Supplementary material
Cartoon Image Processing: A Survey

Yang Zhao[1,2], Diya Ren[1], Yuan Chen[3], Wei Jia[1*], Ronggang Wang[2,4] and Xiaoping Liu[1]

[1*]School of Computer and Information, Hefei University of Technology, Danxia Road, Hefei, 230009, China.
[2]Peng Cheng Laboratory, Dashi Road, Shenzhen, 518000, China.
[3]School of Internet, Anhui University, Feixi Road, Hefei, 230039, China.
[4]School of Electronic and Computer Engineering, Peking University Shenzhen Graduate School, 2199 Lishui Road, Shenzhen, 518055, China.

*Corresponding author(s). E-mail(s): wjia@hfut.edu.cn;
Contributing authors: yzhao@hfut.edu.cn; 2019170962@mail.hfut.edu.cn;
ychen@mail.hfut.edu.cn; rgwang@pkusz.edu.cn; liu@hfut.edu.cn;

In this supplementary material, we will introduce the details of cartoon image processing (CIP) datasets, representative loss functions in the CIP field, and extended experiments of the cartoon image enhancement task.

1 Details of Cartoon Datasets

The overview of the available cartoon datasets is given in TABLE 8 in the manuscript. Related details of each dataset are listed in the following.

1.1 eBDtheque

The L3I group at La Rochelle provided the first publicly available dataset, eBDtheque (Guérin et al. 2013), which included 100 pages from 25 different French, American, and Japanese comics published between 1905 and 2012. The following elements have been labeled on the dataset: 850 panels, 1092 balloons, 1550 characters and 4691 text lines. Even if the number of images is limited, creating such detailed labeled data is time-consuming and very useful for the community.

1.2 Manga109

The Manga109 dataset (Fujimoto et al. 2016) contains 109 manga volumes from 94 different authors, with a total of 21,142 pages drawn by professional manga artists in Japan between the 1970s and 2010s, and are categorized into 12 different genres such as fantasy, humor, sports, etc. The dataset can be used in retrieval, localization, character recognition, colorization, text detection and other research tasks. However, the dataset is limited to Manga, which has its own unique stylistic elements and visual vocabulary.

The complete Manga109 dataset is for academic research only and is not commercially available. The dataset proposers specially established the Manga109 industrial subset Manga109-s for the convenience of industrial users. The subset contains 87 of 109 manga books available for commercial research or development.

1.3 COMICS

COMICS dataset (Iyyer et al. 2017) contains 1,229,664 panels paired with automatic text boxes transcription from 3,948 American comics books published between 1938 and 1954. However, most annotations are automatically created and are not suitable for training or evaluation, with only 500 pages manually annotated.

1.4 BAM!

BAM! (Wilber et al. 2017) is built from Behance, a website containing millions of portfolios from professional and commercial artists. It contains around 2.5 million artistic images such as: 3D computer graphics, comics, oil painting, pen ink, pencil sketches, vector art, and watercolor. The images contain emotion labels (peaceful, happy, gloomy, and scary) and object labels (bicycles, birds, buildings, cars, cats, dogs, flowers, people, and trees), which provide a good basis for further research on the large-scale art image field that has not been fully explored.

1.5 VLRC

The Visual Language Research Corpus (VLRC) (Cohn et al. 2017) is a corpus of annotated comics analyzing the structures in visual languages of the world. At present, the VLRC is made up of 35,000 coded panels from roughly 300 comics from several places (America, Belgium, France, Germany, Hong Kong, Japan, Korea, The Netherlands, Sweden), different time periods (1940-present), and various genres. It includes coding of panel framing, semantic relations between panels, external compositional structure (page layout), multimodality, and a variety of other structures of visual languages.

1.6 GNC

The Graphic Narrative Corpus (GNC) (Dunst et al. 2017) focuses on English-language graphic novels and currently contains about 270 titles with approximately 55,000 pages. All works have a length of at least 64 pages, tell a continuous story, and target an adult readership. The GNC contains both fictional and non-fictional texts from 1970s to 2010s. Human annotations of selected pages of each volume include polygons around panels, main characters, speech balloons, captions, onomatopoeia, and diegetic text, as well as transcriptions of all text objects. The Empirical studies using the GNC include work on illustrator classification, semantic segmentation, and eye movement modeling.

1.7 CartoonSet

CartoonSet (Royer et al. 2020) is a collection of random, 2D cartoon avatar images with two subsets, CartoonSet10k and CartoonSet100k, containing 10000 and 100000 cartoon face images, respectively. Each cartoon face is composed of 16 components including 12 facial attributes (e.g., facial hair, eye shape, etc) and 4 color attributes (such as skin, hair color, etc.) which are chosen from a discrete set of RGB values. CartoonSet helped develop the technology behind the personalized stickers in Google Allo and studied cross-domain image translation.

1.8 selfie2anime

The selfie dataset contains 46,836 selfies annotated with 36 different attributes. Selfie2anime (Li et al. 2019) used only photos of females as training data and test data. The size of the training dataset was 3400, and that of the test dataset was 100, with the image size of 256 x 256. For the anime dataset, 69926 animated character images were first retrieved from Anime-Planet[1]. Among those images, 27023 face images were extracted using an anime-face detector[2]. After selecting only female character images and removing monochrome images manually, Kim et al. collected two datasets of female anime face images, with the sizes of 3400 and 100 for training and test data respectively, which is the same numbers as the selfie dataset. Finally, all anime face images are resized to 256 x 256 by applying a CNN-based image super-resolution algorithm[3].

1.9 GANILLA

Hicsonmez et al. (2020) built on the existing illustrations dataset that was used to classify illustrators, containing almost 9500 illustrations

[1] https://www.anime-planet.com/
[2] https://github.com/nagadomi/lbpcascade_animeface
[3] https://github.com/nagadomi/waifu2x

