CollaGAN: Collaborative GAN for Missing Image Data Imputation

Dongwook Lee[1], Junyoung Kim[1], Won-Jin Moon[2], Jong Chul Ye[1]
[1]: Bio-Imaging, Signal Processing, and Learning (BISPL) Group
Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea
{dongwook.lee, junyoung.kim, jong.ye}@kaist.ac.kr
[2]: Konkuk University Medical Center, Seoul, Korea
mdmoonwj@kuh.ac.kr

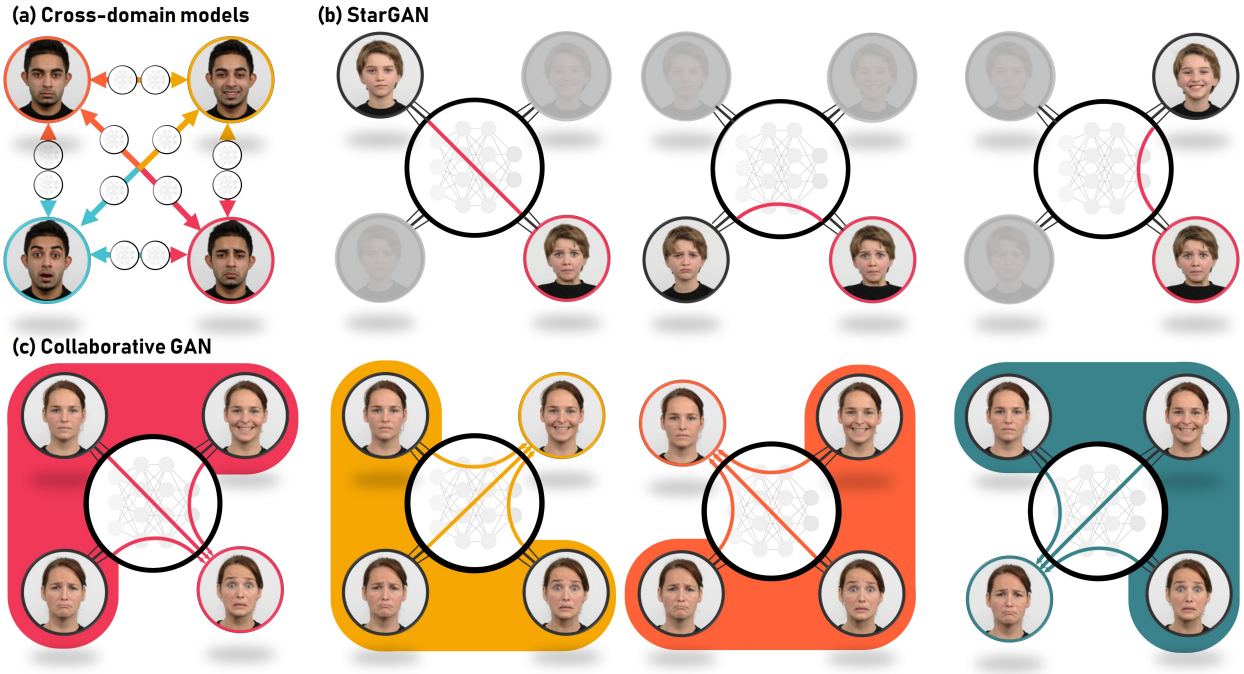

Figure 1: Image translation tasks using (a) cross-domain models, (b) StarGAN, and (c) the proposed collaborative GAN (CollaGAN). Cross-domain model needs large number of generators to handle multi-class data. StarGAN and CollaGAN use a single generator with one input and multiple inputs, respectively, to synthesize the target domain image.

Abstract

In many applications requiring multiple inputs to obtain a desired output, if any of the input data is missing, it often introduces large amounts of bias. Although many techniques have been developed for imputing missing data, the image imputation is still difficult due to complicated nature of natural images. To address this problem, here we proposed a novel framework for missing image data imputation, called Collaborative Generative Adversarial Network (CollaGAN). CollaGAN convert the image imputation problem to a multi-domain images-to-image translation task so that a single generator and discriminator network can successfully estimate the missing data using the remaining clean data set. We demonstrate that CollaGAN produces the images with a higher visual quality compared to the existing competing approaches in various image imputation tasks.

1. Introduction

In many image processing and computer vision applications, multiple set of input images are required to generate the desired output. For example, in brain magnetic reso-

nance imaging (MRI), MR images with T1, T2, or FLAIR (FLuid-Attenuated Inversion Recovery) contrast are all required for accurate diagnosis and segmentation of cancer margin [6]. In generating a 3-D volume from multiple view camera images [5], most algorithms require the pre-defined set of view angles. Unfortunately, the complete set of input data are often difficult to obtain due to the acquisition cost and time, (systematic) errors in the data set, etc. For example, in synthetic MR contrast generation using the Magnetic Resonance Image Compilation (MAGiC, GE Healthcare) sequence, it is often reported that there exists a systematic error in synthetic T2-FLAIR contrast images, which leads to erroneous diagnosis [30]. Missing data can also cause substantial biases, making errorrs in data processing and anlysis and reducing the statistical efficiency [22].

Rather than acquiring all the datasets again in this unexpected situation, which is often not feasible in clinical environment, it is often necessary to replace the missing data with substituted values. This process is often referred to as *imputation*. Once all missing values have been imputed, the data set can be used as an input for standard techniques designed for the complete data set.

There are several standard methods to impute missing data based on the modeling assumption for the whole set such as mean imputation, regression imputation, stochastic imputation, etc [2, 9]. Unfortunately, these standard algorithms have limitations for high-dimensional data such as images, since the image imputation requires knowledge of high-dimensional image data manifold.

Similar technical issues exist in image-to-image translation problems, whose goal is to change a particular aspect of a given image to another. The tasks such as super-resolution, denoising, deblurring, style transfer, semantic segmentation, depth prediction, etc can be treated as mapping an image from one domain to a corresponding image in another domain [10, 3, 7, 8]. Here, each domain has a different aspect such as resolution, facial expression, angle of light, etc, and one needs to know the intrinsic manifold structure of the image data set to translate between the domains. Recently, these tasks have been significantly improved thanks to the generative adversarial networks (GANs) [11]. Specifically, cycleGAN [35] or DiscoGAN [18] have been the main workhorse to transfer image between two domains [17, 21]. These approaches are, however, ineffective in generalizing to multiple domain image transfer, since $N(N\text{-}1)$ number of generators are required for N-domain image transfer (Fig. 1 (a)). To generalize the idea for multi-domain translation, Choi et al [4] proposed a so-called StarGAN which can learn translation mappings among multiple domains by single generator (Fig. 1 (b)). Similar multi-domain transfer network have been proposed recently [33].

These GAN-based image transfer techniques are closely related to image data imputation, since the image translation can be considered as a process of estimating the missing image database by modeling the image manifold structure. However, there are fundamental differences between image imputation and image translation. For example, CycleGAN and StarGAN are interested in transferring one image to another as shown in Fig. 1 (a)(b) without considering the remaining domain data set. However, in image imputation problems, the missing data occurs infrequently, and the goal is to estimate the missing data by utilizing the other clean data set. Therefore, an image imputation problem can be correctly described as in Fig. 1(c), where one generator can estimate the missing data using the remaining clean data set. Since the missing data domain is not difficult to estimate a priori, the imputation algorithm should be designed such that one algorithm can estimate the missing data in *any* domain by exploiting the data for the rest of the domains.

The proposed image imputation technique called Collaborative Generative Adversarial Network (CollaGAN) offers many advantages over existing methods:

- The underlying image manifold can be learned more synergistically from the multiple input data set sharing the same manifold structure, rather than from a single input. Therefore, the estimation of missing data using CollaGAN is more accurate.

- CollaGAN still retains the one-generator architecture similar to StarGAN, which is more memory-efficient compared to CycleGAN.

We demonstrate the proposed algorithm shows the best performance among the state-of-the art algorithms for various image imputation tasks.

2. Related Work

2.1. Generative Adversarial Network

Typical GAN framework [11] consists of two neural networks: the generator G and the discriminator D. While the discriminator tries to find the features to distinguish between fake/real samples during the train process, the generator learns to eliminate/synthesize the features which the discriminator use to judge fake/real. Thus, GANs could generate more realistic samples which cannot be distinguished by the discriminator between real and fake. GANs have shown remarkable results in various computer vision tasks such as image generation, image translation, etc [16, 21, 18].

