人手での再ラベリングは困難であるとして公開停止に踏み切った。さらには，顔画像を含むデータや商用の顔認識アプリケーションの一部も公開停止になるなど，データセットが持つ公平性などの AI 倫理の問題は，無視できないトピックとなった。2021 年に入ってからも，人手によるラベルのミスは，調査の結果，全体の 6% にのぼる [19] と指摘されるなど，人間がデータセットにラベルを付

2) https://groups.csail.mit.edu/vision/TinyImages/

与することで成立する教師あり学習の限界が見え始めてきたと言える。

では，どのようにデータセット関連の問題を解決すればよいのであろうか？その一つは，自然画像を用いずに自然画像認識に必要な視覚特徴表現を獲得することである。不特定多数の自然画像を用いてラベル付けを実施することが権利関係，アノテーション，AI 倫理の側面で問題になるのであれば，いっそのこと自然画像を用いないという選択肢は妥当な策である。この極めて困難だが理にかなった方式を成功させたのが，自然の形成原理の一つであるフラクタル幾何（fractal geometry）を用いて大規模画像データセットを自動生成するプロジェクトである。発想の根底には「ピクセルによる画像空間は膨大なのに，いかにして ImageNet は良好な視覚特徴を捉えているのであろうか」という疑問がある。これは例えば，グレースケールの画像でさえ 256 階調の輝度値があり，画像空間はピクセル数ごとに広がる。30 万画素の画像でさえ 256^{300000} の特徴空間が存在する。ここで，ImageNet はわずか128 万枚程度の学習画像で良好な視覚特徴を獲得していると言われており，その背後には自然の法則が少なからず作用していることが想像できる。ならば自然のパターンを形作る代表的な法則であるフラクタル幾何を直接的に画像に投影すれば，学習はうまくいくのではないか？ これが，この研究が開始された背景である。フラクタル幾何は，草木などの自然物，海岸線や山脈などの風景といった，われわれの現実世界に存在する形状を形作る法則の一つとして有名であり，単純な数式なのでレンダリングが高速で，かつ自然物の再現性が高いため，コンピュータグラフィックスの分野などにおいて頻繁に使用されてきた。そのため，CNN がフラクタル幾何により生成された画像データセットを学習することで，現実世界の物体を認識するために必要な特徴を獲得できる，すなわち，自然画像なしで良好な視覚特徴表現を獲得できるのではないかという発想である。結論から言うと，2021 年の時点では，まだどの場面においても ImageNet による事前学習を超えるに至ってはいないが，一部のデータセットにおいては高い精度を記録している。以下に，数式ドリブン教師あり学習の基本について詳解する。

まず，FractalDB をはじめとした数式ドリブン教師あり学習のコンセプトは，何かしらの法則，関数および数式により，画像パターンとその画像カテゴリの対を自動かつ同時に生成することである。その際，自然画像を認識するために有効な視覚特徴表現を，自然画像を用いない画像データセットによる事前学習から獲得する手段を提供する。通常の教師あり学習では，収集された自然画像に対して人手でアノテーションを行ったデータセットを用いて学習を行う。100 万画像を超えるような大規模な画像収集においては人為的なラベル付けは困難であり，クラウドソーシングや専門家に発注し，数か月から数年単位でデータセットを構築することも珍しくはない。一方で，数式ドリブン教師あり学習は，CNN

を学習するために必要な大規模画像データセットを，人手を介さずに自動構築できる。1,000 カテゴリ，1,000 インスタンスで合計 100 万枚を含む FractalDB の場合，CPU を 40 コア使用すると，1 日以内に画像データセットの構築が完了する。それでいて，数式を用いて画像パターンを生成するので，少なくとも事前学習の文脈においては，著作権やプライバシーといった画像使用に関する権利問題に抵触することがない。人間が映り込むことがない点からも，事前学習の文脈においては，AI 倫理に関する心配もなく学習済みモデルを構築することができる。

フラクタル画像生成

フラクタル幾何とは自己相似性を持つ形状のことである。画像においては，大域的な形状と局所的な形状が類似しており，同形状が再帰的に組み合わさることで擬似的に表現される[3]。それにより生成される画像パターンには，植物や海岸線といった自然物の形状が含まれるとされている。その性質を CNN で直接学習することで，自然画像の認識精度を向上させることができる。FractalDB の画像例を図 5 に示す。フラクタル画像の生成には，確率付き反復関数系（iterated function system; IFS）を使用した。IFS は完備距離関数 $\mathscr{X}$ で定義され，次の式で表される。

[3] 実世界のフラクタルは無限の解像度を持つと言われているが，画像認識におけるフラクタル画像は，画像やピクセルの表現方法による制限を受ける。そのため，実際のフラクタル幾何とは異なることに注意されたい。

$$\mathrm{IFS} = \{\mathscr{X}; w_1, w_2, \ldots, w_N; p_1, p_2, \ldots, p_N\} \tag{6}$$

ここで，$w_i : \mathscr{X} \to \mathscr{X}$ は変換関数，p_i は合計が 1.0 になる確率，N はフラクタル画像 1 枚を生成するために必要な変換関数の数である。フラクタル画像は，反復関数系を用いた $S = \{\boldsymbol{x}_t\}_{t=0}^{\infty} \in \mathscr{X}$ のように，ドットの繰り返しにより描画され

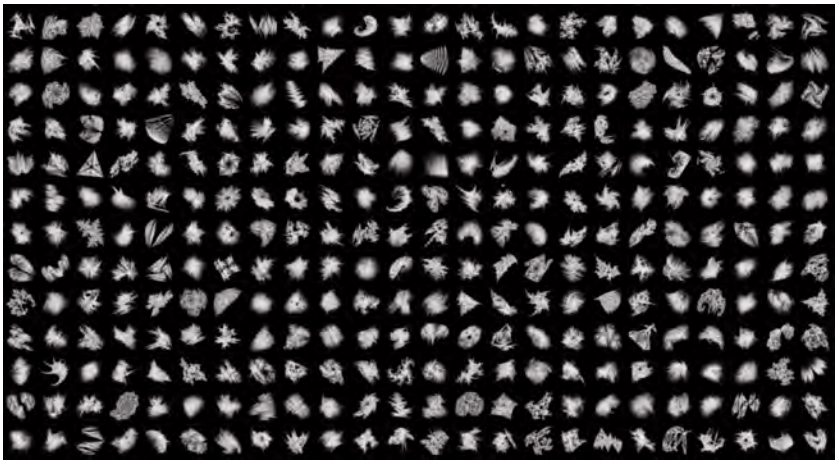

図 5　FractalDB の画像例

る。初期位置 $\boldsymbol{x}_0$ から開始し，$t = 0, 1, 2, \ldots$ と繰り返すうちに，フラクタル画像の各領域を描画する。その際に，(i) 各領域を描画する関数 w^*（$\{w_1, \ldots, w_N\}$）を，あらかじめ生成した確率 $p_i = p(w^* = w_i)$ により選択して，(ii) 新しいドット $\boldsymbol{x}_{t+1} = w^*(\boldsymbol{x}_t)$ を生成しながらフラクタル画像を構成する。

