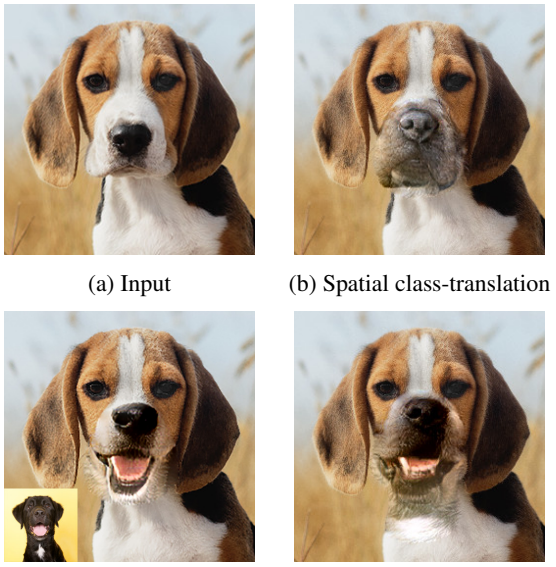
arXiv:1811.10153v1 [cs.CV] 26 Nov 2018

Collaging on Internal Representations: An Intuitive Approach for Semantic Transfiguration

Ryohei Suzuki[1,2*] Masanori Koyama[2] Takeru Miyato[2] Taizan Yonetsuji[2]
[1]The University of Tokyo [2]Preferred Networks, Inc.

Abstract

We present a novel CNN-based image editing method that allows the user to change the semantic information of an image over a user-specified region. Our method makes this possible by combining the idea of manifold projection with spatial conditional batch normalization (sCBN), a version of conditional batch normalization with user-specifiable spatial weight maps. With sCBN and manifold projection, our method lets the user perform (1) ***spatial class translation*** *that changes the class of an object over an arbitrary region of user's choice, and (2)* ***semantic transplantation*** *that transplants semantic information contained in an arbitrary region of the reference image to an arbitrary region in the target image. These two transformations can be used simultaneously, and can realize a complex composite image-editing task like "change the nose of a beagle to that of a bulldog, and open her mouth". The user can also use our method with intuitive copy-paste-style manipulations. We demonstrate the power of our method on various images. Code will be available at* https://github.com/pfnet-research/neural-collage.



(a) Input (b) Spatial class-translation
(c) Semantic transplantation (d) Class + semantic editing

Figure 1: An example result of our editing method.

1. Introduction

Recent advances in deep generative models like generative adversarial networks (GANs) [10] and variational autoencoders (VAEs) [20] make possible the unsupervised learning of rich latent semantic information from images, such as compositions and poses. These models have been extensively studied for the purposes of image editing, and many conditional models have been proposed for various tasks, such as image colorization [14, 17], inpainting [30, 15, 41], domain translation [17, 39, 46, 35, 37], style transfer [12, 42], and object transfiguration [45, 23, 13, 22].

Image conditional GANs [24, 40, 17] based on encoder-decoder architectures have been popular both for their convenient implemention in end-to-end differentiable ML frameworks, and their uncanny ability to produce photo-realistic images. In particular, object transfiguration, where an object in one image is replaced with an object from another, has drawn significant attention for the impact of recent successes. Numerous methods have been proposed for this task. For example, [39, 45] have proposed methods that do not require a paired dataset, [46, 37] have proposed probabilistic transformations that take into account the inherent ambiguity in the translation process, and [13, 21, 1] devised models that can conduct one-to-many domain mapping. All these methods are capable of drastically changing the image while maintaining its contextual information, but may also alter the appearance of regions the user wants to remain unchanged (*e.g.*, CycleGAN may change the background color of an image when transforming one animal into another). Most of these methods, however, are designed to transform the entire image, and little work has addressed the topic of partial image transformation. Previous work has explored attention mechanisms that automatically learn this region of interest [4, 38]. Still others have

*This work was done when the author was at Preferred Networks, Inc.

[33, 6] employed inpainting techniques [30, 15] for interactive partial semantic manipulation of portraits, e.g. changing hairstyles. These methods, however, offer limited freedom in transformation. For instance, the works based on attention mechanisms can transform the user-selected object, but are designed in such that the network automatically determines the region of transformation. Inpainting methods can also perform such object transformation tasks, but current methods are only successful in restricted image domains.

However, in practical image editing tasks like those done with Photoshop, users may want to retain spatial freedom during the transformation; that is, users want fine control over the region of transformation. Also in high demand is a tool for partial translation that exclusively translates an object in the image that is semantically independent from the rests. Tasks like "translate the nose of a beagle into a nose of a bulldog" and "translate rabbit ears into cat ears" are easily described in words, but difficult to formulate in equations. It is indeed not an ordinary task to perform arbitrary, user-selected partial transformations (like those described above) over an arbitrary region of a user's choice.

In this paper, we present a novel CNN-based image editing method that grants the user this very freedom. With our method, the user can transform a user-chosen part of image in a copy-paste fashion–and the user can do this all the while preserving semantic consistency. More precisely, we present a method that features two types of image transformation: (1) spatial class-translation that translates a class category of a region of interest, and (2) semantic transplantation that *transplants* a semantic feature of an user-selected region in an arbitrary image to a region of interest in the target image. To facilitate this editing process, we also propose an efficient optimization method to project images onto the latent space of generator.

We demonstrate on multiple datasets that our method can carry out a diverse set of photorealistic transformations with user-intuitive manipulation. Our method boasts a high object transfiguration fidelity that is on-par with other image translation methods [13, 23]. Moreover, our method does not require task-specific training of the generator; we can use our method with publicly available pre-trained models of class conditional GANs without fine-tuning. This also means that we can *"stand on the shoulders of giants"* and exploit the power of large conditional GANs with high representation power, like those presented in [2].

2. Related Work

In this section, we will briefly describe the ideas of related works on which our method is built, with a particular focus on two approaches from which we will borrow the philosophy.

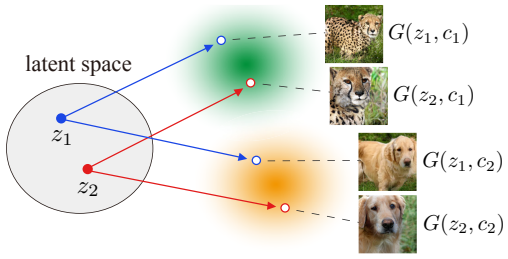


Figure 2: Generator of a class-conditional GAN has latent representation shared among object categories.

Generative adversarial networks (GANs). GAN is a deep generative framework consisting of a generator G and a discriminator D playing a min-max game where G tries to transform a prior distribution *e.g.*, $\mathcal{N}(0, I)$ into the dataset distribution and D tries to distinguish generated (fake) data from true samples [10]. Thanks to the development of techniques against the training instability such as gradient penalty [11] and spectral normalization [25], deep convolutional GANs [34] are becoming the de facto standard for image generation tasks. GANs also excel in representation learning, and numerous studies reports the ability of GANs to capture semantic information. One can produce a sequence of semantically meaningful images by interpolating the latent variable [34, 3, 29, 2].

Class-invariant representation Class-conditional GAN [29, 26, 43, 2] is a framework designed to learn an invariant latent representation among various classes, and it is capable of generating diverse images from a same latent code z by changing class embedding (Figure 2). The work of [26, 2], in particular, succeeded in producing an impressive results by interpolating the parameters of conditional batch normalization layer, which was first introduced in [31, 5]. Conditional batch normalization (CBN) is mechanism that learns conditional information by separately learning condition-specific scaling parameter and shifting parameter for batch normalization. Our method extends the technique used in [26] by restricting the region of interpolation to a region that corresponds to the region of interest in the pixel space. We will refer to our approach *spatial conditional batch normalization* (sCBN). Unlike the manipulation done in style transfer [12], we introduce the conditional information at multiple levels in the network, depending on the style preference of the user. As we will show, sCBN in the lower layers transforms global features, and sCBN at upper layers transforms local features.

