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. 37（2024）
技術ノート

事前深層学習モデルの転移学習による能力比較（第１報）

画像分類のための事前深層学習モデル

廣川　勝久

Transfer Learning and Fine-tuning for Pre-trained Deep-learning Models (Ⅰ)

Image Clustering using Pre-trained Deep-learning Models

HIROKAWA Katsuhisa

近年、PSPNet [arXiv: 1612.01105, (2017)]のようにバックボーンに事前学習モデルを実装した高性能な深層学習モデルが各種提案されている。バックボーンに使用されるような高性能な深層学習モデルは、事前に大量の教師画像と膨大な計算時間を使った学習により提供されている。本報告では、3 種類の事前学習モデルについて、少量の教師画像と少ない処理コストで他に転用するための方法である転移学習とファインチューニングにより学習を行い、その学習能力の評価を行った。

キーワード：転移学習，ファインチューニング，ResNet, DenseNet, ViT, AugMix, Mixup, Cutmix

1. 緒　　言

スキップ接続や ReLU 関数による技術的ブレイクスルーにより大規模に深層化されたニューラルネットワークの学習が可能となり、著しい学習能力の向上が図られた。深層学習による画像認識、自然言語処理、強化学習、画像生成などにおいては、人間の能力を超えるほど高性能なモデルが提案されている。一方このように高性能なモデルでは、大量の学習データ、長時間の学習、高性能なコンピュータが必要となる。このような高性能な深層学習モデルの多くは、事前学習済みの基盤モデルとして提供されている。

これまでの報告では、PSPNet[1]などの画像領域セグメンテーションの能力の評価結果について述べた[2,3]。この PSPNet は学習済み ResNet: Residual Neural Networks[4]をバックックボーンに転用したモデルである。本稿では、PSPNet のバックボーンにも使われた事前学習モデルである ResNet と DenseNet: Densely Connected Convolutional Networks[5]、 ViT: Vision Transformer[6]の 3 モデルを単独に用いて転移学習、ファインチューニングによる学習を行い、その能力を評価した。また、過学習傾向の ViT のファインチューニングに対してデータ拡張を適用した場合の有効性についても報告する。

2. 各種モデルのスキップ接続

2.1 多層化と学習

ニューラルネットワークの初期の研究では、ニューロン層の増加に比例するように学習能力も向上すると考えられていた。しかし、ニューロン層の増加は逆に学習能力の低下を招くことが明らかとなった。これは、学習誤差が複数のニューロン層を逆伝搬するにつれ減衰し、多層化された場合学習時の誤差が入力層まで伝搬されないことが原因であった。誤差伝搬が行われない多層化されたニューラルネットワークは当然のごとく学習困難な状態に陥った。

2.2 ResNet

スキップ接続を実装することによりニューラルネットワークの大規模な多層化を可能としたモデルが ResNet

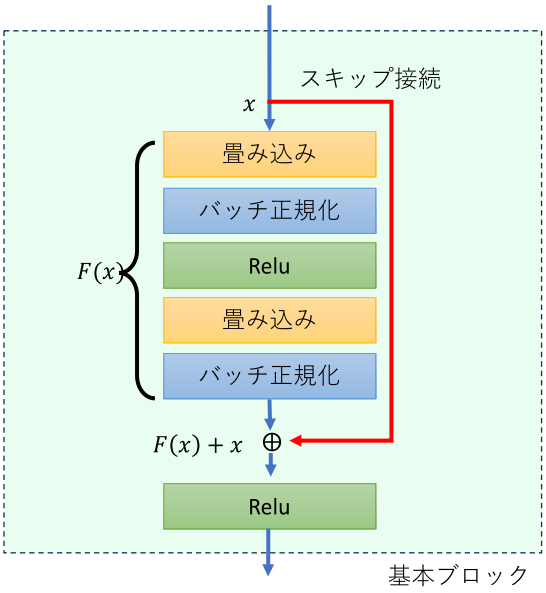

図 1 ResNet の基本ブロック

2024. 6. 27 加工技術研究部　受理

である。スキップ接続により多層化されたニューラルネットワークの誤差を入力層まで伝搬させることが可能となった。これにより ResNet は画像認識の性能を飛躍的に高めることに成功した深層学習モデルである。