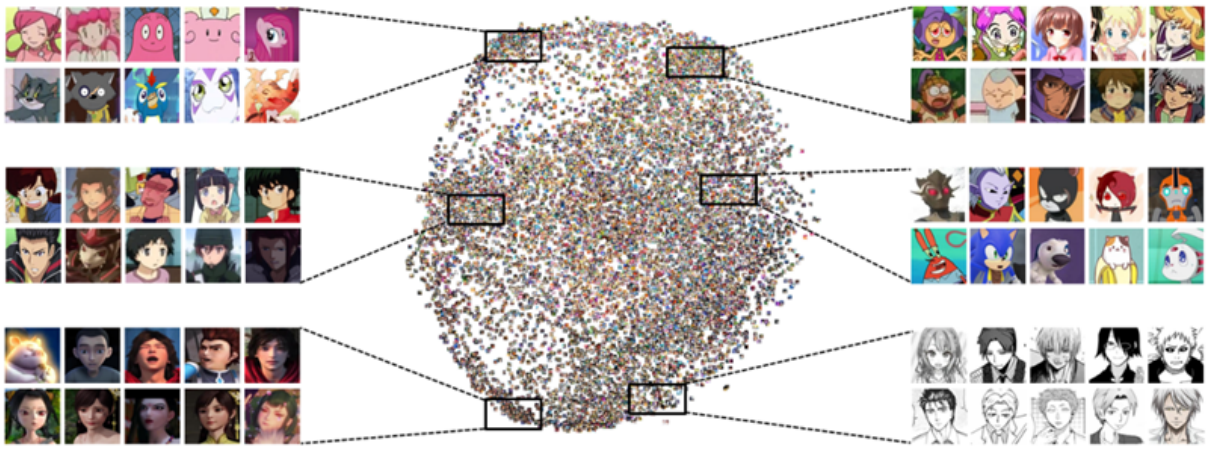

Fig.S- 1 Illustration of iCartoonFace (Zheng et al. 2020) embedding. The proposed dataset consists of diverse data sources for face recognition and detection task

coming from 363 different books and 24 different artists, which is the largest known children's book illustration dataset. They collected new images by scraping the web and scanning books from public libraries. The illustration dataset could be reproduced by scraping web based open libraries.

1.10 IIIT-CFW

As the first large cartoon faces database, IIIT-CFW (Mishra et al. 2016) contains caricatures, paintings, cartoons and sketches of 100 international celebrities (politician, actor, singer, sports person, etc) and 8,928 images in all. The images in this database are harvested from the web. These images contain cartoons drawn in totally unconstrained setting. The IIIT-CFW contains detailed annotations and it can be used for wide spectrum of problems including cartoon face recognition, cartoon face verification, cartoon image retrieval, relative attributes in cartoons, gender or age group estimation from the cartoon faces, and cartoon faces synthesis.

1.11 WebCaricature

The WebCaricature (Huo et al. 2017) database is a large photograph-caricature dataset consisting of 6042 caricatures and 5974 photographs from 252 persons collected from the web. For each image in the dataset, 17 labeled facial landmarks are provided. As all the caricature images are collected from the web, the caricatures are of various artistic styles. Besides, the illumination conditions, pose, expressions, occlusions and age gap are not controlled for both the caricature and photo modalities. The dataset is by far the largest caricature face recognition dataset, filling the gap in the benchmark dataset of caricature face recognition research in the era of deep learning.

1.12 iCartoonFace

As shown in Fig.S-1, the iCartoonFace (Zheng et al. 2020) dataset consists of 389,678 images of 5,013 cartoon persons from public websites and online videos, which are annotated with identity, bounding box, pose, and other auxiliary attributes. It is the largest-scale, high-quality, rich-annotated, and spanning multiple occurrences in the field of image recognition, including near-duplication, inter-class diversity, illumination conditions, and appearance changes.

1.13 Cartoon Sequences in Youku-VESR Dataset

Youku video enhancement and super resolution (Youku-VESR) (2019) dataset includes video samples, data description and evaluation codes. There will be more than 10K samples in Youku-VESR dataset, including the videos of different contents, different noise categories and different noise scales. Among them, there are some 2K cartoon sequences, the length of each video clip is 100 frames (4s).

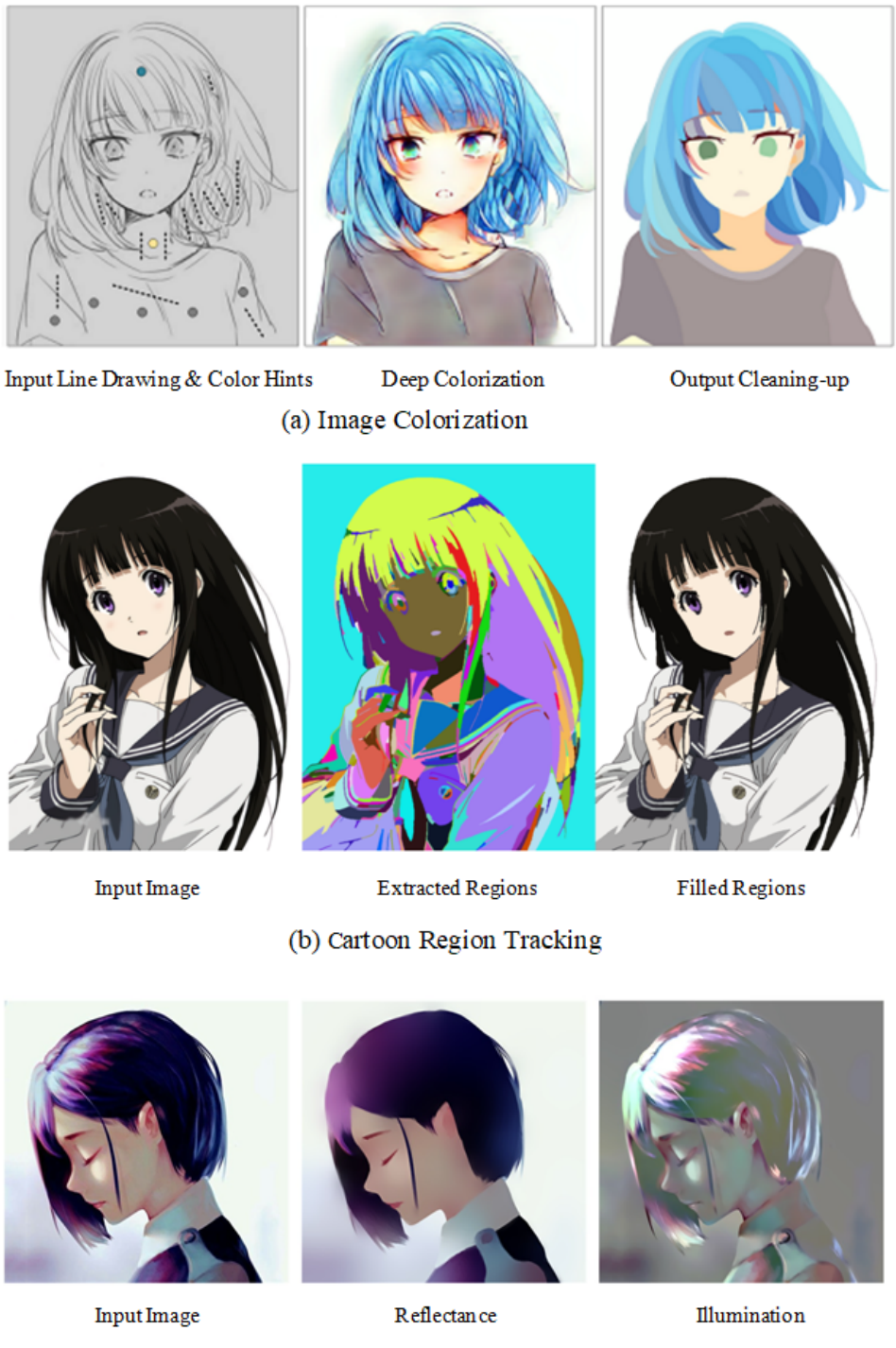

Fig.S- 2 Some applications of DanbooRegion (Zhang et al. 2020b). (a) Image Colorization: Style2Paints (Zhang et al. 2018) used the estimated regions to clean up the sketch colorization results by sampling median color in each region. (b) Cartoon Region Tracking: The user can decompose cartoon image into some regions and then fill colors into the regions to reconstruct the cartoon image. (c) Cartoon Intrinsic Decomposition: The regions can also be used to decompose illumination

