ReMix: Towards Image-to-Image Translation with Limited Data

Jie Cao[1,2], Luanxuan Hou[1,2], Ming-Hsuan Yang[3,4,5], Ran He[1,2]*, Zhenan Sun[1,2]
[1]NLPR, CRIPAC & CEBSIT, CASIA [2]AIR, UCAS
[3]University of California at Merced [4]Google Research [5]Yonsei University
{jie.cao, luanxuan.hou}@cripac.ia.ac.cn, mhyang@ucmerced.edu
{rhe, znsun}@nlpr.ia.ac.cn

Abstract

Image-to-image (I2I) translation methods based on generative adversarial networks (GANs) typically suffer from overfitting when limited training data is available. In this work, we propose a data augmentation method (ReMix) to tackle this issue. We interpolate training samples at the feature level and propose a novel content loss based on the perceptual relations among samples. The generator learns to translate the in-between samples rather than memorizing the training set, and thereby forces the discriminator to generalize. The proposed approach effectively reduces the ambiguity of generation and renders content-preserving results. The ReMix method can be easily incorporated into existing GAN models with minor modifications. Experimental results on numerous tasks demonstrate that GAN models equipped with the ReMix method achieve significant improvements.

1. Introduction

In recent years, Generative Adversarial Networks (GANs) [11] have shown much progress in numerous tasks including image-to-image translation. Well-designed adversarial losses [11, 27, 25, 1, 12, 26] provide effective domain-level supervision, making the translated results indistinguishable from the real samples. The GAN-based methods heavily rely on vast quantities of training examples. For instance, Karras *et al*. [19, 20] use 70K high-quality face images to train their models. However, collecting a large amount of image data can be prohibitively expensive or implausible (*e.g*., for masterpieces by artists). This issue highlights the importance of training GANs with limited data. Unfortunately, reducing the amount of training data often leads to severe model overfitting. Recent findings [18, 42] reveal that GANs easily memorize a small training set and then render drastically degraded results in the testing set.

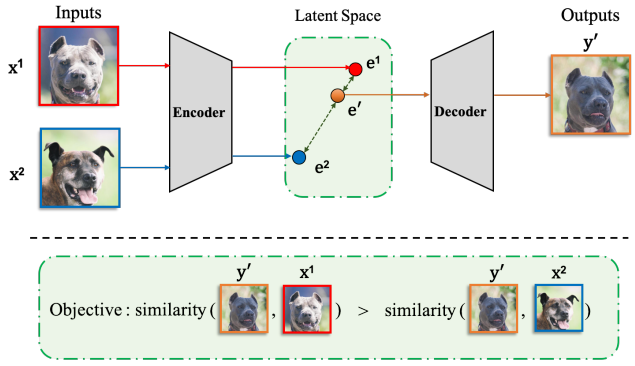


Figure 1. Overview of the proposed data augmentation method. We use the image reconstruction task as an example. The input $\mathbf{x}$ is first encoded into representation $\mathbf{e}$ and then decoded into the output $\mathbf{y}$, and superscript indicates the index of samples. The interpolated data $\mathbf{e}'$ is the convex combination of $\mathbf{e}^1$ and $\mathbf{e}^2$. In this case, we have $d(\mathbf{e}', \mathbf{e}^1) < d(\mathbf{e}', \mathbf{e}^2)$, where d denotes the distance function. We propose to maintain $s(\mathbf{y}', \mathbf{x}^1) > s(\mathbf{y}', \mathbf{x}^2)$, where s is the similarity measure. Here we omit the outputs from $\mathbf{x}^1$ and $\mathbf{x}^2$ for clarity.

Some efforts have recently been made to tackle this problem. The adaption-based approaches [24, 30] use external datasets as an alternative. They first learn a semantically related translation and then adapt it to the translation of interest. Despite the effectiveness, these approaches require additional image collection. Several data augmentation schemes [40, 18, 35, 42, 43] tailored for GANs have been developed to alleviate the need for additional datasets. They use groups of image transformations (*e.g*., cropping, resizing, and cutout [10]) to augment the inputs of the discriminator. Even with limited data, these methods can prevent the discriminator from overfitting, allowing effective adversarial supervision. However, augmenting data for the generator is infeasible due to the problem of leaking [18]. For the image-to-image translation tasks, these methods cannot prevent the generator from memorizing how to trans-

*corresponding author

Interpolate Training Samples at feature level. Content loss is based on perpetual relations of {e', e1, e2}. Try to reconstruct the input image with the helps of distance measure on intermediate features and similarity measure on input and output.