図 1 にスキップ接続を実装した基本ブロックを示す。今、基本ブロックに変数xを入力した場合、畳み込み層等の非線形関数を$F(\cdot)$と表すと畳み込み層等を通過した出力は$F(x)$と表せる。ResNet の基本ブロックでは、畳み込み層等を通過した出力$F(x)$に、入力変数xが畳み込み層等をスキップして加えられる。従って、後段の ReLU 関数への入力を$H(x)$とすると、$H(x)$は

$$H(x) = F(x) + x \tag{1}$$

と記述できる。誤差逆伝搬を考える上で(1)式の微分は、

$$\frac{\partial H(x)}{\partial x} = \frac{\partial F(x)}{\partial x} + \frac{\partial x}{\partial x} = \frac{\partial F(x)}{\partial x} + 1 \tag{2}$$

により与えられる。従って、逆伝搬する学習誤差は（2）式の第 1 項に従い基本ブロックの学習が行われる。一方第 2 項は逆伝搬誤差をそのまま前段の基本ブロックに伝える。この構造が多層化ニューラルネットワークである ResNet の誤差喪失を防ぐ重要な技術である。ResNet では、学習に必要な能力に応じて、この基本ブロックを直列に複数個接続している。

2.3 DenseNet

ResNet は（1）式のように入力変数xを出力$F(x)$に加算することによって誤差喪失を防ぐことが可能となった。DenseNet ではこのスキップ接続の技術の進歩を図った構造となっている。**図 2** に DenseNet の基本ブロックを示す。入力変数xはスキップ接続により、個々の DenseNet ブロック出力にクラスターとして連結され次の DenseNet ブロックの入力変数として利用される。このような構造は、U-Net[2,7]や PSPNet などにも採用されている。DenseNet は入力変数全てを基本ブロック内でスキップ接続することで、より誤差喪失を減少させ、学習能力を向上させている。DenseNet の場合も、基本ブロックを直列に接続し能力の最適化を図っている。

2.4 ViT

進歩の著しい自然言語処理用のニューラルネットワークを画像認識などに応用する試みにより ViT は開発された。**図 3** に ViT の基本ブロックを示す。ViT においても 2 つのスキップ接続が実装されている。入力画像がメッシュ状に分割された後、分割された画像は 1 次元データとして入力される。ViT では画像処理によく用いられる畳み込み層は実装されず、アテンション層が分割された画像間の重み付けを行う。アテンション層の後段には全結合層が実装されている。ViT でも基本ブロックを複数個接続した実装構造となっている。