Semantic transformation. In order to grant the user with wide freedom of semantic transformation, there has to be some mechanism to finely adjust the user-suggested transformation so that the final product becomes natural. Meth-

ods introduced in [44, 3, 19] overcame this challenge by projecting the original image to the manifold on which the generative model is supported, and prompting the user to make a change on the manifold. In algorithm, this is done by identifying the latent variable that can faithfully reproduce the original image, followed by the transformation of the image on the latent space that will, in the original space, manifest as a transformation desired by the user. A major difficulty lies in the finding of the exact transformation in the latent space that will result in a transformation of the user's intention. The works of [44, 3] resorts to an implicitly formulated latent space transformation based on pixel level constraints, and [19] specializes the model for the transformations of pre-defined attributes. With some technicalities omitted, we will transplant a image attribute of source image into the target image by directly conducting the *copy-paste* operation in the feature space of the images. In the semantic transplantation example in Figure 1c, we change the expression of a canine. Our method does not require any training process for pre-defined set of attributes. We will also propose a method of expanding the latent space that speeds up the optimization process of manifold projection. We can in fact conduct both semantic transplantation and spatial class translation together as well (see Figure 1d).

3. Featured Functionalities

We first provide a brief overview for the usages of our method along with the functionalities featured within. Given the input image of interest and the conditional generator G, the process begins by prompting the user to specify the region of an image to be edited, or the region of the image containing the object that the user wants to transform. The user can then apply spatial class translation and semantic transplantation on the selected region.

3.1. Spatial Class-translation

With our spatial class translation, the user can change the class of the object in the user-selected region of interest (ROI). The user can change the class of a part of the target objects in intuitive fashion. Figure 1b shows the result of transforming the nose of a beagle into that of French bulldog. The user will first be prompted to specify the set of subregions on which to conduct the translation. Multiple subregions (corresponding to the target objects) can be specified, and the user can assign different class labels to each subregion. See Figure 3 top, for example. If the user selects the whole ROI as the object of interest, our method will behave like an ordinary object transfiguration. The user can also specify the morphing strength on a continuous scale from 0% to 100%. See the bottom figure in Figure 3, bottom, for example. We see that the strength of the user-selected class features are continuously increasing with the morphing strength.

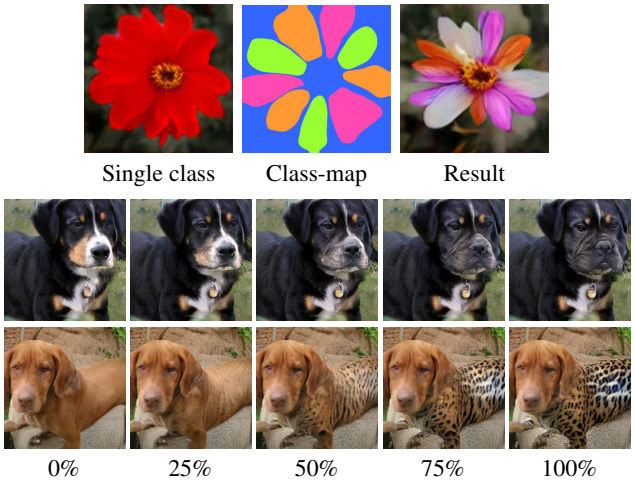


Figure 3: Image generation result of spatial class-translation using multiple classes (top), and results with varying degree of morphing strength (bottom). All the images are generated ones.



Figure 4: An example of semantic transplantation. Input and reference images are real photos, and the result was given by blending the input and the generated patch (post-processing). Feature maps were manipulated in the red area.

3.2. Spatial Semantic Transplantation

With our semantic transplantation, the user can transplant a semantic feature of the user-selected object in the reference image to an object in the target image to be transformed. Our method first prompts the user to specify the region in the target image containing the object of interest, along with the reference image of equal size. The user will be also asked to specify the region in the reference image that contains the semantic information to be transplanted. The method then automatically transplants the semantic information of the specified region of the reference image into the target image. As shown in Figure 4, the user does not have to prepare the reference image containing the object of interest that is precisely aligned with the object in the target image. Also, the object class of the reference image does not have to be the same as the target. For the best performance, the user should select a reference image closely related to the target image. For example, if the user wants to open the mouth of a dog (Figure 1c), using another image of a dog with an opened mouth and otherwise similar pose to the target will achieve the best result.

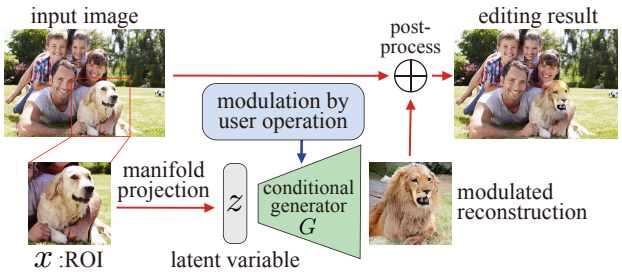


Figure 5: An overview of our image editing workflow.

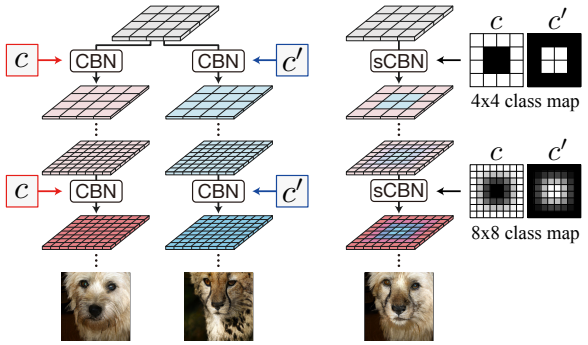


Figure 6: CBN layers gradually add class-specific details to the generated images (left), and the sCBNs transform the style of specific region in an image depending on the given class-map (right).

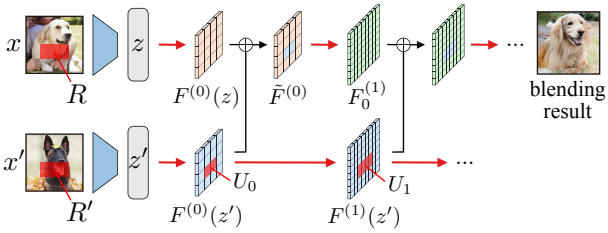


Figure 7: An overview of spatial feature blending.

4. Architecture and Algorithms

In this section, we will provide the mechanism behind our method. Let us first briefly present an overview of the algorithm (Figure 5), We will then articulate the details of each functional component that appear in the procedures in the following subsections.

spatial class translation Our method functions on a trained conditional generator G, paired with the discriminator D with which G was trained. Upon receiving the region of interest x clipped from the target image and the class c of the target object contained in x, the algorithm begins by looking for a latent variable z such that $G(z, c)$ will be close to x in the feature space of D (Manifold Projection step). The class c can either be specified by the user or by a pre-trained classifier. Suppose that the user wants to partially translate a region R in x to a class c', and let V_ℓ be the set of features in ℓ-th conditional batch normalization(CBN) layers that correspond to R in the pixel space. Our method then simply substitutes the parameters governing the shift and mean parameters of V_ℓ with those of c' (Figure 6). This will result in a modification of $G(\cdot, c)$ in which the CBN parameters of V_ℓ exclusively carry the style information of the class c'. A transformed image can be constructed by applying this modified $G(\cdot, c)$ to z.