2.2. Image-to-image translation

Unlike the original GAN, Conditional GAN (CoGAN) [26] controls the output by adding some information labels as an additional parameter to the generator. Here,

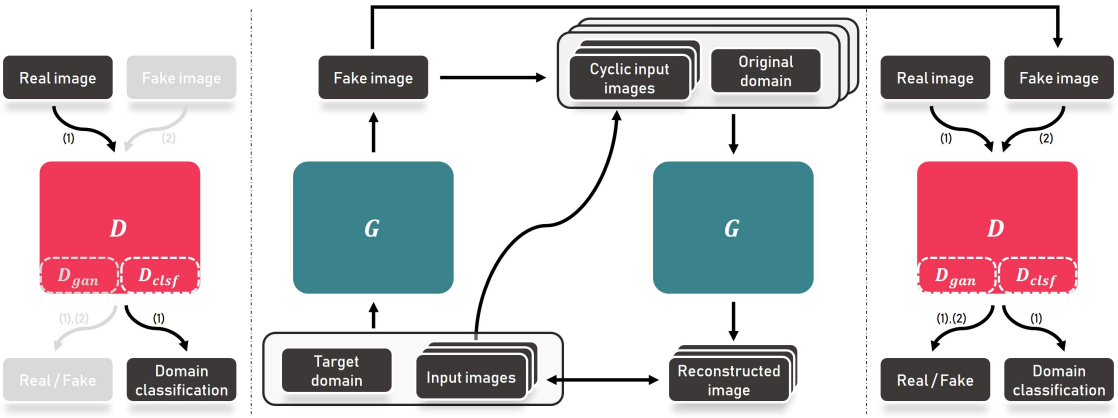

Figure 2: Flow of the proposed method. D has two branches: domain classification D_{clsf} and source classficiation D_{gan} (real/fake). First, D_{clsf} is only trained by (1) the loss calculated from real samples (left). Then G reconstructs the target domain image using the set of input images (middle). For the cycle consistency, the generated fake image re-entered to the G with inputs images and G produces the multiple reconstructed outputs in orignal domains. Here, D_{clsf} and D_{gan} are simultaneously trained by the loss from (1) only real images and (2) both real and fake images, respectively (right).

instead of generating a generic sample from an unknown noise distribution, the generator learns to produce a fake sample with a specific condition or characteristics (such as a label associated with an image or a more detailed tag). A successful application of conditional GAN is for the image-to-image translation, such as pix2pix [17] for paired data, and CycleGAN for unpaired data [23, 35].

CycleGAN [35] and DiscoGAN [18] attempt to preserve key attributes between the input and the output images by utilizing a cycle consistency loss. However, these frameworks are only able to learn the relationships between two different domains at a time. These approaches have scalability limitations when dealing with multi-domains, since each domain pair needs a separate generator-pair and total $N\times(N\text{-}1)$ number of generators are required to handle the N-distinct domains.

StarGAN [4] and Radial GAN [33] are recent frameworks that deal with multiple domains using a single generator. For example, in StarGAN [4], the depth-wise concatenation from input image and the mask vector representing the target domain helps to map the input to reconstructed image in the target domain. Here, the discriminator should be designed to play another role for domain classification. Specifically, the discriminator decides that not only the sample is real or fake, but also the class of the sample.

3. Theory

Here, we explain our Collaborative GAN framework to handle multiple inputs to generate more realistic and more feasible output for image imputation. Compared to StarGAN, which handles single-input and single-output, the multiple-inputs from multiple domains are processed using the proposed method.

3.1. Image imputation using multiple inputs

For ease of explanation, we assume that there are four types ($N = 4$) of domains: a, b, c, and d. To handle the multiple-inputs using a single generator, we train the generator to synthesize the output image in the target domain, $\hat{x}_a$, via a collaborative mapping from the set of the other types of multiple images, $\{x_a\}^C = \{x_b, x_c, x_d\}$, where the superscript ^{C} denotes the complementary set. This mapping is formally described by

$$\hat{x}_\kappa = G\left(\{x_\kappa\}^C; \kappa\right) \tag{1}$$

where $\kappa \in \{a, b, c, d\}$ denotes the target domain index that guides to generate the output for the proper target domain, κ. As there are N number of combinations for multiple-input and single-output combination, we randomly choose these combination during the training so that the generator learns the various mappings to the multiple target domains.

3.2. Network losses

Multiple cycle consistency loss One of the key concepts for the proposed method is the cycle consistency for multiple inputs. Since the inputs are multiple images, the cycle loss should be redefined. Suppose that the output from the forward generator G is $\hat{x}_a$. Then, we could generate $N - 1$ number of new combinations as the other inputs for the backward flow of the generator (Fig. 2 middle). For example, when $N = 4$, there are three combinations of multi-input and single-output so that we can reconstruct the three images of original domains using backward flow of the gen-

erator as:

$$\begin{aligned}\tilde{x}_{b|a} &= G(\{\hat{x}_a, x_c, x_d\}; b)\\ \tilde{x}_{c|a} &= G(\{\hat{x}_a, x_b, x_d\}; c)\\ \tilde{x}_{d|a} &= G(\{\hat{x}_a, x_b, x_c\}; d)\end{aligned}$$

Then, the associated multiple cycle consistency loss can be defined as following:

$$\mathcal{L}_{mcc,a} = ||x_b - \tilde{x}_{b|a}||_1 + ||x_c - \tilde{x}_{c|a}||_1 + ||x_d - \tilde{x}_{d|a}||_1$$

where $||\cdot||_1$ is the l_1-norm. In general, the cycle consistency loss for the forward generator $\hat{x}_\kappa$ can be written by

$$\mathcal{L}_{mcc,\kappa} = \sum_{\kappa' \neq \kappa} ||x_{\kappa'} - \tilde{x}_{\kappa'|\kappa}||_1 \tag{2}$$

where

$$\tilde{x}_{\kappa'|\kappa} = G\left(\{\hat{x}_\kappa\}^C; \kappa'\right) . \tag{3}$$

Discriminator Loss As mentioned before, the discriminator has two roles: one is to classify the source which is real or fake, and the other is to classify the type of domain which is class a, b, c or d. Therefore, the discriminator loss consists of two parts: adversarial loss and domain classification loss. As shown in Fig. 2, this can be realized using a discriminator with two paths D_{gan} and D_{clsf} that share the same neural network weights except the last layers.

Specifically, the adversarial loss is necessary to make the generated images as real as possible. The regular GAN loss might lead to the vanishing gradients problem during the learning process [24, 1]. To overcome such problem and improve the robustness of the training, the adversarial loss of Least Square GAN [24] was utilized instead of the original GAN loss. In particular for the optimization of the discriminatorD_{gan}, the following loss is minimized:

$$\mathcal{L}^{dsc}_{gan}(D_{gan}) = \mathbb{E}_{x_\kappa}[(D_{gan}(x_\kappa)-1)^2] + \mathbb{E}_{\tilde{x}_{\kappa|\kappa}}[(D_{gan}(\tilde{x}_{\kappa|\kappa}))^2],$$

whereas the generator is optimized by minimizing the following loss:

$$\mathcal{L}^{gen}_{gan}(G) = \mathbb{E}_{\tilde{x}_{\kappa|\kappa}}[(D_{gan}(\tilde{x}_{\kappa|\kappa}) - 1)^2]$$

where $\tilde{x}_{\kappa|\kappa}$ is defined in (3).

Next, the domain classification loss consists of two parts: $\mathcal{L}^{real}_{clsf}$ and $\mathcal{L}^{fake}_{clsf}$. They are the cross entropy loss for domain classification from the real images and the fake image, respectively. Recall that the goal of training G is to generate the image properly classified to the target domain. Thus, we first need a best classifier D_{clsf} that should only be trained with the real data to guide the generator properly. Accordingly, we first minimize the loss $\mathcal{L}^{real}_{clsf}$ to train the classifier D_{clsf}, then $\mathcal{L}^{fake}_{clsf}$ is minimized by training G with fixing D_{clsf} so that the generator can be trained to generate samples that can be classified correctly.