画像データセットを構築するため，2 次元画像空間 $\mathscr{X} = \mathbb{R}^2$ にフラクタル画像を描画する。変換関数としては，次の式に示すアフィン変換を採用する。アフィン変換は，回転や移動など 6 個のパラメータ $\theta_i = (a_i, b_i, c_i, d_i, e_i, f_i)$ で構成される。

$$w_i(\boldsymbol{x}; \theta_i) = \begin{bmatrix} a_i & b_i \\ c_i & d_i \end{bmatrix} \boldsymbol{x} + \begin{bmatrix} e_i \\ f_i \end{bmatrix} \tag{7}$$

フラクタル画像 S は，ドットの繰り返しにより表現されている。

フラクタルカテゴリ生成

画像認識の事前学習において画像と対になるカテゴリの自動生成は，困難な課題である。FractalDB では IFS を構築する際に使用する 6 個のパラメータセット a〜f をランダムに生成し，簡単なクオリティチェックを行うことで，自動でカテゴリを追加する枠組みを提案している。ここで，簡単なクオリティチェックとは，画像全体に占めるフラクタル領域の画素数の割合 r を閾値と比較するチェックであり，論文中では r が 20% 以上の際にカテゴリを追加している。IFS により生成されるフラクタルカテゴリの画像パターンは，パラメータとそれに対応する確率の組み合わせ $\Theta = \{(\theta_i, p_i)\}_{i=1}^N$ で変化する。FractalDB-1k/10k は，この Θ を使い，ランダムにパラメータセットを探索およびクオリティチェックした 1,000 および 10,000 カテゴリを持つデータセットである。パラメータセット探索は，まず $\mathbb{N} = \{2, 3, 4, 5, 6, 7, 8\}$ から N を選択し，次にアフィン変換のパラメータ θ_i を $[-1, 1]^6$ の範囲から N 個分決定する。さらに，確率 p_i を以下のように設定する。

$$p_i = \frac{\det A_i}{\sum_{i=1}^N \det A_i} \tag{8}$$

ここで，$A_i = (a_i, b_i; c_i, d_i)$ はアフィン変換の回転行列である。最後に，設定した $\Theta_i = \{(\theta_i, p_i)\}_{i=1}^N$ を用いてフラクタル画像 S を生成し，画像全体に対する画像占有率 r が閾値を超えた際に Θ_i を新たなフラクタルカテゴリを構築するパラメータセットとして登録する。

実験は膨大なパラメータセット探索が実施されたが，事前学習効果があったパラメータは主に下記の 3 種類であった。

- フラクタルカテゴリ/インスタンス数 —— データセットサイズの増加に直結するパラメータであり，現在までの画像認識の知見にかかわらず，よ

り大きくするほど精度向上が見られた。実験では 10,000 カテゴリまで増やしてもまだ右肩上がりであること，および，カテゴリ内のインスタンス数よりも画像カテゴリ数を増やすほうがより効果があることが確認されている。

- パッチによるフラクタル画像描画 —— フラクタル画像を生成する際，1×1 pixel のドットよりも 3×3 pixel のパッチで描画したほうが精度が大きく向上することも確認されている。
- 重みパラメータ（式 (7) の w_i）による画像インスタンス生成 —— フラクタル画像を生成する際の重みパラメータを変動させることで，画像インスタンスを増加させる効果がある。実験では変動幅を大きくすると，カテゴリに対するインスタンスの形状バリエーションを確保でき，より事前学習効果が大きくなることが明らかになった。

最適と思われるパラメータで構成した FractalDB を用いて，事前学習を行った。CNN のベースモデルとして精度と学習コストのバランスが良い ResNet-50 を用いて，FractalDB 事前学習モデルを構築した。現時点の精度は，ImageNet をはじめとした，人間により教師を与えた大規模画像データセットによる事前学習モデルには敵わない[4] ものの，10 万枚程度の規模に縮小した ImageNet 事前学習モデルや，自己教師あり学習の手法である DeepCluster よりも良い結果を得た。このように，FractalDB を用いることで，非常に高い事前学習効果が得られることが確認できており，自然画像を用いずに自然画像を認識するために必要な視覚特徴表現を獲得できることは驚きの結果である。

[4] CIFAR-10 データセットでは，ImageNet 事前学習が 96.8%，FractalDB 事前学習が最大 94.1% であった。

3 イマドキの認識モデル

このようにコンピュータビジョンの最先端を広くお伝えする需要が生じたことには，1 節で触れたように，深層学習によって画像を理解できるレベルが格段に進化し，さらにそれまでスポットライトが当たっていなかった技術についても広く注目が集められるようになったことが寄与している。

そして近年では，世界で毎日のように生じる新技術，報告に，専門家ですら完全には追えない状況になっている。本節の筆者（牛久）も，「2017 年に出てきた Transformer が 2020 年に本格的にコンピュータビジョンに進出してきたと思ったら，最近 Transformer よりも MLP のほうが高性能だという報告が出ている！ どこまで追ったらいいんだ！」という気になっている。

本節では，2012 年に広く脚光を浴びた CNN のあとに登場してきた Transformer や MLP 系の認識モデルについて，最新状態まで素早くキャッチアップできるような解説を試みる。Transformer の論文自体が “Attention is All You

Need″ というおおよそ学術論文らしくないタイトルであったが，「必要なのは自己アテンション機構だ！」「画像では CNN も組み合わせたほうが有効だ！」「いや全部 Transformer で大丈夫だ！」「いや CNN のほうがやっぱり精度が良い！」「むしろ全部 MLP のほうが精度が良い！ Transformer はオワコン！」といった主語の大きいメッセージが飛び交う状況にある。実際はこれらの認識モデルの進化の中で，生き残り続けたモジュールとそうでないモジュールがあり，その連続した歩みがあって初めて現在に至っている。結局何が高精度化に寄与し，何が寄与していないと現時点で考えられているのか，それらを追いながら現在の状況を俯瞰したい。

3.1 Transformer

Transformer は 2017 年の NIPS[1] で提案されたネットワーク構造で，畳み込みも再帰も存在しない。それまで CNN と RNN が席巻していた機械翻訳の分野に突如現れた新星である[2]。しばらくは自然言語処理分野での研究が進んでいたが，2019 年頃からコンピュータビジョンを含む他分野への応用が広がり出している。