Semantic transplantation The procedure of the semantic transplantation also begins by looking for the latent variables z that reconstructs x with G (Figure 7). Only, the algorithm also looks for the latent variable z' that reconstructs the clip x' from the reference image. Let R be the user-selected subregion of x on which to conduct the transformation, and let R' be the region in x' from which the user wants to transplant the semantic information. Algorithm expects that R and R' are roughly aligned and are of same dimension. Let U_k be the set of features in the intermediate layers that correspond to R, and let $F^{(\ell)}$ be the outputs of the ℓ-th layer of G. By the choice of x and x', U_ℓ also corresponds to R'. The algorithm then proceeds by simply conducting a location-dependent mixing of $F^{(\ell)}(z)$ and $F^{(\ell)}(z')$ for each ℓ with heavy weights assigned on U_ℓ.

For realistic image transformation, we also conducted a post processing. Because the model is not designed to handle the background, naively pasting a generated patch on the target image can produce artifacts in the region surrounding the object of interest. In order to brush up the final product, we applied Poisson blending [32] exclusively on the region that was specified by the user as the region pertaining to the object of interest. In principle, our spatial editing method is applicable to any generative model (*e.g.*, GAN, VAE) that is equipped with a machanism to iteratively incorporate class information during its image generation process.

We will next elaborate on the design of our sCBN and spatial semantic implantation, along with the other details omitted in the brief description above.

4.1. Spatial Conditional Batch Normalization

Spatial conditional batch normalization (sCBN) is the core of our spatial class translation. As can be inferred from our naming, sCBN is based on batch normalization (BN) [16], a technique developed for the purpose of reducing the internal covariance shift to accelerate the training of neural network. More precisely, we will borrow our idea from conditional batch normalization (CBN) [8, 5], a variant of BN that incorporates the class specific semantic information in the parameters for BN. Given a set of batches sampled each from a single class, the conditional batch normalization [8, 5] works by modulating the set of intermediate features produced from each batch of inputs so that it follow a normal distribution with mean and variance that are

specific to the corresponding class. Let us fix the layer ℓ, and let $F_{k,h,w}$ represent the feature of ℓ-th layer at channel k, height location h, and width location w. Given a batch $\{F_{i,k,h,w}\}$ of $F_{k,h,w}$s generated from class c, the CBN at layer ℓ then transforms $F_{i,k,h,w}$ by:

$$\hat{F}_{i,k,h,w} \leftarrow \gamma_k(c)\frac{F_{i,k,h,w} - \mathrm{E}\left[F_{\cdot,k,\cdot,\cdot}\right]}{\sqrt{\mathrm{Var}\left[F_{\cdot,k,\cdot,\cdot}\right] + \epsilon}} + \beta_k(c) \qquad (1)$$

where $\gamma_k(c), \beta_k(c)$ are trainable multiplication and bias parameters specific to the class c. sCBN modifies (1) by replacing the $\gamma_k(c)$ with $\tilde{\gamma}_{k,h,w}$ given by

$$\tilde{\gamma}_{k,h,w} := \sum_{c=1}^{N_{\mathrm{class}}} W_{h,w}(c) \cdot \gamma_k(c) \qquad (2)$$

where $W_{h,w}(c)$ is a user-selected non-negative tensor map (class-map) that partitions the unity, that is, $\sum_c W_{h,w}(c) = 1$ for each position (h, w). Number of class can be freely chosen. Spatial CBN also conducts an analogous modification on $\beta_k(c)$ to produce $\tilde{\beta}_{k,h,w}$.

In our implementation, we replaced CBN at each layer with sCBN, with $W_{h,w}$ having support on V_ℓ we described above. For standard usage of our method, we expect the user to use $W_{h,w}$ supported on the region in the feature space produced by simply downsampling M, the region in the pixel space over which the user wants to transform the image. However, applying the sets of $W_{h,w}$ that are differently constructed for each layer can produce interesting results as well (see section 5.3).

4.2. Spatial Feature Blending

For spatial feature blending, let $F_i^{(\ell)}$ be the ℓ-th layer feature map of the i-th reference image x_i. Then, with the understanding that $F_0^{(\ell)}$ stands for the ℓ-th feature map of target image x, we produce the transformed image by recursively replacing $F^{(n)}$ in G from bottom to top by

$$\tilde{F}^{(\ell)} := \sum_{i=0}^{N_{\mathrm{input}}} M_i^{(\ell)} \odot F_i^{(\ell)}, \qquad (3)$$

where $M_i^{(\ell)}$ is a user-selected non-negative tensor (feature blending weights) of same dimension as $F_i^{(\ell)}$ such that $\sum_i M_i^{(\ell)}$ is the tensor whose entries are all 1, and $\odot$ represents the Hadamard product. For our implementation, we choose $M_i^{(\ell)}$ that has support on $R_i^{(\ell)}$, the set of features in the ℓ-th layer that correspond to the region to be transplanted in the i-th image in the pixel space. See Figure 7 for the illustration of this process. In general, feature maps in the earlier layer tend to contain abstract, semantic information that is class invariant. As such, for better performance, the user should choose $M_i^{(\ell)}$ that are more concentrated on $i = 0$ for higher ℓ (see section 5.3). We use the same sCBN class-map to generate $F_0^{(\ell)}$ and $\{F_i^{(\ell)}\}$ to reduce artifacts.

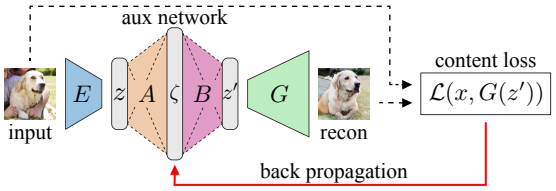


Figure 8: The projection algorithm with auxiliary network.

4.3. Manifold Projection

For the Manifold Projection step, we follow a similar procedure as the one used in [44], using a pre-trained generator and discriminator pair (G, D). Given the clipped image x of interest, the goal of the manifold projection step is to train the encoder E such that $\mathcal{L}(G(E(x)), x)$ is small for some dissimilarity measure $\mathcal{L}$. The choice of $\mathcal{L}$ in our method is the cosine distance on the final feature space of the discriminator D. That is, if $\mathbf{\Psi}(x)$ is the normalized version of the image of x in the final layer of D,

$$\mathcal{L}(x_1, x_2) := 1 - \mathbf{\Psi}(x_1) \cdot \mathbf{\Psi}(x_2). \qquad (4)$$

After training the encoder, one can produce the reconstruction of x by applying G to $z = E(x)$. In the reconstructed image, however, semantically independent objects are often dis-aligned. We therefore calibrate z by backpropagating the loss $\mathcal{L}$. After some rounds of calibration, we can use the resulting z for the image transformation.

