Multi-Domain Image Completion for Random Missing Input Data (Supplementary)

APPENDIX I
IMPLEMENTATION DETAILS

A. Hyperparameters

In our algorithm, we use the Adam optimizer with $\beta_1 = 0.5, \beta_2 = 0.999$. The learning rate is 0.0001. We set the loss weights in the total loss (Equation 7 in main text) as $\lambda_{adv} = 1, \lambda^x_{cyc} = 10, \lambda^c_{cyc} = 1, \lambda^s_{cyc} = 1, \lambda_{rec} = 20$, and $\lambda_{seg} = 1$ in the unified model for image completion and segmentation. For comparison purpose, we train the model with batch size 1 and 100,000 iterations for image generation task, and compare the results across MUNIT, StarGAN, CollaGAN, and ours ReMIC in all the three datasets In ReMIC, we set the dimension of the style code as 8 for comparison purpose with MUNIT. For image generation during testing, we use a fixed style code of 0.5 in each dimension for both MUNIT and ReMIC to compute quantitative results.

B. Network Architectures

The network structure of ReMIC is developed on the backbone of MUNIT model. We describe the details of each module here.

1) Unified Content Encoder: : consists of a down-sampling module and residual blocks to extract contextual knowledge from all available domain images in inputs. The down-sampling module contains a 7×7 convolutional block with stride 1 and 64 filters, and two 4×4 convolutional blocks with stride 2 and, 128 and 256 filters respectively. The convolutional layers downsample the input to features maps of size $W/4 \times H/4 \times 256$, where W and H are the width and height of input image. Next, there are four residual blocks, each of which contains two 3×3 convolutional blocks with 256 filters and stride 1. We apply Instance Normalization (IN) after all the convolutional layers. Note that the proposed unified content encoder accepts images of all domains as input (missing domains are filled up with zeros padding in the initialization), and learns a universe content code complementarily and collaboratively, which are different from MUNIT.

2) Style Encoder: : contains a similar down-sampling module and several residual blocks, which is followed by a global average pooling layer and a fully connected layer to learn the verteorized style code. The down-sampling module is developped using the same structure as that in the unified content encoder above, and two more 4×4 convolutional blocks with stride 2 and 256 filters are followed. The final fully connected layer generates style code as a 8-dim vector. There is no IN applied to the style encoders to keep the original feature means and variances with style information.

3) Generator: : includes four residual blocks, each of which contains two 3×3 convolutional blocks with 256 filters and stride 1. Two nearest-neighbor upsampling layers and a 5×5 convolutional block with stride 1 and, 128 and 64 filters respectively are followed to up-sample content codes back to the original image size. Finally, there is a a 7×7 convolutional block with stride 1 to output the reconstructed image. In order to incorporate the style code in the generation process, the Adaptive Instance Normalization (AdaIN) is applied to each residual block as follows

$$AdaIN(z, \gamma, \beta) = \gamma\Big(\frac{z - \mu(z)}{\sigma(z)}\Big) + \beta \tag{1}$$

where z is the activation from the last convolutional layer. $\mu(z)$ and $\sigma(z)$ are the channel-wise mean and standard deviations of the activation. γ and β are the affine parameters in the AdaIN layers that are generated from style codes via a multi-layer perceptron (MLP). In this way, the input style code controls the generated style information through the affine transformation in the AdaIN layers in all generators.

4) Discriminator: : includes four 4×4 convolutional blocks with stride 2 and, 64, 128, 256, and 512 filters in sequence. The Leaky ReLU activation with slope 0.2 is applied after convolutional layers. A multi-scale discriminator is used to include the results at three different scales together. In adversarial training, we adopt LSGAN objective as the adversarial loss to learn to generate realistic images.

5) Segmentor: : We adopt a segmentation net with a U-Net shape . In order to build a joint model with the image generation modules, we build a variant of U-Net, that is, the downsampling part shares the same structure as the content encoder aforementioned while the upsampling part has the same layers as the generator as described above. Similar to the original U-Net , we also adopt the skip connections between the downsampling and upsampling layers in our segmentation module.

