複数スタイルの融合と部分的適用を可能とする Multi-style Feed-forward Network の提案

Multi-style Fast Style Transfer Network with Linear-weighting Style Fusion and Spatial Style Fusion

丹野 良介 *1　下田 和 *2　柳井 啓司 *3
Ryosuke Tanno　Wataru Shimoda　Keiji Yanai

*1*2*3電気通信大学 大学院情報理工学研究科
Department of Informatics, The University of Electro-Communications

In 2015, Gatys et al. proposed an algorithm on neural artistic style transfer using Convolutional Neural Network(CNN). This method enables us to modify the style of an image keeping the content of the image easily. However, since the method proposed by Gatys et al. required forward and backward computation many times, the processing time tends to become longer even using GPU. Then, several methods using only feed-forward computation of CNN to realize style transfer have been proposed so far. One of them is the method proposed by Johnson et al. They proposed perceptual loss functions to train the ConvDeconvNetwork as a feed-forward style transfer network. However, the ConvDeconvNetwork trained by their method can treat only one fixed style. If transferring ten kinds of styles, we have to train ten different ConvDeconvNetwork independently. Then, we propose a multi-style feed-forward network which can transfer not only one of the multiple trained styles but also their mixture.

1. はじめに

2015 年，Gatys ら [1, 2] は Convolutional Neural Network(CNN) を用いて，図 1 のように 2 種類の画像を合成するアルゴリズム “Neural Style Transfer”(スタイル変換) を提案した．この手法では画像のコンテンツ (形状) を保持したまま，スタイル (画風) を変化させることができ，例えば，写真を絵画風に変換可能であるなど，芸術的な画像をコンピュータが生成できるとして一世を風靡している．

Gatys らの手法では，CNN の特徴マップ間の相関行列であるグラム行列により表現されたスタイル行列を導入している．これにより，コンテンツ画像の信号が CNN の各層を順伝播している間に劣化する情報を，スタイル画像から抽出されたスタイル情報により置き換え，入力に与えたスタイル画像と同じスタイルに変換されたコンテンツ画像を生成することができる．

しかしながら，Gatys らの手法では，順伝搬及び誤差逆伝播を複数回繰り返して画像を生成するため，GPU を使う場合でも数十秒程度要するなど，処理に時間がかかる．この問題を解決するために，feed-forward のみを使ってスタイル変換を高速に行う研究が世界中で多くなされている．

Johnson ら [3] は feed-forward style transfer network として ConvDeconvNetwork を学習する “perceptual loss” を提案した．特定のスタイルの変換を順伝播で行う CNN を学習しておくことで，生成時には順伝搬のみで高速に画像を変換することができる．しかし，この手法では，1 つのモデルで単一のスタイルしか学習できず，消費メモリの増大，学習に時間がかかる，変換の質が画像の質に依存する，などの問題点があった．

そこで本研究では，[3] のネットワークを拡張し，複数スタイルの融合と部分的適用を可能とする multi-style feed-forward network を提案した．本ネットワークを用いることで 1 回の学習で複数のスタイルを同時に学習でき，学習時間の削減やモデルのメモリ量の効率化に寄与する．また，部分的適用を可能にすることで画像の領域毎に異なるスタイルを転送することができるようになる．

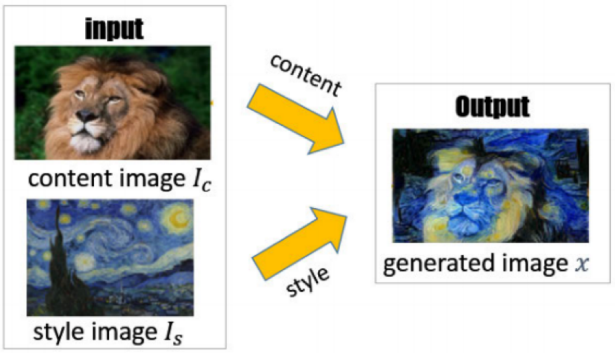

図 1: Neural Style Transfer アルゴリズム．スタイル画像の画風をコンテンツ画像に転送可能．

2. 関連研究

2.1 画風変換アルゴリズム

Gatys らが提唱した “Neural Style Transfer”(スタイル変換) を発端として，この分野に関する研究は急速に進んでいる．Gatys らの手法では，図 2 のように，コンテンツ画像，スタイル画像，目的画像をそれぞれ，$\vec{p}, \vec{a}, \vec{x}$ として，一様乱数で初期化した目的画像 $\vec{x}$ を入力した時の中間層 $Layer_{n_m}$ の出力の i 番目のチャンネルの座標 j の値を $conv_{n_m}(\vec{x})_{i,j}$ とした時，学習済み CNN(VGG-16 or VGG-19) の特徴マップ間の相関行列は式 1 のグラム行列 G を用いて

$$G(\vec{x})^l_{i_1,i_2} = \sum_k (conv_l(\vec{x})_{i_1,k} conv_l(\vec{x})_{i_2,k}) \quad (1)$$