Specifically, to optimize the D_{clsf}, the following $\mathcal{L}^{real}_{clsf}$ should be minimizied with respect to D_{clsf}:

$$\mathcal{L}^{real}_{clsf}(D_{clsf}) = \mathbb{E}_{x_\kappa}[-\log(D_{clsf}(\kappa; x_\kappa))] \tag{4}$$

where $D_{clsf}(\kappa; x_\kappa)$ can be interpreted as the probability to correctly classify the real input x_κ as the class κ. On the other hand, the generator G should be trained to generate fake samples which are properly classified by the D_{clsf}. Thus, the following loss should be minimized with respect to G:

$$\mathcal{L}^{fake}_{clsf}(G) = \mathbb{E}_{\hat{x}_{\kappa|\kappa}}[-\log(D_{clsf}(\kappa; \hat{x}_{\kappa|\kappa}))] \tag{5}$$

Structural Similarity Index Loss Structural Similarity Index (SSIM) is one of the state-of-the-art metrics to measure the image quality [32]. The l_2 loss, which is widely used for the image restoration tasks, has been reported to cause the blurring artifacts on the results [21, 25, 34]. SSIM is one of the perceptual metrics and it is also differentiable, so it can be backpropagated [34]. The SSIM for pixel p is defined as

$$\text{SSIM}(p) = \frac{2\mu_X\mu_Y + C_1}{\mu_X^2 + \mu_Y^2 + C_1} \cdot \frac{2\sigma_{XY} + C_2}{\sigma_X^2 + \sigma_Y^2 + C_2} \tag{6}$$

where μ_X is an average of X, σ_X^2 is a variance of X and σ_{XX^*} is a covariance of X and X^*. There are two variables to stabilize the division such as $C_1 = (k_1 L)^2$ and $C_2 = (k_2 L)^2$. L is a dynamic range of the pixel intensities. k_1 and k_2 are constants by default $k_1 = 0.01$ and $k_2 = 0.03$. Since the SSIM is defined between 0 and 1, the loss function for SSIM can be written by:

$$\mathcal{L}_{\text{SSIM}}(X, Y) = -\log\left(\frac{1}{2|P|}\sum_{p \in P(X,Y)}(1 + \text{SSIM}(p))\right) \tag{7}$$

where P denotes the pixel location set and $|P|$ is its cardinality. The SSIM loss was applied as an additional multiple cycle consistency loss as follows:

$$\mathcal{L}_{mcc-\text{SSIM},\kappa} = \sum_{\kappa' \neq \kappa} \mathcal{L}_{\text{SSIM}}\left(x_{\kappa'}, \tilde{x}_{\kappa'|\kappa}\right). \tag{8}$$

3.3. Mask vector

To use the single generator, we need to add the target label as a form of mask vector to guide the generator. The mask vector is a binary matrix which has same dimension with the input images to be easily concatenated. The mask vector has N class number of channel dimension to represent the target domain as one-hot vector along the channel dimension. This is the simplified version of mask vector which was originally introduced in StarGAN [4].

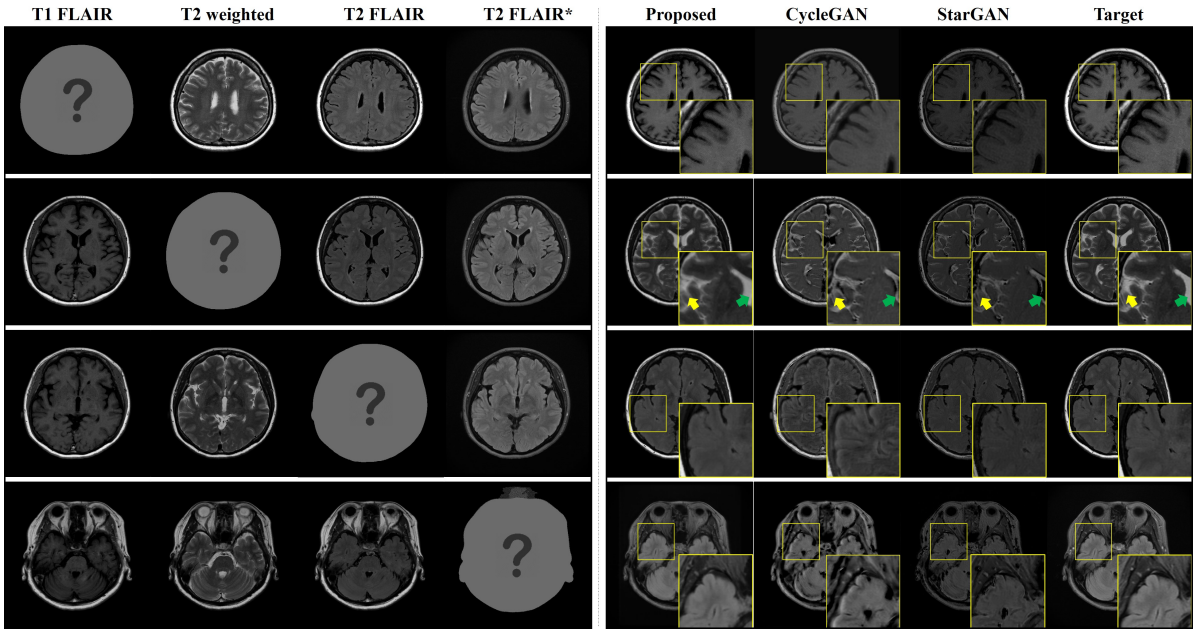

Figure 3: MR contrast imputation results. The generated images (right) were reconstructed from the other contrast inputs (left). The yellow and green arrows point out the remarkable parts of the results. For CycleGAN and StarGAN, the T2-FLAIR* contrast was used as an input for the T1-FLAIR/ T2-weighted/ T2-FLAIR contrast imputation, and the T1-FLAIR contrast was used as an input for the T2-FLAIR* contrast imputation. The image to impute is marked as the question mark.

4. Method

4.1. Datasets

MR contrast synthesis Total 280 axis brain images were scanned by multi-dynamic multi-echo sequence and the additional T2 FLAIR (FLuid-Attenuated Inversion Recovery) sequence from 10 subjects. There are four types of MR contrast images in the dataset: T1-FLAIR (T1F), T2-weighted (T2w), T2-FLAIR (T2F), and T2-FLAIR* (T2F*). The first three contrasts were acquired from MAGnetic resonance image Compilation (MAGiC, GE Healthcare) and T2-FLAIR* was acquired by the addtional scan with different MR scan parameter of the third contrast (T2F). The details of MR acquisition parameters are available in Supplementary Material.

CMU Multi-PIE For the illumination translation task, the subset of Carnegie Mellon Univesity Multi-Pose Illumination and Expression face database [12] was used. There were 250 participants in the first session and the frontal face of neutral expression were selected with the following five illumination conditions: -90°(right), -45°, 0°(front), 45°and 90°(left). The images were cropped by 240×240 where the faces are centered as shown in Fig. 4.

RaFD The Radboud Faces Database (RaFD) [20] contains eight different facial expressions collected from the 67 participants; neutral, angry, contemptuous, disgusted, fearful, happy, sad, and surprised. Also, there are three different gaze directions and therefore total 1,608 images were divided by subjects for train, validation and test set. We crop the images to 640×640 and resize them to 128×128.

4.2. Network Implementation

The proposed method consists of two networks, the generator and the discriminator (Fig. 2). To achieve the best performance for each task, we redesigned the generators and discriminator to fit for the property of each task, while the general network architecture are similar.

Generators

The generators are based on the U-net [27] structure. U-net consists of the encoder/decoder parts and the each parts between encoder/decoder are connected by contracting paths [27]. The instance normalization [31] and Leaky-ReLU [13] was used instead of batch normalization and ReLU, respectively. We also redesigned the architecture of the networks to fit for each task as described in the followings. ./

MR contrast translation There are various MR contrasts such as T1 weight contrast, T2 weight contrast, etc. The specific MR contrast scan is determined by the MRI scan parameters such as repetition time (TR), echo time (TE) and so on. The pixel intensities of the MR contrast image are decided based on the physical property of the tissues called MR parameters of the tissues, such as T1, T2, proton density, etc. The MR parameter is the voxel-wise

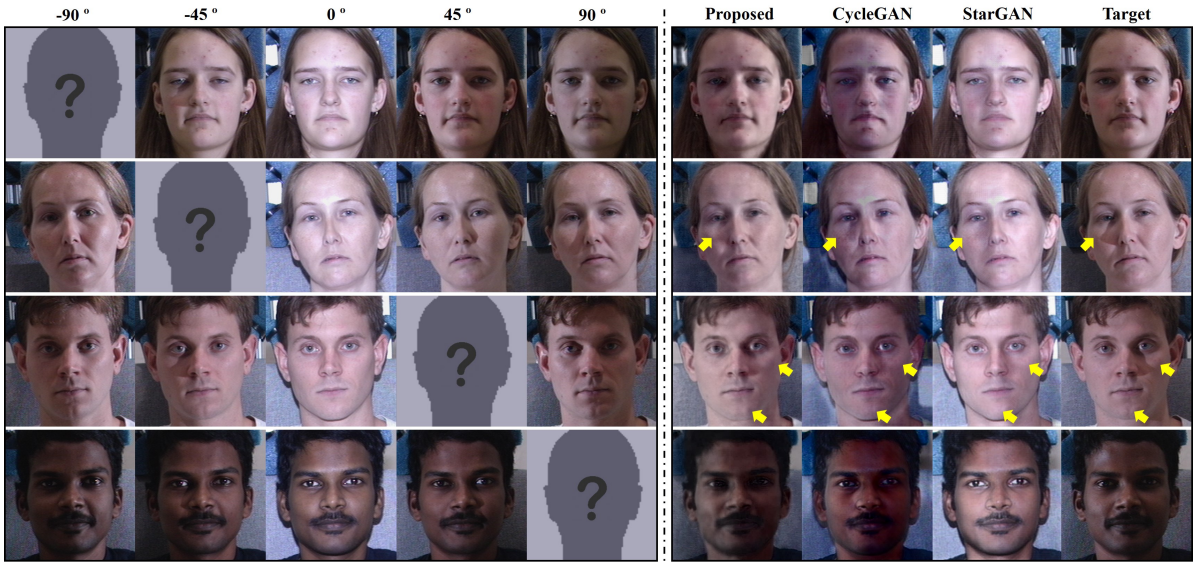

Figure 4: Illumination imputation results at (-90°, -45°, 45°and 90°). The imputed images (right) were reconstructed from the inputs with multiple illuminations (left). The yellow arrows shows remarkable parts. The frontal illumination (0°) image was given as the input of CycleGAN and StarGAN. The image to impute is marked as the question mark.