APPENDIX II
EXTENDED RESULTS AND ABLATIVE STUDY FOR MULTI-DOMAIN IMAGE COMPLETION

A. Multi-sample learning

Based on the proposed model as shown in Fig. 3 in the main text, we further propose a training strategy when multiple samples are inputted at one time to facilitate learning disentangled representations. Specifically, based on the assumption of partially shared latent space, we assume that the factorized latent code can represent the corresponding content and style

TABLE I
EXTENDED RESULTS OF MULTI-DOMAIN IMAGE COMPLETION FOR BRATS DATASET

Methods	T1	T1Gd
	MAE(↓) / NRMSE(↓) / PSNR(↑) / SSIM(↑)	MAE(↓) / NRMSE(↓) / PSNR(↑) / SSIM(↑)
ReMIC	0.0187 / 0.2008 / 28.5508 / 0.9618	0.0153 / 0.2375 / 29.1628 / 0.9521
ReMIC+Multi-Sample	**0.0180** / **0.1942** / **28.8354** / **0.9634**	**0.0127** / **0.2070** / **30.2444** / **0.9555**
ReMIC+Seg	0.0195 / 0.2033 / 28.5679 / 0.9597	0.0142 / 0.2285 / 29.2134 / 0.9468
ReMIC+Joint	0.0214 / 0.2128 / 27.9944 / 0.9568	0.0140 / 0.2251 / 29.3624 / 0.9484
Methods	T2	FLAIR
	MAE / NRMSE / PSNR / SSIM	MAE / NRMSE / PSNR / SSIM
ReMIC	0.0190 / **0.2481** / 27.4829 /0.9457	0.0198 / 0.2469 / 27.1540 / 0.9367
ReMIC+Multi-Sample	0.0195 / 0.2493 / **27.5168** / **0.9463**	**0.0192** / **0.2456** / **27.3598** / **0.9385**
ReMIC+Seg	**0.0193** / 0.2525 / 27.2864 / 0.9431	0.0206 / 0.2553 / 26.9191 / 0.9333
ReMIC+Joint	0.0197 / 0.2596 / 26.9954 / 0.9429	0.0220 / 0.2651 / 26.5068 / 0.9302

TABLE II
EXTENDED RESULTS OF MULTI-DOMAIN IMAGE COMPLETION FOR PROSTATEX DATASET

Methods	T2	ADC
	MAE(↓) / NRMSE(↓) / PSNR(↑) / SSIM(↑)	MAE(↓) / NRMSE(↓) / PSNR(↑) / SSIM(↑)
ReMIC	0.0840 / 0.4908 / 18.6200 / 0.5427	0.0253 / 0.2179 / 26.6150 / 0.9232
ReMIC+Multi-Sample	0.0810 / **0.4742** / 18.8986 / **0.5493**	**0.0250** / **0.2171** / **26.7024** / **0.9263**
ReMIC+Seg	**0.0871** / 0.5024 / 18.4236 / 0.5336	0.0272 / 0.2322 / 26.0828 / 0.9107
ReMIC+Joint	0.0881 / 0.5071 / 18.3206 / 0.5353	0.0288 / 0.2403 / 25.8024 / 0.9064
Methods	HighB	
	MAE(↓) / NRMSE(↓) / PSNR(↑) / SSIM(↑)	
ReMIC	**0.0254** / **0.3894** / 24.7927 / **0.9150**	
ReMIC+Multi-Sample	0.0268 / 0.3945 / **24.8066** / 0.9116	
ReMIC+Seg	0.0272 / 0.4110 / 24.3277 / 0.9061	
ReMIC+Joint	0.0286 / 0.4359 / 23.8270 / 0.9006	

information in the input image. Therefore, by exchanging the style codes from two independent samples in all available domains, it should be able to reconstruct the original input images by recombining the original content and the new style codes from the other sample. Based on this idea, we build a comprehensive model with cross-sample training between two samples. Similarly as the framework in Fig. 3 in main text, the image and latent consistency loss and image reconstruction loss are also constrained through the encoding and decoding procedure. The results of multi-sample learning are shown in Table I and Table II denoted as "ReMIC+Multi-Sample".