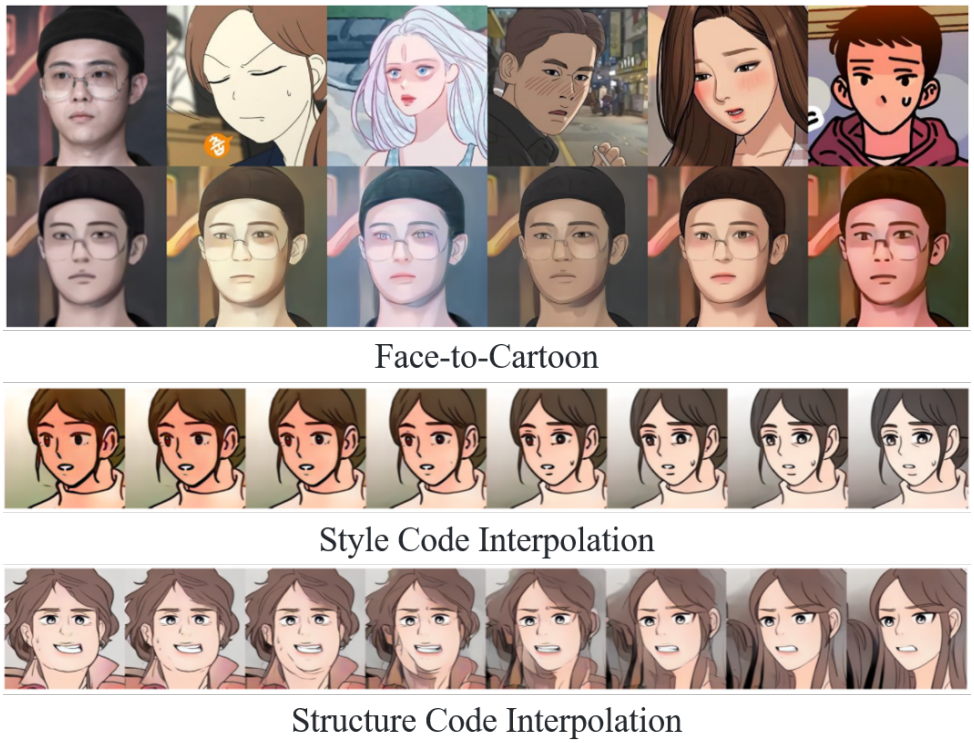

Fig.S- 3 Some applications of Naver Webtoon Data (2021)

1.14 Danbooru2020

Danbooru2020 (2021) is a large-scale anime image database with 4.2million+ images annotated with 130 million+ tags covering Danbooru from 2005 to 2020. It can be useful for machine learning purposes such as image recognition, generation, categorization and multi-label classification (tagging).

1.15 DanbooRegion

DanbooRegion (Zhang et al. 2020b) is a project conducted by ToS2P (the Team of Style2Paints), aiming at finding a solution to extract regions from illustrations and cartoon images, so that many region-based image processing algorithms can be applied to in-the-wild illustration and digital paintings. The dataset was randomly downloaded from the Danbooru2018 and manually annotated by 12 artists, consisting of 5,377 region annotation pairs, all samples are provided as RGB images at 1024-px resolution and 8-bit depth. The main uniqueness of this project is that the dataset was created by real artists in a semi-automatic manner, as shown in Fig.S-2, which can be used for image Colorization (a), Cartoon region tracking (b), and cartoon intrinsic decomposition (c).

1.16 Nico-Illust

Nico-Illust dataset (2016) contains over 400,000 images (illustrations) from two illustration communities of Niconico Seiga[1] and Niconico Shunga[2].

1.17 Naver Webtoon Data

Bryandlee (2021) collected 453846 cartoon facial images with 256×256 resolution from the NAVER Webtoon[4], as shown in Fig.S-3, and used this dataset for several tasks such as cartoon style transfer and frame interpolation.

2 Representative Loss Functions in CIP Tasks

In this section, we first introduce some commonly used loss functions in the CIP tasks, and

[1] http://seiga.nicovideo.jp/
[2] http://seiga.nicovideo.jp/shunga/
[4] https://comic.naver.com/index

then introduce several loss functions specifically designed for cartoons.

2.1 Typical Loss Functions

2.1.1 Pixel-level loss

Pixel-level loss measures element-wise difference between two images and mainly includes L_1 loss (i.e., mean absolute error) and L_2 loss (i.e., mean square error). Comparing with L_1 loss, the L_2 loss penalizes larger errors but is more tolerant to small errors, and thus often results in too smooth results, as follows:

$$L_1 = \sum_{i,j}^{n} |y^{(i,j)} - G(x)^{(i,j)}| \tag{1}$$

$$L_2 = \sum_{i,j}^{n} \left(y^{(i,j)} - G(x)^{(i,j)}\right)^2 \tag{2}$$

where y is the real sample and $G(x)$ is the generated sample, respectively. The pixel loss gradual becomes the most widely used loss function. However, the pixel loss usually lacks high-frequency details and is perceptually unsatisfying with over-smooth textures.

2.1.2 Total Variation Loss

Total Variation Loss L_{tv} is used to impose spatial smoothness on generated images. It also reduces high-frequency noises such as salt-and-pepper noise. It is defined as the sum of the absolute differences between neighboring pixels and measures how much noise is in the images:

$$L_{tv} = \sqrt{\left(G(x)^{(i,j+1)} - G(x)^{(i,j)}\right)^2 + \left(G(x)^{(i+1,j)} - G(x)^{(i,j)}\right)^2} \tag{3}$$

2.1.3 Content/Perceptual Loss

Besides the high quality in image generation, an important goal in cartoon image processing is to ensure the generated cartoon image retains the semantic content of the input image. To achieve this, it measures the semantic differences between images using a pre-trained image classification network ϕ, The content loss defined at layer l is the mean square error between the feature maps of two images:

$$L_{cont} = \sqrt{\sum_{i=1}^{N} \left(\phi^l(y) - \phi^l(G(x))\right)^2} \tag{4}$$

where $\phi^l(y)$ denotes the feature maps at level l.