property. This means that for the convolutional neural network, the pixel-by-pixel processing is just as important as processing with the information from neighborhood and/or a large FOV. Thus, instead of using single convolution, the generator uses two convolution branches with 1x1 and 3x3 filters to handle the multi-scale feature information. The two branches of the convolutions are concatenated similar to the inception network [29].

Illumination translation For the illumination translation task, the original U-net structure with instance normalization [31] was used instead of batch normalization.

Facial expression translation For the facial expression translation task, the inputs are multiple facial images with various facial expressions. Since there exists the head movements of the subjects between the facial expressions, the images are not strictly aligned pixel-wise manner. If we use the original U-net for the facial expresion images-to-image task, the generator show poor performance because the informations from the multiple facial expressions are mixed up in the very early stage of the network. From the intuition, the features from the facial expressions should be mixed up in the middle stage of the generator where the features are calculated from the large FOV or already downsampled by pooling layers. Thus, the generators are redesigned with eight branches of encoders for each eight facial expressions and they are concatenated after the encoding process at the middle stage of the generator. The structure of the decoder is similar to decoder parts of U-net except for the use of the residual blocks [14] to add more convolutional layers. The more details about the generator are available in Supplementary Material.

Discriminator

The discriminators commonly composed of a series of convolution layer and Leaky-ReLU [13]. As shown in Fig. 2, the discriminator has two output headers: one is the classification header for real or fake and the other is classification header for the domain. PatchGAN [17, 35] was utilized to classify whether local image patches are real or fake. The dropout [15, 28] was very effective to prevent the overfitting of the discriminator. Exceptionally, the discriminator of MR contrast translation has branches for multi-scale processing. The details of the specific discriminator architecture is available in Supplementary Material.

4.3. Network Training

All the models were optimized using Adam [19] with a learning rate of 0.00001, $\beta_1 = 0.9$ and $\beta_2 = 0.999$. As mentioned before, the performance of the classfier should be associated only to real labels which means it should be trained only using the real data. Thus, we first trained the classifier on real images with its corresponding labels for the first 10 epochs, and then we trained the generator and the discriminator simultaneously. Training takes about six hours, half a day, and one day for the MR contrast translation task, illumination translation, and facial expression translation task, respectively, using a single NVIDIA GTX 1080 GPU.

For the illumination translation task, YCbCr color coding was used instead of RGB color coding. YCbCr coding consists of the Y-luminance and CbCr-color space. There are five different illumination images. They almost share the CbCr codings and the only difference is Y-luminance

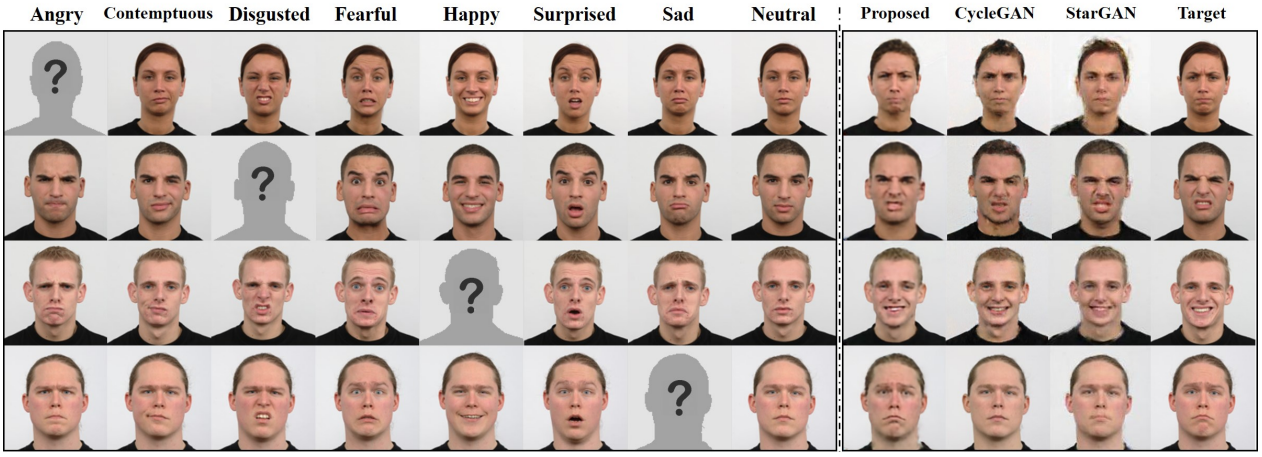

Figure 5: Four facial expression imputation results. The generated images (right) were reconstructed from the inputs with the multiple facial expressions (left). The image of neutral facial expression was used as an input in CycleGAN and StarGAN.

channel. Thus, the only Y-luminance channels were processed for the illumination translation tasks and then the reconstructed images coverted to RGB coded images. We used RGB channels for facial expression translation task, and the MR contrast dataset consists of single-channel images.

5. Experimental Results

For all three image imputation tasks, each datasets were divided into the train, validation and test sets by the subjects. Thus, all our experiments were performed using the unseen images during the training phase. We compared the performance of the proposed method with CycleGAN [35] and StarGAN [4] which are the representative models for image translation tasks.

5.1. Results of MR contrast imputation

First, we trained the models on MR contrast dataset to learn the task of synthesizing the other contrasts. In fact, this was the original motivation of this study that was inspired by the clinical needs. There are four different MR contrasts in the dataset and the generator learns the mapping from one contrast to the other contrast.

Qualitative evaluation As shown in Fig. 3, the proposed method reconstructed the four different MR contrasts, which are very similar to the targets, while StarGAN shows poor results. Since there are so many variables that affect the pixel intensity of MR images, it is necessary to use the pixels from at least three different contrast to accurately estimate the intensity of the other contrast. Thus, there exists a limitation on CycleGAN or StarGAN, since they uses a single input contrast. For example, consider the reconstruction of T2 weighted image from the T2 FLAIR* input in Fig. 3. The cerebrospinal fluid (CSF) in the T2-weighted image should be bright, while in the T2-FLAIR* it should be dark (yellow and green arrows in Fig. 3). When StarGAN tries to generate the T2 weighted image from the T2 FLAIR*, this should be difficult because the input pixels are close to zero. StarGAN somehow reconstructed the CSF pixels near the gray matter (yellow arrow in Fig. 3) with the help of the neighborhood, but the larger CSF area (green arrow in Fig. 3) cannot be reconstructed because the help of neighborhood pixels is limited. The proposed method, however, utilized the combination of the inputs to accurately reconstruct every pixel.

Quantitative evaluation Since the contrast images were aligned with very high accuracy, we can calculate the quantitative indices for each reconstruction. For the quantitative evaluation, a normalized mean squared error (NMSE) and SSIM were calculated between the reconstruction and the target (Table. 1). Compared to the results of StarGAN, the four different contrast MR images were reconstructed with minimum errors using the proposed method.

		T1F	T2w	T2F	T2F*	Mean
CycleGAN	NMSE	0.150	0.378	0.354	0.313	0.299
	SSIM	0.860	0.724	0.720	0.757	0.765
StarGAN	NMSE	0.583	0.472	0.539	0.442	0.509
	SSIM	0.668	0.790	0.790	0.855	0.776
Proposed	NMSE	0.0326	0.109	0.0238	0.110	0.0689
	SSIM	0.918	0.904	0.942	0.740	0.876

Table 1: Quantitative results of MR contrast imputation. The normalized mean squared error (NMSE) and structural similarity index (SSIM) are displayed.