However, image features are generally entangled in the latent space of generative model, and the optimization on the complex landscape of loss can be time-consuming. To speed up the process, we construct an auxiliary network that embeds z in higher dimensional space. The auxiliary network consists of an embedding map A that converts z into high dimensional ζ, and a projection map B that converts ζ back to z (Figure 8). That is, instead of calibrating the latent variable z by backpropagating $\mathcal{L}$ through G, we will calibrate ζ by backpropagating $\mathcal{L}$ through G and B. The goal of training this auxiliary network is to find the map B that grants us with a representation of the landscape of $\mathcal{L}$ that is more suitable for optimization, together with the map A that can embed z in a way that is well-suited for the learning on the landscape. Let ζ_j be the variable in the high dimensional latent space after j rounds of calibration. The update rule of ζ_j we use here is

$$\zeta_{j+1} = \zeta_j - \alpha_j \nabla_{\zeta_j} \mathcal{L}(G(B(\zeta_j), x),\ \zeta_0 = A(E(x)), \qquad (5)$$

where α_j is the length parameter at the j-th round. We train the networks A and B using the following loss function:

$$\mathcal{L}_{\mathrm{AB}} := \sum_{j=0}^{N_{\mathrm{iter}}} w_j \mathcal{L}(G(B(\zeta_j)), x) + \lambda \|B(A(z)) - z\|_2^2 \qquad (6)$$

where constants $\{w_j\}$ determine the importance of the j-th round of the calibration. For the first term, A and B are updated through the backpropagation from ζ_j. The second term makes sure that z can be reconstructed from ζ. This process can be seen as a variant of meta-learning [9, 27].

5. Experiments

In this section, we present the results of our experiments together with the design of our network architecture and the experimental settings. For further details, please see the supplementary material.

5.1. Experimental Settings

The generator used in our study is a ResNet-based generator trained as part of a conditional DCGAN. Each residual block in our generator contains the conditional batch normalization (CBN) layer. At the time of inference, these CBN layers are replaced by the aforementioned sCBN layers that are tailored to the user's preference. We base our architectures on those used in previous work [25, 26], and used the pre-trained model from [26]. Our generator is designed to receive a low dimensional signal (128 in our experiments). The generator maps the input into feature maps of dimension 4×4. These feature maps are to be upsampled to create new maps, doubling the resolution at every ResBlock, ultimately mapping to RGB images of user-specified dimensions (128×128, 256×256 in our experiments). For the auxiliary maps A and B, we used a 5-layer MLP with hidden units of size $(1000, 10000, 1000)$ at the respective layers, and treated the 10000-dimensional layer as extended latent variable ζ. For the input of $\mathbf{\Psi}$ in equation (4) for the transformation of 128×128 (256×256) dimensional image, we used 512 (768) dimensional feature vectors of the discriminator D prior to the final global pooling, which is expected to capture the semantic image features [7, 18].

5.2. Results

We demonstrate the representation power of our algorithm using the set of DCGANs trained on different image dataset: (1) dogs+cats (subset of ImageNet, 143 classes), (2) birds (200 classes) [36], (3) flowers (102 classes) [28]. For the dog+cat images, we reused the pre-trained model publicized in [26][1]. For the images of other classes, we trained DCGANs on our own. Auxiliary networks and encoders were trained after the preparation of DCGANs. In order to verify the power of the DCGANs and our manifold projection method, we conducted an experimental study for *non-spatial* translation as well (see supplementary material), and confirmed that the generators used in our study are indeed capturing the class-invariant intermediate features.

[1] https://github.com/pfnet-research/sngan_projection

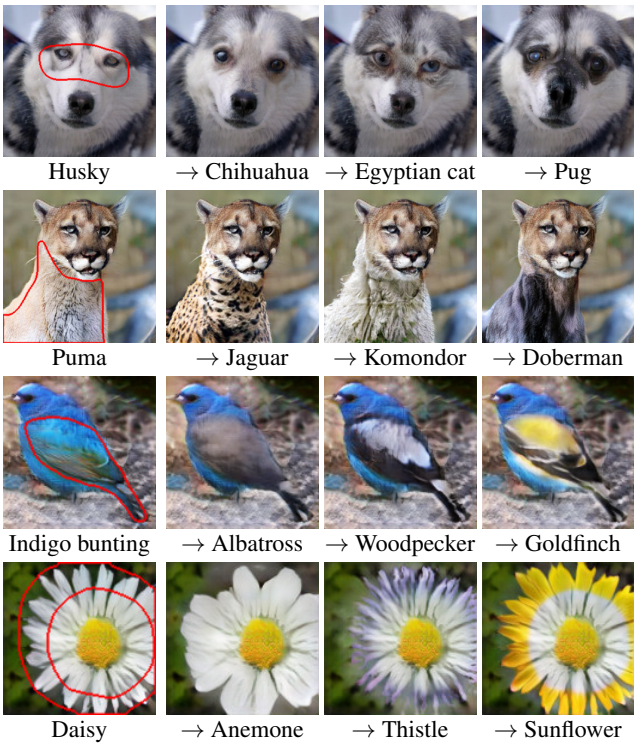


Figure 9: Examples of spatial class-translation. The leftmost column is the single-class generation results and morphing regions. The other columns show morphing results.

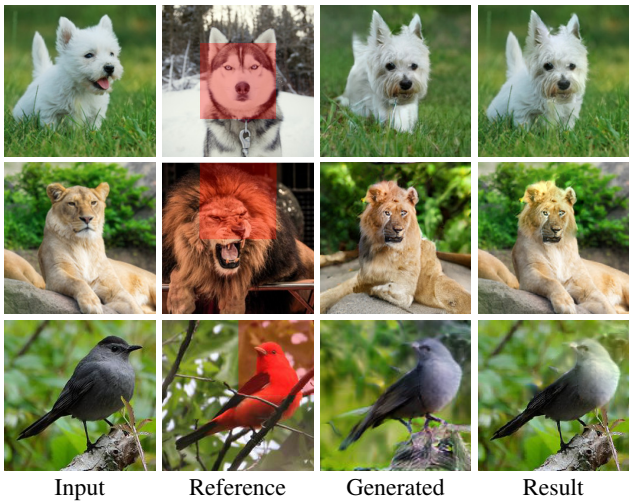


Figure 10: Examples of semantic transplanting. The feature maps were blended in the red regions. Post-processing was applied using manually prepared masks.

Figure 9 shows the example results of our spatial class-translation for various object categories. We see that our translation is modulating both local information like color and texture together with global information in a way that renders the final image semantically consistent. We can dis-



Figure 11: More image editing results on wild images.



Figure 12: Comparison between the results of latent space interpolation (upper rows) and feature blending (lower rows) using the same pairs of latent variables. Feature blending was applied to the feature maps at the first layer in the red regions.

cern the user-selected region from the translated image if we pay attention to the texture and color. On the other hand, the user-selection region is not too reflective of global features like the face of the Husky and the shape of the petal. In the light of the fact that user is prompted to specify the target region in the pixel space, this observation is indicative of the fact that the global features are governed by the layers that are closer to the latent space. Figure 10 shows the examples of our semantic transplanting. Here our method succeeds in modulating the semantic features like the postures of a dog and a bird, and the sex of a lion. We also can see that the performance is robust against the alignment of the region to be transformed.

Note the difference in results between our method and the interpolation in the latent space (Figure 12). For a pair of latent variable z_1, z_2, our method uses z_2 as a reference and interpolates the feature map only in a specified region, while the latent-space interpolation directly changes the value of z. We can see that, our method does not change the semantic context outside the specified region, and preserves the object identity. For more editing results, see Figure 11.

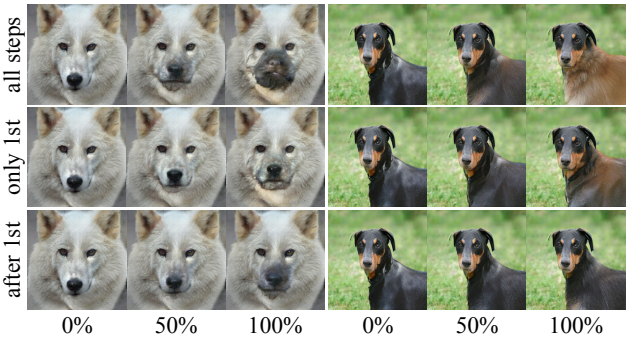


Figure 13: Spatial class-translation results depending on the modulated layers (best seen zoomed).