と表される．このグラム行列 G を用いてコンテンツ損失 $L_{content}$ とスタイル損失 L_{style} は式 2, 式 3 で表される．式 3 中の A_l は各層の画素数などの差異を吸収する係数を表す．

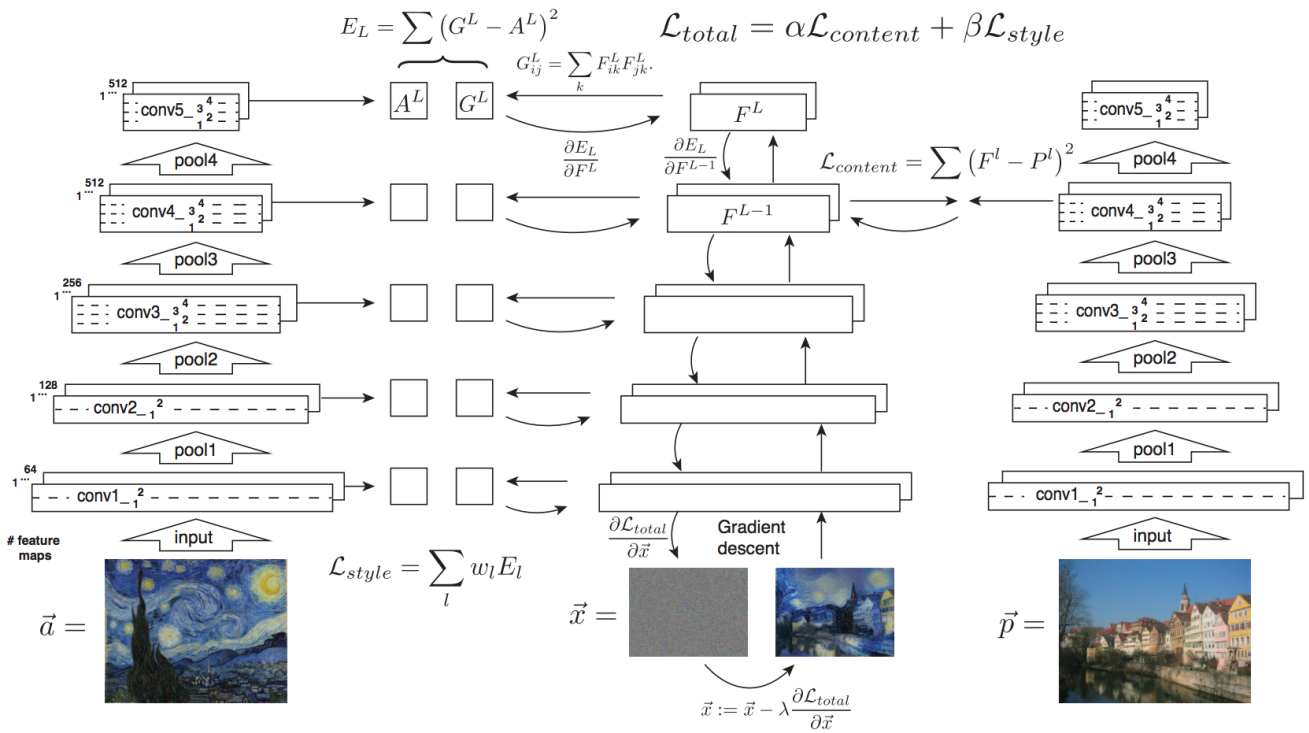

図 2: 順伝搬及び誤差逆伝播を複数回繰り返すため計算コストが高い ([1] より引用)

$$L_{content}(\vec{x}, \vec{p}) = \sum_{i,j} (conv_{4_2}(\vec{x})_{i,j} conv_{4_2}(\vec{p})_{i,j})^2 \tag{2}$$

$$L_{style}(\vec{x}, \vec{a}) = \frac{1}{A_l} \sum_{n=1}^{5} \sum_{i_1, i_2} (G(\vec{x})^{n_1}_{i_1,i_2} - G(\vec{a})^{n_1}_{i_1,i_2})^2 \tag{3}$$