5.2. Results of illumination imputation

We trained CycleGAN, StarGAN and the proposed method using CMU Multi-PIE dataset for the illumination imputation task. Given five different illumination directions, the input domain for CycleGAN and StarGAN was fixed as the frontal illumination (0°).

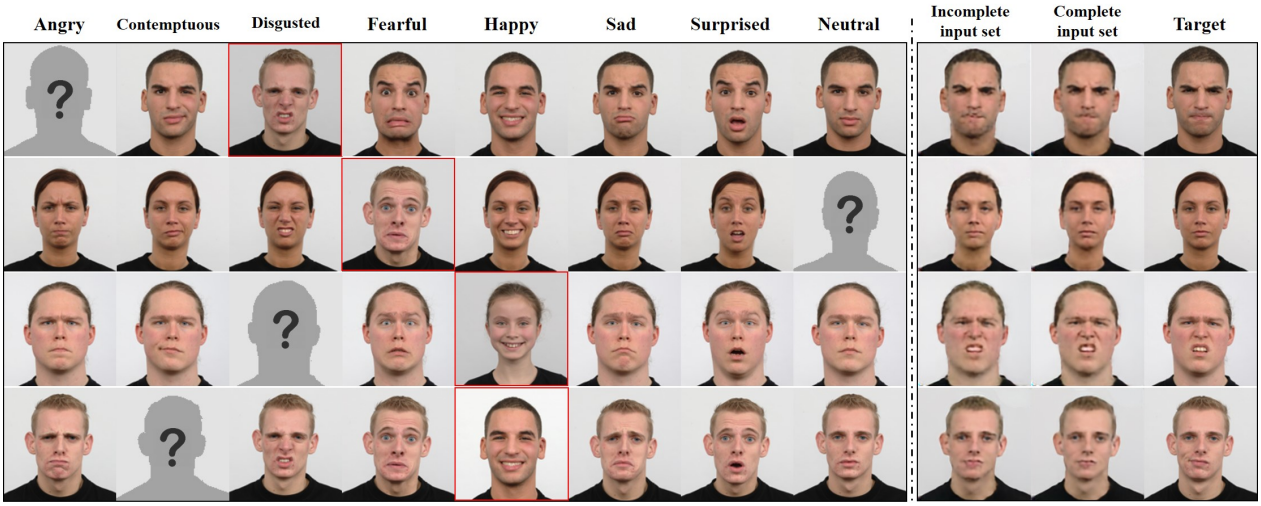

Figure 6: Comparison of incomplete and complete input data set for image imputation results by CollaGAN. For the incomplete input cases, the generated images (right) were reconstructed from the inputs with multiple facial expressions (left) with one substituted facial expression from another person (red box). The image to impute is marked as the question mark.

Qualitative evaluation As shown in Fig. 4, the proposed method clearly generate the natural illuminations while properly maintaining the color, brightness balance, and textures of the facial images. The CycleGAN and StarGAN also generate the four different illumination images from the frontal illumination input. In the result of CycleGAN, however, we can see the emphasis of the red channel and the image looks reddish overall. Also the resulting image looks like a graphic model or a drawing, rather than a photo. The resulting image of StarGAN was only adjusted to the left and right of the illumination smoothly, but did not reflect detailed illumination such as the structure of the face. And unnatural lighting changes were observed on the result of StarGAN.

The proposed method shows the most natural lighting images among the three algorithms. While CycleGAN and StarGAN had simply adjusted the brightness of the left and right sides of the images, the shadow caused by the shape of the nose, the cheek and the jaw is expressed naturally in the proposed method (Fig. 4 yellow arrows).

5.3. Results of facial expression imputation

The eight facial expressions in RaFD were used to train the proposed model for facial expression imputation. The input domain for CycleGAN and StarGAN was defined as a neutral expression among the eight different facial expressions.

Qualitative evaluation Different facial expressions were reconstructed naturally using the proposed method as shown in Fig. 5. Compared with the results of StarGAN, which uses only the single input, the proposed method utilizes as much information as possible from the combinations of facial expressions. As shown in the generated results of CycleGAN and StarGAN (Fig. 5), the generated results of 'sad' were very similar to the generated image of 'neutral' which was the input of them, while the proposed method expressed the 'sad' very well. With a help of multiple cycle consistency, the proposed method clearly generates the natural facial expressions while preserving the identity correctly.

5.4. Effect of incomplete input set

In order to investigate the robustness of the proposed method, we demonstrated CollaGAN results from incomplete input set. If there are two missing facial expressions (eg. 'happy' and 'neutral') and one is interested in reconstruct the missing image (eg. 'happy'), one can substitute one image (eg.'neutral') from the other subject as one of the input for the CollaGAN. As shown in Fig. 6, the generated image from incomplete input set with the substitute data from others shows similar results compared to the complete input set. CollaGAN utilized the other subject's facial information (eg. 'neutral') to impute the missing facial expression (eg. 'happy').

6. Conclusion

In this paper, we presented a novel CollaGAN architecture for missing image data imputation by synergistically combining the information from the available data with the help of a single generator and discriminator. We showed that the proposed method produces images of higher visual quality compared to the existing methods. Therefore, we believe that CollaGAN is a promising algorithm for missing image data imputation in many real world applications.

References

[1] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein GAN. *arXiv preprint arXiv:1701.07875*, 2017.

[2] A. N. Baraldi and C. K. Enders. An introduction to modern missing data analyses. *Journal of school psychology*, 48(1):5–37, 2010.

[3] T. Chen, M.-M. Cheng, P. Tan, A. Shamir, and S.-M. Hu. Sketch2photo: Internet image montage. In *ACM Transactions on Graphics (TOG)*, volume 28(5), page 124. ACM, 2009.

[4] Y. Choi, M. Choi, M. Kim, J.-W. Ha, S. Kim, and J. Choo. StarGAN: Unified generative adversarial networks for multi-domain image-to-image translation. *arXiv preprint*, 1711, 2017.

[5] C. B. Choy, D. Xu, J. Gwak, K. Chen, and S. Savarese. 3D-R2N2: A unified approach for single and multi-view 3D object reconstruction. In *European conference on computer vision*, pages 628–644. Springer, 2016.

[6] A. Drevelegas and N. Papanikolaou. Imaging modalities in brain tumors. In *Imaging of Brain Tumors with Histological Correlations*, pages 13–33. Springer, 2011.

[7] A. A. Efros and W. T. Freeman. Image quilting for texture synthesis and transfer. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 341–346. ACM, 2001.

[8] D. Eigen and R. Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2650–2658, 2015.

[9] C. K. Enders. *Applied missing data analysis*. Guilford press, 2010.

[10] R. Fergus, B. Singh, A. Hertzmann, S. T. Roweis, and W. T. Freeman. Removing camera shake from a single photograph. In *ACM transactions on graphics (TOG)*, volume 25(3), pages 787–794. ACM, 2006.

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

[12] R. Gross, I. Matthews, J. Cohn, T. Kanade, and S. Baker. Multi-PIE. *Image and Vision Computing*, 28(5):807–813, 2010.

[13] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

[14] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[15] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

[16] X. Huang, Y. Li, O. Poursaeed, J. E. Hopcroft, and S. J. Belongie. Stacked generative adversarial networks. In *CVPR*, volume 2, page 3, 2017.

[17] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. *arXiv preprint*, 2017.

[18] T. Kim, M. Cha, H. Kim, J. K. Lee, and J. Kim. Learning to discover cross-domain relations with generative adversarial networks. *arXiv preprint arXiv:1703.05192*, 2017.

[19] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[20] O. Langner, R. Dotsch, G. Bijlstra, D. H. Wigboldus, S. T. Hawk, and A. Van Knippenberg. Presentation and validation of the radboud faces database. *Cognition and emotion*, 24(8):1377–1388, 2010.

[21] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. P. Aitken, A. Tejani, J. Totz, Z. Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *CVPR*, volume 2(3), page 4, 2017.

[22] R. J. Little and D. B. Rubin. *Statistical analysis with missing data*, volume 333. John Wiley & Sons, 2014.

[23] M.-Y. Liu, T. Breuel, and J. Kautz. Unsupervised image-to-image translation networks. In *Advances in Neural Information Processing Systems*, pages 700–708, 2017.

[24] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. P. Smolley. Least squares generative adversarial networks. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, pages 2813–2821. IEEE, 2017.

[25] M. Mathieu, C. Couprie, and Y. LeCun. Deep multi-scale video prediction beyond mean square error. *arXiv preprint arXiv:1511.05440*, 2015.

[26] M. Mirza and S. Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[27] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[28] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[29] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[30] L. N. Tanenbaum, A. J. Tsiouris, A. N. Johnson, T. P. Naidich, M. C. DeLano, E. R. Melhem, P. Quarterman, S. Parameswaran, A. Shankaranarayanan, M. Goyen, et al. Synthetic MRI for clinical neuroimaging: Results of the Magnetic Resonance Image Compilation (MAGiC) prospective, multicenter, multireader trial. *American Journal of Neuroradiology*, 2017.