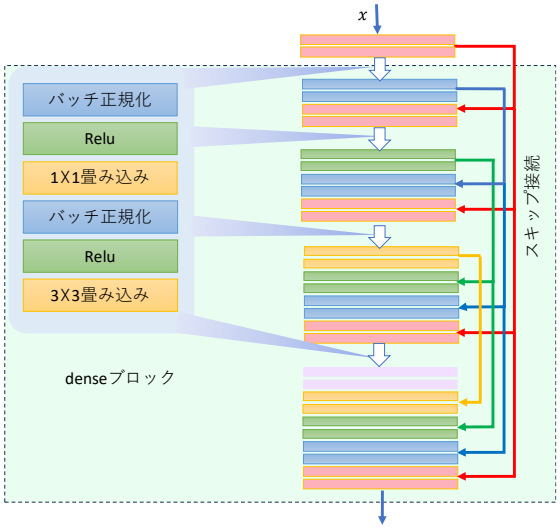

図 2 DenseNet の基本ブロック

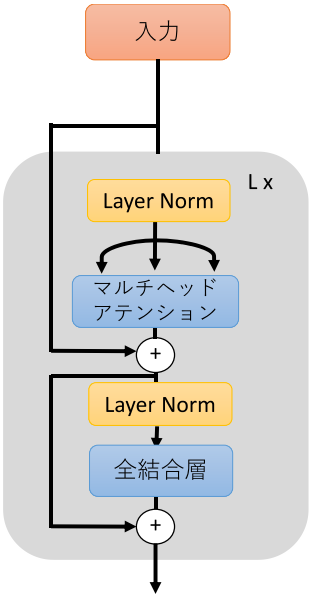

図 3 ViT の基本ブロック

3. 各種モデルの実装

3.1 画像データ

STL-10 の画像データを各種モデルの能力比較のために使用した。STL-10 はカラー画像からなる 10 種類のクラスで構成されており、学習用データ 8000 枚、評価用データ 5000 枚が用意されている。画像サイズは 96 ピクセル×96 ピクセルとなっているが ResNet、DenseNet、ViT の入力画像のサイズを統一するため、224 ピクセル×224 ピクセルに拡大した画像を用いた。

3.2 ResNet の実装

図 4 に ResNet18 の構造を示す。ResNet では、入力、出力の畳み込み層については、多層化の割合が変わった場合でも共通の構造を使用している。多層化の調整はレ

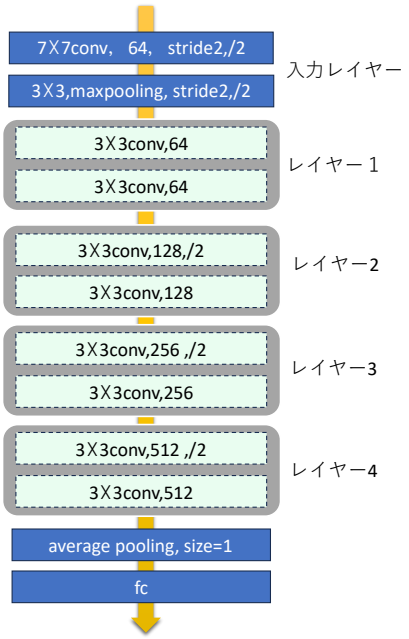

図 4 ResNet18 の構造

イヤー 1 からレイヤー4により行われる。図では各レイヤーに基本ブロックを 2 組実装する構成となっており、入出力層を含め 18 層構造を実装している。使用した PyTorch のフレームワークには、ResNet18 に加えて、ResNet32、ResNet50、ResNet101、ResNet152 の事前学習モデルが用意されている。転移学習とファインチューニングには ResNet50 の事前学習モデルから実装を行った。転移学習のためには、学習が不要な重みが再学習されず、必要な重みのみが学習されるように設定する必要がある。実装にあたり、まず全ての重みの学習を止め、最終層のみが 10 種類のクラスターリングを学習できる設定とした。

```
weights = ResNet50_Weights.DEFAULT
model = resnet50(weights=weights)
for param in model.parameters():
    param.requires_grad = False
model.fc = torch.nn.Linear(model.fc.in_features, 10)
```

STL-10 の ResNet50 に必要な転移学習パラメータは
Total params: 23,528,522
Trainable params: 20,490
となった。ファインチューニングでは、部分的な重みの学習を停止することなく、すべてのパラメータの再学習が行われる。

3.3 DenseNet の実装

図 5 は DenseNet 全体の構成である。ResNet 同様、層数の調整は 4 箇所の dense レイヤーによって行われる。PyTorch には densenet121、 densenet161、 densenet169、 densenet201 事前学習モデルが用意されている。本稿では ResNet50 のパラメータ数に合わせるため, dense201 を選択した。

