Colorization for Line Art Anime with Cross - Attention and GAN ResUnet

Trang Hoang Khang[1], Duong Nguyen Gia Huy[2], Dinh Viet Vinh Khanh[3], and Phan Khanh Bang[4]

FPT University, Ho Chi Minh City, Vietnam
khangthse182228@fpt.edu.vn
khanhdvvse181518@fpt.edu.vn

Abstract. The anime and manga industry is booming and attracting a a lot of attention, leading to the need for efficient tools to support automatic line art coloring. In this paper, we focus on reference-based coloring and propose a model that combines Attention Mechanism with Generative Adversarial Networks (GANs). This ensures that color information is recovered effectively while maintaining the sketch image's overall structure. To evaluate the effectiveness of our method, we compare the results with traditional GANs models without using attention mechanisms. This highlights the advantage of attention learning to give improved color blending outputs, avoiding unwanted artifacts such as color bleeding.

Keywords: Anime Line Art Colorization, Reference-Based colorization, Cross-Attention Mechanism, Generative Adversarial Networks (GANs), Deep Learning for Image Processing, Computer Vision in Animation.

1 Introduction

Image coloring in general has attracted extensive research attention [1–3] in the field of Computer Graphics and Multimedia. In particular, automatic line drawing coloring is receiving much attention in the research community when traditional coloring methods are too time-consuming and laborious and are not suitable for large-scale mass production, for example in the production of cartoons, anime, and manga. Previously, methods [4–6] have been proposed to speed up the traditional coloring process. Line image coloring seems to be more challenging than grayscale image coloring, when the images do not contain any information about brightness, images with complex lines make the coloring more difficult and easy to bleed. This problem is a conditional image-to-image translation. The advent of Deep Learning has opened up many research directions to help automate line art coloring. The work uses neural networks to automatically colorize animated images with random colors [5]. However, it often requires a lot of interaction to fine-tune the colored results to meet what the user specifies. To effectively control the color of the results, many user-suggestion-based methods have been proposed successively, such as spot coloring [7–9], doodle coloring [10],

text hinting [11], and language-based [12]. These user-suggestion-based methods are still not convenient or intuitive, especially for amateur users who are not trained in aesthetics. Reference-based coloring methods [13–15] provide a more convenient way, can automatically complete the coloring process without any other manual intervention. The user enters a drawing and reference image to control the painting process, see Figure 1 for illustration.

Fig. 1: Sample images of line drawing, reference color image, and generated colored result of our model.

Recent advances in Deep Learning have enabled the development of powerful neural networks for automated colorization. In particular, Diffusion Models (DMs) [11], such as Stable Diffusion (SD) [16], have emerged as state-of-the-art and efficient methods for generating high-quality images. These models gradually refine noise into structured photos through denoising steps. However, this also results in computationally intensive models, often relying on large-scale GPUs or TPUs to generate images [17]. In contrast, GANs [18] remain a classic but powerful approach for image-to-image conversion tasks, including anime line art colorization. While GAN-based U-Net models [19] are effective, current methods still suffer from color inconsistencies, poor semantic alignment, and lack of flexibility in transferring colors from reference images. Some approaches attempt to mitigate these issues by using attention mechanisms or feature alignment strategies. To address these limitations, this paper proposes an enhanced reference-based coloring framework that integrates the Cross-Attention [20] Mechanism into the GAN-ResUNet architecture [18]. Our main contributions can be summarized as follows:

- **Encoder:** We use Residual Blocks to extract essential features from both sketch and reference images, effectively capturing detailed information about contours and colors.
- **Cross-Attention Module:** We integrate into the deep layers of ResUNet, optimizing the process of matching color features from reference images to sketch images.

– **Decoder:** We use Decoder the extracted information to reconstruct colorized images with high fidelity while preserving the original structure.

– **PatchGAN Discriminator:** We use PatchGAN to evaluate the authenticity of the colorized images, ensuring that the generated results appear natural, and consistent with artistic coloring styles, and improving the overall realism of the final output.