目的画像の生成は，まず，$\vec{x}$ を一様乱数で初期化を行い，$L_{content}, L_{style}$ の線形和 (式 4) を最小化するよう勾配降下法 (式 5) により最適化を行う．

$$L_{total}(\vec{x}, \vec{p}, \vec{a}) = \alpha L_{content}(\vec{x}, \vec{p}) + \beta L_{style}(\vec{x}, \vec{a}) \tag{4}$$

$$\vec{x} \leftarrow \vec{x} - \lambda \frac{\partial L_{total}}{\partial \vec{x}} \tag{5}$$

Gatys らの手法の発想の着眼点は，コンテンツ画像の信号が CNN の各層を順伝播している間に劣化する情報を，スタイル画像から抽出されたスタイル情報により置き換えることにある．

しかしながら，Gatys らの手法では，図 2 にあるように feed-forward 及び back propagation を複数回繰り返すため，GPU を使う場合でも，画像の生成に数十秒程度要するなど，処理に時間がかかる．

この問題を解決するために，feed-forward のみを使ってスタイル変換を高速に行う研究が世界中で多くなされている．

Johnson ら [3] は図 3 に挙げるような feed-forward style transfer network として，Downsampling 層，複数の Residual block，Upsampling 層から構成される ConvDeconvNetwork f_w を学習する “perceptual loss” を提案した．まず，図 3 のように変数が設定されているとして，層の Feature Loss は式 6 で表される．この時，式 6 中の C, H, W はそれぞれ Channel, Height, Width を表す．

$$l^{\phi,j}_{feat}(\hat{y}, y_c) = \frac{1}{C_j H_j W_j} \|\phi_j(\hat{y}) - \phi_j(y_c)\|^2_2 \tag{6}$$

また，各層におけるグラム行列 (式 7) を用いて Style Loss は式 8 で表される．

$$G^{\phi}_j(x)_{c,c'} = \frac{1}{C_j H_j W_j} \sum_{h=1}^{H_j} \sum_{w=1}^{W_j} \phi_j(x)_{h,w,c} \phi_j(x)_{h,w,c'} \tag{7}$$

$$l^{\phi,j}_{style}(\hat{y}, y_s) = \|G^{\phi}_j(\hat{y}) - G^{\phi}_j(y_s)\|^2_F \tag{8}$$

式 6 と式 8 の線形和に $\hat{y}$ の分散を減少させる正則化項 $\lambda_{TV} l_{TV}(y)$ を追加した式 9 を最小化するように f_w の学習を行う．

$$\hat{y} = \arg\min_y \lambda_c l^{\phi,j}_{feat}(y, y_c) + \lambda_s l^{\phi,J}_{style}(y, y_s) + \lambda_{TV} l_{TV}(y) \tag{9}$$

これにより，特定のスタイルの変換を feed-forward で行う CNN を学習しておくことで，Gatys らの手法と比較して約 1000 倍 (500 iterations) ほど高速に画像を変換可能にする．しかし，この手法では，スタイル変換ネットワーク f_w はスタイル毎に学習する必要があり，1 つのモデルで単一のスタイルしか表現することができない．そのため，消費メモリの増大，学習に時間がかかる，変換の質が画像の質に依存する，などの問題点があった．