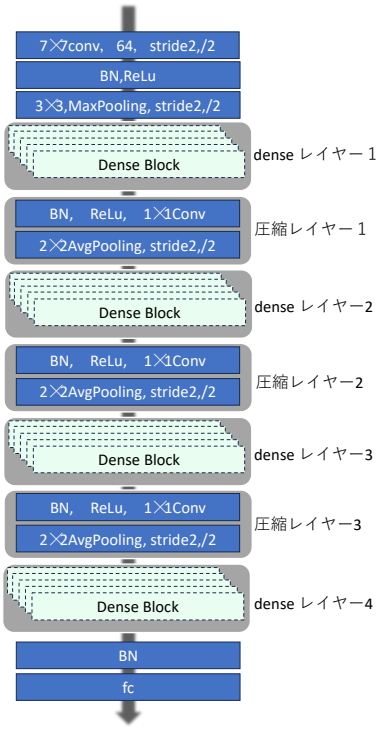

図 5 DenseNet の構造

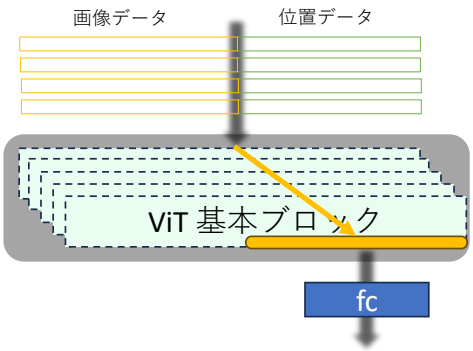

図 6 ViT の構造

```
weights = DenseNet201_Weights.DEFAULT
model = densenet201(weights=weights)
for param in model.parameters():
    param.requires_grad = False
model.classifier = torch.nn.Linear(model.classifier.in_features, 10)
```

STL-10 の densnet201 に必要な転移学習パラメータは
Total params: 18,112,138
Trainable params: 19,210
であった。

3.4 ViT の実装

自然言語処理用 Transformer[8] を画像処理に応用した構造を図 6 に示す。ViT では、Transformer のエンコーダ部分のみを使い、画像の識別を行う。入力画像はメッシュ状に分割された後、1 次元データに変換され、各データにはそれぞれ異なる位置データが付与される。位置データが付与された 1 次元画像データは、ViT の基本ブロ

ックにより関係性の高い画像エリアが紐づけられ、学習が行われる。ViT では、この基本ブロック層の数を増減することにより学習能力を調整している。本稿では最もパラメータ数が少ない vit_b_16 のモデルを選択した。

```
weights = ViT_B_16_Weights.DEFAULT
model = vit_b_16(weights=weights)
for param in model.parameters():
    param.requires_grad = False
model.heads.head =  torch.nn.Linear(model.heads.head.in_features, 10)
```

学習に必要なパラメータ数は

Total params: 85,806,346

Trainable params: 7,690

となり、総パラメータ数は多いが、学習パラメータは 3 つの事前学習モデルの中では最少となった。

4. 各種モデルの学習能力

4.1 ResNet の学習能力評価

ResNet の学習能力を比較するために、ResNet50 の転移学習、ファインチューニング、および事前学習無しの状態の 3 種類の学習を行った。画像データのバッチサイズを 50 とし、5000 枚の学習データの 50 回の学習による学習誤差を評価した。8000 枚の未学習データによる汎化能力も評価した。最適化関数には SGD 関数を使い、その学習係数に *lr*=0.001 を、モーメンタム係数には 0.9 を選んだ。**図 7** に学習に対する誤差を示す。転移学習、ファインチューニングともに高い学習能力を示した。事前学習が行われていない場合については、過学習状態に陥った。未学習データに対する転移学習、ファインチューニングの誤差を比較すると、転移学習の最小誤差は 0.1166、ファインチューニングが 0.0781 とファインチューニングの学習誤差の方が少ない結果となった。

4.2 DenseNet の学習能力評価

ResNet と同条件により DenseNet の能力評価を行った。**図 8** にその評価結果のグラフを示す。事前学習が行われていない場合は、過学習の状態ではあるが ResNet と比較すると学習回数に対する誤差の増加は少ない。また、転移学習、ファインチューニングの学習誤差は、同程度となり、転移学習の最小誤差は 0.0986、ファインチューニングが 0.0799 となった。但し、ファインチューニングの最小誤差は学習が 15 回の時のものであり、以降の学習ではわずかに過学習状態となった。

4.3 ViT の学習能力評価

事前学習されていないモデルでは、学習初期から過学習状態となり学習が困難となった。そのため、**図 9** には、転移学習とファインチューニングによる学習結果を示す。転移学習、ファインチューニングの誤差は、転移学習の最小誤差は 0.0635、ファインチューニングが 0.0546 と 3 つのモデルの中で最も学習誤差が少ない結果となった。しかしファインチューニングでは、5 回目の学習時に最小