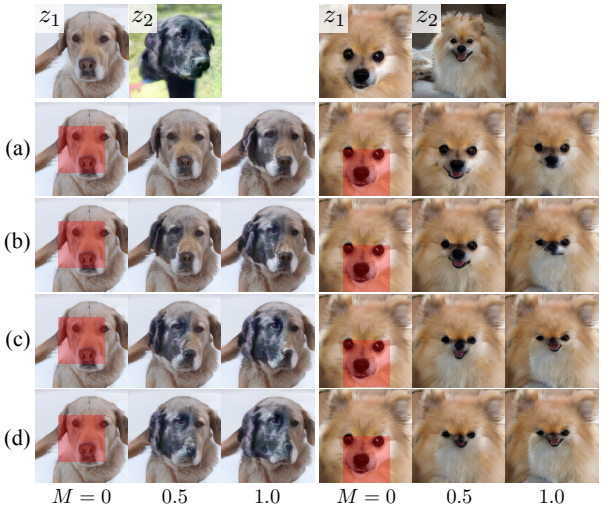


Figure 14: Feature blending results depending on the applied layers: (a) $\ell = 0$, (b) $\ell = 0, 1$, (c) $\ell = 0, 1, 2$, and (d) $\ell = 0, 1, 2, 3$. Weight for reference image was set M in the red regions.

5.3. More Customized Image Transformation

As mentioned in the method section, we can assign different modulation parameters for each ResBlock. In particular, the user is given freedom to independently control feature blending weights and the choice of class-map for spatial class translation. Here, we demonstrate the power of this freedom in the form of highly-customized image transformations. Figure 13 illustrates the result obtained by applying spatial class translation applied to (1) all layers, (2) the layer closest to the input (first layer), and (3) all the layers except the first layer. As we implied in the previous section, the lower layers tend to regulate global features, and the high layers tend to regulate local features. As we can see in the figure, the features affected by the manipulation of the lower layers (face, body shape) are somewhat independent from those of the higher layers (texture). We can therefore choose the level of transformation at each level of locality.

Method	cat → big cat	cat → dog	dog → dog
Ours	7.8%	21.1%	20.8%
UNIT	14.8%	N/A	36.2%
MUNIT	26.0%	55.4%	N/A

Table 1: Comparison of top-5 category classification error rate after class translation between two domains.

Figure 14 shows the result of applying feature blending done at different layers ($\ell \in 0, 1, 2, 3$) with different blending weights. When the blending is done exclusively at the layer $\ell = 0$, the transformation is globally smooth and the extent of the change is rather contained. When the blending is done for the higher layers, local features like fur textures are transplanted as well. However, when the reference image is significantly different from the target image in terms of its topology, the transplantation at higher layers tend to produce artifacts. This tendency become stronger for the transplantation in the layers higher than $\ell = 3$.

5.4. Transfiguration Fidelity

In order to provide some quantitative measurement for the quality of our translation, we evaluate the transfiguration fidelity of our method. We conducted the translation of (1) cat → big cat, (2) cat → dog, and (3) dog species → another dog species, and evaluated the classification accuracy by inception-v3 classifier trained on ImageNet to classify 1000 classes. For all experiments, we selected four classes from both source and target domains, and conducted 1000 translation tasks from a randomly selected class from the source domain to a randomly selected class from the target domain. We used UNIT [23] and MNUIT [13] for baselines. For their evaluations, we used the models described in their publications. Because MUNIT is not designed for class to class translation, we conducted MUNIT using the set of images in the target class as the reference images. Table 1 summarizes the result. For each set of translation task, our method achieved better top-5 error rate than the other methods. This result confirms the efficacy of the combination of the class conditional GANs and the manifold projection. Our method is also capable of many to many translation over a set of 100 or more classes.

5.5. Speed-up by Latent Space Expansion

We also conducted an experiment to verify that our auxiliary network can indeed speed up the learning process. We used the DCGANs trained on dog+cat dataset, and evaluated the average loss for 1000 images randomly selected from ImageNet. The optimization was done with Adam, implemented with the best learning rate α in the search-grid that achieved the fastest loss decrease. Figure 15 compares the transition of the loss function learned on z-latent space against the loss transition produced on ζ-latent space. To

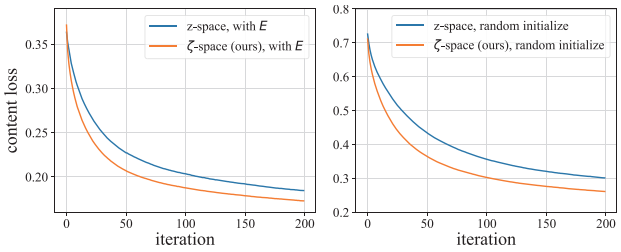


Figure 15: Loss transition during optimization on z-space and ζ-space using the encoder (left) and random initialization (right).

decrease the loss by the same amount, the learning of ζ required only less than $2/3$ the number of iterations required by the learning of z. Calculation overhead due to the latent space expansion was negligible ($< 1\%$). Indeed, the learning process depends on the initial value of the latent variable z or ζ. When compared to random initialization, the optimization process on both z space and ζ space proceeded faster when we set the initial value at $E(x)$ and $E(A(x))$. This result is indicative of the importance of training A.

6. Discussion and Conclusions

Limitations Needless to say, our method can only handle image expressions in the scope of the used generator. While it can spatially translate images of a wide variety, its capability is limited by the diversity of the dataset used to train the models. Also, our method is not necessarily suitable for the transformation of images for which the manifold projection is difficult, since some information is bound to be lost in the process. Transformation of the face of a specific person can be difficult unless the model is trained with an ample set of images for the target person, for example.

Future Work The conditional batch normalization is capable of handling not only class conditions, but other types of information like verbal statements as well. One might be able to conduct even more flexible transformation by making use of them. Also, one might be able to take advantage of the high dimensionality of the extended latent space to disentangle the semantic features. Further experimental and theoretical exploration of the properties of the ζ space might enable even more flexible image manipulations. Finally, we might be able to improve the post-processing by introducing less heuristic machinery like an attention mechanism.

Conclusion In this paper, we introduced a novel image transformation method that allows the user to translate the class of an object and transplant semantic features over a user-specified pixel region of the image. Indeed, there is still much room left for the exploration of the semantic information contained in the intermediate feature spaces of CNNs. We were, however, able to show that we can manipulate this information in a somewhat intuitive manner and produce customized photorealistic images.

Acknowledgments

We thank Jason Naradowsky and Masaki Saito for helpful discussions.