「MLP を，すべての関数を近似できる [20] ものとして学んだのに，実は汎化性能が出にくいと言われ，畳み込みという信号処理分野以外には聞き慣れない概念や逆の畳み込み（deconvolution）というさらに謎の概念を消化し，Long Short-Term Memory（LSTM）の複雑怪奇な構造や再帰のために複雑になる誤差逆伝播にもようやく慣れてきたと思ったら，また見慣れないネットワーク構造が出てきてしまった。綺麗な図ではあるけれど，何をやっているのかわかりにくい…」，そう思っている人は少なくないのではないだろうか。少なくとも筆者は，Transformer の出現当初にそのようなことを思った。

そこでまず，この項では，Transformer の基本的な構造と働きについて解説していきたい。

[1] NIPS は，2018 年から NeurIPS と呼ばれるようになった。

[2] ちなみに，この論文は Google から発表されたが，Google 翻訳の製品版を開発しているチームからではない。このあたりの層の厚さが Google の強さの一端に表れているように感じるが，これは別の話題。

Transformer の大まかな形

早速であるが，図 6 に Transformer の大まかな形を示す。2017 年の Vaswani らの論文の Figure 1 を覚えている方は「だいぶん見た目が違う」と思うかもしれないが，まずはいったん新たな気持ちでこの図をご覧いただきたい。Transformer のみで CNN より優れた画像認識精度を達成した Vision Transformer（ViT）[21] は，コンピュータビジョン分野でもかなり話題になったが，実は Vision Transformer は元の Transformer よりもずっと簡単な形をしている。これはなぜかと言うと，Vaswani らのオリジナル Transformer は機械翻訳のための Encoder-Decoder 型の Transformer であるのに対して，ViT は Encoder のみ

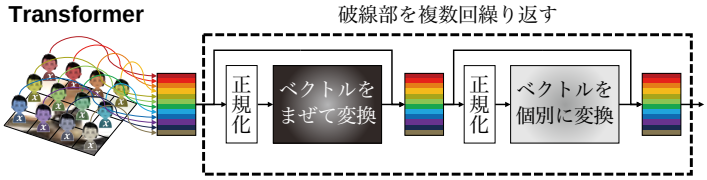

図 6　Transformer の大まかな形。ViT（Vision Transformer）などはまさにこの形である。ジグソーパズルによる自己教師あり学習の例のように，画像を格子状の線でバラバラにし，各パッチ内の縦×横×RGB の各ピクセル値をベクトル状にする。さらに各パッチの縦と横の位置を表す情報（Positional Encoding）をベクトルに足して，これに破線部分のモジュールを繰り返し適用していく。出力は入力と同数のベクトルであるが，例えば ViT のように識別させたい場合は，機械的に最初のベクトル（図中であれば赤いベクトル）にさらに MLP を繋いでクラス事後確率を算出させる。

の Transformer だからだ。

図 1 と同様に画像を格子状の線で切断して，各パッチ内の縦×横×RGB の各ピクセル値をベクトル状にする。例えば一つのパッチが 16 ピクセル四方のカラー画像だった場合，16×16×3 = 768 次元のベクトルがたくさん出てくる。このベクトルと，各パッチの縦と横の位置を表す情報（Positional Encoding）をベクトルに足して，これに破線部分のモジュールを繰り返し適用していく。出力は入力と同数のベクトルであるが，例えば ViT のように識別させたい場合は，機械的に最初のベクトル（図中であれば赤いベクトル）にさらに MLP を繋いでクラス事後確率を算出させる[3]。

Transformer の要点は以上である。説明にいっさいアテンションが出てこないのを疑問に思った読者も多いと思うが，最近の MLP 系ネットワーク構造の台頭を踏まえると，重要だったのは必ずしもアテンションだけではない。

[3] ViT の厳密な説明をすると，ViT に入力する最初のベクトルは画像によらない定数が入った「カラ」のベクトルである。したがって，ViT にはパッチ数 + 1 本のベクトルが入力される。

破線部分の中身を考える

さて，次に破線部分の中身を見ていこう。図のとおり，破線部分の中身は 2 回のスキップ接続を含んでいる。これは ResNet と同じなので，なじみ深い読者も多いのではないだろうか。そしてスキップしていないほうのパスで何をやっているかと言うと，一つ目が正規化と「**ベクトルをまぜて変換**」，二つ目が正規化と「**ベクトルを個別に変換**」，というかなりシンプルなパイプラインになっている[4]。

都合上，二つ目の「ベクトルを個別に変換」というブロックから説明する。これは Vaswani らの論文で言うところの Feed-Forward Network，つまりは MLP である。この論文では線形変換→ ReLU →線形変換という形であり，各ベクト

[4] Vaswani らが最初に発表した論文では正規化があとだったが，その後の複数の論文で正規化を最初にやったほうが精度が良いという報告がなされており，ViT でもこの順番が採用されている。したがって，本稿では，この順序が標準の Transformer であるとして解説している。

ルに対して共通のパラメータの MLP が適用される。

そして次に，一つ目の「ベクトルをまぜて変換」というところである。ここが Transformer では虎の子の Multi-Head Attention を適用するところである。図 6 でのカラフルなベクトルを並べてみたのが図 7 である。図 7 および図 8 では，この中のあるベクトル（左から 6 個目の緑のベクトル）を取り上げながら，Self-Attention の計算方法についての概要を示している。以下では，これらの図に即して具体的な式展開を述べる。

[5] 正規化には Layer Normalization [22] を用いる。

1) **クエリの計算**：この正規化済み[5] のベクトル $\boldsymbol{x} \in \mathbb{R}$ に対して行列 W^{Q} を乗じて，他のベクトルを「検索」するためのクエリ

$$\boldsymbol{q} = W^{\mathrm{Q}}\boldsymbol{x} \tag{9}$$

を計算する。

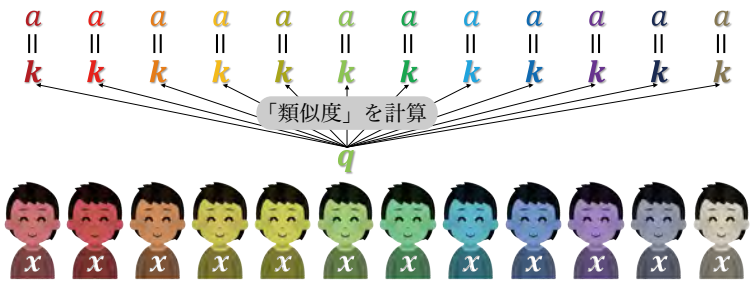

図 7　Self-Attention の計算。ここでは，1) クエリの計算から，2) キーの計算，3) アテンション（正規化された類似度）の計算までを図示している。