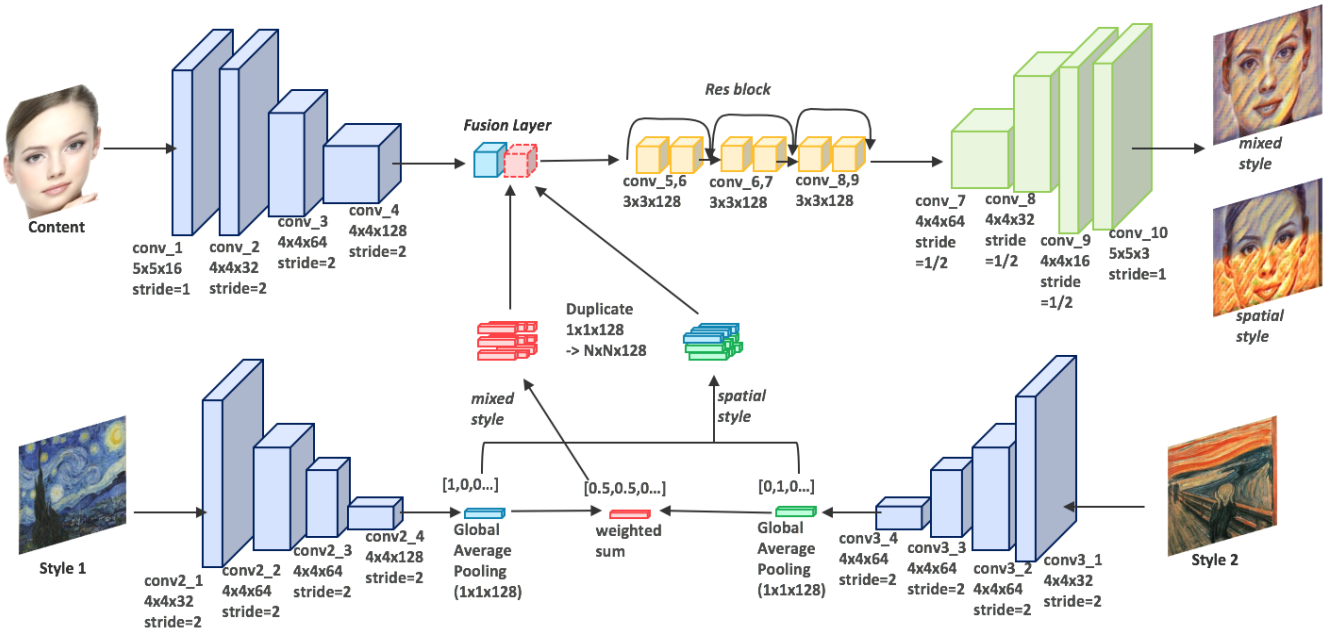

図 3: 複数スタイルの融合と部分的適用を可能とする multi-style feed-forward network

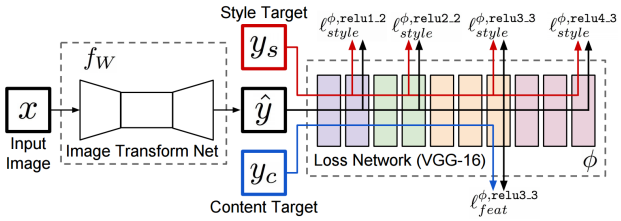

図 4: 特定のスタイルの変換を feed-forward で行う CNN を学習することで高速スタイル変換を実現 ([3] より引用)

2.2 本研究の位置付け

本研究では，Johnson ら [3] のネットワークを拡張し，複数スタイルの融合と部分的適用を可能とする multi-style feed-forward network を提案した．本ネットワークを用いることで 1 回の学習で複数のスタイルを同時に学習でき，学習時間の削減やモデルのメモリ量の効率化に寄与する．また，部分的適用を可能にすることで画像の領域毎に異なるスタイルを転送することができるようになる．CNN の具体的な実装の詳細については [4] に記載してある．

3. Multi-style Feed-forward Network

本研究が提案する複数スタイルの融合と部分的適用を可能とする multi-style feed-forward network を図 4 として以下に示す．

まず，図 4 上段のメインネットワークである ConvDeconvNet に図 4 下段にある通りスタイル転送 Net を追加する．各スタイルをスタイル転送 Net に入力し，128 次元実数値スタイルベクトルを事前に算出し，線形重ね合せにより複数のスタイル重みを表現するスタイルベクトルを求める．その後，ConvDeconvNet の中間層における特徴量マップに，事前に求めたスタイルベクトルをメインネットワークの特徴量マップと同じサイズ分コピーを作成し，Fusion Layer で結合させることで，複数スタイルの同時学習を可能にした．この Fusion Layer は飯塚ら [5] の研究により発想を得ている．スタイル転送 Net の最後の層に Global Average Pooling(GAP) を用いることで，各チャネルの平均 ($\simeq$ スタイル表現) を出力し，変換後の画像の質が向上することから (Li ら [6])，本ネットワークでも GAP を用いている．

また，本研究で提案するネットワークは図 4 上の spatial style とある通り，ConvDeconvNet の中間層に結合させる際のスタイルベクトルを異なるスタイルベクトルでコピーをすることで，転送するスタイルを部分的に異なるスタイルにすることが可能である．

ネットワークの学習方法は基本的には Johnson ら [3] と同じで，損失関数として pre-trained VGG-16 を用いて，各スタイル画像と 5 万枚のコンテンツ画像 (MS-COCO) により学習を行う．

4. モバイル実装

今回提案した multi-style feed-forward network を iOS アプリとしてモバイル上に実装した．アプリの動作例を図 5 として以下に示す．本研究では [3] のネットワークの拡張を行っているが，モバイル上で高速に動作させるために，

1. down-sampling layer と up-sampling layer を追加
2. 最初と最後の conv layer のカーネルを 9x9 を 5x5 に変更
3. 5 つの residual element を 3 つに変更