References

[1] A. Almahairi, S. Rajeswar, A. Sordoni, P. Bachman, and A. Courville. Augmented cyclegan: Learning many-to-many mappings from unpaired data. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 195–204, 2018. 1

[2] A. Brock, J. Donahue, and K. Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv:1809.11096*, 2018. 2

[3] A. Brock, T. Lim, J. M. Ritchie, and N. Weston. Neural photo editing with introspective adversarial networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017. 2, 3

[4] X. Chen, C. Xu, X. Yang, and D. Tao. Attention-gan for object transfiguration in wild images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 167–184, 2018. 1

[5] H. De Vries, F. Strub, J. Mary, H. Larochelle, O. Pietquin, and A. C. Courville. Modulating early visual processing by language. In *Advances in Neural Information Processing Systems (NIPS)*, pages 6594–6604, 2017. 2, 4

[6] B. Dolhansky and C. C. Ferrer. Eye in-painting with exemplar generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7902–7911, 2018. 2

[7] A. Dosovitskiy and T. Brox. Generating images with perceptual similarity metrics based on deep networks. In *Advances in Neural Information Processing Systems*, pages 658–666, 2016. 6

[8] V. Dumoulin, J. Shlens, and M. Kudlur. A learned representation for artistic style. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017. 4

[9] C. Finn, P. Abbeel, and S. Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 1126–1135, 2017. 6

[10] 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 (NIPS)*, pages 2672–2680, 2014. 1, 2

[11] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville. Improved training of wasserstein gans. In *Advances in Neural Information Processing Systems (NIPS)*, pages 5767–5777, 2017. 2

[12] X. Huang and S. J. Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 1510–1519, 2017. 1, 2

[13] X. Huang, M.-Y. Liu, S. Belongie, and J. Kautz. Multimodal unsupervised image-to-image translation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 179–196, 2018. 1, 2, 8

[14] S. Iizuka, E. Simo-Serra, and H. Ishikawa. Let there be color!: joint end-to-end learning of global and local image priors for automatic image colorization with simultaneous classification. *ACM Transactions on Graphics (TOG)*, 35(4):110:1–110:11, 2016. 1

[15] S. Iizuka, E. Simo-Serra, and H. Ishikawa. Globally and locally consistent image completion. *ACM Transactions on Graphics (TOG)*, 36(4):107:1–107:14, 2017. 1, 2

[16] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 448–456, 2015. 4

[17] P. Isola, J. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5967–5976, 2017. 1

[18] J. Johnson, A. Alahi, and L. Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 694–711, 2016. 6

[19] T. Kaneko, K. Hiramatsu, and K. Kashino. Generative attribute controller with conditional filtered generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7006–7015, 2017. 3

[20] D. P. Kingma and M. Welling. Auto-encoding variational bayes. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2014. 1

[21] H.-Y. Lee, H.-Y. Tseng, J.-B. Huang, M. Singh, and M.-H. Yang. Diverse image-to-image translation via disentangled representations. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 36–52, 2018. 1

[22] X. Liang, H. Zhang, L. Lin, and E. Xing. Generative semantic manipulation with mask-contrasting gan. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 558–573, 2018. 1

[23] M.-Y. Liu, T. Breuel, and J. Kautz. Unsupervised image-to-image translation networks. In *Advances in Neural Information Processing Systems (NIPS)*, pages 700–708, 2017. 1, 2, 8

[24] M. Mirza and S. Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014. 1

[25] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida. Spectral normalization for generative adversarial networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018. 2, 6, 11

[26] T. Miyato and M. Koyama. cGANs with projection discriminator. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018. 2, 6, 11, 15

[27] T. Munkhdalai, X. Yuan, S. Mehri, and A. Trischler. Rapid adaptation with conditionally shifted neurons. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 3661–3670, 2018. 6

[28] M.-E. Nilsback and A. Zisserman. Automated flower classification over a large number of classes. In *Proceedings of the Indian Conference on Computer Vision, Graphics and Image Processing*, 2008. 6

[29] A. Odena, C. Olah, and J. Shlens. Conditional image synthesis with auxiliary classifier GANs. In *Proceedings of the International Conference on Machine Learning (ICML)*, volume 70 of *Proceedings of Machine Learning Research*, pages 2642–2651, International Convention Centre, Sydney, Australia, 2017. PMLR. 2

[30] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros. Context encoders: Feature learning by inpainting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2536–2544, 2016. 1, 2

[31] E. Perez, F. Strub, H. De Vries, V. Dumoulin, and A. Courville. Film: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pages 3942–3951, 2018. 2

[32] P. Pérez, M. Gangnet, and A. Blake. Poisson image editing. *ACM Transactions on graphics (TOG)*, 22(3):313–318, 2003. 4

[33] T. Portenier, Q. Hu, A. Szabó, S. A. Bigdeli, P. Favaro, and M. Zwicker. Faceshop: Deep sketch-based face image editing. *ACM Transactions on Graphics (TOG)*, 37(4):99:1–99:13, 2018. 2

[34] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2016. 2

[35] P. Sangkloy, J. Lu, C. Fang, F. Yu, and J. Hays. Scribbler: Controlling deep image synthesis with sketch and color. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 5400–5409, 2017. 1

[36] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011. 6

[37] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8798–8807, 2018. 1

[38] C. Yang, T. Kim, R. Wang, H. Peng, and C.-C. J. Kuo. Show, attend and translate: Unsupervised image translation with self-regularization and attention. *arXiv preprint arXiv:1806.06195*, 2018. 1

[39] Z. Yi, H. R. Zhang, P. Tan, and M. Gong. Dualgan: Unsupervised dual learning for image-to-image translation. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 2868–2876, 2017. 1

[40] D. Yoo, N. Kim, S. Park, A. S. Paek, and I. S. Kweon. Pixel-level domain transfer. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 517–532, 2016. 1

[41] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang. Generative image inpainting with contextual attention. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5505–5514, 2018. 1

[42] H. Zhang and K. Dana. Multi-style generative network for real-time transfer. *arXiv preprint arXiv:1703.06953*, 2017. 1

[43] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena. Self-attention generative adversarial networks. *arXiv preprint arXiv:1805.08318*, 2018. 2

[44] J.-Y. Zhu, P. Krähenbühl, E. Shechtman, and A. A. Efros. Generative visual manipulation on the natural image manifold. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 597–613, 2016. 3, 5

[45] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 2242–2251, 2017. 1

[46] J.-Y. Zhu, R. Zhang, D. Pathak, T. Darrell, A. A. Efros, O. Wang, and E. Shechtman. Toward multimodal image-to-image translation. In *Advances in Neural Information Processing Systems (NIPS)*, pages 465–476, 2017. 1

A. Global Transltation

In order to verify the sheer ability of our translation method, we conducted a translation task for the entire image as well (as opposed to the translation for a user-specified subregion). For each of selected images from the ImageNet, we calculated a latent encoding variable z, and applied a set of spatial uniform class-condition c to the layers of the decoder. The results are listed in the Figure 18. The Figure 19 contains the translation of an image (designated as *original*) to all 143 object classes that were used for the training of the dog+cat model. We can see that the semantic information of the original image is naturally preserved in most of the translations.

B. Experimental Details

This section provides further details for the experimental settings.

B.1. Implementation of networks

Our experimental setup is based on that of snGAN-projection [25, 26], and our models were implemented in chainer.

We trained the following three components of the model separately, in order. We first prepared a generator and discriminator pair following the training procedure of conditional GANs described in [25, 26]. We then trained the encoder network for the trained generator using the objective function defined based on the trained discriminator (see the main section in the article about the manifold projection). Finally, fixing the encoder and the generator we trained, we trained the auxiliary networks to enhance the manifold-embedding optimization. For the evaluation of $\mathcal{L}_{\mathrm{AB}}$, we used $N_{\mathrm{iter}} = 2$, $w_0 = 20.0, w_1 = 2.0, w_2 = 1.0$, and $\lambda = 100.0$.

We used the batch size of 8 for the training of all network components. For the training of the cGANs and the encoder, we used the Adam optimizer. The parameters β_1, β_2 were set at $0, 0.9$ respectively for the training of both architecture. The learning rate was set at $2 \cdot 10^{-4}$ for the training of cGANs, and was set at $5 \cdot 10^{-5}$ for the training of the encoder and the auxiliary networks.

For the training of the auxiliary network, we used AdaGrad with adaptive learning rate for the gradient descent to calculate $\{\zeta_i\}_{i=1}^{N_{\mathrm{iter}}}$. We used AdaGrad for this procedure because other methods (*e.g.*, Adam) causes numerical instability in double backpropagation for network parameter update. We trained each model over $450,000$ iterations. For the training of auxiliary networks, we applied the gradient clipping with the threshold of 100.0 and the weight decay at the rate of 0.0001. We trained the networks on GeForce GTX TITAN X for about a week for each component.