2.1.4 Adversarial Loss

Because the L_{cont} is regularization in feature spaces. If there is not explicit restriction in image space, the generated images tend to be inconsistent among different parts, and usually contain some small line segments. The L_{adv} introduces the adversarial training strategy, the generator network G tries to produce image which look the same as images from domain B, and meanwhile D aims to distinguish between synthetic samples $G(x)$ and real samples y:

$$\begin{aligned} L_{adv} = {} & \mathbb{E}_{y \sim P_{\text{data}}(B)}[\log D(y)] + \\ & \mathbb{E}_{x \sim P_{\text{data}}(A)}[\log(1 - D(G(x)))] \end{aligned} \tag{5}$$

where $y \sim P_{data}(B)$ and $x \sim P_{data}(A)$ denotes the data distribution.

2.1.5 Cycle Consistency Loss

Cycle Consistency Loss L_{adv} cannot guarantee that the learned function can map an individual input x to a desired output y. L_{cyc} further helps constrain the mapping solution from the input to the output. For each image x from domain A, the image translation cycle should be able to bring x back to the original image, i.e., $x \to G(x) \to F(G(x)) \approx x$, where $G : A \to B$ and $F : B \to A$:

$$\begin{aligned} L_{cyc} = {} & \mathbb{E}_{x \sim P_{\text{data}}(A)}[\|F(G(x)) - x\|_1] \\ & + \mathbb{E}_{y \sim P_{\text{data}}(B)}[\|G(F(y)) - y\|_1] \end{aligned} \tag{6}$$

2.1.6 Style Loss

The style loss L_{style} guides the output to have the same cartoon style as the input cartoon image. Since AdaIN layer only transfers the mean and standard deviation of the style features, L_{style} matches these statistics:

Table S- 1 Specially designed loss functions and their characteristics in cartoon image processing tasks

Loss	Full name	Characteristics	Method
L_{sur}	Surface Loss	Obtain a smooth surface for an image	Cartoonize (Wang et al. 2020)
L_{stru}	Structure Loss	Get global structure information and sparse colour blocks	Cartoonize (Wang et al. 2020)
L_{tex}	Texture Loss	Get details and edges of the image	Cartoonize (Wang et al. 2020)
L_{dann}	Domain-Adversarial Loss	Bridge the domain gap at the semantic level	XGAN (Royer et al. 2020)
L_{sem}	Semantic Consistency Loss	Ensure the input semantics are preserved after domain translation	XGAN (Royer et al. 2020)
L_{land}	Landmark Consistency Loss	Effectively solve the problem of structural mismatching between unpaired training data	Landmark-assisted CycleGAN (Wu et al. 2019)
L_{idt}	Identity Loss	Further encourage feature preservation	U-GAT-IT (Kim et al. 2019b)
L_{cam}	CAM Loss	Get the biggest difference between two domains	U-GAT-IT (Kim et al. 2019b)
L_{IPA}	Identity-Preservation Adversarial Loss	Distinguish between different identities and styles	WarpGAN (Shi et al. 2019)
L_{attr}	Attribute Matching Loss	Reserve important visual features of the input	StyleCariGAN (Jang et al. 2021)
L_{reg}	Smoothness Regularization Loss	Encourage the warping field to be smooth	AutoToon (Gong et al. 2020)
L_{DT}	Distance Transform Loss	Lead to improved strokes in portrait drawing	APDrawingGAN (Yi et al. 2019)
L_{SP}	Similarity Preserving Loss	Improve the similarity between domains of photo and manga	MangaGAN (Su et al. 2020)
L_{SS}	Structural Smoothing Loss	Make the structure of stroke lines smooth	MangaGAN (Su et al. 2020)
L_{temp}	Temporal Loss	Enforce the temporal consistency between adjacent frames	Huang et al. (2017)
L_{class}	Classification Loss	Train the discriminator for evaluating the amount of information associated with tags in the input image	Tag2Pix (Kim et al. 2019a)
L_{tr}	Triplet Loss	Play a beneficial role in generating realistic images by directly supervising semantic correspondence	Lee et al. (2020)
L_{latent}	Latent Constraint Loss	Constraints on intermediate results of the network	Shi et al. (2020)
L_{MBC}	Margin Binary Classification Loss	Enhance the discrimination of network to better distinguish faces from the dense predictions	ACFD (Zhang et al. 2020a)

$$\begin{aligned}\mathcal{L}_{\text{style}} &= \sum_{\lambda} \left\|\sigma\left(f_{\lambda}(G(x))\right) - \sigma\left(f_{\lambda}(x)\right)\right\|_2 \\ &+ \sum_{\lambda} \left\|\mu\left(f_{\lambda}(G(x))\right) - \mu\left(f_{\lambda}(x)\right)\right\|_2\end{aligned} \tag{7}$$

where $\sigma(x)$ and $\mu(x)$ are the channel-wise mean and variance of input x and $f_{\lambda}(x)$ represents the λ-th layer's feature response of input x.

2.2 Loss Functions Specially Designed for Cartoon

In addition to these representative loss functions, many specific losses are designed for cartoon scenarios, which are summarized in Table S-1.

2.2.1 Surface Loss

To smooth images while maintaining the global semantic structure, a differentiable guided filter F_{dgf} is adopted for edge-preserving filtering in Cartoonize (Wang et al. 2020). It takes an image I as input, and returns the extracted surface representation $\mathcal{F}_{dgf}(I, I)$ with textures and details removed. A discriminator D is introduced to judge whether the model outputs and reference cartoon images have similar surfaces, and guide the generator G to learn the information stored in the extracted surface representation:

$$\begin{aligned}\mathcal{L}_{\text{surface}} &= \log D\left(\mathcal{F}_{dgf}\left(I_c, I_c\right)\right) + \\ &\log\left(1 - D\left(\mathcal{F}_{dgf}\left(G\left(I_p\right), G\left(I_p\right)\right)\right)\right)\end{aligned} \tag{8}$$

where I_p denotes the input photo and I_c denotes the reference cartoon image.

2.2.2 Structure Loss

Cartoonize (Wang et al. 2020) used high-level features extracted by pre-trained VGG16 network to enforce spatial constrain between the results and the extracted structure representation:

$$\begin{aligned}\mathcal{L}_{\text{structure}} = \|&VGG_n(G(I_p)) - \\ &VGG_n(\mathcal{F}_{\text{st}}\left(G(I_p)\right))\|\end{aligned} \tag{9}$$

where F_{st} denotes the structure representation extraction, which emulates flattened global content, sparse color blocks, and clear boundaries in celluloid style cartoon.