の変更を行い，[3] の ConvDeconvNet の縮小を行っている．[3] との差分は表 1 の通りである．ネットワークを縮小している

図 5: multi-style feed-forward network を用いたアプリの動作例．画面右側のスライダーを動かすことで任意の重みでスタイルを合成し，リアルタイムに変換可能である．

が，変換後の画像の質には影響がなく，[3] のネットワークが過剰であることがわかった．実装上の高速化手法は，

1. Deep Neural Network を直接 C コードに変換 (Chainer2C) し，コンパイラ的に実行．
2. Multithread 化による NEON/BLAS の効率的な利用．
3. CNN に掛かる演算の可能な限りの事前計算の実行．

などであり，実装の詳細については [4] や [7] を参照願いたい．

表 1: [3] とのネットワーク構成の比較

Johnson [3]	Ours
Reflection Padding (40x40)	5x5x16 conv, stride 1
9x9x32 conv, stride 1	4x4x32 conv, stride 2
3x3x64 conv, stride 2	4x4x64 conv, stride 2
3x3x128 conv, stride 2	4x4x128 conv, stride 2
Residual block, 128 filters	Fusion layer
Residual block, 128 filters	Residual block, 128 filters
Residual block, 128 filters	Residual block, 128 filters
Residual block, 128 filters	Residual block, 128 filters
Residual block, 128 filters	4x4x64 conv, stride 1/2
3x3x64 conv, stride 1/2	4x4x32 conv, stride 1/2
3x3x32 conv, stride 1/2	4x4x16 conv, stride 1/2
9x9x3 conv, stride 1	5x5x3 conv, stride 1

5. おわりに

本研究では，Johnson ら [3] のネットワークを拡張し，複数スタイルの融合と部分的適用を可能とする multi-style feed-forward network を提案した．本ネットワークを用いることで 1 回の学習で複数のスタイルを同時に学習でき，学習時間の削減やモデルのメモリ量の効率化を行うことができる．また，部分的適用を可能にすることで画像の領域毎に異なるスタイルを転送することができるようになる．更に，提案した multi-style feed-forward network を iOS アプリとしてモバイル実装を行い，そのために CNN の高速化やネットワークの縮小を行うことでリアルタイムに画風変換できるアプリを作成した．

今後の課題としては，領域分割 [8] や物体検出 [9, 10] などを組み合わせて，モバイル上での部分的な画風変換システムなどを構築したい．そのために，更なる高速化やスタイルを合成する際のスタイル重みの最適な組合せを調べるなどは重要である．

参考文献

[1] L.A. Gatys, A.S. Ecker, and M. Bethge. Image Style Transfer Using Convolutional Neural Networks. In *Proc. of IEEE Computer Vision and Pattern Recognition*, 2016.

[2] L.A. Gatys, A.S. Ecker, and M. Bethge. A Neural Algorithm of Artistic Style. In *Proc. of arXiv:1508.06576*, 2015.

[3] J. Johnson, A. Alahi, and L.F. Fei. Perceptual Losses for Real-Time Style Transfer and Super-Resolution. In *Proc. of European Conference on Computer Vision*, 2016.

[4] K. Yanai, R. Tanno, and K. Okamoto. Efficient Mobile Implementation of A CNN-based Object Recognition System. In *Proc. of ACM Multimedia*, 2016.

[5] S. Iizuka, E. Simo-Serra, and H. Ishikawa. Let there be Color!: Joint End-to-end Learning of Global and Local Image Priors for Automatic Image Colorization with Simultaneous Classification. *ACM Transactions on Graphics (Proc. of SIGGRAPH 2016)*, Vol. 35, No. 4, 2016.

[6] Y. Li, N. Wang, J. Liu, and X. Hou. Demystifying Neural Style Transfer. In *Proc. of arXiv:1701.01036*, 2017.

[7] R. Tanno and K. Yanai. Caffe2C: A Framework for Easy Implementation of CNN-based Mobile Applications. In *Proc. of International Workshop On Mobile Ubiquitous Systems, Infras- tructures, Communications, And AppLications (MUSICAL 2016)*, 2016.

[8] W. Shimoda and K. Yanai. Distinct Class-specific Saliency Maps for Weakly Supervised Semantic Segmentation. In *Proc. of European Conference on Computer Vision*, 2016.

[9] J. A. Divvala, S. A. Girshick, R. A. Farhadi, and A. . You Only Look Once: Unified, Real-Time Object Detection. In *Proc. of IEEE Computer Vision and Pattern Recognition*, 2016.

[10] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Y. Fu, and A. C. Berg. SSD: Single Shot MultiBox Detector. In *Proc. of European Conference on Computer Vision*, 2016.