B.2. Manifold projection

For the optimization process about z and ζ in the manifold projection at the time of inference, we conducted backpropation from $\mathcal{L}$ and used Adam optimizer for the iterative updates.

We conducted a grid search about the choice of α. The choice of $\alpha = 0.075$ achieved the fastest learning rates for the optimization about z, and the choice of $\alpha = 0.015$ achieved the fastest learning rates for the optimization about ζ.

We also implemented naïve GD, AdaGrad, and L-BFGS-B for the optimization, but Adam performed the best among all methods we tried.

B.3. Transfiguration fidelity study

For the transfiguration fidelity experiment, we conducted a set of automatic spatial class translations. For each one the selected images, we (1) used a pre-trained model to extract the region of the object to be transformed (dog/cat), (2) conducted the manifold projection to obtain the z, (3) passed z to the generator with the class map corresponding to the segmented region, and (4) conducted a post-processing over the segmented region. For the semantic segmentation, we used a TensorFlow implementation of DeepLab v3 Xception model trained on MS COCO dataset[2]. See Figure 17 for the transfiguration examples.

[2] https://github.com/tensorflow/models/tree/master/research/deeplab

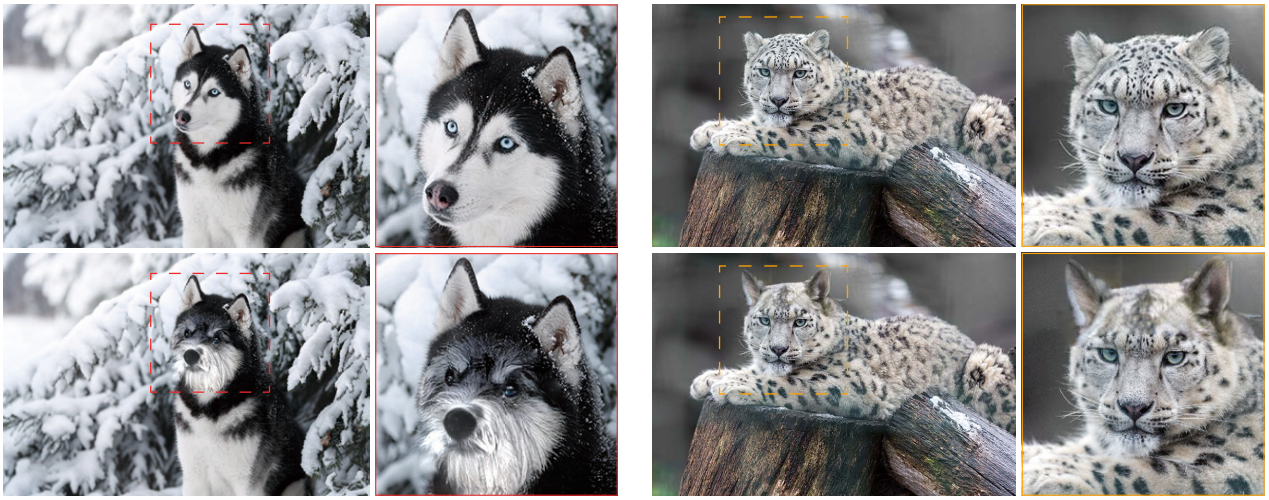


Figure 16: Additional spatial image editing results given by the proposed method. The face of a Husky was changed to that of a Miniature Schnauzer (left), and the ears of a snow leopard were changed to that of an Elkhound (right).

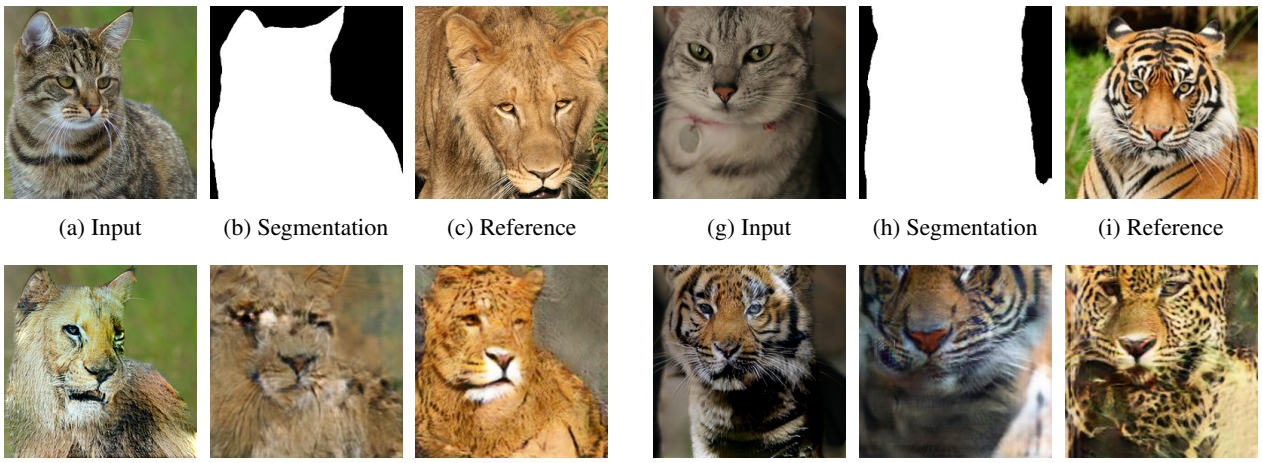


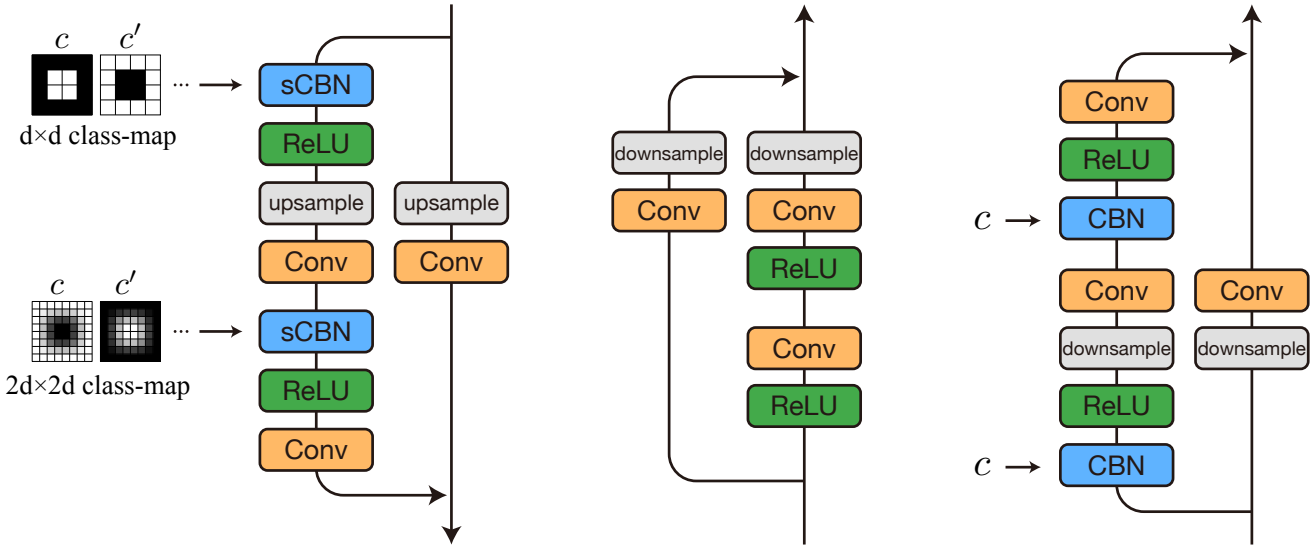
Figure 17: Example inputs and outputs for the fidelity study. The reference images (c, i) were used by MUNIT to specify the target class. Color artifact in our result was caused by Poission blending.