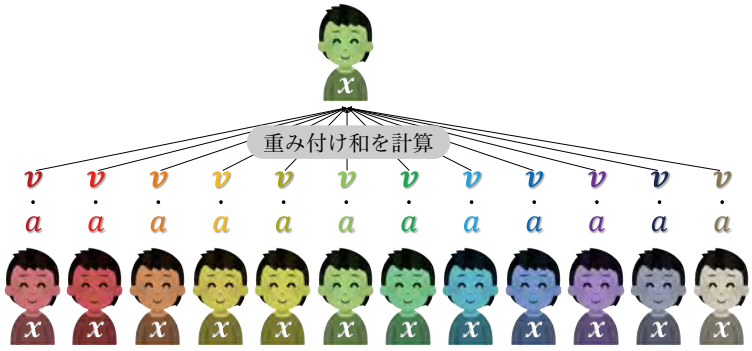

図 8　Self-Attention の計算の続き。ここでは，4) バリューの計算と，5) ベクトルの変換を示している。ここまでの計算は，内積（dot-product）を定数倍（scaled）して用いているので，Scaled Dot-Product Attention とも呼ばれている。

2) **キーの計算**：このベクトルを含むすべてのベクトル $\boldsymbol{x}$ を別の行列 W^K と掛け合わせることで，「検索」の対象であるキーが求められる。

$$\boldsymbol{k} = W^K \boldsymbol{x} \tag{10}$$

3) **各ベクトルへのアテンションの計算**：アテンション a は，このクエリとキーの内積を次元数と softmax で正規化した形で算出される。つまり，クエリとキーの正規化された類似度がアテンションに相当するわけである。

$$a = \mathrm{softmax}\left(\boldsymbol{q}^\top \boldsymbol{k} / \sqrt{d}\right) \tag{11}$$

4) **バリューの計算**：新たな行列 W^V を乗じることで，すべてのベクトルのバリューを計算する。

$$\boldsymbol{v} = W^V \boldsymbol{x} \tag{12}$$

5) **アテンションによるベクトルの変換**：上記 3) で求めたアテンション a とバリュー $\boldsymbol{v}$ の重み付け和を加算することによって，更新分のベクトル $\delta\boldsymbol{x}$ を得る。

$$\delta\boldsymbol{x} = \sum a\boldsymbol{v} \tag{13}$$

そして最後に，スキップ接続の部分で，元の正規化していないベクトル $\boldsymbol{x}$ に $\delta\boldsymbol{x}$ を加える。

$$\boldsymbol{x} \to \boldsymbol{x} + \delta\boldsymbol{x} \tag{14}$$

ここまでが, Self-Attention によって単一のベクトルを更新する部分の説明である。これをすべてのベクトルに対して行うのが，Scaled Dot-Product Attention である。内積（dot-product）を次元数の平方根で割るところが，Scaled という名前がついている所以である。あらためて先ほどのベクトル $\boldsymbol{x}$ を列ベクトルとして横に並べた行列 X としてみると，

$$X \to X + VA \tag{15}$$

と書ける。ここで，V は各ベクトルのバリュー v を横に並べた行列，A は i 番目のベクトルから他のベクトルへのアテンションを格納した列ベクトル $\boldsymbol{a}_i$ を横に並べた行列である。V は W^V と X の積，A は W^Q と X の積，W^K と X の積をさらに掛け合わせて計算していることを思い出すと，VA は X について 3 次の項と見なせる。

さて，この Scaled Dot-Product Attention は，行列 W^Q, W^K, W^V をそれぞれ一つずつ用意して計算している。図 9 でも示しているように，これらの行列

図 9　Multi-Head Attention。図 8 の Scaled Dot-Product Attention を h 個の行列 W^Q, W^K, W^V で行い，得られた h 本のベクトル更新結果（$\sum av$）を直列して行列 W^O で射影変換し，元の d 次元に戻す。

を複数ずつ用意して，各ベクトルに加算する $\sum av$ も複数出てくるものを直列し，新たな重み行列 W^O で d 次元に戻すのが，Multi-Head Attention である。

このようにして，Transformer では Self-Attention を活用して「ベクトルをまぜて変換」することを可能にしている。ViT では，ImageNet の画像だけを学習しても ResNet や EfficientNet といった最近の CNN ネットワーク構造よりは精度が劣るが，JFT-300M[6] を学習すれば，それらの CNN に比肩する精度が出ると報告されている。ここからより “少数” のデータでも良い精度の出る Transformer 画像分類の研究がにわかに活気づき，例えば Data-efficient image Transformers（DeiT）[23] は，知識蒸留などを利用して，ImageNet だけを学習した場合でも従来の CNN と比肩する精度を達成している。

[6] JFT-300M は ViT でも用いられたデータセット。ImageNet の典型的な訓練データセットは通常 1,000 クラスで 100 万枚超程度だが，全体では 2 万クラスで 1 千万枚超の画像を含んでいる。これに対して，JFT-300M は 3 億枚程度の画像を保持する。

画像分類は ViT や DeiT などによって，Transformer のみで State-of-the-art な精度を出せることがわかってきた。そして，そのほかのコンピュータビジョンタスクでも，CNN と Transformer の組み合わせによって，それまでの State-of-the-art が急速に塗り替えられている。画像データとテキストデータを融合的に理解するビジョン&ランゲージ分野では，ViLBERT [24] などに代表される表現学習手法が数多く提案されているし，物体検出では DETR（DEtection TRansformer）[25]，領域分割では FPT（Feature Pyramid Transformer）[26] などを筆頭とした手法が次々と提案されている。

3.2　MLP の逆襲

そんな中，2021 年 5 月に立て続けに MLP 系のネットワーク構造を提案するプレプリントが arXiv 上に出現し，機械学習界隈が騒然となった。まず，Google の ViT を提案していた著者ら（とほぼ同一のチーム）から，MLP-Mixer [27] という MLP しか使わないネットワークが提案された。図 10 にあるように，ViT