See the summary for details.

Try to reconstruct image with interpolating intermediate features and finding output on the basis of distance mesaure and similarity measure.

late the given source images.

To facilitate training GANs with limited data in image-to-image translation, we propose a data augmentation strategy named ReMix. We mix source images in the feature space using convex combinations. The generator learns to map the mixed samples to the target space against overfitting. In addition, the discriminator is improved in the process of distinguishing the augmented fake samples. We present a novel content loss that maintains the perceptual relations among the samples. The proposed loss avoids the model from producing ambiguous results from the augmented data. In Figure 1, the image reconstruction task is illustrated as an example. We aim to reconstruct two samples $\mathbf{x}^1$ and $\mathbf{x}^2$, and synthesize a virtual input $\mathbf{e}'$ by interpolating the intermediate features $\mathbf{e}^1$ and $\mathbf{e}^2$. However, the reconstruction target for the input $\mathbf{e}'$ is unknown, so the corresponding output $\mathbf{y}'$ requires additional constraints on image content. To this end, we propose to constrain the perceptual relationships among $\{\mathbf{x}^1, \mathbf{x}^2, \mathbf{y}'\}$ based on the relationships among $\{\mathbf{e}^1, \mathbf{e}^2, \mathbf{e}'\}$. Concretely, if $\mathbf{e}'$ is closer to $\mathbf{e}^1$ (or $\mathbf{e}^2$), we then enforce the output $\mathbf{y}'$ to be more similar to $\mathbf{x}^1$ (or $\mathbf{x}^2$) than the other one. In this manner, we provide effective supervision and neatly sidestep estimating the targets for the interpolated inputs.

Main Concept

The ReMix method can be incorporated into existing methods easily. Only a few lines of codes are required to modify the original loss function. In the experiments, we evaluate the proposed method on several tasks, including cross-spectrum face translation on the CASIA dataset [23], animal face translation on the AFHQ dataset [7], and image synthesis from semantic label maps on the Cityscapes dataset [8]. We use the state-of-the-art models [37, 28, 7, 20] on these tasks as the baselines. Experimental results demonstrate that the models equipped with the ReMix method achieve significant improvements. We also train these models with 10% available data and still get comparable performances.

The main contributions are summarized as follows:

- We propose a data augmentation strategy based on feature-level interpolation. Our method reduces the overfitting problem of GANs, particularly for the image-to-image translation tasks.
- We propose to maintain the perceptual relations among samples to optimize the interpolated translations. Our scheme reduces the ambiguity of generation and forces the model to learn content-preserving translations.
- We achieve significant improvements in multiple image synthesis tasks. In addition, we produce plausible results with only 10% training data.

2. Related Work

Unsupervised image-to-image (I2I) translation. These methods aim to learn the mapping from the source domain to the target domain without paired data. Since this problem is inherently ill-posed, the translated results will be ambiguous without additional constraints. To tackle this issue, existing I2I methods are constrained to preserve the image content based on pixel-level values [4, 31], semantic features [34, 15, 22], or attribute labels [4]. The proposed loss functions, *e.g.*, reconstruction loss and cycle consistency loss [44], serve as the objective for content-preserving translation. Existing I2I methods heavily rely on large collections of high-quality images. In this work, we propose an interpolation-based augmentation scheme for image-to-image under limited data. To avoid ambiguous generations from the interpolated input, we develop a new loss function to preserve image content.

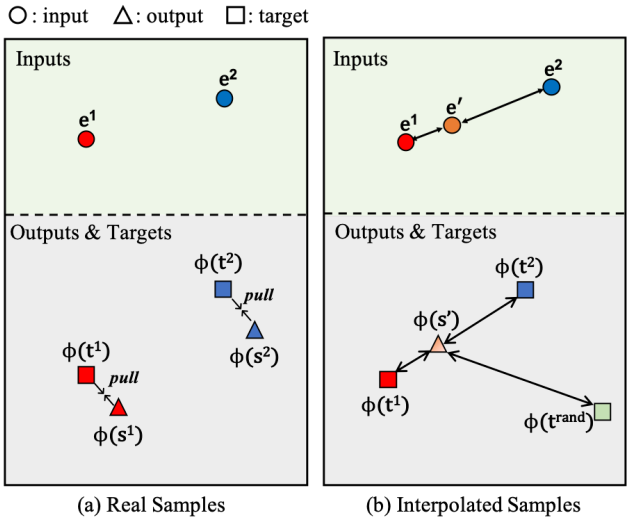
Data augmentation. Numerous methods have been developed to increase the amount of data for training deep learning models without overfitting. Applying some content-preserving operations (*e.g.*, flipping, rotation, and cropping) has become a routine data pre-processing step. To augment data for GANs, some recent approaches use adaptive [18, 40] or automatic [42] strategies to combine these operations. However, these schemes can only be applied to the discriminator and do not address the overfitting problem of the generator.