2 Related Work

2.1 Line Drawing Colorization

Line drawings mainly contain information about the overall structure through sparse line sets without luminance information, which means that traditional coloring methods designed for grayscale images cannot be applied directly. Most of the conventional early line drawing coloring methods [4, 6] are optimization-based, allowing users to apply color to specific regions with a brush. With the development of deep learning, many user-suggested coloring methods [8, 11, 12] have emerged to provide better control over color applications. However, these user-suggested methods become cumbersome when the number of line drawings increases and require more interactions. Therefore, some reference-based coloring methods [13–15] have been introduced, which are especially effective for coloring line drawing sets or videos containing anime characters. The coloring model we propose falls into this reference-based category.

2.2 Sematic Correspondence

Semantic correspondence [21] is a fundamental problem in computer vision that aims to establish the relationship between pixels in different images depicting the same type of object or sharing similar semantic information. In this context, the Reference-based Line Art Colorization problem can be viewed as a cross-domain correspondence issue. The primary challenge lies in the texture differences between the line art image and the reference image, which is in color. Most of the previously developed methods for semantic correspondence learning have focused on the grayscale image coloring problem. In our work, we employ an attention mechanism to identify the semantic connections between the line art image and the reference image.

2.3 Attention Mechanism

The introduction of attention mechanisms [22, 23] has demonstrated effective results in extracting and aggregating features in image perception tasks, bringing neural models closer to how humans perceive visual information. Recently, several methods for colorizing line drawings using attention-based techniques [9, 14, 15] have emerged to enhance a model's ability to add color. In the context

of grayscale image colorization, Kumar et al. [24] developed the Colorization Transformer, which relies solely on self-attention for this task. Building on this idea, we incorporate the Cross-Attention Mechanism [20] into selected deep layers of the ResUNet [18] architecture to enhance the color transfer process from reference images to line drawings. To evaluate the effectiveness of this approach, we compare our method with a model without any attention mechanisms.

3 Methodology

3.1 Model Architecture

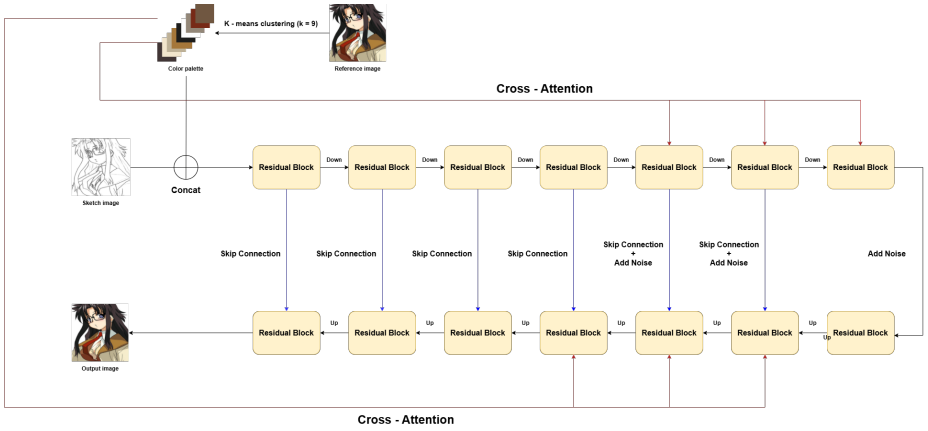

Fig. 2: The overview training pipeline of our network.

Given a line drawing I_{line} and a reference color image I_{ref}, the reference image is processed using the K-means clustering algorithm with $k = 9$ to extract dominant color information. This process generates a 27-channel color tensor I_{color}, which is then combined with the 3-channel line drawing I_{line} to form a 30-channel input tensor I_{input} for the model:

$$I_{\text{input}} = I_{\text{line}} \oplus I_{\text{color}}, \quad I_{\text{color}} \in \mathbb{R}^{27}, \quad I_{\text{line}} \in \mathbb{R}^{3}, \quad I_{\text{input}} \in \mathbb{R}^{30}$$