2.2.3 Texture Loss

Cartoonize (Wang et al. 2020) proposed a random color shift algorithm F_{rcs} to extract single-channel texture representation from color images, which retains high-frequency textures and decreases the influence of color and luminance.

$$\begin{aligned}\mathcal{F}_{rcs}\left(I_{rgb}\right) = (1-\alpha)(&\beta_1 * I_r + \\ &\beta 2 * I_g + \beta_3 * I_b) + \alpha * Y\end{aligned} \tag{10}$$

where I_{rgb} represents 3-channel RGB color images, I_r, I_g and I_b represent three color channels, and Y represents standard grayscale image converted from RGB color image. Wang et al. (2020) set $\alpha = 0.8$, β_1 β_1 and $\beta_3 \sim U(-1, 1)$. Subsequently, a discriminator D is introduced to distinguish texture representations extracted from the model outputs and the reference cartoons, and guide the generator to learn the clear contours and fine textures stored in the texture representations.

$$\begin{aligned}\mathcal{L}_{\text{texture}} &= \log D_t\left(\mathcal{F}_{rcs}\left(I_c\right)\right) \\ &+ \log\left(1 - D\left(\mathcal{F}_{rcs}\left(G\left(I_p\right)\right)\right)\right)\end{aligned} \tag{11}$$

2.2.4 Domain-Adversarial Loss

The domain-adversarial loss L_{dann} (Royer et al. 2020) pushes embeddings from domain A and domain B to lie in the same subspace, thus bridging the domain gap at the semantic level. This is achieved by training a binary classifier c_{dann} on top of the embedding layer to classify the encoded image as coming from domain A or B. c_{dann} is trained to maximize the classification accuracy while the encoders e_A and e_B strive to confuse the domain-adversarial classifier by minimizing it, as follows:

$$\begin{aligned}\mathcal{L}_{\text{dann}} &= \mathbb{E}_{P_{data}(A)}\ell\left(A, c_{dann}\left(e_A(x)\right)\right) \\ &+\mathbb{E}_{P_{data}(B)}\ell\left(B, c_{dann}\left(e_B(x)\right)\right)\end{aligned} \tag{12}$$

where ℓ denotes a classification loss function (e.g., cross-entropy).

2.2.5 Semantic Consistency Loss

The semantic consistency loss L_{sem} (Royer et al. 2020) ensures that input semantics are preserved after domain translation. To be specific, the semantics of input $x \in \mathcal{D}_A$ should be preserved when x is translated to the other domain $G(x) \in \mathcal{D}_B$, and vice versa. However this consistency property is hard to be assessed at the pixel-level because there are no paired data and pixel-level metrics are sub-optimal for image comparison. Instead, they introduce a feature-level semantic consistency loss, which encourages the network to preserve the learned embedding during domain translation, as follows,

$$\begin{aligned}\mathcal{L}_{sem} &= \mathbb{E}_{x\sim P_{\text{data}(A)}}\left\|e_A(x) - e_B(G(x))\right\| \\ &+\mathbb{E}_{y\sim P_{\text{data}(B)}}\left\|e_B(y) - e_A(F(y))\right\|\end{aligned} \tag{13}$$

where $\|\cdot\|$ denotes a distance between vectors.

2.2.6 Landmark Consistency Loss

Landmark-assisted CycleGAN (Wu et al. 2019) designed L_{land} to enforce the similarity of facial structures. Specifically, they first give constraints on the real landmark and predicted landmark as follows,

$$\mathcal{L}_{land} = \left\|R_B\left(G_{(A,L)\to Y(x,l)}\right) - l\right\|_2 \tag{14}$$

where L indicates the input landmark heatmap set ($l \in L$) and R refers to a pre-trained U-Net like landmark regressor with 5-channel output for respective domain, while R_B are used for domain B.

2.2.7 Identity Loss

To ensure that the color distributions of input image and output image are similar, U-GAT-IT (Kim et al. 2019b) proposes an identity consistency constraint to the generator:

$$\begin{aligned}L_{\text{ide}} &= \mathbb{E}_{x\sim P_{data(A)}}\left[\|x - F(x)\|_1\right] \\ &+\mathbb{E}_{y\sim P_{data(B)}}\left[\|y - G(y)\|_1\right]\end{aligned} \tag{15}$$

2.2.8 CAM Loss

Zhou et al. (2016)have proposed Class Activation Map (CAM) using global average pooling in a CNN. Subsequently, U-GAT-IT (Kim et al. 2019b) uses the CAM to distinguish two domains. By exploiting the information from the auxiliary classifiers η_A and η_D, given an image $y \sim P_{data}(B)$ or $x \sim P_{data}(A)$, G and D get to know where they need to improve or what makes the most difference between two domains in the current state:

$$\begin{aligned}L_{\text{cam}}^{A\to B} &= -\left(\mathbb{E}_{x\sim P_{\text{data}\ (A)}}\left[\log\left(\eta_A(x)\right)\right]\right. \\ &\left.+\mathbb{E}_{y\sim P_{\text{data}\ (B)}}\left[\log\left(1 - \eta_A(y)\right)\right]\right)\end{aligned} \tag{16}$$

$$L_{cam} = L_{cam}^{A\to B} + L_{cam}^{B\to A} \tag{17}$$

$$\begin{aligned}L_{cam}^{D} &= \mathbb{E}_{y\sim P_{data(B)}}\left[\left(\eta_D(y)\right)^2\right] + \\ &\mathbb{E}_{x\sim P_{data(A)}}\left[\left(1 - \eta_D\left(G(x)\right)^2\right]\right.\end{aligned} \tag{18}$$

2.2.9 Attribute Matching Loss

To constrain the shape exaggeration blocks to produce valid caricature deformations, StyleCariGAN (Jang et al. 2021) used facial attribute classifiers for photos and caricatures, respectively. The attribute matching loss L_{attr} is defined by

using binary cross entropy losses between photo attributes and caricature attributes:

$$\begin{aligned}\mathcal{L}_{attr}^{p\to c} = -\mathbb{E}_{w\sim\mathcal{W}}\left[\phi_p\left(G_p(w)\right)\log\phi_c\left(G_{p\to c}(w)\right)\right. \\ \left.+\left(1-\phi_p\left(G_p(w)\right)\right)\log\left(1-\phi_c\left(G_{p\to c}(w)\right)\right)\right]\end{aligned} \tag{19}$$

$$\begin{aligned}\mathcal{L}_{attr}^{c\to p} = -\mathbb{E}_{w\sim\mathcal{W}}\left[\phi_c\left(G_c(w)\right)\log\phi_p\left(G_{c\to p}(w)\right)\right. \\ \left.+\left(1-\phi_c\left(G_c(w)\right)\right)\log\left(1-\phi_p\left(G_{c\to p}(w)\right)\right)\right]\end{aligned} \tag{20}$$

$$\mathcal{L}_{attr} = \mathcal{L}_{attr}^{p\to c} + \mathcal{L}_{attr}^{c\to p} \tag{21}$$

where ϕ_p is a photo attribute classifier, ϕ_c is a caricature attribute classifier, G_p is the photo StyleGAN, G_c is the caricature StyleGAN, $G_{p\to c}$ is p2c-StyleCariGAN, and $G_{c\to p}$ is c2p-StyleCariGAN.