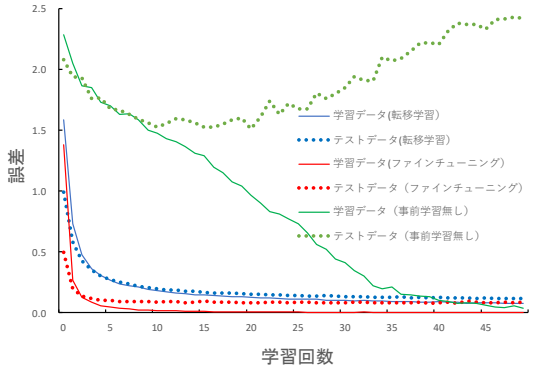

図 7 ResNet の学習能力比較

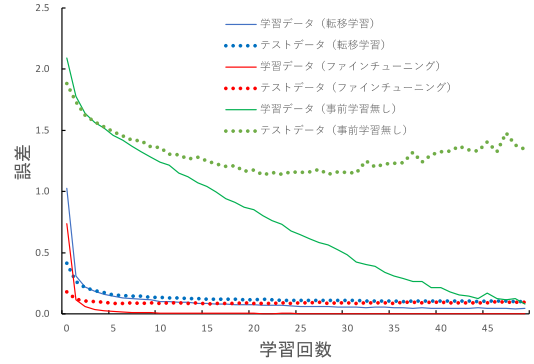

図 8 DenseNet の学習能力比較

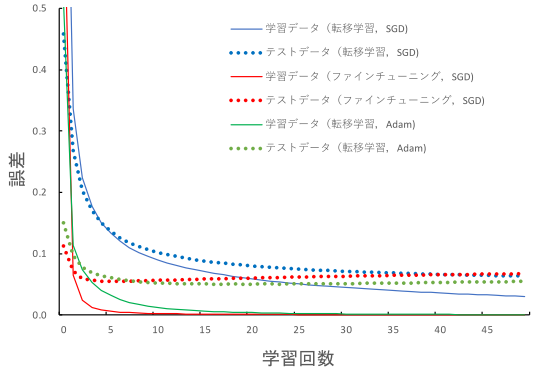

図 9 ViT の学習能力比較

値となった以降誤差が増加する、過学習状態となった。また、最適化関数が Adam の場合は転移学習の最小誤差が SGD のファインチューニングの誤差より少ない結果となった。

図 10 は 3 モデルの転移学習時の未学習データに対する誤差を示したグラフである。ResNet、DenseNet の学習能力は同程度であったが、ViT は学習パラメータが他の

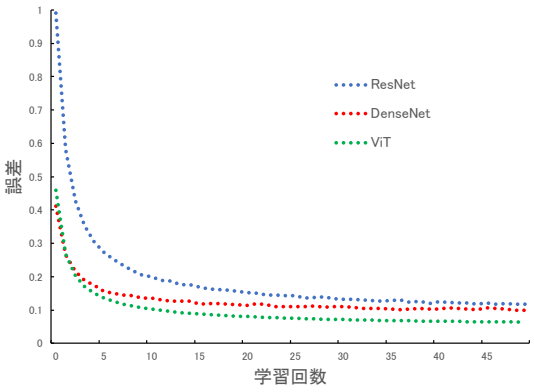

図 10 3 モデルの転移学習能力

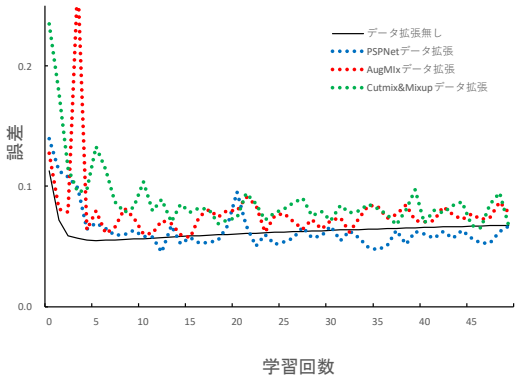

図 11 データ拡張によるファインチューニング

２つのモデルの 1/3 程度しか無いにもかかわらず他のモデルより高い学習能力を示した。

4.4 ViT のファインチューニング学習に対するデータ拡張の有効性

ViT をファインチューニングにより学習を行った場合、学習パラメータが他の２つのモデルと比較して多いためか、初期の学習段階で過学習状態となった。この学習誤差を減少させるため、データ拡張による学習の有効性について確認を行った。データ拡張については、次の 3 種類の方法を用いた。