Interpolation-based augmentation methods [5, 9, 39, 2, 3] focus on mixing training samples at the feature-level or image-level. Linear interpolation is simple but powerful in improving the generalization. For image synthesis, generating plausible interpolated results is also a desired property. However, it remains difficult to determine supervisory signals for the interpolated inputs. The mixup method [39] assumes that the relationship between the training data and supervisory signal is linear. KNN interpolation algorithms [5, 36] only choose the neighbors from the same class to interpolate. The regularization [33] and penalty [29] methods can also be applied to estimate the supervisory signals. For the image-to-image translation problems where the supervision signal is high-dimensional data, these estimations can be prone to errors. In contrast, our method maintains the perceptual relation among samples, which does not require the estimation of supervisory signals.

3. Proposed Method

We aim to learn the mapping function from the source domain $\mathbb{X}$ to the target domain $\mathbb{Y}$. First, we train a generator, $G : \mathbb{X} \mapsto \mathbb{Y}$, for this task. Our goal is two-fold: 1) given $\mathbf{x} \in \mathbb{X}$, $G(\mathbf{x})$ should be indistinguishable from the samples in $\mathbb{Y}$, and 2) $G(\mathbf{x})$ should preserve certain content information. To this end, we optimize the adversarial loss $\mathcal{L}_{\text{gan}}$ and content loss $\mathcal{L}_{\text{con}}$. We formulate the objective functions for

Basic principle of image reconstruction.



(a) Real Samples (b) Interpolated Samples

Figure 2. Illustration of the proposed ReMix method. We use colors to indicate different samples. (a) For each feature $\mathbf{e}$ extracted from a real input $\mathbf{x}$, we minimize the distance between the output $\mathbf{s}$ and corresponding content target $\mathbf{t}$. (b) For the interpolated feature $\mathbf{e}' = \lambda \cdot \mathbf{e}^1 + (1-\lambda) \cdot \mathbf{e}^2$, we constrain the relative similarity of the output $\mathbf{s}'$. Concretely, if $\mathbf{e}'$ is closer/further to $\mathbf{e}^1$ than $\mathbf{e}^2$, we enforce $\phi(\mathbf{s}')$ to be closer/further to $\phi(\mathbf{t}^1)$ than $\phi(\mathbf{t}^2)$. We let ϕ denote the function to extract content representations. In addition, we enforce $\phi(\mathbf{s}')$ to be closer to $\phi(\mathbf{t}^2)$ than $\phi(\mathbf{t}^{\text{rand}})$, and $\mathbf{t}^{\text{rand}}$ is an arbitrary sample except $\mathbf{t}^1$ and $\mathbf{t}^2$.

Content Level supervision through losses in eqn (1) and (2).

the generator G and the discriminator D as:

$$\mathcal{L}^G = \sum_{(\mathbf{x},\mathbf{t})\sim\mathbb{X},\ \mathbf{y}\sim\mathbb{Y}} \mathcal{L}_{\text{gan}}(G(\mathbf{x}),\mathbf{y}) + \mathcal{L}_{\text{con}}(\phi(G(\mathbf{x})),\phi(\mathbf{t})), \tag{1}$$

$$\mathcal{L}^D = \sum_{\mathbf{x}\sim\mathbb{X},\ \mathbf{y}\sim\mathbb{Y}} -\mathcal{L}_{\text{gan}}(G(\mathbf{x}),\mathbf{y}), \tag{2}$$

where ϕ denotes the function to extract content representations. The generator is trained to produce realistic samples that confuse the discriminator. In addition, we enforce the content of $G(\mathbf{x})$ to match the content of $\mathbf{t}$, as illustrated in Figure 2(a). Assigning $\mathbf{t}$ identical to $\mathbf{x}$ is the most common scheme for unpaired image-to-image translation, whereas other choices are also permitted by our method. The forms of $\mathcal{L}_{\text{gan}}$ and $\mathcal{L}_{\text{con}}$ are determined according to specific tasks. In the following, we introduce the ReMix method to augment training data for the GAN-based framework.