Figure 18: Example results of *non-spatial* class-translation using a cGAN trained on dog+cat dataset and the proposed manifold projection algorithm. The pictures in the leftmost column are sampled from the validation dataset of ImageNet, and the right columns contain translated images. Target categories are Sydney Silky, Labrador Retriever, Collie, EntleBucher, Newfoundland dog, Red Fox, Siamese Cat, and Lion, respectively.

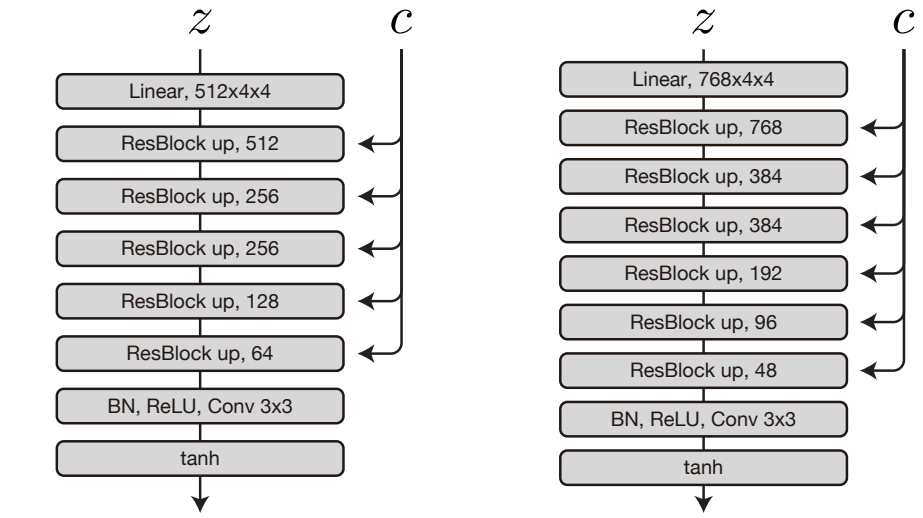


Figure 19: One-to-many *non-spatial* class-translation result. The most upper left image is the input sampled from the validation dataset of ImageNet, and the rest images are translation results to all the 143 dog+cat classes of ImageNet. All the translation results are produced using a same latent variable calculated by the proposed algorithm.



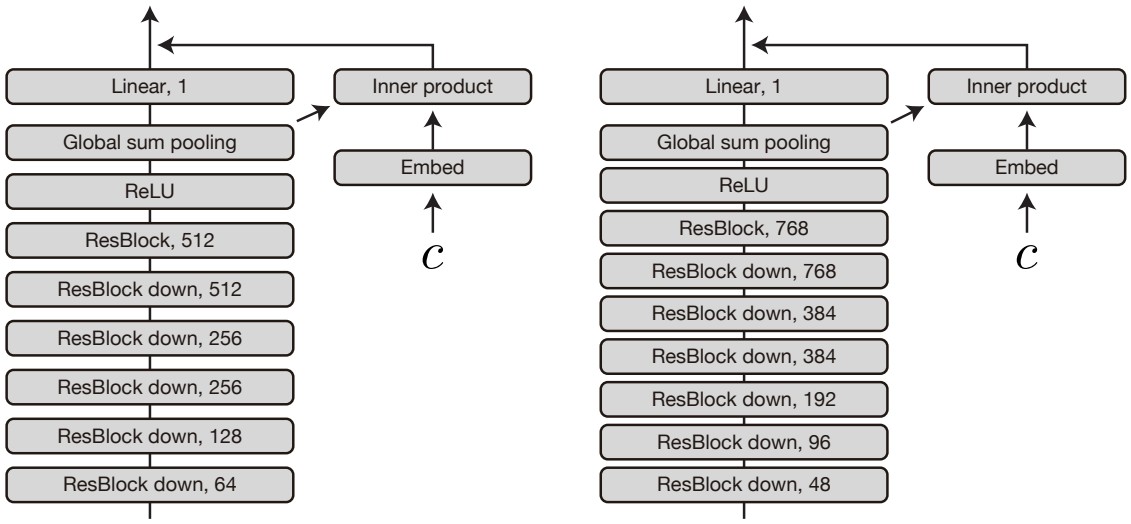
(a) ResBlock for generator (b) ResBlock for discriminator (c) ResBlock for encoder

Figure 20: Architectures of ResBlocks used in the experiments. ResBlocks for generator and discriminator are identical to those used in [26], except that we replaced CBNs with sCBNs at the time of inference. A ResBlock for encoder accepts class-information of the input image given by a pre-trained classifier or the user using CBNs. We used 3×3 convolution for *conv* layers in the residual connections, and 1×1 convolution for those in the shortcut connections. We performed average pooling as the downsampling operation for discriminator and encoder, and nearest neighbor upsampling for generator. Downsampling operations were removed from the last ResBlock of discriminator.



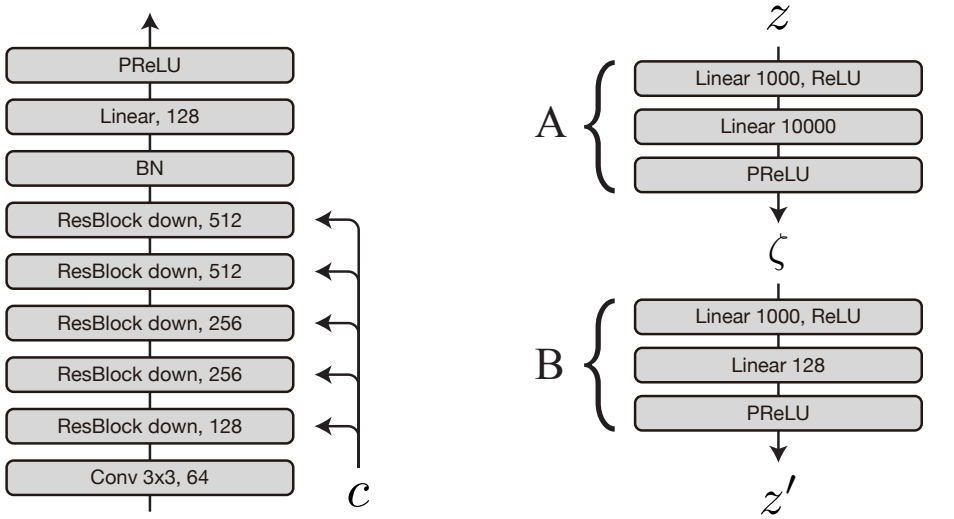
(a) generator for 128×128 pixels (b) generator for 256×256 pixels

Figure 21: Generator architectures for 128×128 pixels and 256×256 pixels image generation tasks. The design is essentially same as that of [26] except for the difference of ResBlock described in Figure 20.



(a) discriminator for 128×128 pixels (b) discriminator for 256×256 pixels

Figure 22: Discriminator architectures for 128×128 pixels and 256×256 pixels image generation tasks.



(a) encoder for 128×128 pixels (b) auxiliary network

Figure 23: Encoder and auxiliary network architectures. We used the encoder for 128×128 pixels input even for 256×256 pixels image projection task by firstly downsampling the input image, because the pixel-level fine detail of images is not important in the encoding process.