① PSPNet で提案されたデータ拡張 [1)]
② AugMix[9)]
③ Cutmix と Mixup のランダム拡張 [10)]

Cutmix と Mixup の実装では下記のように、2 つの関数をランダムに呼び出す方式とした。

```
cutmix = v2.CutMix(num_classes=NUM_CLASSES)
mixup = v2.MixUp(num_classes=NUM_CLASSES)
cutmix_or_mixup = v2.RandomChoice([cutmix, mixup])
```

実験時の最適化関数には SGD を用いた。通常のファインチューニング学習と 3 種類のデータ拡張によるファインチューニング学習を行った場合の未学習のテストデータに対する学習誤差を図 11 に示す。データ拡張を行った場合でも大幅な性能向上は見られない。AugMix データ拡張と Cutmix と Mixup によるデータ拡張はむしろ学習能力が低下した。PSPNet 提案のデータ拡張ではわずかに学習能力が向上し、かつ過学習の状態に陥っていない。ファインチューニング学習時の未学習テストデータに対する PSPNet、 AugMix、 Cutmix & Mixup それぞれの最小学習誤差は 0.0451、 0.0575、 0.0653 となった。

5. 結　　言

事前学習モデルである ResNet、 DenseNet、 ViT の 3 モデルの転移学習、ファインチューニングによる学習能力を比較した。転移学習については、学習パラメータがもっとも少ないにもかかわらず ViT が最も優れた学習能力を示した。また、STL-10 の画像データの学習においては、3 モデルとも転移学習よりファインチューニングが学習誤差の少ない結果となった。ただし、ViT のファインチューニングでは総パラメータ数の多さから、過学習の傾向を示した。データ拡張を適用した ViT のファインチューニングでは、PSPNet 提案のデータ拡張のみが過学習の軽減し、学習誤差を小さくすることができた。PSPNet の学習では Cutmix & Mixup を使った学習が効果的であったが ViT のファインチューニングでは異なる結果となった [3)]。これは、ViT が採用しているアテンション構造に対しては PSPNet に採用されたデータ拡張の有効性が高い可能性が示された。

本実験結果から実用化のために画像分類に事前学習モデルを転移学習による転用を行う場合，どのモデルを用いてもほぼ同程度の性能が得られることが予想される。ただし，この結果は、学習データ量が本稿と同程度に限った場合であり、用意可能な学習データによっては再検討が必要である。一般的には、学習データが少ない場合は，転移学習を選び，転移学習では学習が不可能なほど学習データが多い時はファインチューニングを選択するとされている。

文　　献

1) Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia: Pyramid Scene Parsing Network, arXiv: **1612.01105**, (2017).
2) 廣川 勝久，花房 龍男，中濱 久雄：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，深層学習による画像の領域分割（第１報）**36**（2023）.
3) 廣川 勝久，花房 龍男，中濱 久雄：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，深層学習による画像の領域分割（第２報）**37**（2024）.
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) 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).

7) laf Ronneberger, Philipp Fischer, and Thomas Brox: U-net: Convolutional networks for biomedical image segmentation, arXiv:**1505.04597**, (2015).

8) 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).

9) Dan Hendrycks, Norman Mu, Ekin D. Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan: AugMix: A Simple Data Processing Method to Improve Robustness and Uncertainty, arXiv:**1912.02781** (2019).

10) WWW : How to use CutMix and MixUp, http://pytorch.org/vision/main/auto_examples/transforms/plot_cutmix_mixup.html.