Stylized-Colorization for Line Arts

Tzu-Ting Fang\textsuperscript{1} Duc Minh Vo\textsuperscript{2} Akihiro Sugimoto\textsuperscript{3} Shang-Hong Lai\textsuperscript{1}

\textsuperscript{1}National Tsing Hua University Hsinchu, Taiwan
\textsuperscript{2}SOKENDAI Tokyo, Japan
\textsuperscript{3}National Institute of Informatics Tokyo, Japan
Introduction

- There are 3 major steps to finish an anime character illustration:
  1. Drawing a sketch
  2. Converting the sketch to a clean line-art
  3. Colorizing the line-art
• Inspired by the coloring practice table available for the online artist community, the colorizing step is more than filling with appropriate colors.

• There are 5 typical “coloring styles”.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Image" /></td>
<td><img src="image2.png" alt="Image" /></td>
<td><img src="image3.png" alt="Image" /></td>
<td><img src="image4.png" alt="Image" /></td>
<td><img src="image5.png" alt="Image" /></td>
<td><img src="image6.png" alt="Image" /></td>
</tr>
</tbody>
</table>

• It can be observed that the styles give different impression to the same character.

• We aim at colorizing line-arts in different coloring styles.
Our task is challenging because it has 2 aspects: (1) colorization and (2) style transfer.

Different from style transfer
- Methods for style transfer usually bring distortion of object contours to reflect painting style. Our styles more take into account lightness, shading, and saturation of colors.
- Existing style transfer methods can not handle multi-domain of styles.
- We directly specify the styles, unlike style transfer where styles are given as images.

We propose a GAN-based end-to-end model to address a novel stylized-colorization problem, which can be stated as a multi-domain image translation task.
Overview of proposed method

Diagram showing the architecture of the proposed method, including the generator (G), color discriminator (Dc), and style discriminator (Ds). The generator takes line-art and color guide (optional) as input and produces a generated image. The color discriminator and style discriminator are used to distinguish between the generated image and the ground-truth color image and style, respectively. The diagram also includes loss functions ($L_{\text{pix}}$, $L_{D_c}$, $L_{\text{style}}$, $L_{D_s}$) for training the generator and discriminators.
Network architecture

- **Generator** \((G)\)
  - U–Net architecture.
  - Skip connections are added between mirrored layers of the encoder and decoder.
Network architecture

• **Color discriminator** (Dc)
  - Adopting PatchGAN
  - Dc receives either a ground-truth colored image or a generated image.
  - Distinguish if each NxN patch is real or fake.

• **Style discriminator** (Ds)
  - We condition the coloring style following the cGAN architecture.
  - Ds receives either a ground-truth stylized image or a generated image.
• We do not use any batch normalization layer in our network in order to keep the flexibility of colors.

• Stabilize the training and avoid collapsing our model
  – We add spectral normalization in both of Dc and Ds.
  – Adopt shared-weights at the first three layers between two discriminators.
Objective function

- **Adversarial loss**
  - The hinge loss: It helps the adversarial learning to be strong and stable.
  - Minimize the loss function for $D_s$ and $D_c$:
    \[
    \mathcal{L}_{D_c} = \mathbb{E}_{y_c \sim p_{data}} \left[ \max(0, 1 - D_c(y_c)) \right] \\
    \quad + \mathbb{E}_{(x,s) \sim p_{data}} \left[ \max(0, 1 + D_c(G(x, s))) \right], \tag{1}
    \]
    \[
    \mathcal{L}_{D_s} = \mathbb{E}_{(y_s,s) \sim p_{data}} \left[ \max(0, 1 - D_s(y_s, s)) \right] \\
    \quad + \mathbb{E}_{(x,s) \sim p_{data}} \left[ \max(0, 1 + D_s(G(x, s), s)) \right], \tag{2}
    \]
  - Minimize the loss function for $G$:
    \[
    \mathcal{L}_G = - \mathbb{E}_{(x,s) \sim p_{data}} \left[ D_c(G(x, s)) \right] \\
    \quad - \mathbb{E}_{(x,s) \sim p_{data}} \left[ D_s(G(x, s), s) \right]. \tag{3}
    \]
Objective function

• Per-pixel loss
  – L1 distance between the generated image and ground truth colored illustration.
  – Per-pixel loss enforces the generated image to be similar to the ground truth which is helpful in learning the color and keeping the image structure.

\[ \mathcal{L}_{\text{pix}} = \mathbb{E}_{(y_c, x, s) \sim p_{\text{data}}} \| y_c - G(x, s) \|_1 \quad (4) \]
Objective function

• **Style feature loss**
  
  – We introduce the style feature loss to learn the styles of images.
  