3.1. Interpolation-Based Data Augmentation

We augment the training data based on the interpolation at the feature-level. Let $G = G_2 \circ G_1$, where $\circ$ denotes function composition. We mix the intermediate features extracted by G_1. The interpolated data is given by:

$$\mathbf{e}' = \lambda \cdot \mathbf{e}^1 + (1-\lambda) \cdot \mathbf{e}^2, \tag{3}$$

where $\mathbf{e}^1 = G_1(\mathbf{x}^1)$ and $\mathbf{e}^2 = G_1(\mathbf{x}^2)$. Here, $\mathbf{x}^1$ and $\mathbf{x}^2$ are two random samples from the source domain, and $\lambda \in [0,1]$ is the interpolation weight. Note that directly interpolating on the raw input $\mathbf{x}$ is a particular case in our method.

For the interpolated inputs to be useful for training, we need to translate them into content-preserving results. But calculating the content loss $\mathcal{L}_{\text{con}}$ for the interpolated input $\mathbf{e}'$ requires the unknown content target $\mathbf{t}'$. We only know $\mathbf{t}^1$ and $\mathbf{t}^2$, which are the corresponding content targets of $\mathbf{e}^1$ and $\mathbf{e}^2$, respectively. Instead, let $\mathbf{s}' = G_2(G_1(\mathbf{e}'))$, and we constrain the perceptual relationships among $\{\mathbf{t}^1, \mathbf{t}^2, \mathbf{s}'\}$ in the metric space. Without loss of generality, we assume that $\mathbf{e}_1$ weighs more in Equation 3. We then enforce the result $\mathbf{s}'$ to satisfy the following constraint:

$$\mathcal{L}_{\text{con}}\big(\phi(\mathbf{s}'), \phi(\mathbf{t}^1)\big) < \mathcal{L}_{\text{con}}\big(\phi(\mathbf{s}'), \phi(\mathbf{t}^2)\big). \tag{4}$$

That is, the interpolated $\mathbf{e}'$ is closer to $\mathbf{e}_1$ than $\mathbf{e}_2$ in interpolation space, and we enforce the outputs to have an analogous relation: the corresponding output $\mathbf{s}'$ should be closer to $\mathbf{t}^1$ than $\mathbf{t}^2$ in the metric space. Figure 2(b) shows a visualized illustration.

Although Equation 4 provides supervision to generate content-preserving results, the term $\mathcal{L}_{\text{con}}\big(\phi(\mathbf{s}'), \phi(\mathbf{t}^2)\big)$ does not have an upper bound yet. This means simply pushing the output $\mathbf{s}'$ away from $\mathbf{t}^2$ can satisfy the constraint, which is less desirable. Let $\mathbf{e}^{\text{rand}} = G_1(\mathbf{x}^{\text{rand}})$, where $\mathbf{x}^{\text{rand}}$ is an arbitrary sample other than $\mathbf{x}^1$ or $\mathbf{x}^2$. We further propose the following constraint:

clear from fig 2

$$\mathcal{L}_{\text{con}}\big(\phi(\mathbf{s}'), \phi(\mathbf{t}^2)\big) < \mathcal{L}_{\text{con}}\big(\phi(\mathbf{s}'), \phi(\mathbf{t}^{\text{rand}})\big), \tag{5}$$

where $\mathbf{t}^{\text{rand}}$ is the content target of $\mathbf{x}^{\text{rand}}$. Since $\mathbf{x}^{\text{rand}}$ does not contribute to the interpolation, $\mathbf{t}^{\text{rand}}$ should be irrelevant to the output $\mathbf{s}'$. Therefore, we enforce $\mathbf{s}'$ to be closer to $\mathbf{t}^2$ than $\mathbf{t}^{\text{rand}}$.

We refer to the above-described scheme as ReMix for data augmentation. We constrain the relative position of the output based on the perceptual relations among the inputs. Our approach provides effective supervision while allowing diverse generations. The translated results can be multimodal as long as the content constraints are satisfied.