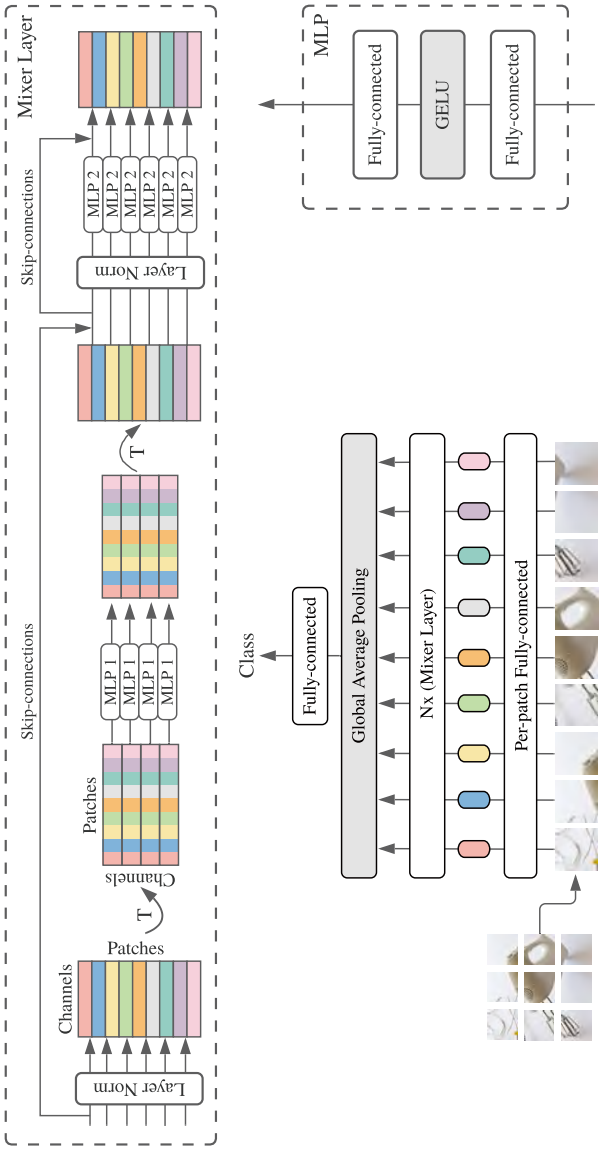

図 10　MLP-Mixer のネットワーク構造。主要モジュールである Mixer Layer と全体の画像認識パイプライン，MLP モジュールの概要からなる (https://arxiv.org/abs/2105.01601 の Figure 1)。一見すると，CNN とも Transformer とも大きく違うネットワーク構造に見える。

で CNN から畳み込みが消えたように，ViT からアテンション機構すらなくなってしまった。代わりに Mixer Layer というところでは，ベクトルが並んだ行列を転置して MLP に入力し，さらに転置して（戻して）加算する，という謎なことをやっているように見える。

このプレプリントでは，画像識別課題で ImageNet や JFT-300M を用いて学習した結果，MLP-Mixer は ViT に比べて少し精度で劣るものの，推論や学習がかなり速いというメリットがあると報告されている。例えば，16 ピクセル四方のパッチを用いた 24 層 3 億パラメータの ViT-L/16 という ViT の大きなモデルを用いて JFT-300M を学習したとき，ImageNet 上での 1,000 クラス識別の正解率が 87.76% となっている。一方で，同じパッチサイズと層数（ただし，そうするとパラメータ数は 2 億くらいである）の MLP-Mixer モデル Mixer-L/16 で学習した結果は，正解率が 86.65% である。なお，ViT-L/16 が 1 秒に TPU1 コアで推論できる画像枚数が 32 枚なのに対して，Mixer-L/16 は TPU1 コアで 1 秒に 105 枚の画像を推論できる。また，JFT-300M を 14 エポック学習するのにかかった時間は ViT-L/16 が 650TPU 日，Mixer-L/16 が 450TPU 日となっている。

MLP-Mixer が arXiv に公開されたと思ったら，同様に CNN や Transformer ではない MLP 系のアーキテクチャを提案するプレプリントが瞬く間に増えた。また，MLP-Mixer に影響を受けた手法についてのプレプリントも，早速出てきている。2021 年 7 月 31 日現在までに観測されているだけでも，表 2 に挙げるように 10 本以上のプレプリントが公開されている。

すべてに触れる紙幅は残念ながらないが，表中にもあるとおり，MLP-Mixer に非常によく似た手法が多い。そこで，本稿では，その中でもユニークかつ高性能なネットワーク構造として，gMLP を紹介したい。gMLP のプレプリントの Figure 1 で紹介しているメインモジュールの引用を図 11 に示す。MLP-Mixer は CNN 系論文に多い左から右の情報の流れで図示されていたのに対して，gMLP は Transformer と同様に下から上の情報の流れで図示されている。ここでは，Channel Proj というものを二つと Spatial Proj というものを含む Spatial Gating Unit なるユニークなユニットが用いられている。

gMLP のプレプリントでは，コンピュータビジョンと自然言語処理の実験を行っている。コンピュータビジョンの実験としては，ImageNet の 1,000 クラス分類が行われている。MLP-Mixer のように，ImageNet の全 21k クラスの画像 1 千万枚超や，JFT-300M で学習した結果は存在しない。ViT では CNN の画像認識手法の精度に比肩するためには，ImageNet では足りず，JFT-300M が必要だった。一方，gMLP では，ImageNet の 1,000 クラス画像 100 万枚だけでも，EfficientNet や Transformer に比肩する性能が出ている。この流れは，ViT

表 2　MLP 系のネットワーク構造を提案するプレプリントの一覧。すべて arXiv 上に公開されている。またもちろん公開年はすべて 2021 年である。米中を中心とした世界中の機関から同時多発的に同じような手法が提案されていることがわかる。

公開日	手法名	概　要
5 月 4 日	MLP-Mixer	Google で ViT を開発したチームによる MLP 系論文の草分け的存在。
5 月 5 日	RepMLP [28]	中国精華大を中心としたチームから。CNN と MLP の組み合わせ。
5 月 5 日	EA [29]	これも中国精華大によるチーム。EA（External Attention）はほぼ行列積。
5 月 6 日	FF [30]	Oxford の学生が単著で公開。MLP-Mixer と非常に似ていて，“Do You Even Need Attention?” という挑戦的なタイトルの論文である。
5 月 7 日	ResMLP [31]	Facebook を中心としたチーム。MLP-Mixer と非常に似ている。
5 月 17 日	gMLP [32]	Google で LSTM による機械翻訳や EfficientNet を開発した Quoc V. Le らによる手法。
6 月 2 日	Container [33]	アレン人工知能研究所と香港中文大学によるチーム。CNN と Transformer と MLP を一般化。
6 月 14 日	S^2-MLP [34]	百度の研究所によるチームから。MLP-Mixer に画像空間上の位置変動を与える手法。
6 月 23 日	ViP [35]	シンガポールの産学チームから。MLP-Mixer のベクトル変換に置換も入れる。
7 月 18 日	AS-MLP [36]	上海科技大と Tencent のチーム。動機は S^2-MLP に似ている。

からより少数のデータで訓練可能な Transformer 型画像認識手法が盛んに研究されている流れと似ていて，他の MLP 系のネットワーク構造を提案するプレプリントでも，ImageNet の学習だけで従来の CNN や Transformer 系ネットワーク構造と比肩する精度を達成できることを長所の一つに挙げているものが多い[7]。

さらに付け加えると，MLP-Mixer を含む MLP 系の論文は，Transformer で用いられていた Positional Encoding を採用していない。Attention もいらない，Positional Encoding もいらない，やっていることは MLP だけ，と言うと，Transformer がオワコン化しているように見えても無理はないと思う。