  – First we train a style classifier $C$ by employing the **center loss** [9].
    
    • Penalizing the distances between the style features and their corresponding center.
    
    • Reduce the intra–class differences.
Objective function

- **Style feature loss**
  - After the training $C$, we obtained the feature representation $c_s$ for coloring style $s$.
  - We employ the pre-trained classifier $C$ to define our style feature loss:

$$
\mathcal{L}_{\text{style}} = \mathbb{E}_{(x,s) \sim p_{\text{data}}} \left[ \|c_s - C(G(x, s))\|_2^2 \right],
$$  

$x$: input line-art
$s$: the coloring style
$c_s$: feature representation of style $s$
$C(G(\cdot))$: the style feature of generated image
In summary, our full objective function is:

$$
\mathcal{L} = \mathcal{L}_G + \lambda_{\text{pix}} \mathcal{L}_{\text{pix}} + \lambda_{\text{style}} \mathcal{L}_{\text{style}}; \\
$$

where $\lambda_{\text{pix}}$ and $\lambda_{\text{style}}$ are the hyper-parameters.
## Visual comparison

<table>
<thead>
<tr>
<th>Coloring styles</th>
<th>Line-art</th>
<th>Realistic</th>
<th>Galgame</th>
<th>Watercolor</th>
<th>Anime</th>
<th>Mono chrome</th>
</tr>
</thead>
<tbody>
<tr>
<td>pix2pix</td>
<td><img src="pix2pix_images" alt="Images" /></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>StarGAN</td>
<td><img src="StarGAN_images" alt="Images" /></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Asymmetric GAN</td>
<td><img src="AsymmetricGAN_images" alt="Images" /></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AAMS</td>
<td><img src="AAMS_images" alt="Images" /></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Petalica + AAMS</td>
<td><img src="PetalicaAAMS_images" alt="Images" /></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ours</td>
<td><img src="Ours_images" alt="Images" /></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ground truth</td>
<td><img src="Groundtruth_images" alt="Images" /></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Quantitative comparison

- We adopted three metrics:
  - PSNR (larger is better), SSIM (larger is better), and FID (smaller is better) for evaluation.

- R, G, W, A, M stand for realistic, galgame, watercolor, anime, monochrome, respectively.

<table>
<thead>
<tr>
<th>Metric</th>
<th>PSNR</th>
<th>SSIM</th>
<th>FID</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>R</td>
<td>G</td>
<td>W</td>
</tr>
<tr>
<td>pix2pix [10]</td>
<td>8.378</td>
<td>11.028</td>
<td>11.488</td>
</tr>
<tr>
<td>StarGAN [12]</td>
<td>4.978</td>
<td>5.062</td>
<td>6.288</td>
</tr>
<tr>
<td>AsymmetricGAN [13]</td>
<td>5.180</td>
<td>5.095</td>
<td>6.047</td>
</tr>
<tr>
<td>AAMS [23]</td>
<td>8.995</td>
<td>9.379</td>
<td>10.867</td>
</tr>
<tr>
<td>Petalica+AAMS</td>
<td>9.266</td>
<td>9.504</td>
<td>10.764</td>
</tr>
<tr>
<td>Ours</td>
<td>10.753</td>
<td>11.511</td>
<td>12.957</td>
</tr>
</tbody>
</table>
Conclusion

• We present a GAN–based end–to–end model to address a novel problem of stylized–colorization for line–arts.

• Our two discriminators work for judging in colorization and the coloring styles separately, and stylized–colorization results are produced by the generator.

• Our experiments demonstrate that our model successfully coloring line–arts with different styles, which is difficult to achieve with just combining existing colorization and style transfer methods.