[31] D. Ulyanov, A. Vedaldi, and V. Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016.

[32] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to

structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

[33] J. Yoon, J. Jordon, and M. van der Schaar. RadialGAN: Leveraging multiple datasets to improve target-specific predictive models using generative adversarial networks. *arXiv preprint arXiv:1802.06403*, 2018.

[34] H. Zhao, O. Gallo, I. Frosio, and J. Kautz. Loss functions for image restoration with neural networks. *IEEE Transactions on Computational Imaging*, 3(1):47–57, 2017.

[35] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *arXiv preprint*, 2017.

[36] S. Remedios, D. L. Pham, J. A. Butman, and S. Roy. Classifying magnetic resonance image modalities with convolutional neural networks. In *Medical Imaging 2018: Computer-Aided Diagnosis*, volume 10575, 2018.

./
./

A. Collaborative training

This section describes the experiments that further analyze the importance of multi-inputs by providing additional qualitative results.

A.1. Effects of input dropout

The input of the proposed method is much more informative than StarGAN [4]. In other words, there might exist some wasted inputs since there is some redundancy of the inputs. For example on RaFD [20], if 'Happy' image plays a major role for reconstructing 'Angry' images, the other facial expressions may contribute little on the output, which is not collaborative. To achieve the collaborative learning, it is imporatant to use random nulling on the inputs (control of the number of missing imputs). Thus, the random nulling of the input images helps to increase the contribution of the other facial expressions evenly. It could be treated as a dropout [28] layer on the input images. The contribution of the input dropout is as shown in Fig. 7. The input dropout increases the performance of the reconstruction quality for all 'Missing N' since the inputs contribute more evenly to the reconstruction.

A.2. Incompelete input datasets

To investigate the effects of the number of inputs, we compared the reconstruction results with the control of the missing number of inputs. Figure. 8 shows the reconstruction results 'Happy' and 'Angry' using the inputs with different missing values from seven to one. As the amount of input information increased, the reconstruction results improved qualitatively as shown in Fig. 8.

B. Implementation Details

B.1. Details of MR acquisition parameter

Among the four different contrasts, three of them were synthetically generated from the MAGiC sequence (T1F, T2w and T2F) and the other was additionally scanned by conventional T2-FLAIR sequence (T2F*). The MR acquisition parameters are shown in Table. 2.

	TR(ms)	TE(ms)	TI(ms)	FA(deg)
T1F	2500	10	1050	90
T2w	3500	128	-	90
T2F	9000	95	2408	90
T2F*	9000	93	2471	160
Common parameters	FOV:220×220mm, 320×224 matrix, 4.0 mm thickness			

Table 2: MR acquisition parameters for each contrast. T1F, T2w, T2F and T2F* represent MAGiC synthetic T1-FLAIR, MAGiC synthetic T2-weighted, MAGiC synthetic T2-FLAIR and conventional T2-FLAIR, respectively. Four contrasts share the field of view (FOV), acquisition matrix, and slice thickness as shown in the common parameters row.

B.2. Network Implementation

The proposed method consists of two networks, the generator and the discriminator. There are three tasks (MR contrasts imputation, illumination imputation and facial expression imputation) and each task has its own property. Therefore, we redesigned the generators and discriminator for each tasks to achieve the best performance for each task while the general network architecture are similar.

MR contrast translation Instead of using single convolution, the generator uses two convolution branches with 1x1 and 3x3 filters to handle the multi-scale feature information. The two branches of the convolutions are concatenated similar to the inception network [29]. We called this series of two convolution, concatenation, instance normalization [31] and leaky-ReLU [13], CCNR unit, as shown in Table. 3. These CCNR units help the pixel-by-pixel processing of the CNN as well as the processing with a large FOV. The architecture of the generator describes in Table. 3 and Fig. 9.

To classify the MR contrast, multi-scale (multi-resolution) processing is important. The discriminator has three branches that each has different scales as shown in Table. 4. A branch handles the feature on the original resolution. Another branch process the features on the quater-resolution scales ($height/4, width/4$). The other one sequentially reduces the scales for extract features. Three branches are concatenated to process multi-scale features.

Figure 7: Facial expression imputation results changing the missing number of input images from seven to one. 'Sad' (up) and 'Contemptuous' (down) facial expressions were reconstructed using various number of inputs. Each 1st row was the results trained by input dropout and the other was not. Each column represents the results from the incomplete input set which has 'Missing N' inputs. To impute each facial expression, other (8-N) facial expressions were collaboratively used as inputs.

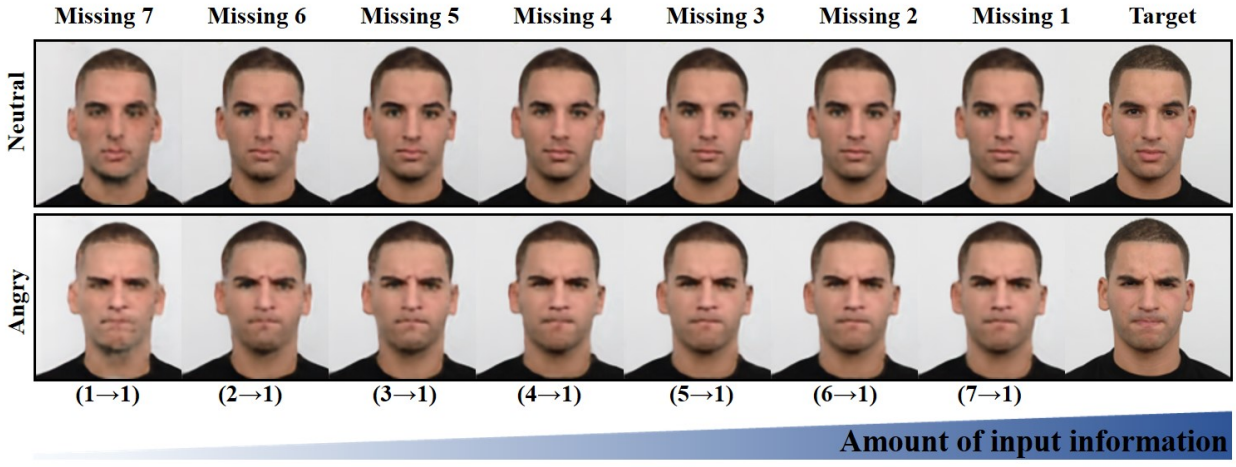

Figure 8: Facial expression imputation results changing the missing number of input images from seven to one. 'Neutral' (1st row) and 'Angry' (2nd row) facial expressions were reconstructed using various number of inputs. Each column represents the results from the incomplete input set which has 'Missing N' inputs. To impute each facial expression, other (8-N) facial expressions were collaboratively used as inputs. More information was used to the right column and the quality of the reconstruction improved. The numbers below the images, ($N_{in} \rightarrow N_{out}$), explain the number of input images and output images, respectively.

Unit	Layers						nCh
Main		CCNL×2	(skip) / (Blck#1)	Cat	CCNL×2	C'	16
Blck#1	P	CCNL×2	(skip) / (Blck#2)	Cat	CCNL×2	T	32
Blck#2	P	CCNL×2	(skip) / (Blck#3)	Cat	CCNL×2	T	64
Blck#3	P	CCNL×2	(skip) / (Blck#4)	Cat	CCNL×2	T	128
Blck#4	P	CCNL×2				T	256
CCNL		Conv(k1,s1) / Conv(k3,s1)	Cat-InstanceNorm-LeakyReLU				

Table 3: Architecture of the generator used for MR contrast translation. The U-net [27] structure was redesigned with the proposed CCNR units which includes instance normalization (N) and leaky-ReLU (L). Conv, P, Cat and T represent convolution, average pooling with strides 2, concatenate, and convolution transpose with strides 2 and kernel size 2×2, respectively. While k and s refer to the kernel size and the stride, C' is 1×1 convolution layer, Conv(k1,s1).

Similar architecture with this kind of multi-scale approach works well to classify the MR contrast [36].

Order	Layers					k
1a	C(n4,s1)-L	C(n4,s1)-L	C(n4,s1)-L	C(n4,s1)-L	C(n16,s4)-L	4
1b	C(n4,s1)-L	C(n8,s2)-L	C(n8,s1)-L	C(n16,s2)-L	C(n16,s1)-L	4
1c	C(n16,s4)-L	C(n16,s1)-L	C(n16,s1)-L	C(n16,s1)-L	C(n16,s1)-L	4
2	1a / 1b / 1c	Cat	C(n32,s2)-L	C(n64,s2)-L	C(n128,s2)-L	4
3a	C(n1,s1)	Sigmoid (D_{gan})				3
3b	FC(n4)	Softmax (D_{cls})				8

Table 4: Architecture of the descriminator used for MR contrast translation. k is the kernel size for the convolution and C(n,s) represents the convolution layer with n channels and s strides. Cat, L and FC represent the concatenate layer, the leaky-ReLU layer and the fully-connected layer, respectively.