Our Generator network is designed based on the ResUNet architecture [18], which incorporates residual blocks to improve feature propagation and stabilize training. It follows an Encoder - Decoder structure, where both the Encoder and Decoder consist of 7 layers, each corresponding to a Residual Block, as illustrated in Figure 3. Since the training batch size is set to 4, Batch Normalization (BN) becomes ineffective in stabilizing training due to the small number of samples per batch. Instead, following the approach in [25], we replace BN with Group Normalization (GN) and apply Weight Standardization (WS) to all convolutional

layers. According to [25], this change has a similar or better effect on the use of BN. The sampling mechanism in our architecture operates in two modes:

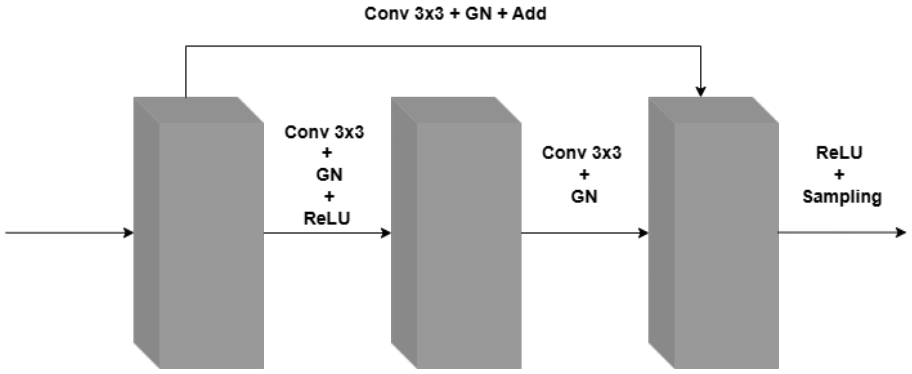

Fig. 3: Residual Block.

Our model integrates Cross-Attention [20] into the Generator's deep layers, as illustrated in Figure 2, to facilitate the transfer of semantic color information from the reference image while maintaining the structural integrity of the target line drawing. The attention mechanism follows the standard query-key-value (Q-K-V) formulation, computed as follows:

- **Query(Q)**: Extracted from the target line drawing feature map using a learned linear projection:
$$Q = W_Q X$$
where W_Q is a learnable weight matrix, and X represents the reshaped feature representation of the sketch.
- **Key(K)**: Computed from the reference color image feature map via another linear projection:
$$K = W_K I_{\text{ref}}$$
where W_K is a weight matrix, and I_{ref} represents the encoded reference image.
- **Value(V)**: Also derived from the reference feature map, using the mapping:
$$V = W_V I_{\text{ref}}$$
where W_V ensures that the color distribution in the reference image is properly aligned to be transferred.

The attention mechanism is responsible for transferring color features from the reference image while preserving the structural details of the target sketch.

The attention weights are computed based on the similarity between Query (Q) and Key (K):

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

where d_k is the dimensionality of the Key matrix, used for scaling the dot product.

After computing the attention weights, the output is projected back into the feature space:

$$\text{Output} = \text{Norm}\left(X + \text{Attention}(Q, K, V)\right)$$

where Norm represents Group Normalization (GN).

To enhance feature richness and improve fine details in the decoding process, random noise is scaled per channel and injected to feature in some layers of the Decoder, as illustrated in Figure 2. This task gives 'Stochastic variation' on result [26]. The weights of noise scaling are the trainable parameters. Figure 4 shows a block diagram of the Decoding Block.