[7] おいそれと JFT-300M のような巨大なデータを何度も学習できる計算基盤がある機関が限られている，というのもありそうだが…。

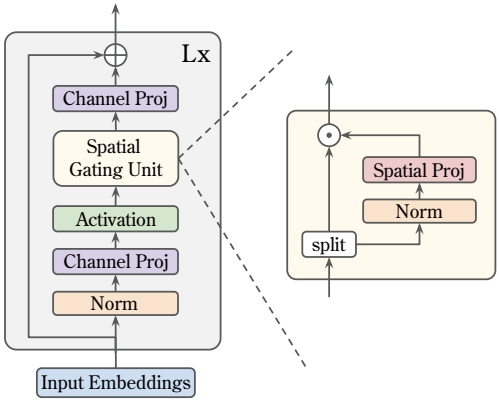

図 11　gMLP のネットワーク構造の一部。MLP-Mixer が左から右に情報処理の流れを記載しているのに対して，gMLP では Transformer と同じように下から上の方向で記載がある（https://arxiv.org/abs/2105.01601 の Figure 1 の一部）。

3.3　Transformer とそんなに変わらない MLP 系手法たち

では，本当に Transformer と MLP 系のネットワーク構造は似て非なるものだろうか。図 6 に示した Transformer の大まかな形と同じ解釈の仕方で MLP-Mixer を図示してみた結果が，図 12 である。画像を格子状の線でバラバラにし，各パッチ内の縦×横×RGB の各ピクセル値をベクトル状にして，破線部分のモジュールを繰り返し適用していく。出力は入力と同数のベクトルであるが，例えば ViT のように識別させたい場合は，機械的に最初のベクトル（図中であれば赤いベクトル）にさらに MLP を繋いでクラス事後確率を算出させる[8]。ここまでの説明が図 6 のキャプションとほとんど変わらないことに気づく読者も多いと思うが，実際ほとんど違わない。強いて言えば，先ほど述べたとおり，Positional Encoding がなくなったことも違いになる。

そして，「ベクトルを個別に変換」する部分は，線形変換 → GELU（Gaussian Error Linear Unit）[37] → 線形変換である。Transformer では線形変換 → ReLU → 線形変換だった。つまり，活性化関数が ReLU から GELU になっただけであ

[8] 厳密には，ViT ではクラス識別用の「カラ」のベクトルを画像パッチのベクトル群に追加している。また，gMLP では，Global Average Pooling でベクトル群を集約して識別している。このあたりの選択肢は Transformer でも MLP 系でも同様である。

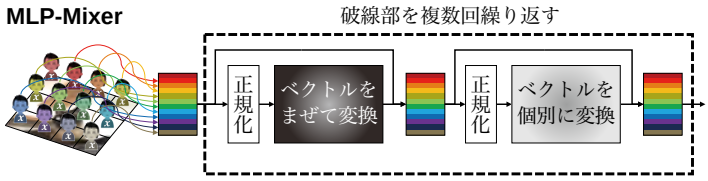

図 12　MLP-Mixer を図 6 と同じように解釈した図。図 6 との違いは，実は左上の手法名だけである。

る。ReLU と GELU の違いはと言えば，入力が 0 のときの勾配が ReLU では非連続であるのに対して，GELU では滑らかになっている，というだけで，他の部分はほぼ同じ概形である。さらに，Transformer でも BERT や GPT-3，ViT で GELU が使われているので，これはまったく同じと言って差し支えない[9)]。

[9)] さらに付記すると，Transformer の ReLU が GELU になったからと言って，精度が向上するわけではないという実験的報告 [38] もなされている。

一番大きな違いは，「ベクトルをまぜて変換」する部分である。式 (15) でも示したとおり，Transformer ではベクトルの集合を表現した行列 X が $X \to X + VA$ と更新されるのが「ベクトルをまぜて変換」する方法であった。MLP-Mixer では，X を転置して「ベクトルを個別に変換」する方法とまったく同じ方法を適用して変換した後，転置し直して X に加算する。すなわち，

$$X \to X + (W_2\,\mathrm{GELU}(W_1 X^\top))^\top \tag{16}$$

と更新する。ここで W は射影行列であり，GELU は要素ごとに GELU を適用する関数である。式 (15) では X の各ベクトルの 3 次の積が登場するのに対して，ここでは X 自体は一度しか用いられないのが大きな違いである。

しかしながら，ここでおもしろい Transformer のバリエーションとして，Synthesizer [39] を紹介したい。Synthesizer は Transformer の Self-Attention の計算が重いことを指摘し，かつその計算を大幅に省略できるのではないかという提案を行っている。Transformer で $X \to X + VA$ と更新される際に

$$A = \mathrm{softmax}(W_2\,\mathrm{ReLU}(W_1 X)) \tag{17}$$

とするのが Dense Synthesizer であり，さらに A を乱数にしてしまうのが Random Synthesizer である。V が $W^V X$ で計算されることを思い出すと，Dense Synthesizer は X の 2 次の情報を含んでいる。一方で，Random Synthesizer は結局

$$X \to X + W^V X R \tag{18}$$

と，A と同じ大きさの乱数行列 R で A を代用してしまう。つまり，MLP-Mixer が「ベクトルをまぜて変換」する式 (16) の線形なバージョンが，Random Synthesizer で「ベクトルをまぜて変換」する式 (18) に相当するわけである。

そして，Transformer で採用されていた Positional Encoding が MLP 系で採用されない理由は簡単で，Transformer では「ベクトルをまぜて変換」するのも「ベクトルを個別に変換」するのも位置情報を利用したいと考えていて，かつそれぞれのモジュールはベクトルの順番に依存しなかった。一方で，MLP 系の手法では，「ベクトルをまぜて変換」するにはベクトルの位置情報が使えていればそれで十分と考えており，「ベクトルを個別に変換」する際は Positional Encoding のないベクトル $\boldsymbol{x}$ に対して共通のパラメータの MLP を適用してい

る。そして，MLP 系の手法で「ベクトルをまぜて変換」する際は，X を転置して W で射影変換するので，ベクトル $\boldsymbol{x}$ の行列 X 内でのインデックスに応じて，$\boldsymbol{x}$ を変換するパラメータがそれぞれ異なっている。つまり，ベクトルに位置情報を埋め込まずとも，ベクトル $\boldsymbol{x}$ の行列 X 内の位置に応じて異なる線形変換が学習される。これであれば，やはり「ベクトルをまぜて変換」する際にも Positional Encoding は不要となる。

このように，Transformer と MLP-Mixer の本質的な違いは「ベクトルをまぜて変換」する方法だけであり，他は同一と言って差し支えない。さらには，MLP-Mixer における変換は，Random Synthesizer の非線形かつパラメータ化された変換と見なせるのである。