3.2. Learning GAN Models with Limited Data

We show how to apply the proposed ReMix method to the batch-wise training of GAN models. For each iteration, we feed an interpolated data batch to the model with a probability of p. If the batch is not interpolated, we directly train the model with the original settings. Otherwise, we draw two data batches, $\{(\mathbf{x}_i^1, \mathbf{t}_i^1)\}_{i=1}^n$ and $\{(\mathbf{x}_i^2, \mathbf{t}_i^2)\}_{i=1}^n$, where n denotes the batch size. Similar to the mixup method [39], we calculate the interpolation weight by:

Sample μ from Beta distribution

$$\mu = \text{Beta}(\alpha, \alpha), \tag{6}$$

$$\lambda = \max(\mu, 1-\mu), \tag{7}$$

where Beta(α, α) denotes the beta distribution parameterized by α. We then obtain the augmented inputs $\{\mathbf{e}'_i\}_{i=1}^n$ by the interpolation scheme formulated in Equation 3. Note that $\mathbf{e}_i^1$ always weighs more in the interpolation because we have $\lambda \geq 0.5$.

We compute the adversarial loss $\mathcal{L}_{\text{gan}}$ using the augmented batch for domain-level supervision. For the content supervision described in Equations 4 and 5, we have:

Domain Level Supervision

$$\mathcal{L}_p = \sum_i \max\{0,\ \mathcal{L}_{\text{con}}\big(\phi(\mathbf{s}'_i), \phi(\mathbf{t}_i^1)\big) - \mathcal{L}_{\text{con}}\big(\phi(\mathbf{s}'_i), \phi(\mathbf{t}_i^2)\big)\}, \tag{8}$$

$$\mathcal{L}_n = \sum_i \max\{0,\ \mathcal{L}_{\text{con}}\big(\phi(\mathbf{s}'_i), \phi(\mathbf{t}_i^2)\big) - \bar{a}\}. \tag{9}$$

We minimize $\mathcal{L}'_{con} = \mathcal{L}_p + \mathcal{L}_n$ for the interpolated inputs, which is referred to as the relative form of $\mathcal{L}_{con}$. We initialize $\bar{a}$ to 0 and update it dynamically during training. Concretely, we first compute:

$$a = \sum_i \mathcal{L}_{\text{con}}\big(\phi(\mathbf{s}'_i), \phi(\mathbf{t}_j^2)\big), \tag{10}$$

where $j \neq i$, so $\mathbf{e}_j^2$ does not contribute to the interpolation of $\mathbf{e}'_i$. Hence, a denotes the mean distance of the unrelated output-target pairs within the training batch. Then, we adopt a momentum update of $\bar{a}$:

$$\bar{a} \leftarrow m \cdot \bar{a} + (1 - m) \cdot (a - \bar{a}), \tag{11}$$

where we set the momentum coefficient m to 0.99. Algorithm 1 shows the main step to train the generator with the ReMix data augmentation method. For the discriminator, the process is similar. We compute one single content loss in the relative form, whereas the ReMix method can also be applied to the case with multiple content losses. Each loss can be calculated in the relative form independently.

3.3. Comparison with Existing Methods

In contrast to existing approaches, the ReMix method does not rely on estimating the corresponding target $\mathbf{t}'$ for each interpolated input $\mathbf{e}'$. For example, the scheme by Zhang *et al*. [39] assumes the relationship between the training data and supervision signal is linear. Hence, given the interpolation weight λ for the inputs, this scheme [39] computes:

$$\mathbf{t}' = \lambda \cdot \mathbf{t}^1 + (1 - \lambda) \cdot \mathbf{t}^2. \tag{12}$$

In addition, this method proposes to directly use the supervisory signal of the sample that weighs more with:

$$\mathbf{t}' = \begin{cases} \mathbf{t}^1, & \text{if } \lambda \geq 0.5, \\ \mathbf{t}^2, & \text{otherwise.} \end{cases} \tag{13}$$

Furthermore, regularizations can be used in the estimation the content target. For instance, based on the LSR method [33], we can clamp the weight λ into a predefined range $[\lambda_{\text{min}}, \lambda_{\text{max}}]$ to interpolate the content target. Other tricks like noise injection, nearest-neighbor interpolation [5, 36] can also be used.

The approaches described above use the estimated input-target pairs to augment the training data. For the classification tasks where the target $\mathbf{t}'$ is a label, they are shown to be effective. However, in the image-to-image translation tasks, we use raw images or high-dimensional features as supervision signals, which are substantially more difficult to estimate. Inaccurate estimations may negatively affect the quality of the augmented training data. We evaluate the ReMix method against these schemes for multiple image-to-image translation tasks.