2.2.10 Characteristic Loss

CariGANs (Cao et al. 2018) proposed characteristic loss, with the underlying idea that the differences from a face to the mean face represent its most distinctive features and thus should be kept after exaggeration. Specifically, it penalizes the cosine differences between the input landmark $x \sim P_{data}(A)$ and the predicted one $G(x)$ after subtracting its corresponding means:

$$\begin{aligned}\mathcal{L}_{\text{cha}}^{B}(G) = \mathbb{E}_{x\sim P_{\text{data}(A)}}[1-\cos(x- \\ \overline{P_{\text{data}(A)}}, G(x)-\overline{P_{\text{data}(B)}})]\end{aligned} \tag{22}$$

where $\overline{P_{\text{data}(A)}}$ (or $\overline{P_{\text{data}(B)}}$) denotes the average of $P_{\text{data}(A)}$ (or $P_{\text{data}(B)}$). The characteristic loss in the reverse direction $\mathcal{L}_{\text{cha}}^{A}(F)$ is defined similarly.

2.2.11 Smoothness Regularization Loss

AutoToon (Gong et al. 2020) used a cosine similarity regularization loss L_{reg} that encourages the warping field to be smooth. This can be described as:

$$\begin{aligned}\mathcal{L}_{reg} = \textstyle\sum_{i,j\in\hat{\mathbf{F}}}\left(2-\frac{\left\langle\hat{\mathbf{F}}_{i,j-1},\hat{\mathbf{F}}_{i,j}\right\rangle}{\left\|\hat{\mathbf{F}}_{i,j-1}\right\|\left\|\hat{\mathbf{F}}_{i,j}\right\|}\right. \\ \left.-\frac{\left\langle\hat{\mathbf{F}}_{i-1,j},\hat{\mathbf{F}}_{i,j}\right\rangle}{\left\|\hat{\mathbf{F}}_{i-1,j}\right\|\left\|\hat{\mathbf{F}}_{i,j}\right\|}\right)\end{aligned} \tag{23}$$

where $\left\langle\hat{\mathbf{F}}_{i,j-1},\hat{\mathbf{F}}_{i,j}\right\rangle$ denotes the dot product of the upsampled warping field $\hat{\mathbf{F}}$ at pixel indices i, $j-1$ and i, j.

2.2.12 Distance Transform Loss

L_{DT} is a novel measure to better learn stroke lines by using distance transformation (DT) and chamfer matching. Given an input x, APDrawingGAN (Yi et al. 2019) defined two DTs of x as images $I_{DT}(x)$ and $I'_{DT}(x)$: assuming $\hat{x}$ is the binarized image of x, each pixel in $I_{DT}(x)$ stores the distance value to its nearest black pixel in $\hat{x}$ and each pixel in $I'_{DT}(x)$ stores the distance value to its nearest white pixel in $\hat{x}$. They trained two CNNs to detect black and white lines in APDrawings, denoted as Θ_b and Θ_w respectively. The Chamfer matching distance between x_1 and x_2 was defined as:

$$\begin{aligned}d_{CM}(x_1,x_2) = \sum_{(j,k)\in\Theta_b(x_1)} I_{DT}(x_2)(j,k) \\ + \sum_{(j,k)\in\Theta_w(x_1)} I'_{DT}(x_2)(j,k)\end{aligned} \tag{24}$$

where $I_{DT}(x)(j,k)$ and $I'_{DT}(x)(j,k)$ are distance values at the pixel (i,j) in the images $I_{DT}(x)$ and $I'_{DT}(x)$, respectively. $d_{CM}(x_1,x_2)$ measures the sum of distances from each line pixel in x_1 to the closest pixel with the same type (black or white) in x_2. Then L_{DT} is defined as:

$$\begin{aligned}L_{DT} = \mathbb{E}_{(p_i,a_i)\sim S_{\text{data}}}\left[d_{CM}(a_i, G(p_i))\right. \\ \left.+d_{CM}(G(p_i), a_i)\right]\end{aligned} \tag{25}$$

2.2.13 Similarity Preserving Loss

MangaGAN (Su et al. 2020) designed a Similarity Preserving (SP) module with an SP loss L_{SP} to constrain the similarity in lower resolution space. The SP module leverages a pre-trained network ϕ to extract feature maps in different latent spaces and resolutions. For the forward mapping $\Psi_{app}^{\delta}: \widehat{m}^{\delta} = G_M^{\delta}(p^{\delta})$, input p^{δ} and $G_M^{\delta}(p^{\delta})$ to SP module, and optimize G_M^{δ} by minimizing the loss functions L_{SP} defined as:

$$\begin{aligned}\mathcal{L}_{SP} = \sum_{i\in\phi}\lambda_i\mathcal{L}_{feat}^{\phi,i}\left[f_i^{\phi}(p^{\delta}), f_i^{\phi}(G_M^{\delta}(p^{\delta}))\right] \\ +\lambda_I\mathcal{L}_{\text{pixel}}^{I}\left[p^{\delta}, G_M^{\delta}(p^{\delta})\right]\end{aligned} \tag{26}$$

where λ_i and λ_I control the relative importance of each objective, $\mathcal{L}_{\text{pixel}}^{I}$ and $\mathcal{L}_{feat}^{\phi,i}$ are used to keep

the similarity on pixel-wise and different feature-wise respectively:

$$\mathcal{L}_{feat}^{\phi,i} = \left\| f_i^{\phi}\left(p^{\delta}\right) - f_i^{\phi}\left(G_M^{\delta}\left(p^{\delta}\right)\right)\right\|_2^2 \tag{27}$$

$$\mathcal{L}_{\text{pixel}}^{I} = \left\| p^{\delta} - G_M^{\delta}\left(p^{\delta}\right)\right\|_2^2 \tag{28}$$

where $f_i^{\phi}(x)$ is a feature map extracted from i-th layer of network phi when x as the input.

2.2.14 Structural Smoothing Loss

The structural smoothing loss L_{SS} (MangaGAN, Su et al. 2020) is designed for encouraging networks to produce manga with smooth stroke-lines, defined as:

$$\begin{aligned}\mathcal{L}_{SS} = &\frac{1}{\sqrt{2\pi}\sigma}\left[\sum \exp\left(\frac{-\left(G_P^{\delta}\left(m^{\delta}\right)_j - \mu\right)^2}{2\sigma^2}\right)\right. \\ &\left.+\sum \exp\left(-\frac{\left(G_M^{\delta}\left(p^{\delta}\right)_k - \mu\right)^2}{2\sigma^2}\right)\right]\end{aligned} \tag{29}$$

where $j \in \{1, 2, \ldots, N\}$, $k \in \{1, 2, \ldots, N\}$, and L_{SS} is based on a Gaussian model with $\mu = \frac{255}{2}$, $G_P^{\delta}\left(m^{\delta}\right)_j$ or $G_M^{\delta}\left(p^{\delta}\right)_k$ is the j-th or k-th pixel of $G_P^{\delta}\left(m^{\delta}\right)$ or $G_M^{\delta}\left(p^{\delta}\right)$.