B. Multi-task learning

For the jointly trained model of image completion and segmentation, the generated images are also evaluated using the same metrics as shown in Table I and Table II. Similarly to Table 3 in the main text, "ReMIC+Seg" stands for using separate content encoders for image generation and segmentation tasks in our proposed unified framework, while "ReMIC+Joint" indicates sharing the weights of content encoder for both two tasks. The results indicate that adding segmentation branch does not bring an obvious benefit for image generation. This is because the segmentation sub-module mainly focuses on the tumor region which takes up only a small part among the whole slice image. Besides, we use dice loss as the segmentation training objective which might not be consistent with the metrics used to evaluate generated image quality, which mainly emphasize the whole-slice pixel-level similarity.

TABLE III
MISSING-DOMAIN SEGMENTATION WITH INFERENCE ON PRE-TRAINED SEGMENTATION MODEL (DICE SCORES ARE REPORTED)

Methods	BraTS				ProstateX		
	T1	T1Gd	T2	FLAIR	T2	ADC	HighB
Oracle	0.822				0.908		
Zero	0.651	0.473	0.707	0.454	0.528	0.243	0.775
Average	0.763	0.596	0.756	0.671	0.221	0.692	0.685
NN	0.769	0.540	0.724	0.606	0.759	0.850	0.854
MUNIT	0.783	0.537	0.782	0.492	0.783	0.708	0.858
StarGAN	0.799	0.553	0.746	0.613	0.632	0.653	0.832
CollaGAN	0.753	0.564	0.798	0.674	0.472	0.760	0.842
ReMIC	**0.819**	**0.641**	**0.823**	**0.784**	**0.863**	**0.907**	**0.903**

TABLE IV
MISSING-DOMAIN SEGMENTATION WITH RE-TRAINING SEGMENTATION MODEL (DICE SCORES ARE REPORTED)

Methods	BraTS				ProstateX		
	T1	T1Gd	T2	FLAIR	T2	ADC	HighB
Oracle	0.822				0.908		
Zero	0.811	0.656	0.823	0.775	0.868	0.899	0.897
Average	0.796	0.604	0.788	0.759	0.856	0.885	0.897
ReMIC	0.789	0.655	0.805	0.765	0.871	0.898	0.891
ReMIC+Seg	0.806	0.674	0.822	0.771	**0.872**	**0.909**	**0.905**
ReMIC+Joint	**0.828**	**0.693**	**0.828**	**0.791**	0.867	0.904	0.904

APPENDIX III
EXTENDED RESULTS AND ABLATIVE STUDY FOR MISSING-DOMAIN SEGMENTATION

A. Missing-domain segmentation with inference on pre-trained segmentation model

Suppose we have trained an oracle segmentation model on a complete dataset with all domain images. Then this pre-trained model would be used to predict segmentation results for new

TABLE V
MISSING-DOMAIN SEGMENTATION WITH INFERENCE ON PRE-TRAINED 2D AND 3D SEGMENTATION MODEL (PER-CLASS DICE SCORES ARE REPORTED)

Methods		T1	T1Gd	T2	FLAIR
		WT / TC / ET	WT / TC / ET	WT / TC / ET	WT / TC / ET
2D	Oracle	0.910 / 0.849 / 0.708			
	Zero	0.771 / 0.609 / 0.572	0.872 / 0.539 / 0.008	0.755 / 0.690 / 0.677	0.458 / 0.468 / 0.435
	Average	0.870 / 0.744 / 0.674	0.882 / 0.603 / 0.303	0.849 / 0.732 / 0.686	0.655 / 0.710 / 0.648
	NN	0.883 / 0.765 / 0.660	0.871 / 0.564 / 0.186	0.811 / 0.720 / 0.642	0.534 / 0.669 / 0.614
	MUNIT	0.886 / 0.785 / 0.679	0.872 / 0.552 / 0.187	0.882 / 0.781 / 0.682	0.408 / 0.541 / 0.527
	StarGAN	0.897 / 0.795 / 0.704	0.886 / 0.588 / 0.184	0.851 / 0.725 / 0.661	0.570 / 0.664 / 0.604
	CollaGAN	0.860 / 0.747 / 0.651	0.864 / 0.576 / 0.252	0.882 / 0.811 / 0.700	0.663 / 0.697 / 0.663
	ReMIC	**0.909** / **0.834** / **0.714**	**0.899** / **0.669** / **0.354**	**0.905** / **0.855** / **0.709**	**0.853** / **0.807** / **0.691**
3D	Oracle	0.909 / 0.867 / 0.733			
	Zero	0.876 / 0.826 / 0.694	0.884 / 0.574 / 0.020	0.901 / 0.865 / 0.728	0.661 / 0.730 / 0.643
	Average	0.880 / 0.814 / 0.640	0.854 / **0.618** / **0.282**	0.838 / 0.801 / 0.695	0.713 / 0.732 / 0.675
	NN	0.890 / 0.829 / 0.703	0.859 / 0.538 / 0.081	0.790 / 0.799 / 0.704	0.472 / 0.686 / 0.607
	ReMIC	**0.905** / **0.864** / **0.722**	**0.888** / 0.614 / 0.273	**0.902** / **0.871** / **0.734**	**0.855** / **0.850** / **0.724**

