機械学習による白飛び画像の明瞭化

2018SS051 小川功城
指導教員：小市俊悟

1　はじめに

本研究では写真を撮影する際の失敗例の中から，「白飛び」という現象に着目した．「白飛び」とは，日光の強い場所や白色の背景など明るいものがあるところで，デジタルカメラなどで撮影した際に，明るい部分が色を失い，白くなる現象のことを指す．本研究では，機械学習，特に敵対的生成ネットワーク（GAN）と呼ばれる手法を使って，白飛びしている画像を復元することを試みる．特に，白飛びした部分に写る物体の輪郭を明瞭化することを目指す．また，Generator の損失関数に用いる学習済みモデルの変更や，層の追加といった要素も加えることで，どの条件が一番良く復元できるのかを検証する．

2　GAN の説明

2.1　GAN とは

GAN とは，Generative Adversarial Networks の略であり，敵対的生成ネットワークと訳される．GAN では，図 1 のように，Generator（生成ネットワーク）と Discriminator（識別ネットワーク）と呼ばれる 2 つの深層ネットワークを互いに競い合わせることで，所望の機能について，その精度を高める．簡単に言えば，Generator は偽物を作り出す役割を，Discriminator は本物かどうか見極める役割を担い，この 2 つが競合関係の中で学習を進める．

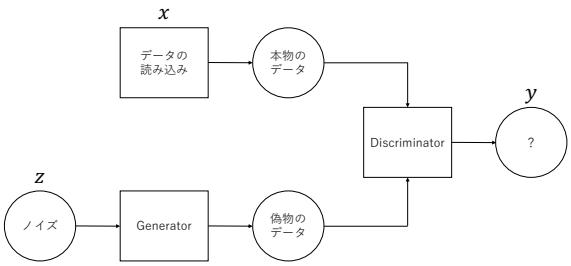

図 1　GAN の構造

2.2　SRGAN とは

SRGAN[1] とは，Generative Adversarial Network for Super-Resolution の略である．Super-Resolution は，超解像と訳されるもので，低解像度の画像を高解像度の画像に変換することを言う．SRGAN では GAN を利用することで，明細な高解像度画像を生成することが可能となっている．SRGAN などによる超解像によって得られる高解像度画像は，実際に高解像度で撮影された画像と一致するとは限らない．この点には注意が必要であるが，低解像度画像しか入手できない状況においては，GAN による超解像度から得られる情報も有用であると考えられる．本研究もこのような情報の補完を目的としている．

3　研究の方法

3.1　学習データの準備

SRGAN を白飛び画像の明瞭化に転用するため，その学習データとして，白飛びしている画像と白飛びしていない画像を一組にして，全部で 50 組用意する．白飛びしている画像は，Raspberry Pi のカメラ機能を使って，明度を意図的に大きくすることで，強制的に白飛び現象を引き起こすことで収集した．図 2 は白飛びしていない画像，図 3 はそれに対応する白飛びした画像である．

図 2　明度が適切な画像

図 3　白飛びしている画像

3.2　SRGAN に基づくプログラムの流れ

本研究では，PyTorch[2] を使用してプログラムを実装する．PyTorch とは，Python のオープンソース機械学習ライブラリの 1 つである．

本研究は SRGAN を基に，それを白飛び画像の明瞭化に転用するので，Generator への入力は一般的な GAN の場合のようにノイズ（乱数）ではなく，SRGAN が低解像度画像であるように，白飛びした画像とする．Generator には入力された画像を元に，白飛びしていない適正な画像に近い画像を出力させる．一方，Discriminator には，Generator が出力した画像と白飛びしていない適正な画像を判別させる．そのために，Discriminator には，白飛び画像と白飛び画像に対応する適正な画像をペアにして渡す．

