# Neural Style Transfer と Cycle GAN を利用したフォント変換

成沢 淳史[1,a)]　柳井 啓司[,b)]

## 1. はじめに

文字にまつわる分野においてディープラーニングのおかげで様々な研究タスクを考えられるようになっている. 情景文字認識タスクではリアルタイムで多言語への変換を行うほか, 古文書の解析のように複雑な文字の読み取りを行うことができるようになった. このように従来では難しいとされた文字認識タスクが比較的容易に実現できる状況に現在ある. こうした流れから, 文字認識の新しい価値の創出に注目が集まりつつある. 特にコンピュータビジョンにおいて画像生成タスクが盛り上がりを見せており, 当然, フォントの形状変換や新しいフォントの生成を目的とする実験が散見される.

## 2. 目的

本研究でもフォント生成に注目している. 日本語フォントは英字フォントに比べて様々な文字種を含んでおり, フォントの製作にはコストがかかる. したがって画像生成技術を利用して英字フォントのデザインから日本語フォントを生成することでコストの削減に繋げることが期待される.

しかし, 英字フォントからのフォント生成は難しいとされる. その理由は英字の場合には大文字, 小文字合わせて高々 50 あまりのサンプル画像しか用意できないからである. 本研究の目的とするところはこの, 少量のサンプルからそのフォントの特徴たる部分を効果的に学習し, 日本語のような多様な文字を持つ言語に対するフォント生成を行うことにある.

## 3. 関連研究

### 3.1 Neural Style Transfer

近年の画像生成タスクへのアプローチの先駆けとなった研究に Neural Style Transfer [2] が挙げられる. この研究を執り行った Gatys らは当初テクスチャ合成 [1] を行なっていた. 注目すべき点はスタイル画像 $x$ は CNN の順伝搬により画像中から特徴を複数求め, それら特徴間の尤度をグラム行列 $G^L$ で表現した点である. 出力テクスチャはノイズ画像 $\hat{x}$ から徐々に生成される. ノイズ画像はまず パラメータがシェアされた CNN から複数の特徴を得たのちグラム行列 $\hat{G^L}$ を求める. このグラム行列がスタイル側とのグラム行列に近くなるよう, 逆伝搬によりノイズ画像を更新するプロセスを繰り返すことでテクスチャ合成を実現している.

グラム行列により最適化を行う理由がある. 特徴マップ間での相関が考慮され, 似た特徴部分を共起させることで生成されたテクスチャがスタイル画像それ自体が出力されることを防ぐことができる. また, CNN には VGG アーキテクチャー [10] を用いる場合が多く, 選択的なレイヤー $(L)$ においてスタイルグラム $(G^L)$ を求め以下の定式化 1 により $E_L$ の最小化を行う. この スタイルロス $L(\vec{x}, \hat{\vec{x}})$ の最適化に伴い, 逐次更新式 3 により $\vec{x}$ を更新する.

$$E_L = \sum (\hat{G^L} - G^L)^2 \tag{1}$$

また, スタイルグラムの取得を複数のレイヤーで行う場合には, レイヤー間での重み $(w_l)$ を定義し式 2 の最小化を解くこととなる.

$$L(\vec{x}, \hat{\vec{x}}) = \sum_{l=0}^{L} w_l E_l \tag{2}$$

$$\hat{\vec{x}} = \hat{\vec{x}} - \alpha \frac{\sigma Ł}{\sigma \hat{\vec{x}}} \tag{3}$$

そして, この研究を発展させた Neural Style Transfer [2] ではある画像を別画像の画風に変換するタスクで従来の手法よりも複雑な変換ができるようになった. これを達成するためスタイルロス $L_{style}$ に加え, さらにコンテンツロス $L_{content}$ が導入された. このコンテンツロス $L_{content}$ は式 4 に示すようにあるレイヤー $(l)$ でのコンテンツ画像 $(\vec{p})$ の特徴マップ $(P^l)$ とノイズ画像 $(\vec{x})$ の特徴マップ $(F^l)$ の L2 ロスの総和で求められる. そして, 最適化のための最終的

[1] 電気通信大学
[a)] narusawa-a@mm.inf.uec.ac.jp
[b)] yanai@cs.uec.ac.jp

な評価値 ($L_{total}$) は, ハイパーパラメータ $\alpha$ と $\beta$ を用いて式 5 のように求められる.