samples during the inference. For new subjects, some domains might be missing. Straightforward solutions to complete the missing domains include zero filling, average image computed from the existing domains, and the nearest neighbor (NN) searching among available training samples. We show the dice scores for these baseline methods in Table III. Oracle results give the average testing dice score when all the domains are available in the inference. Each column shows the dice scores of segmentation predictions when the current domain is missing during inference. Moreover, based on image translation methods, we can generate fake images for missing domain imputation, and the results for different methods are shown in Table III. We show that our proposed method achieves the best dice score compared with all aforementioned baselines and other GAN-based image translation methods. This also indicates that our method could generate better images by preserving a better content representation. Furthermore, from the results in Table III, we know that the T1Gd modality and the T2 modality are the most significant contrasts in the segmentation of BraTS and ProstateX data, missing of which will cause a severe performance decrease in dice score. Our method could alleviate such a loss to a large extent. Here, the dice score for BraTS is the average for the three segmentation categories: enhancing tumor (ET), tumor core (TC), and whole tumor (WT). Please see Table V for a full table with per-class dice scores.

B. Missing-domain segmentation with re-training segmentation model

Suppose we would like to train a segmentation model for a new data set, but most patients in this cohort just contain a random subset of all required domains. In this scenario, it is definitely not efficient to just use the most common domain overlapped by most patients. One simple solution is to complete all the missing images in training set by some imputation method, such as zero-filling image, average image, or generating images via image translation model. The results for these methods are shown in Table IV. More advanced, based on the content code learned in our model, we could develop a join model for multi-task learning of both generation and segmentation. By optimizing the generation loss and segmentation loss simultaneously, the unified model could learn how to generate missing images to promote segmentation performance. The results of jointly learned model as shown in Table IV achieve the best dice score in both BraTS and ProstateX datasets. "ReMIC+Seg" stands for using separate content encoders for generation and segmentation tasks, while "ReMIC+Joint" indicates sharing the weights of content encoder for the two tasks. We note that the baseline methods get better results after retraining the model on the missing data, since the model is trained to fit to the exact missing inputs format by optimizing the segmentation objective under the supervision of segmentation labels, which makes it more robust to missing inputs. However, our method can still get the best results through adaptive learning model.

C. 3D image segmentation with missing domains

Furthermore, we validate that our method could not only work for 2D image segmentation but also 3D image segmentation. When a 3D volumetric image is missing in some domain, we deploy our method to generate 2D images per slice and stack them to build the whole 3D volumetric image in the corresponding missing domain. As shown in Table V, we evaluate the per-class dice score for missing-domain imputation with the oracle model trained from complete-domain 3D image segmentation. The results show our method could give a better performance in most domains. During experiments, we find that the smoothness among different slices in the 3D image generation might be an issue that needs to be further improved. Besides, we also show that the per-class dice scores for BraTS segmentation results in Table V. Compared with WT and TC classes, ET class is definitely more challenging in the brain tumor segmentation, since enhancing tumor usually just covers a very small region among the whole tumor. Particularly in the ET class segmentation, we can see our method outperforms the other methods to a large extent.