3.3　SRGAN からの変更点

SRGAN を転用するにあたっては，次の 2 つを大きな変更点とした．

・Generator の損失関数に用いる学習済みモデルを，VGG16 以外のものも検討した．

・Generator に転置畳み込み層とバッチ正規化層を 1 つずつ追加し，SRGAN より多層化した．

なお，追加を更にもう一度繰り返し，それぞれ計２つの転置畳み込み層とバッチ正規化層を追加した場合も検証している．これらの変更を加え学習を行い，作成された Generator の性能を比較する．

4　実行結果と比較

結果の評価のために，白飛びしていない適正画像と，白飛びした画像を元に Generator が生成した画像の比較を SSIM を用いて行う．SSIM[3] は，structural similarity の略であり，構造的類似性と訳される．SSIM は，画像について，人間が感じる違いを正確に指標化することができるとされ，１に近い値ほど２つの画像は似ているとされる．表中の範囲は，最大値と最小値の差を表す．

4.1　学習済みモデルの変更とその結果

学習済みモデルを VGG16 から MNASNet に変更した場合の結果は，表１のようになった．その他の学習済みモデルも試したが，MNASNet が最もふさわしいということが分かった．また VGG16 は，MNASNet に劣るものの，白飛び画像を明瞭化するには十分な機能が備わっていることが分かった．図 4 は MNASNet を使用した場合の画像である．

図 4　（左）適正な画像（中）MNASNet で生成された画像（右）白飛び画像

表 1　適正な画像と GAN で生成した画像を SSIM で評価した結果

画像	VGG16	MNASNet
平均値	0.513	0.689
最大値	0.65	0.76
最小値	0.22	0.61
範囲	0.43	0.15

4.2　Generator に層を追加した場合

表 2 から，層の数は増やさずに学習を行うよりも，転置畳み込み層とバッチ正規化層を１つずつ追加して学習を行う方が，より良い結果になることが分かった．しかし，層を２つずつ追加した際は，学習済みモデルによって結果が異なり，MNASNet で検証を行った際は，１つだけ追加したときよりも良い結果になり，逆に VGG16 で検証を行った際は，１つだけ追加したときよりも悪い結果になった．MNASNet は，層を１つだけ追加したときよりも良い結果ではあったものの，その違いはほとんどなかったので，層を１つだけ追加するのが汎用的には最もふさわしい方法だということが分かった．図 5，図 6 は Generator に層を追加した場合の画像である．

図 5　（左）適正な画像（中）２層を１度追加した時の画像（右）白飛び画像

図 6　（左）適正な画像（中）２層を２度追加した時の画像（右）白飛び画像

表 2　適正な画像と多層化した GAN で生成した画像を SSIM で評価した結果

画像 層の数	VGG16 １度追加	VGG16 ２度追加	MNASNet １度追加	MNASNet ２度追加
平均値	0.666	0.417	0.798	0.827
最大値	0.79	0.49	0.85	0.87
最小値	0.22	0.12	0.74	0.77
範囲	0.57	0.37	0.11	0.1

5　おわりに

本研究では，機械学習を使って白飛び画像を明瞭化することに取り組んだ．本研究の独自性として，学習済みモデルの変更や層の追加などを行い，その効果を検証した．

また本研究では，SRGAN の構造を，白飛び画像の明瞭化を専門とした構造に改良して検証を行っているので，まだまだ研究の余地があると感じた．特に学習済みモデルは，本研究で取り扱ったもの以外にもまだ沢山あり，層の増やす量も３つ以上は検証していないので，その点も機会があれば検証していきたいと思う．

参考文献

[1] C. Ledig, L. Theis, F. Huszán, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi: Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network. *arXiv:1609.04802v5*, 19 p., 2017.

[2] WELCOME TO PYTORCH TUTORIALS https://pytorch.org/tutorials/index.html（アクセス日: 2021/9/23）

[3] 画質評価アラカルト　～SSIM ってすごい！～．株式会社デジタル・フロンティア https://dftalk.jp/?p=18111　（アクセス日: 2021/9/23）