Algorithm 1 Pseudocode of the ReMix method.

```
# D : Discriminator, (N * C * H * W) -> N
# G: Generator, which consists of G1 and G2
# G1: (N * C * H * W) -> (N * C' * H' * W')
# G2: (N * C' * H' * W') -> (N * C * H * W)
# gan : the adversarial loss, N -> N
# phi: extracting content, (N * C * H * W) -> (N * E)
# con: the content loss, (N * E) -> N

for batch1, batch2 in data_loader:
   # the probability of augmentation is p
   if p > rand(0, 1):
      # x, t : input and target, (N * C * H * W)
      x1, t1 = batch1
      x2, t2 = batch2
      e1, e2 = G1.forward(x1), G1.forward(x2)

      # interpolating the input
      mu = beta.draw() # the beta distribution
      lambda = max(mu, 1-mu)
      e_prime = lambda * e1 + (1 - lambda) * e2

      # calculating the adversarial loss
      s_prime = G2.forward(e_prime)
      prediction = D.forward(s_prime)
      loss_gan = gan(prediction).mean()

      # calculating the relative content loss
      d1 = con(phi(s_prime), phi(t1))
      d2 = con(phi(s_prime), phi(t2))
      # clamp : clamp all elements into [0, Infinity]
      l_p = clamp(d1 - d2).mean()
      l_n = clamp(d2 - a_mean).mean()
      loss_con = l_p + l_n

      # update of Generator
      loss = loss_gan + loss_con
      loss.backward()
      update(G.parameters)

      # momentum update of a_mean
      # shuf : shuffle data along the batch axis
      a = con(phi(s_prime), phi(shuf(t2))).mean()
      a_mean = m * a_mean + (1 - m) * (a - a_mean)
```

4. Experiments and Analysis

We consider three practical tasks, *i.e*., NIR-to-VIS face translation, animal face translation, and image synthesis from semantic label maps. We first introduce the datasets and implementation details.

The Animal Faces-HQ dataset (AFHQ) [7] provides animal faces of three domains: cat, dog, and wildlife. Each