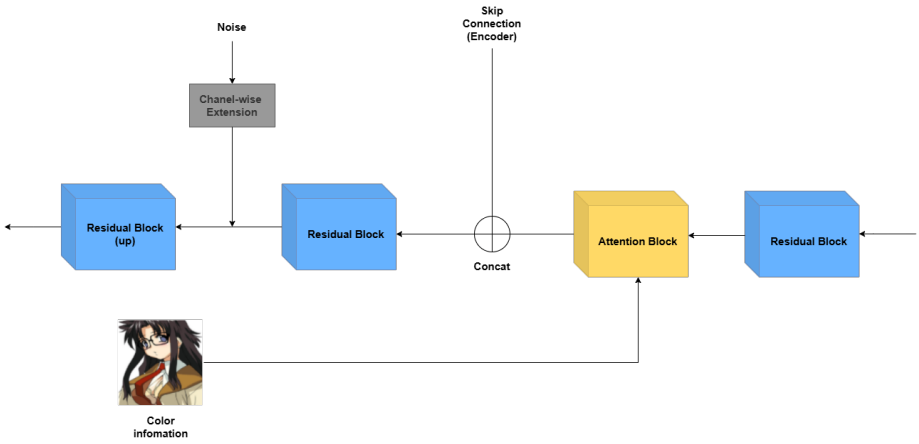

Fig. 4: Decoder Processing Pipeline with Noise Injection and Attention Mechanism.

3.2 PatchGAN

Adversarial learning in image colorization provides a normalization for the image to avoid local minima that produce an unrealistic image. We use Patch GAN rather than the normal GAN for this model. The output is a patch-based feature map instead of a single scalar value. The Binary Cross Entropy (BCE) loss is

computed for each patch, and the final discriminator loss is obtained by averaging the log loss across all patches. The discriminative architecture consists of 5 convolutional layers with Spectral normalization (SN) for stable training and leaky ReLU activation. The first three layers reduce the spatial size by a total factor of 6, while the last two layers further reduce it by 1 pixel each. The final output is a feature map of size 30 by 30 with a single output filter.

3.3 Loss Function

The total loss objective of the Generator is the weighted sum of L1 loss, feature matching loss, total variance loss, and adversarial loss. In this case, we set $\lambda_{l1} = 50$, $\lambda_{fm} = 0.05$, $\lambda_{tv} = 5 \times 10^{-4}$, and $\lambda_{adv} = 1$ with a learning rate of 2×10^{-4}.

$$\mathcal{L}_G = \lambda_{l1}\mathcal{L}_{l1} + \lambda_{fm}\mathcal{L}_{fm} + \lambda_{tv}\mathcal{L}_{tv} + \lambda_{adv}\mathcal{L}_{adv} \tag{1}$$

L1 loss, also known as mean absolute error, estimates the anticipated pixel's absolute divergence from the actual image. L1 loss is preferable for image colorization jobs over L2 loss because L2 loss is more likely to produce a dull-colored image, minimizing the loss. The loss can be calculated as follows.

$$\mathcal{L}_{l1} = \|y - G(x)\|_1 \tag{2}$$

Feature matching loss is commonly used in image colorization tasks as it helps the generated image retain structural consistency and perceptual similarity to the ground truth. Feature matching loss is the L2 distance between high-level features of the fake and real image in a pre-trained model. In this case, we use VGG16 as a pre-trained model. For image colorization, it minimizes the difference between deep feature representations extracted from a pre-trained discriminator, encouraging the generator to produce more realistic and coherent colorization results.

$$\mathcal{L}_{fm} = \sum_{layer} \frac{1}{N_{layer}} \|\phi_{layer}(y) - \phi_{layer}(G(x))\|_2^2 \tag{3}$$

The Total Variation (TV) Loss is commonly used in image generation tasks to enforce spatial smoothness and reduce unwanted noise or artifacts in the output. It penalizes high-frequency variations by measuring the difference between neighboring pixels in both horizontal and vertical directions. The loss is computed as the sum of squared differences between adjacent pixels, encouraging the generated image to have more natural and visually coherent transitions.

$$\mathcal{L}_{tv} = \lambda_{tv} \sum_{i,j} (|I_{i+1,j} - I_{i,j}| + |I_{i,j+1} - I_{i,j}|) \tag{4}$$

The adversarial loss in the Generator is derived from the PatchGAN discriminator, which encourages the Generator to produce images indistinguishable from real ones. This loss measures how effectively the Generator deceives the discriminator by computing the log loss of the generated image under the assumption

that it belongs to the real class. The overall adversarial loss is obtained by averaging the log losses across all patches in the discriminator's output feature map. Notice that the adversarial loss contributes only a minor portion to the total loss function, as its primary role is to act as a normalization to ensure the image is realistic enough.