Illumination translation Architecture of the generator used for illumination translation. It is similar to original U-net structure with instance normalization (N) and leaky-ReLU (L) instead of batch normalization and ReLU, respectively, as shown in Table. 5 and Fig. 10.

The discriminator is consists of convolutions with strides 2 and instance normalization. At the end of the discriminator, there are two branch [4]: one for discriminating real/fake and the other for the domain classification. Here, patchGAN [17, 35] was utilized to classify the source (real/fake).

Facial expression translation For the generator of facial expression translation, we designed a multi-branched U-net which has individual encoder for each input images (Fig. 11). The default architecture is based on U-net structure. The generator consists of two part: encoder and decoder. In the encoding step, each image are encoded separately by eight branches. Here, the mask vector is concatenated to every input images to extract the feature for the target domain. Then, the encoded features are concatenated in the decoder and the decoder shares the structure of the modified U-net as explained in Table. 5. The discriminator shares the architecture with the one used for the illumination translation task (Table. 6) except fot the last fully-connected layer has eight channels for eigth facial expression classification.

Unit	Layers						nCh
Main	-	CNL×2	(skip) / (Blck#1)	Cat	CNL×2	C'	64
Blck#1	P	CNL×2	(skip) / (Blck#2)	Cat	CNL×2	T	128
Blck#2	P	CNL×2	(skip) / (Blck#3)	Cat	CNL×2	T	256
Blck#3	P	CNL×2	(skip) / (Blck#4)	Cat	CNL×2	T	512
Blck#4	P	CNL×2				T	1024
CNL		Conv(k3,s1)	Cat-InstanceNorm-LeakyReLU				

Table 5: Architecture of the generator used for illumination translation. Conv, P, Cat and T represent convolution, average pooling with strides 2, concatenate, and convolution transpose with strides 2 and kernel size 2×2, respectively. k and s refer to the kernel size and the stride. C' is 1×1 convolution layer, Conv(k1,s1).

Order	Layers
1	C(n64,k4,s2)-L
2	C(n128,k4,s2)-L
3	C(n256,k4,s2)-L
4	C(n512,k4,s2)-L
5	C(n1024,k4,s2)-L
6	C(n2048,k4,s2)-L
7a	C(n1,k3,s1)-Sigmoid (D_{gan})
7b	FC(n5)-Softmax (D_{cls})

Table 6: Architecture of the generator used for facial expression translation.

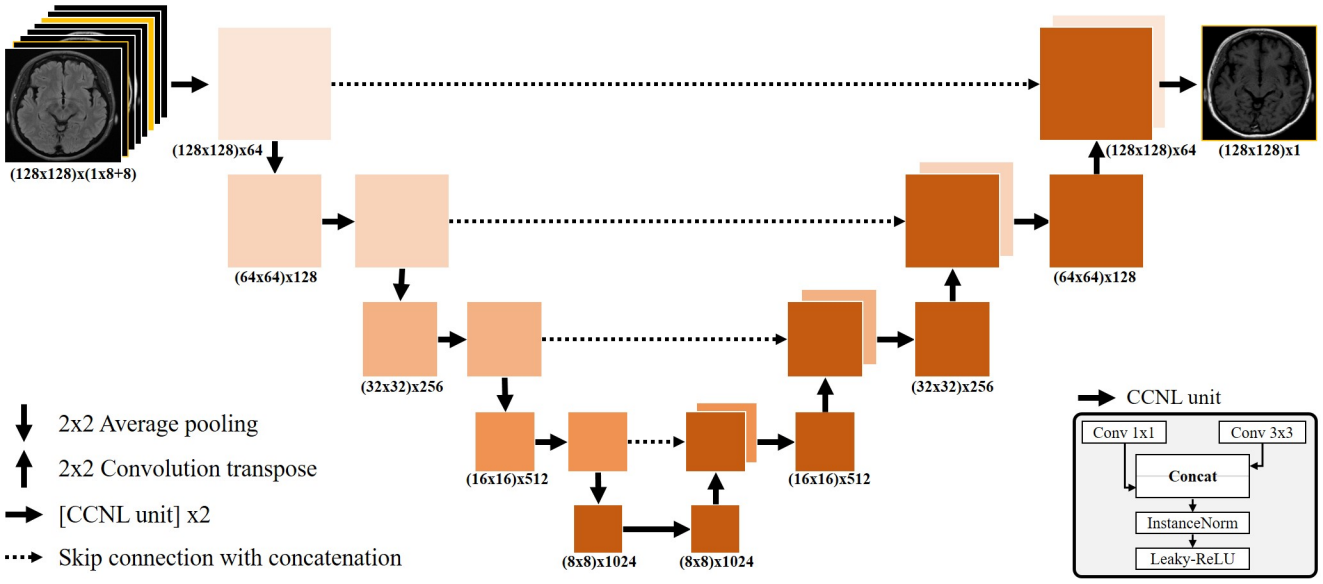

Figure 9: Architecture of the generator used for MR contrast imputation. It is the modified U-net architecture with CCBR unit which consists of two branches of convolution (3×3 and 1×1), concatenation, instance normalizatoin and leaky-ReLU. The input images were concatenated with mask vector which represents the target domain. The downward arrows, upward arrows, right arrows and dashed arrows represent 2×2 average pooling, 2×2 convolution transpose, two repetition of CCNL unit and skip connection with concatenation, respectively, as explained in Table. 3

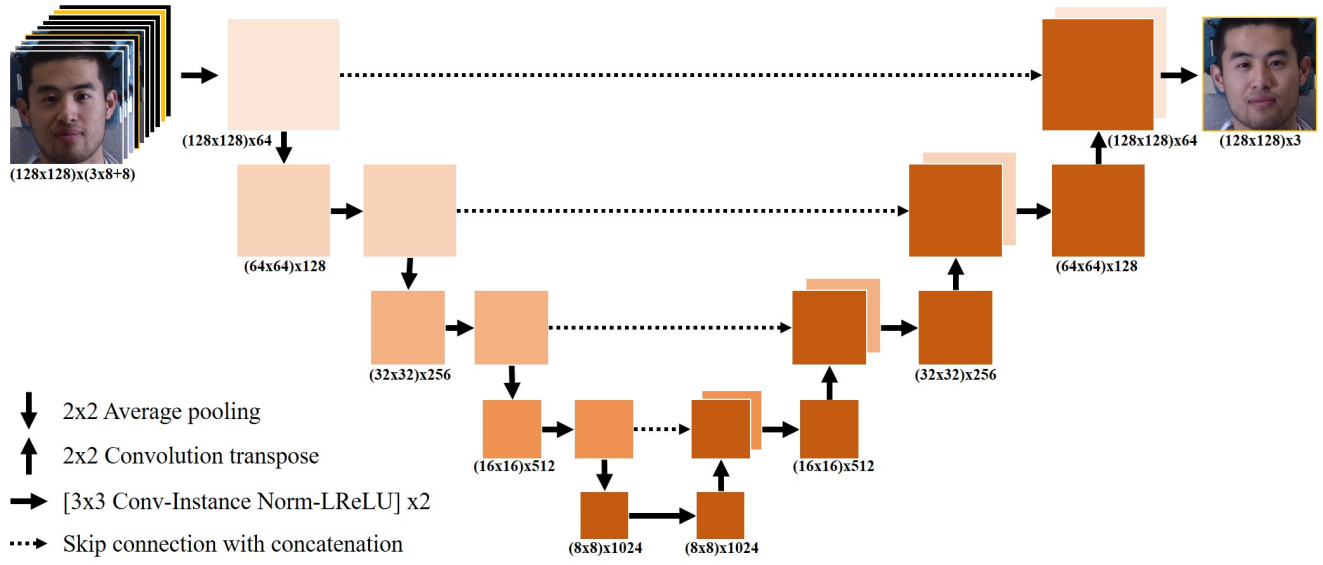

Figure 10: Architecture of the generator used for illumination imputation. U-net structure with instance normalization and leaky-ReLU was used. The input images were concatenated with mask vector which represents the target domain. The downward arrows, upward arrows, right arrows and dashed arrows represent 2×2 average pooling, 2×2 convolution transpose, two repetition of [3x3 convolution, instance normalization and leaky-ReLU], and skip connection with concatenation, respectively, as explained in Table. 5