次に，gMLP を考えてみよう。以下の結論として，gMLP の大まかな流れを先に図示すると，図 13 のようになっている。なるほど，見た目でもさすがに Transformer や MLP-Mixer と少し違っている。まず目につくのが，「ベクトルを個別に変換」するところが 2 回あることだろう。ただ実は，これらの個別に変換する部分は，1 回目が線形変換 → ReLU で，2 回目が線形変換のみとなっている。つまり，Transformer/MLP-Mixer では，線形変換 → ReLU → 線形変換と，一度に「ベクトルを個別に変換」する部分を実行するのに対して，gMLP ではその真ん中あたりで変換を 2 ステップに分割し，その間に「ベクトルを個別に変換」するモジュールを割り込ませているに過ぎない。

やはり，本質的な違いは「ベクトルをまぜて変換」するところにある。ここの変換（Spatial Gating Unit; SGU）s は，ベクトル $\boldsymbol{x}$ が横に並んだ行列 X に対して以下のように計算する。

$$X \to X \odot s(X) = X \odot (XW) \tag{19}$$

ここで，$\odot$ は行列同士の要素積であり，W は X 内のベクトルの線形和をとる（つまりベクトルをまぜる）ための正方行列である。この手法の着想は LSTM や GRU（Gated Recurrent Unit）などから得ている。LSTM では，短期記憶と長期記憶を保持するために，長期記憶についての誤差逆伝播時に昔の時点の入力

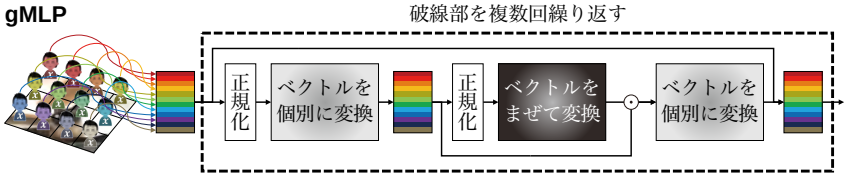

図 13　gMLP を図 6 と同じように解釈した図。途中にある ⊙ は，ここの合流部分だけは，要素ごとの和ではなく要素ごとの積であることを意味している。

まで誤差の勾配を失わないような工夫を入れてある。これがゲートと呼ばれる部品であり，通常だと変換をスタックして $f \circ g(X)$ と計算していくために勾配が爆発したり消失したりするところが，$f(X) \odot g(X)$ の形で表現される。このため，ゲートを用いると各変換 f や g で勾配が残りやすいという性質がある。いわば，ResNet などに見られるスキップ接続の掛け算版に相当するのが，このSGU である。

そして，gMLP の論文中でも議論されているように，LSTM などのゲートを他のネットワークに持ってくる SGU のアイデアは GLU（Gated Linear Unit）[40] として提案されており，Transformer との親和性も報告されている [41]。何が違うかと言えば，GLU はベクトルごとの活性化関数として使われるのが基本であるのに対して，SGU は各ベクトルのある次元の要素に対して使われる。言い換えれば，GLU は行列 X の各列に対して作用するのに対して，SGU は各行に作用する。

3.4 Transformer は過去の遺物になるのか？

MLP 系のプレプリントが大量に出現し，そのいずれでもアテンション機構のないさらにシンプルなネットワーク構造が提案されていることはわかった。その一方で，その変換方法，つまり “Transform” する方法のパイプラインは，Transformer のそれとほぼ同じ形で MLP 系の手法でも生き続けていることもわかった。画像や文をパッチや単語の集合として扱うという概念は，2000 年前後の Bag-of-Words の考え方から生まれたものであり，また，勾配の爆発や消失を防ぐ正規化およびスキップ接続やゲート関数は，近年の CNN や RNN での進歩の中で見出されたテクニックである。Transformer は，こうした技術体系からベクトルの集合を決まった順序で繰り返し “Transform” するパイプラインを見出した点に価値がある。

そして，gMLP を提案している論文では，完全に MLP のみで構築した提案手法である gMLP のほかに，ごくわずかな回数だけアテンション機構を導入した aMLP を提案している。ベクトル間のインタラクションが推論に必要となる自然言語処理のタスクにおいて，gMLP は Transformer よりやや精度が劣るものの，aMLP であれば Transformer を上回る優れた精度を叩き出せることをさらっと報告している。

もともとアテンション機構は，機械翻訳や画像キャプション生成の際に出力する単語を一つ一つ推定する際に 1 回ずつ呼び出されていたものである。Transformer は，このアテンション機構を「ベクトルをまぜて変換」するためのモジュールとして重用したわけであるが，少量だけ呼び出す分には今後の MLP 系のネットワーク構造でも寄与が認められる可能性が高いと筆者は見ている。

Transformer が “Attention Is All You Need” という浅いタイトルを付けてしまったがゆえに，“○○ Is All You Need” という別の浅い論文が続出したり，アテンション機構だけに注目が集まる（がゆえにアテンションがいらない，つまり Transformer ＝オワコンと思われる）リスクが大きかったり，という状況になっている。タイトルの良し悪しはともかくとして，貢献したのが何であり，何が今後も生き残るかについては，冷静に判断していく必要がある。

未知の有機化合物を測定（質量分析）した系列データから，その化合物が何であるかを当てる。その際には，データを用いた学習と人間の知識を組み合わせる —— こう言うと「すわ最新の深層学習の成果か？！」となりそうだが，これは今から半世紀以上前に取り組まれた Dendral という人工知能プロジェクトのことである。そして，そのプロジェクトの中でも，新たな観測情報に対して「関連する経験」をもとに化合物が何であるかを推論することが求められた。そのような汎用性を人工知能システムに持たせるために必要なものとして，当時ラトガーズ・ニュージャージー州立大学に在職していた Tom M. Mitchell 博士（現 CMU 教授）が説いたのが，inductive leap（後の inductive bias，つまり帰納バイアス）である。

機械学習に期待されるのは，学習時に見たことがあるデータに対する推論ではなく，学習時に見たことがないデータに対しても「それらしく」なされる推論である。この「それらしさ」が帰納バイアスであり，何も深層学習に特有の概念ではない一般的な概念である。そして，この帰納バイアスが強い機械学習手法ほど，少ない学習データからでも何らかの「関連する経験」を想起できるような学習済みモデルを獲得できる。もちろんこうしたモデルでも，「関連する」と思って想起した「経験」が誤った推論結果になる（例えば，「似ている」と思った系列データがまったく異なる化合物のものである）リスクは存在する。

ここまでの機械学習手法の発展の歴史は，この帰納バイアスを絞ってきた歴史でもある。帰納バイアスを小さくして，代わりに大量のデータで大量のパラメータを学習すればよい，というような傾向を生んだのが今の Transformer であり，MLP 系のネットワーク構造である。一方で，かつてのようにすべての画像の画素を 1 列に並べて，2 層の全結合型ニューラルネットワークを学習すればよい，といったような状況には戻らないであろう。ここに辿り着くまでに得られた知識 —— すなわち，画像や文をそれらの構成要素の集合として扱うのがセオリーで，正則化とスキップ接続を用いると深層学習が安定し，そこでは構成要素ごとの変換とインタラクションが必要になる，という知見 —— は，引き続き維持され，磨き続けられるだろう。

謝辞

「コンピュータビジョン（CV）の動向 2021」[1] を一部参考にさせていただいた。感謝する。

[1] https://gihyo.jp/dev/column/newyear/2021/computer-vision-trends?page=1

参考文献

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems (NIPS) 25*, pp. 1097–1105, 2012.