$$\mathcal{L}_{adv} = -\mathbb{E}_{i,j,c}\left[\log\left(D_{i,j,c}\left(G(x)\right)\right)\right] \tag{5}$$

Discriminator loss is the opposite of adversarial loss in Generator. It aims to distinguish the real and generated images, hence computing the log loss of both real and generated images. Similar to adversarial loss, the loss is the mean of all log losses across all patches and filters.

$$\mathcal{L}_D = -\mathbb{E}_{i,j,c}\left[\log\left(D_{i,j,c}(y)\right) + \log\left(1 - D_{i,j,c}(G(x))\right)\right] \tag{6}$$

4 Experiments

4.1 Dataset

We trained our model using an anime dataset that contains 17769 pairs of sketches and colorful images on Kaggle [27]. The dataset is divided into 14224 samples for training and 3,545 samples for testing. The model input is a tensor with 30 channels (3 channels for the line art image and 27 channels for the 9 color features extracted from the reference image using K-means clustering). All images are resized to 256 x 256. We apply simple data augmentation techniques by using a random cropping and a random flip to the image.

4.2 Implementation Details

We implemented our method using the PyTorch framework and the model was trained on a NVIDIA 4070 super GPU with a batch size of 4. We set the total number of epochs as 40, and both the Generator and the Discriminator were alternately updated in every epoch. The input size of all images is 256 x 256 and the pixel value was normalized to the range of [-1, 1]. We used the Adam optimizer with $\beta_1 = 0.5$ and $\beta_2 = 0.99$. The learning rate for both the Generator and Discriminator was initially set to 2×10^{-4}. To further refine the training process, we incorporated a learning rate decay strategy using a StepLR scheduler with a step size of 20 epochs and a decay factor of 0.5, reducing the learning rate by half every 20 epochs.

4.3 Qualitative Evaluation

Figure 5 presents a qualitative comparison of the generated results using our proposed model. Each row in the figure represents a different reference image and its corresponding outputs. The results demonstrate that our model effectively captures the overall color distribution and maintains semantic consistency between

the reference and generated images with strong semantic correspondences. Our approach excels at transferring complex color patterns while preserving structural integrity, ensuring that key regions such as facial features, clothing, and background elements closely resemble the original reference.

Fig. 5: Qualitative result comparison: (a) reference image, (b) line drawing, (c) result image, (d) ground truth.

However, minor deviations in shading and fine details can still be observed in certain cases, highlighting potential areas for further refinement. These variations are most noticeable in highly textured areas, where the model may struggle to replicate intricate patterns exactly. Despite this, our method shows robust generalization across a diverse set of references, adapting well to different styles and color schemes.

Our method offers good semantic correspondence in the parts of hair, eyes, and clothes. The enhanced ability to maintain color consistency is largely attributed to the integration of Cross-Attention [20], which facilitates efficient feature alignment between the reference and target images. Additionally, we observed that the incorporation of the Convolutional Attention mechanism not only improves the quality of the generated outputs but also accelerates the model's convergence during training. Compared to baseline models without attention

mechanisms, our approach achieves faster stabilization of loss curves, reducing the number of required epochs while still achieving high-quality results.

Furthermore, subjective evaluation indicates that our model generates more aesthetically pleasing and visually coherent outputs compared to previous approaches. The natural blending of colors and the reduced occurrence of artifacts, such as color bleeding or unwanted hue shifts, further highlight the effectiveness of our proposed framework. Future improvements could focus on refining high-frequency details and enhancing adaptability to extreme variations in lighting and texture.

4.4 Quantitative Evaluation

We evaluated our model's performance in two cases: self-reference colorization and random-reference colorization. In the self-reference, we pair a line drawing with its corresponding reference image, and ideally, the colorized output will match the reference image. To assess the quality of colorization, we calculated the structural similarity and perceptual similarity between the colored output and the reference image using metrics such as Peak Signal-to-Noise Ratio (PSNR), Multi-Scale Structural Similarity Index Measure (MS-SSIM), and Learned Perceptual Image Patch Similarity (LPIPS) [28]. In contrast, random-reference colorization resembles typical practical applications of exemplar-based colorization methods. To evaluate the generative capability of a GAN-based network, we conducted a quantitative analysis by computing the Frechet Inception Distance (FID) score between the colorized output and the reference image. A lower FID score indicates that the distribution of the colored image is more aligned with that of the reference color image.