2.2.15 Temporal Loss

By simply applying the cartoon style transfer method to video frames, flicker artifacts will be inevitably introduced. Therefore, Huang et al. (2017) incorporated a temporal loss to enforce the temporal consistency between adjacent frames. The temporal loss is defined as the mean square error between the stylized output at time t and the warped version of the stylized output at time $t-1$:

$$\mathcal{L}_{\text{temp}} = \frac{1}{D}\sum_{k=1}^{D} \mathbf{c}_k \left(\hat{x}_k^t - f\left(\hat{x}_k^{t-1}\right)\right)^2 \tag{30}$$

where $\hat{x}^t$ and $\hat{x}^{t-1}$ are the stylized results of the current frame and the previous one, respectively. $f\left(\hat{x}_k^{t-1}\right)$ is a function that warps the stylized output at time $t-1$ to time t according to a pre-computed optical flow. $D = H \times W \times C$ is the dimension of the output. $\mathbf{c} \in [0,1]^D$ denotes the per-pixel confidence of the optical flow: 0 in occluded regions and at motion boundaries, and 1 otherwise.

2.2.16 Classification Loss

The tag classification loss L_{class} is introduced by Tag2Pix (Kim et al. 2019a), which makes the generator and discriminator learn colorization based on a better understanding of object shape and location:

$$\begin{aligned}\mathcal{L}_{class} = &\mathbb{E}_{y,c_v,c_i}\left[-\log D_{cls}\left(c_v, c_i \mid y\right)\right] + \\ &\mathbb{E}_{x,c_v,c_i}\left[-\log D_{cls}\left(c_v, c_i \mid G_f\left(x, c_v\right)\right)\right]\end{aligned} \tag{31}$$

where c_i is the CIT(color invariant tags) values and $D_{cls}\left(c_v, c_i \mid y\right)$ denotes the binary classification for each tag by giving y.

2.2.17 Similarity-based Triplet Loss

Utilizing pixel-level correspondence information, Lee et al. (2020) proposed a similarity-based triplet loss, which is a variant of triplet loss (Schroff et al. 2015), it directly supervised the affinity between the pixel-wise query and key vectors used to compute the attention map, which is computed as:

$$\mathcal{L}_{tr} = \max\left(0, \left[-S\left(v_q, v_k^p\right) + S\left(v_q, v_k^n\right) + \gamma\right]\right) \tag{32}$$

where $S(\cdot,\cdot)$ computes the scaled dot product. Given a query vector v_q as an anchor, v_k^p indicates a feature vector sampled from the positive region, and v_k^n is a negative sample. γ denotes a margin, which is the minimum distance that $S\left(v_q, v_k^p\right)$ and $S\left(v_q, v_k^n\right)$ should maintain. L_{tr} encourages the query representation to be close to the correct (positive) key representation, while penalizing it it to be far from the wrong (negatively sampled) one.

2.2.18 Latent Constraint Loss

In order to improve the stability of the generated effect, Shi et al. (2020) introduced extra constraints on the intermediate results of the network. Specifically, they added multi-supervised

constraints to the similarity-based color transform layer output f_{sim} and middle module output f_{mid}. To make perceptual similarity measured more easily, f_{sim} and f_{mid} were first passed through the latent decoders to output 3-channel color images $\hat{y}_{\text{sim}}$ and $\hat{y}_{\text{mid}}$. Then they used L_1 loss to measure their similarity with the ground truth as follows:

$$\mathcal{L}_{\text{latent}} = \|y - \hat{y}_{\text{sim}}\|_1 + \|y - \hat{y}_{\text{mid}}\|_1 \quad (33)$$

2.2.19 Margin Binary Classification Loss

To improve the classification ability of network so that it can distinguish faces who are similar to the background, ACFD (Zhang et al. 2020a) applied the widely used margin-based loss function (Deng et al. 2019) to cartoon face classification. The margin-based losses share the same idea that maximizing the inter-class variance, minimizing the intra-class variance, and enhance the capacity of discrimination by adding an extra hard margin. In margin binary classification application, suppose p_i is the output of the network, then the margin-based prediction is formulated as follow:

$$p_i^m = [p_i^o = 1] \cdot (p_i - m) + [p_i^o = 0] \cdot p_i \quad (34)$$

where p_i^m is the corresponding one-hot label, m is a hard margin, and p_i^m is used for the computation of classification loss.

3 Experimental Results of Cartoon Enhancement Task

For cartoon enhancement task, we select 2D cartoon video sequences from the Youku-VESR (2019) dataset, and then obtain the degraded input videos by means of 4× downsampling and H.264 video compression (with random "-crf" parameters from 30 to 38). Hence, this testing mainly compare the super-resolution and compression artifacts removal results, which are common degradations in cartoon videos.

In this experiment, we selected single-frame natural image enhancement models of DBPN (Haris et al. 2018), EDSR (Lim et al. 2017), ESRGAN (Wang et al. 2018), and a multi-frame model EDVR (Wang et al. 2019), and two cartoon image enhancement methods of Anime4K (2019) and waifu2x (2018). The visualization results are shown in Fig.S-4. Table S-2 listed the objective results of these methods. Note that none of these algorithms have been retrained, and by comparing the visual quality and the PSNR/SSIM values, it can be clearly seen that the waifu2x and Anime4K are more robust for cartoon images than directly applying state-of-the-art natural image enhancement methods. It is worth pointing out that these deep models have a high learning capacity and can achieve better results if they have specifically designed and retrained for real cartoon sequences.