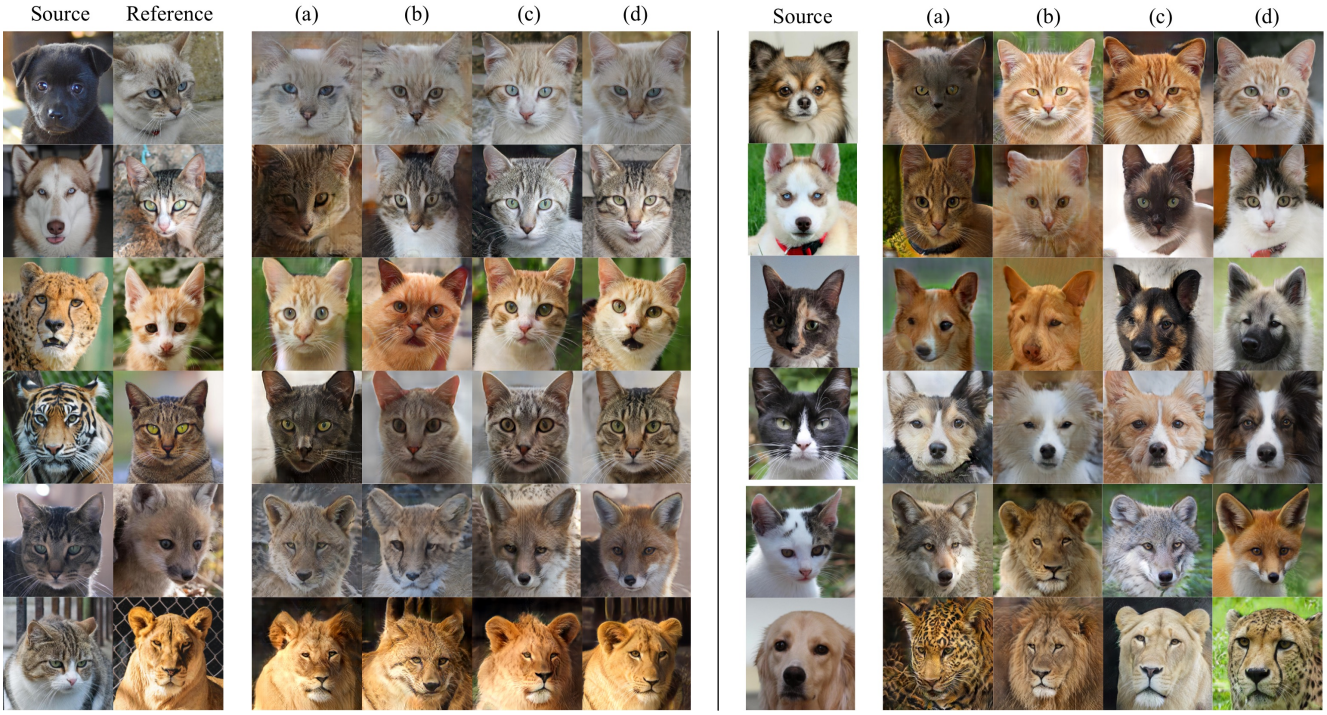


Figure 3. Visual examples synthesized by different methods with 10 % training data on the AFHQ dataset [7]. The left part is the results of reference-guided translation, and the right part is the results of latent-guided translation. The column of results are (a) StarGAN v2 [6] (baseline), (b) baseline + WM (Equation 13), (c) baseline + mixup [39], and (d) baseline + ReMix (ours).

category contains about 5,000 images. We aim to train a single model to learn the translations among these domains. The StarGAN v2 [7] is used as the baseline for this task. We interpolate the output of the style encoder in the baseline model [7]. In our ReMix method, we modify the style reconstruction loss to the relative form.

The CASIA NIR-VIS 2.0 Face Dataset [23] contains near-infrared (NIR) and visible (VIS) images of 725 subjects. There are large variations of the same identity, including lighting, expression, pose, and accessories. For the NIR-to-VIS face translation, we use the LightCNN-29v2 [37] and StyleGAN2 [20] to build an encoder-decoder network. The LightCNN [1] is pre-trained, and we choose to interpolate its outputs. We train the StyleGAN from scratch using the default settings [20]. In addition, we add the L1 distance loss [16] in the pixel space as the content supervision. When learning GAN models with the interpolated data, we use the relative form of the L1 distance loss.

The Cityscapes dataset [8] contains 3,500 street scene images and the corresponding semantic label maps. We use the SPADE Net [28] for translating the label maps to scenes. We directly interpolate the raw inputs in this task. The baseline model uses the perceptual loss [17] guided by VGGNet [32]. For ReMix, we modify this loss to the relative form.

We only modify the mentioned losses for the ReMix method, and the other losses remain the same. We set the probability of augmentation to 0.25 for each iteration. Similar to the mixup method [39], we set the hyper-parameter $\alpha = 0.2$ for the beta distribution.

We implement these baselines using the released source codes. The input resolution is 512×256 on the Cityscapes dataset [8] and 256×256 for the others. We change the dimension of the input latent in StyleGAN2 [20] to 256. Except for this point, we do not make any modifications to the network architectures. We use the recommended training settings in the original work for each baseline model, including the batch size, optimizer, training iterations, and loss weights. To determine the value of the augmentation probability in our ReMix method, we conduct a grid search on the AFHQ dataset and use the FID score as the metric. We use the found value for all the experiments without hyper-parameter tuning.

4.1. Animal Face Translation

The first task is to change the species of the given animal face. If a reference image is available, an encoder extracts the style representation from it. Then, the generator mixes the style with the content of the input, producing the

[1] https://github.com/AlfredXiangWu/LightCNN

Table 1. Fréchet Inception Distance (FID, lower is better) and Learned Perceptual Image Patch Similarity (LPIPS, higher is better) of different methods on the AFHQ dataset [7]. The WM method is described in Equation 13.

Method	Latent-guided translaion				Reference-guided translation			
	100% data		10% data		100% data		10% data	
	FID↓	LPIPS↑	FID↓	LPIPS↑	FID↓	LPIPS↑	FID↓	LPIPS↑
Baseline : StarGAN v2 [7]	16.18	0.450	46.02	0.431	19.78	0.432	38.42	0.402
Baseline + WM	20.03	0.484	41.36	**0.477**	23.64	0.475	45.88	0.455
Baseline + mixup [39]	15.91	0.453	28.15	0.466	18.67	0.453	27.34	0.451
Baseline + ReMix (ours)	**15.22**	**0.491**	**21.82**	0.471	**15.56**	**0.481**	**22.92**	**0.460**