[2] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *The IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

[3] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis. Mastering the game of Go with deep neural networks and tree search. *Nature*, Vol. 529, No. 7587, pp. 484–489, 2016.

[4] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pp. 5998–6008, 2017.

[5] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A large-scale hierarchical image database. In *The IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 248–255, 2009.

[6] B. Thomee, D. A. Shamma, G. Friedland, B. Elizalde, K. Ni, D. Poland, D. Borth, and L.-J. Li. Yfcc100m: The new data in multimedia research. *Commun. ACM*, Vol. 59, No. 2, pp. 64–73, 2016.

[7] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *The IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–9, 2015.

[8] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations (ICLR)*, 2015.

[9] B. Zoph and Q. V. Le. Neural architecture search with reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2017.

[10] M. Tan and Q. Le. EfficientNet: Rethinking model scaling for convolutional neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, Vol. 97 of *Proceedings of Machine Learning Research*, pp. 6105–6114. PMLR, 09–15 Jun 2019.

[11] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *The International Conference on Learning Representations (ICLR)*, 2021.

[12] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning (ICML)*, 2020.

[13] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick. Momentum contrast for unsupervised visual representation learning. In *The IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[14] X. Chen and K. He. Exploring simple siamese representation learning. In *The IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[15] M. U. Gutmann and A. Hyvärinen. Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics. *Journal of Machine Learning Research*, Vol. 13, No. Feb, pp. 307–361, 2012.

[16] H. Kataoka, K. Okayasu, A. Matsumoto, E. Yamagata, R. Yamada, N. Inoue, A. Nakamura, and Y. Satoh. Pre-training without natural images. In *ACCV*, 2020.

[17] K. Yang, K. Qinami, L. Fei-Fei, J. Deng, and O. Russakovsky. Towards fairer datasets: Filtering and balancing the distribution of the people subtree in the ImageNet hierarchy. In *Conference on Fairness, Accountability, and Transparency*, 2020.

[18] A. Birhane and V. U. Prabhu. Large image datasets: A pyrrhic win for computer vision? Winter Conference on Applications of Computer Vision (WACV), 2021.

[19] S. Yun, S. J. Oh, B. Heo, D. Han, J. Choe, and S. Chun. Re-labeling ImageNet: From single to multi-labels, from global to localized labels. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2021.

[20] G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, Vol. 2, No. 4, pp. 303–314, 1989.

[21] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2020.

[22] J. L. Ba, J. R. Kiros, and G. E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[23] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, pp. 10347–10357. PMLR, 2021.

[24] J. Lu, D. Batra, D. Parikh, and S. Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *Advances in Neural Information Processing Systems*, Vol. 32, pp. 13–23, 2019.

[25] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko. End-to-end object detection with transformers. In *European Conference on Computer Vision*, pp. 213–229. Springer, 2020.

[26] D. Zhang, H. Zhang, J. Tang, M. Wang, X. Hua, and Q. Sun. Feature pyramid transformer. In *European Conference on Computer Vision*, pp. 323–339. Springer, 2020.

[27] I. Tolstikhin, N. Houlsby, A. Kolesnikov, L. Beyer, X. Zhai, T. Unterthiner, J. Yung, D. Keysers, J. Uszkoreit, M. Lucic, et al. Mlp-mixer: An all-MLP architecture for vision.

arXiv preprint arXiv:2105.01601, 2021.

[28] X. Ding, X. Zhang, J. Han, and G. Ding. RepMLP: Re-parameterizing convolutions into fully-connected layers for image recognition. *arXiv preprint arXiv:2105.01883*, 2021.

[29] M.-H. Guo, Z.-N. Liu, T.-J. Mu, and S.-M. Hu. Beyond self-attention: External attention using two linear layers for visual tasks. *arXiv preprint arXiv:2105.02358*, 2021.

[30] L. Melas-Kyriazi. Do you even need attention? A stack of feed-forward layers does surprisingly well on ImageNet. *arXiv preprint arXiv:2105.02723*, 2021.

[31] H. Touvron, P. Bojanowski, M. Caron, M. Cord, A. El-Nouby, E. Grave, A. Joulin, G. Synnaeve, J. Verbeek, and H. Jégou. ResMLP: Feedforward networks for image classification with data-efficient training. *arXiv preprint arXiv:2105.03404*, 2021.

[32] H. Liu, Z. Dai, D. R. So, and Q. V. Le. Pay attention to mlps. *arXiv preprint arXiv:2105.08050*, 2021.

[33] P. Gao, J. Lu, H. Li, R. Mottaghi, and A. Kembhavi. Container: Context aggregation network. *arXiv preprint arXiv:2106.01401*, 2021.

[34] T. Yu, X. Li, Y. Cai, M. Sun, and P. Li. S^2-MLP: Spatial-shift MLP architecture for vision. *arXiv preprint arXiv:2106.07477*, 2021.

[35] Q. Hou, Z. Jiang, L. Yuan, M.-M. Cheng, S. Yan, and J. Feng. Vision permutator: A permutable MLP-like architecture for visual recognition. *arXiv preprint arXiv:2106.12368*, 2021.

[36] D. Lian, Z. Yu, X. Sun, and S. Gao. As-MLP: An axial shifted MLP architecture for vision. *arXiv preprint arXiv:2107.08391*, 2021.

[37] D. Hendrycks and K. Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

[38] S. Narang, H. W. Chung, Y. Tay, W. Fedus, T. Fevry, M. Matena, K. Malkan, N. Fiedel, N. Shazeer, Z. Lan, et al. Do transformer modifications transfer across implementations and applications? *arXiv preprint arXiv:2102.11972*, 2021.

[39] Y. Tay, D. Bahri, D. Metzler, D.-C. Juan, Z. Zhao, and C. Zheng. Synthesizer: Rethinking self-attention for transformer models. In *International Conference on Machine Learning*, pp. 10183–10192. PMLR, 2021.

[40] Y. N. Dauphin, A. Fan, M. Auli, and D. Grangier. Language modeling with gated convolutional networks. In *International Conference on Machine Learning*, pp. 933–941. PMLR, 2017.

[41] N. Shazeer. Glu variants improve transformer. *arXiv preprint arXiv:2002.05202*, 2020.

うしく よしたか（オムロン サイニックエックス株式会社/株式会社 Ridge-i）
いのうえ なかまさ（東京工業大学）
かたおか ひろかつ（産業技術総合研究所）