Table S- 2 PSNR and SSIM results of different enhancement methods for cartoon images

Models	PSNR	SSIM
DBPN	29.765	0.846
EDSR	29.742	0.844
ESRGAN	27.390	0.725
EDVR	29.760	0.843
Anime4K	29.811	0.887
waifu2x	30.719	0.902

References

(2018) waifu2x. URL https://github.com/nagadomi/waifu2x

(2019) Anime4k. URL https://github.com/bloc97/Anime4K

(2019) Youku video super-resolution and enhancement challenge(youku-vsre2019). [Online], Available:https://tianchi.aliyun.com/dataset/dataDetail?dataId=39568 dataset, 2019

(2021) naver-webtoon-faces. URL https://github.com/bryandlee/naver-webtoon-faces

Anonymous, community D, Branwen G (2021) Danbooru2020: A large-scale crowdsourced and tagged anime illustration dataset. https://www.gwern.net/Danbooru2020, URL https://www.gwern.net/Danbooru2020

Cao K, Liao J, Yuan L (2018) Carigans: Unpaired photo-to-caricature translation. *arXiv preprint arXiv:181100222*

Cohn N, Taylor R, Pederson K (2017) A picture is worth more words over time: Multimodality and narrative structure across eight decades of american superhero comics. *Multimodal Communication* 6(1):19–37

Deng J, Guo J, Xue N, Zafeiriou S (2019) Arcface: Additive angular margin loss for

Fig.S- 4 Reconstruction results of several enhancement methods for cartoon images

deep face recognition. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp 4690–4699

Dunst A, Hartel R, Laubrock J (2017) The graphic narrative corpus (gnc): design, annotation, and analysis for the digital humanities. In: *2017 14th IAPR international conference on document analysis and recognition (ICDAR)*, IEEE, vol 3, pp 15–20

Fujimoto A, Ogawa T, Yamamoto K, Matsui Y, Yamasaki T, Aizawa K (2016) Manga109 dataset and creation of metadata. In: *Proceedings of the 1st international workshop on comics analysis, processing and understanding*, pp 1–5

Gong J, Hold-Geoffroy Y, Lu J (2020) Autotoon: Automatic geometric warping for face cartoon generation. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp 360–369

Guérin C, Rigaud C, Mercier A, Ammar-Boudjelal F, Bertet K, Bouju A, Burie JC, Louis G, Ogier JM, Revel A (2013) ebdtheque: a representative database of comics. In: *2013 12th International Conference on Document Analysis and Recognition*, IEEE, pp 1145–1149

Haris M, Shakhnarovich G, Ukita N (2018) Deep back-projection networks for super-resolution. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp 1664–1673

Hicsonmez S, Samet N, Akbas E, Duygulu P (2020) Ganilla: Generative adversarial networks for image to illustration translation. *Image and Vision Computing* 95:103886

Huang H, Wang H, Luo W, Ma L, Jiang W, Zhu X, Li Z, Liu W (2017) Real-time neural style transfer for videos. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp 783–791

Huo J, Li W, Shi Y, Gao Y, Yin H (2017) Webcaricature: a benchmark for caricature recognition. *arXiv preprint arXiv:170303230*

Ikuta H, Ogaki K, Odagiri Y (2016) Blending texture features from multiple reference images for style transfer. In: *SIGGRAPH ASIA 2016 Technical Briefs*, pp 1–4

Iyyer M, Manjunatha V, Guha A, Vyas Y, Boyd-Graber J, Daume H, Davis LS (2017) The amazing mysteries of the gutter: Drawing inferences between panels in comic book narratives. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern recognition*, pp 7186–7195

Jang W, Ju G, Jung Y, Yang J, Tong X, Lee S (2021) Stylecarigan: caricature generation via stylegan feature map modulation. *ACM Transactions on Graphics (TOG)* 40(4):1–16

Kim H, Jhoo HY, Park E, Yoo S (2019a) Tag2pix: Line art colorization using text tag with secat and changing loss. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp 9056–9065

Kim J, Kim M, Kang H, Lee K (2019b) U-gat-it: unsupervised generative attentional networks with adaptive layer-instance normalization for image-to-image translation. *arXiv preprint arXiv:190710830*

Lee J, Kim E, Lee Y, Kim D, Chang J, Choo J (2020) 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*, pp 5801–5810

Li X, Zhang W, Shen T, Mei T (2019) Everyone is a cartoonist: Selfie cartoonization with attentive adversarial networks. In: *2019 IEEE International Conference on Multimedia and Expo (ICME)*, IEEE, pp 652–657

Lim B, Son S, Kim H, Nah S, Mu Lee K (2017) Enhanced deep residual networks for single image super-resolution. In: *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pp 136–144

Mishra A, Rai SN, Mishra A, Jawahar C (2016) Iiit-cfw: A benchmark database of cartoon faces in the wild. In: *European Conference on Computer Vision*, Springer, pp 35–47

Royer A, Bousmalis K, Gouws S, Bertsch F, Mosseri I, Cole F, Murphy K (2020) Xgan: Unsupervised image-to-image translation for many-to-many mappings. In: *Domain Adaptation for Visual Understanding*, Springer, pp 33–49

Schroff F, Kalenichenko D, Philbin J (2015) Facenet: A unified embedding for face recognition and clustering. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp 815–823

Shi M, Zhang JQ, Chen SY, Gao L, Lai YK, Zhang FL (2020) Deep line art video colorization with a few references. *arXiv preprint arXiv:200310685*

Shi Y, Deb D, Jain AK (2019) Warpgan: Automatic caricature generation. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp 10762–10771

Su H, Niu J, Liu X, Li Q, Cui J, Wan J (2020) Unpaired photo-to-manga translation based on the methodology of manga drawing. *arXiv preprint arXiv:200410634*

Wang X, Yu J (2020) Learning to cartoonize using white-box cartoon representations. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp 8090–8099

Wang X, Yu K, Wu S, Gu J, Liu Y, Dong C, Qiao Y, Change Loy C (2018) Esrgan: Enhanced super-resolution generative adversarial networks. In: *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, pp 63–69

Wang X, Chan KC, Yu K, Dong C, Change Loy C (2019) Edvr: Video restoration with enhanced deformable convolutional networks. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp 1954–1963

Wilber MJ, Fang C, Jin H, Hertzmann A, Collomosse J, Belongie S (2017) Bam! the behance artistic media dataset for recognition beyond photography. In: *Proceedings of the IEEE international conference on computer vision*, pp 1202–1211

Wu R, Gu X, Tao X, Shen X, Tai YW, et al. (2019) Landmark assisted cyclegan for cartoon face generation. *arXiv preprint arXiv:190701424*

Yi R, Liu YJ, Lai YK, Rosin PL (2019) Apdrawinggan: Generating artistic portrait drawings from face photos with hierarchical gans. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp 10743–10752

Zhang B, Li J, Wang Y, Cui Z, Xia Y, Wang

C, Li J, Huang F (2020a) Acfd: Asymmetric cartoon face detector. *arXiv preprint arXiv:200700899*

Zhang L, Li C, Wong TT, Ji Y, Liu C (2018) Two-stage sketch colorization. *ACM Transactions on Graphics (TOG)* 37(6):1–14

Zhang L, JI Y, Liu C (2020b) Danbooregion: An illustration region dataset. In: *European Conference on Computer Vision (ECCV)*, pp 137–154

Zheng Y, Zhao Y, Ren M, Yan H, Lu X, Liu J, Li J (2020) Cartoon face recognition: A benchmark dataset. In: *Proceedings of the 28th ACM International Conference on Multimedia*, pp 2264–2272

Zhou B, Khosla A, Lapedriza A, Oliva A, Torralba A (2016) Learning deep features for discriminative localization. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp 2921–2929