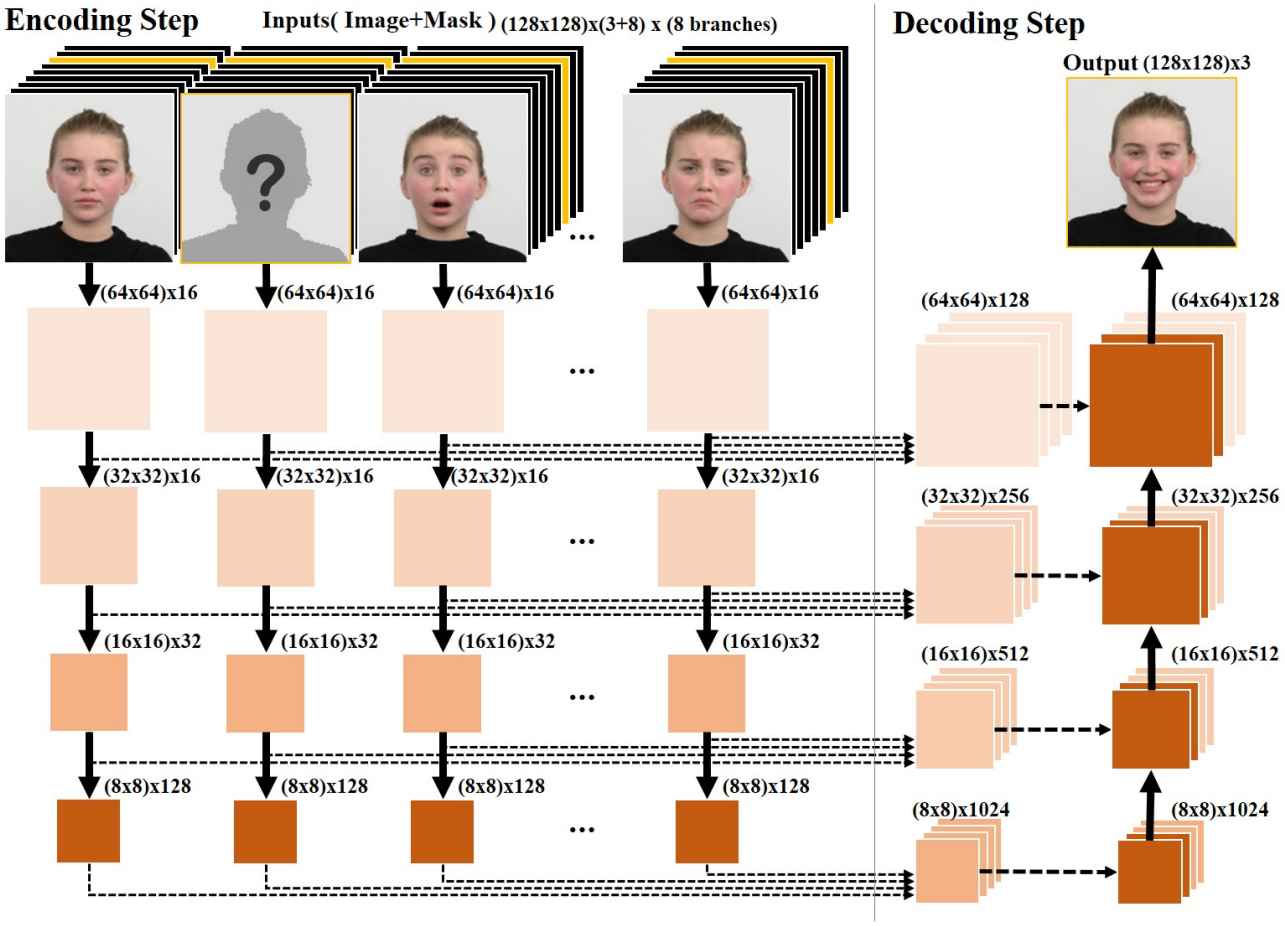

Figure 11: Architecture of the generator used for facial expression translation. It has multi-branched encoder for individual feature extraction of each input images. The ./encoded features are concatenated in the decoder and the decoder structure shares with the discriminator used for the illumination translation. $(h \times w) \times N_{ch}$ represents the dimension of the features/images where h, w and N_{ch} is height, width and number of channels. The dashed arrow means skip connections. The downward, upward and right arrows represent [CNL$\times$2-P] layers, [T-CNL$\times$2] layers and [CNL$\times$2] layers, respectively, as explained in Table. 5

B.3. Additional Qualitative Results

Figure 12: Additional results for facial expression imputation. To impute each facial expression, the other seven facial expressions were collaboratively used as inputs.

./

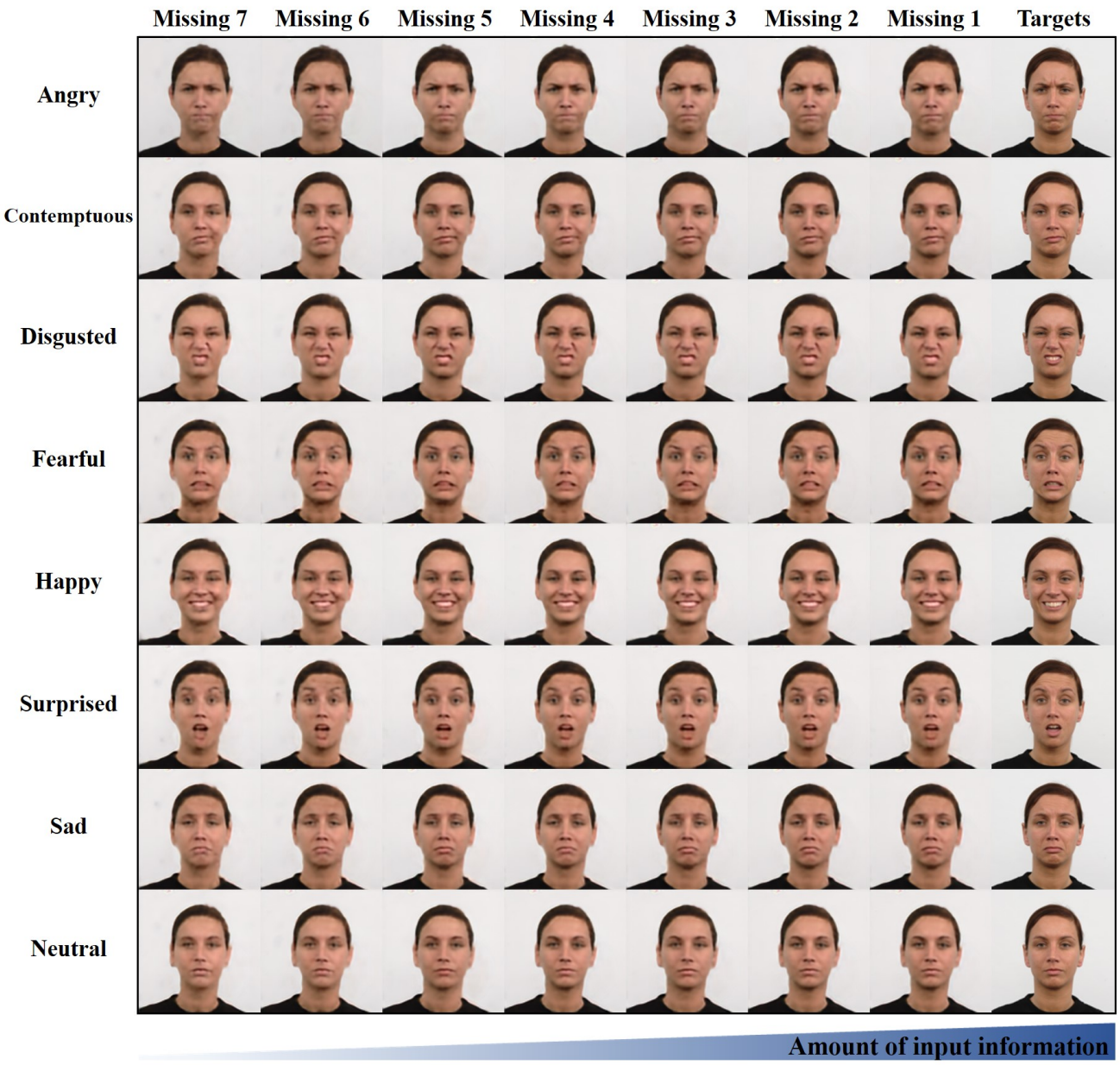

Figure 13: Additional results for facial expression imputation from incomplete input sets. Each column represents the results from the incomplete input set which has 'Missing N' inputs. To impute each facial expression, other (8-N) facial expressions were collaboratively used as inputs. More information was used to the right column.