PSNR measures the fidelity of the colorized output with respect to the reference image, and is defined as:

$$\text{PSNR} = 10 \cdot \log_{10} \left(\frac{MAX_I^2}{\text{MSE}} \right) \tag{7}$$

where MSE (Mean Squared Error) is given by:

$$\text{MSE} = \frac{1}{mn} \sum_{i=1}^{m} \sum_{j=1}^{n} \left(I(i,j) - K(i,j) \right)^2 \tag{8}$$

MS-SSIM evaluates the structural similarity between the reference image and the generated image at multiple scales:

$$\text{MS-SSIM}(I,K) = \prod_{j=1}^{M} \left[\text{SSIM}_j(I,K) \right]^{\alpha_j} \tag{9}$$

where SSIM at scale j is defined as:

$$\text{SSIM}(I,K) = \frac{(2\mu_I\mu_K + C_1)(2\sigma_{IK} + C_2)}{(\mu_I^2 + \mu_K^2 + C_1)(\sigma_I^2 + \sigma_K^2 + C_2)} \tag{10}$$

LPIPS measures the perceptual similarity between two images using deep network features:

$$\text{LPIPS}(I, K) = \sum_l w_l \frac{1}{H_l W_l} \sum_{h=1}^{H_l} \sum_{w=1}^{W_l} \|\phi_l(I)_{hw} - \phi_l(K)_{hw}\|_2^2 \tag{11}$$

FID computes the distance between the feature distributions of real and generated images:

$$\text{FID}(X, Y) = \|\mu_X - \mu_Y\|_2^2 + \text{Tr}\left(\Sigma_X + \Sigma_Y - 2(\Sigma_X \Sigma_Y)^{\frac{1}{2}}\right) \tag{12}$$

where (μ_X, Σ_X) and (μ_Y, Σ_Y) represent the mean and covariance of real and generated image features.

Table 1: **Quantitative comparison with GAN without Attention.**

Method	**FID ↓**	**PSNR ↑**	**MS-SSIM ↑**	**LPIPS ↓**
Traditional GAN without Attention	40.08	24.70	0.92	0.09
Ours	**23.34**	**27.21**	**0.96**	**0.05**

As shown in Table 1, our proposed method achieves the best FID score, with a 41.77% improvement compared to the traditional GAN without Attention. Moreover, our method also surpasses the baseline across all evaluation metrics, achieving higher PSNR and MS-SSIM, while obtaining the lowest LPIPS. This demonstrates that our model effectively preserves semantic consistency and produces visually coherent colorized results. The integration of attention mechanisms enhances the model's ability to capture complex structures and details, leading to significant improvements in perceptual quality.

5 Conclusion

In this paper, we propose a reference-based anime line art colorization model that integrates Cross-Attention Mechanism into a GAN-ResUNet architecture. Our approach aims to improve color consistency, semantic accuracy, and creative flexibility in transferring colors from reference images to line drawings using incorporating attention mechanisms into deep layers of the network. We conducted extensive qualitative and quantitative experiments to validate the effectiveness of our method. The results demonstrate that our model surpasses traditional approach in terms of both FID, PSNR, MS-SSIM, and LPIPS, indicating significant improvements in perceptual quality and semantic coherence.