$$L_{content} = \sum (F^l - P^l)^2 \tag{4}$$

$$L_{total} = \alpha \mathrm{L}_{content} + \beta \mathrm{L}_{style} \tag{5}$$

### 3.2 Generative Adversarial Networks

画像生成の代表格として Deep Convolutional Generative Adversarial Netoworks (DCGAN) [9] が挙げられる. これは Generative Adversal Netoworks (GAN) [3] の Generator 部分に CNN を用いることで画像生成に対する精度を上げた研究である. Adversal Net は Discriminator($D$) 部分と Generator($G$) 部分から成り, 生成画像はランダムベクトル $z$ から $G$ を通して生成される. 式 6 のように, $D$ は学習データである確率を出力し, $logD(x)$ の最大化を目的として学習を行う. 一方, $G$ は $log(1 - D(G(z)))$ を最小化するよう学習を行い, $D$ は徐々に識別限界に近づくようになる. この状態において, 生成データ $G(z)$ は学習データ $x$ に極めて近しい分布のもと生成されることになる.

$$\begin{aligned}\min_G \max_D V(G,D) = E_{x\sim p_{data}(x)}[logD(x)] \\ +E_{z\sim p_z(z)}[log(1 - D(G(z)))]\end{aligned} \tag{6}$$

また, GAN における Generator 部分を Auto-Encoder に置き換えることで入力を画像とする構造 [7] が画像生成では普通となっている. GAN に対する拡張として GAN を多層に積むもの [12][5] や, 新しいロス関数を定義することで精度の向上を報告する研究 [8] が数多く上がっている.

### 3.3 Cycle GAN

その一例である Cycle GAN [13] は異なるドメインにある画像間のマッピング [11] において有効なアーキテクチャーである. 従来研究 [6] では, 学習時にドメイン間で近しい画像ペアを用意する必要があったが, Cycle GAN [13] ではソースドメイン ($X$) のとターゲットドメイン ($Y$) 内でペアを用意する必要がない. このクロスドメイン学習は, 図 2 のように ソースドメイン ($X$) の入力 $x$ は関数 $G$ によりターゲットドメイン ($Y$) の $\hat{y} = F(G(x))$ に写像される. さらに $\hat{y}$ は関数 $F$ によりソースドメイン $G$ に写像され $\hat{x} = G(F(y))$ となる. ソースドメインからの変換に加え, ターゲットドメインからの変換から定義される Cycle Consistency Loss ($L_{cyc}$) は式 7 のようになる.

$$\begin{aligned}L_{cyc}(G,F) = E_{x\sim p_{data}(x)}[F(F(x)) - x]_l \\ +E_{y\sim p_{data}(y)}[G(F(y) - y)]_l\end{aligned} \tag{7}$$


**図 1** スタイル変換結果 (左からコンテンツ画像、スタイル画像、合成画像)

この $L_{cyc}$ と Adversal Loss とが組み合わされ, 式 8 のロスから Cycle GAN は学習される.

$$\begin{aligned}L(G,F,D_x,D_y) = L_{GAN}(G,D_Y,X,Y) \\ +L_{GAN}(G,D_X,X,Y) + \lambda L_{cyc}(G,F)\end{aligned} \tag{8}$$

## 4. 手法

Neural Style Transfer ではコンテンツ側とスタイル側それぞれの重みを調整しながら、画像生成が安定するフォントの組み合わせを探求する。

英字のように文字種が少ない場合にはサンプル画像の生成が難しいため, サンプル画像を補填する方法を考える. 本研究では英字を特定の直線, 曲線パターンに従い分解を行いルールベースでパターン画像の生成を行うことで対処する. 用意されたデータセットは Cycle GAN [13] を用いて訓練され, 英字フォントから日本語の文字画像に対してデザインを転写する.

また, スタイル変換により英字から日本語の文字画像を生成し Cycle GAN における学習時での学習サンプルの不足を解消することを試みる.

## 5. 実験

### 5.1 Neural Style Transfer によるフォント変換の例

まずは Neural Style Transfer によりあるフォントにスタイルを転送した場合にどのような出力が得られるか 3 つのフォントでテストを行った. 結果を図 1 に示す. スタイル側のフォントの特徴を得るため画像中に複数の文字を置き入力画像をグラムマトリクスから逆伝搬により更新していく. 出力結果をみるとコンテンツ側の大局的な特徴がスタイル側のフォント特徴と置き換わっている様子が観察できる. 今回のケースでは生成画像自体はアーティファクトの影響が少なく綺麗な画像を生成することができた.

### 5.2 Cycle GAN によるフォント変換の例

次に Cycle Gan [13] に従い、二種類の日本語フォント間

での変換を学習する。片方のフォントは Simsum フォント (A) に固定する。他方のフォント (B) は五種類用意しそれぞれの組み合わせで A から B, B から A のネットワークの学習を行い、A, B, A の変換の際の B を合成画像としている. 入力画像は一文字毎とし A, B それぞれのフォントから文字画像をランダムに生成しネットワークを学習する.

学習時の最適化手法は [4] に従い, Adam とし, 式 9 中の Gradient Penalty $\lambda$ の値を 10 として実験を行なっている. 学習時のハイパーパラメータの設定は表 2 のように一般的な値を使っている.

$$L = E_{\hat{x}\sim P_g}[D(\tilde{x})] - E_{\hat{x}\sim P_r}[D(x)] + \lambda E_{\hat{x}\sim p_{\hat{x}}}(\|\nabla_{\hat{x}} D(\hat{x})\|_2 - 1)^2 \quad (9)$$

実験に用いたフォントの一例を図 3 に示す. 各フォントごとに学習用に 741 枚, テストに 186 枚用意し学習を行なった. 学習時のステップ数は表 1 のようにフォントごとに適当な回数で学習を打ち切った. ベースになるドメインはシステムフォント中に含まれる Simsum フォントとし図中にみられる 5 種類のフォントへの変換実験を行った. 変換実験の一例を図 4 に示す.

まず, Gothic 体への変換は簡単なタスクと言うことができる. 一方で Aoyagi フォントへの変換において「苛」の字をオリジナルのものと比較してみると分かるようにこのフォントの特徴的な部分は筆を滑らせたようなストロークの厚みとして表現されていることが分かる. 次に Gathee フォントへの変換はその雰囲気を再現する結果を得られている. Misaki フォントに関してはドット風であり, オリジナルの文字でさえ識別が難しいため評価が難しい. Mofuji フォントについても同様の評価であるが, 文字の一部分が太く強調される特徴が生成された画像からも確認することができる.

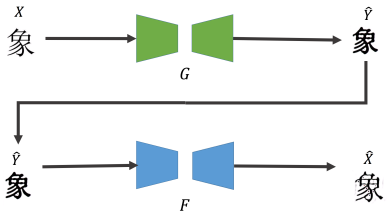


**図 2** Cycle GAN 概要

## 6. 終わりに

クロスドメイン学習はフォント変換タスクに最適と考えられる. しかしながら, 少ないサンプルで安定した出力を得るための方法を今後考えていく必要があり, まずフォント変換が最適なターゲットとして研究がなされていくだろう.

**表 1** 実験におけるフォントごとの学習ステップの違い

| フォント名 | 学習ステップ数 |
|---|---|
| Gothic | 1500 |
| Aoyagi | 12000 |
| Gathee | 3500 |
| Misaki | 7000 |
| Mofuji | 7000 |

**表 2** ハイパーパラメーターの設定

| Optimizer | Adam |
|---|---|
| 学習率 | 0.0001 |
| beta_1 | 0.5 |
| beta_2 | 0.9 |
| lambda | 10 |
| lambda_cycle | 10 |

## 参考文献


[1] Gatys, L. A., Ecker, A. S. and Bethge, M.: Texture Synthesis Using Convolutional Neural Networks, *Proc. of IEEE Computer Vision and Pattern Recognition* (2015).

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

[3] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. and Bengio, Y.: Generative adversarial nets, *Advances in neural information processing systems*, pp. 2672–2680 (2014).

[4] Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V. and Courville, A.: Improved Training of Wasserstein GANs, *arXiv preprint arXiv:1704.00028* (2017).

[5] Huang, X., Li, Y., Poursaeed, O., Hopcroft, J. and Belongie, S.: Stacked Generative Adversarial Networks, *Proc. of IEEE Computer Vision and Pattern Recognition* (2017).

[6] Isola, P., Zhu, J.-Y., Zhou, T. and Efros, A. A.: Image-to-image translation with conditional adversarial networks, *arXiv preprint arXiv:1611.07004* (2016).

[7] Larsen, A. B. L., Sønderby, S. K., Larochelle, H. and Winther, O.: Autoencoding beyond pixels using a learned similarity metric, *arXiv preprint arXiv:1512.09300* (2015).

[8] Odena, A., Olah, C. and Shlens, J.: Conditional image synthesis with auxiliry classifier gans (2017).

[9] Radford, A., Metz, L. and Chintala, S.: Unsupervised representation learning with deep convolutional generative adversarial networks (2015).

[10] Simonyan, K. and Zisserman, A.: Very deep convolutional nwtworks for large-scale image recognition (2014).

[11] Taigman, Y., Polyak, A. and Wolf, L.: Unsupervised Cross-Domain Image Generation, *arXiv preprint arXiv:1611.02200* (2016).

[12] Zhang, H., Xu, T., Li, H., Zhang, S., Huang, X., Wang, X. and Metaxas, D.: StackGAN: Text to Photo-realistic Image Synthesis with Stacked Generative Adversarial Networks, *Proc. of arXiv:1612.03242* (2016).

[13] Zhu, J.-Y., Park, T., Isola, P. and Efros, A. A.: Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks, *arXiv preprint arXiv:1703.10593* (2017).


**図 3** SimSum フォントからの変換先フォント一覧 (2 段目以下のフォント)


**図 4** 変換結果 (SimSum, ターゲットフォント)