References

1. Menghan Xia, Wenbo Hu, Tien-Tsin Wong, and Jue Wang. Disentangled image colorization via global anchors. *ACM Transactions on Graphics (TOG)*, 41(6):204:1–204:13, 2022.
2. Zhitong Huang, Nanxuan Zhao, and Jing Liao. Unicolor: A unified framework for multi-modal colorization with transformer. *arXiv preprint*, arXiv:2209.11223, 2022.
3. Peng Lu, Jinbei Yu, Xujun Peng, Zhaoran Zhao, and Xiaojie Wang. Gray2colornet: Transfer more colors from reference image. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 3210–3218, 2020.
4. Yingge Qu, Tien-Tsin Wong, and Pheng-Ann Heng. Manga colorization. *ACM Transactions on Graphics (TOG)*, 25(3):1214–1220, 2006.
5. Domonkos Varga, Csaba Attila Szabo, and Tamas Sziranyi. Automatic cartoon colorization based on convolutional neural network. In *Proceedings of the 15th International Workshop on Content-Based Multimedia Indexing*, pages 1–6, 2017.
6. Daniel Sykora, John Dingliana, and Steven Collins. Lazybrush: Flexible painting tool for hand-drawn cartoons. *Computer Graphics Forum*, 28:599–608, 2009.
7. Adeleine. Adeleine colorization, 2021. Available at: `https://github.com/SerialLain3170/Colorization/tree/master/Adeleine`.
8. Yuanzheng Ci, Xinzhu Ma, Zhihui Wang, Haojie Li, and Zhongxuan Luo. User-guided deep anime line art colorization with conditional adversarial networks. In *Proceedings of the 26th ACM International Conference on Multimedia*, pages 1536–1544, 2018.
9. Mingcheng Yuan and Edgar Simo-Serra. Line art colorization with concatenated spatial attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3946–3950, 2021.
10. Petalica. Petalica paint, 2019. Available at: `https://petalica.com/index_en.html`.
11. Hyunsu Kim, Ho Young Jhoo, Eunhyeok Park, and Sungjoo Yoo. Tag2pix: Line art colorization using text tag with secat and changing loss. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9056–9065, 2019.
12. Changqing Zou, Haoran Mo, Chengying Gao, Ruofei Du, and Hongbo Fu. Language-based colorization of scene sketches. *ACM Transactions on Graphics (TOG)*, 38(6):1–16, 2019.
13. Shu-Yu Chen, Jia-Qi Zhang, Lin Gao, Yue He, Shihong Xia, Min Shi, and Fang-Lue Zhang. Active colorization for cartoon line drawings. *IEEE Transactions on Visualization and Computer Graphics*, 28(2):1198–1208, 2020.
14. Junsoo Lee, Eungyeup Kim, Yunsung Lee, Dongjun Kim, Jaehyuk Chang, and Jaegul Choo. Reference-based sketch image colorization using augmented-self reference and dense semantic correspondence. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5801–5810, 2020.
15. Zekun Li, Zhengyang Geng, Zhao Kang, Wenyu Chen, and Yibo Yang. Eliminating gradient conflict in reference-based line-art colorization. *arXiv preprint*, arXiv:2207.06095, 2022.
16. Hey Amit. Implement self-attention and cross-attention in pytorch, 2024. Accessed: 2025-03-15.
17. Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *arXiv preprint arXiv:2105.05233*, 2021.

18. Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, volume 27, 2014.
19. Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1125–1134, 2017.
20. 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. *International Conference on Learning Representations (ICLR)*, 2021.
21. Taihong Xiao, Sifei Liu, Shalini De Mello, Zhiding Yu, Jan Kautz, and Ming-Hsuan Yang. Learning contrastive representation for semantic correspondence. *International Journal of Computer Vision*, 130(5):1293–1309, 2022.
22. Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7132–7141, 2018.
23. Miao Hu, Yali Li, Lu Fang, and Shengjin Wang. A2-fpn: Attention aggregation based feature pyramid network for instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15343–15352, 2021.
24. Manoj Kumar, Dirk Weissenborn, and Nal Kalchbrenner. Colorization transformer. *arXiv preprint arXiv:2102.04432*, 2021.
25. Siyuan Qiao, Huiyu Wang, Chenxi Liu, Wei Shen, and Alan L. Yuille. Weight standardization. *arXiv preprint arXiv:1903.10520*, 2019.
26. Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. *arXiv preprint arXiv:1812.04948*, 2019.
27. Taebum Kim. Anime sketch colorization pair, 2021. Accessed: 2025-03-15.
28. Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 586–595, 2018.