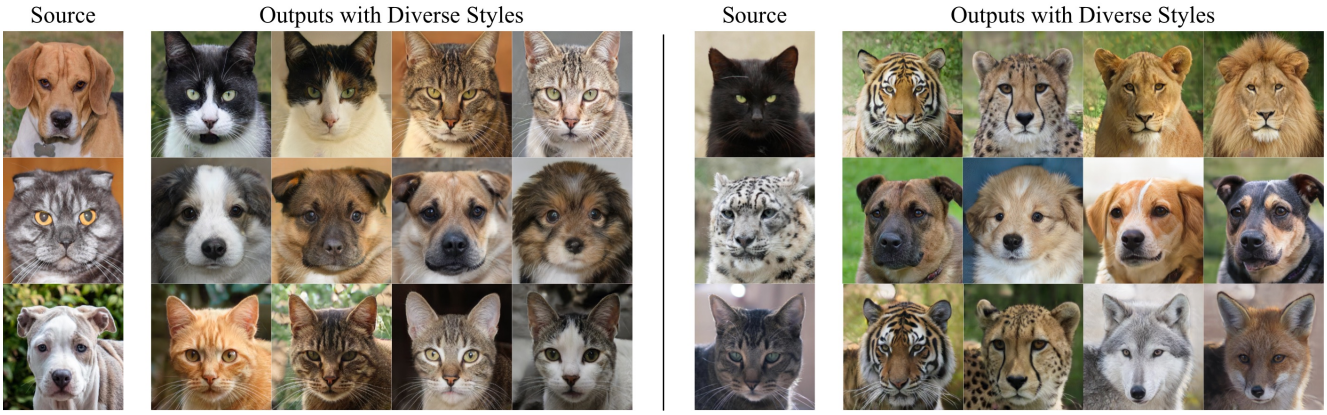


Figure 4. Diverse translation results on the AFHQ dataset [7]. Our model can learn to generate diverse high-quality results using only 10% data in the training set.

translated result. Otherwise, given a one-hot class label, the generator draws a latent code from a prior distribution as the style representation. The two types of tasks are referred to as reference-guided translation and latent-guided translation, respectively.

We randomly choose 500 images for each class, which is about 10% of the full training set. Then, we train the models under the 10% data settings. We evaluate our method against existing interpolation approaches, including the mixup [39] and WM (Equation 13) schemes. Figure 3 shows some synthesized images by the evaluated methods. The baseline model suffers from the overfitting problem and generates some unrealistic texture details. Overall, the proposed method synthesizes images with higher visual quality than other schemes. Figure 4 shows diverse translation results by our method. Given a source image, we generate diverse results by randomly sampling multiple reference images. These results show that our approach can generate distinctive styles while preserving content information.

We also evaluate the quality of synthesized images using Fréchet Inception Distance (FID) [14] and Learned Perceptual Image Patch Similarity (LPIPS) [41]. The FID metric [14] measures the Wasserstein distance between two image sets. We extract the features from the last average pooling layer of the Inception-v3 model [33] to calculate the FID score. The LPIPS score [41] measures the diversity of images using the L1 distance in the feature space, and the pre-trained AlexNet [21] is used as the feature extractor. We compute the FID and LPIPS scores for every pair of the image domains (*e.g.*, dog→cat, cat→wildlife) and report the average values.

Table 1 shows the FID and LPIPS scores. We evaluate the methods under both the 10% and 100% data settings. Our approach performs favorably against existing augmentation methods in terms of these quantitative metrics. The FID scores indicate that our results are more similar to the real data. The LPIPS score of our method with 10% data is higher than that of the baseline trained with the full training set. These results demonstrate the proposed method is effective for diverse and realistic image translation.

4.2. Cross-Spectrum Face Translation

The second task is to translate the input NIR face into the VIS domain and preserve the identity (content) infor-

Table 2. Rank-1 accuracy (%) and verification rate (%, VR) of different methods on the CASIA NIR-VIS 2.0 dataset (the first fold). FAR denotes the false acceptance rate. The performances are evaluated according to the "recognition via generation" protocol [13]. We use LightCNN-29v2 [37] and StyleGAN2 [20] to build an encoder-decoder network as the baseline. The WM method is described in Equation 13. "raw input" means we directly use the LightCNN model to match the NIR faces with the VIS faces.

Method	Rank-1	VR@FAR=1%	VR@FAR=0.1%
Raw Input	96.84	99.10	94.68
Baseline [37, 19]	93.13	94.22	88.79
Baseline + WM	91.32	92.57	81.27
Baseline + mixup [39]	97.66	99.38	97.59
Baseline + ReMix (ours)	**98.18**	**99.63**	**98.11**

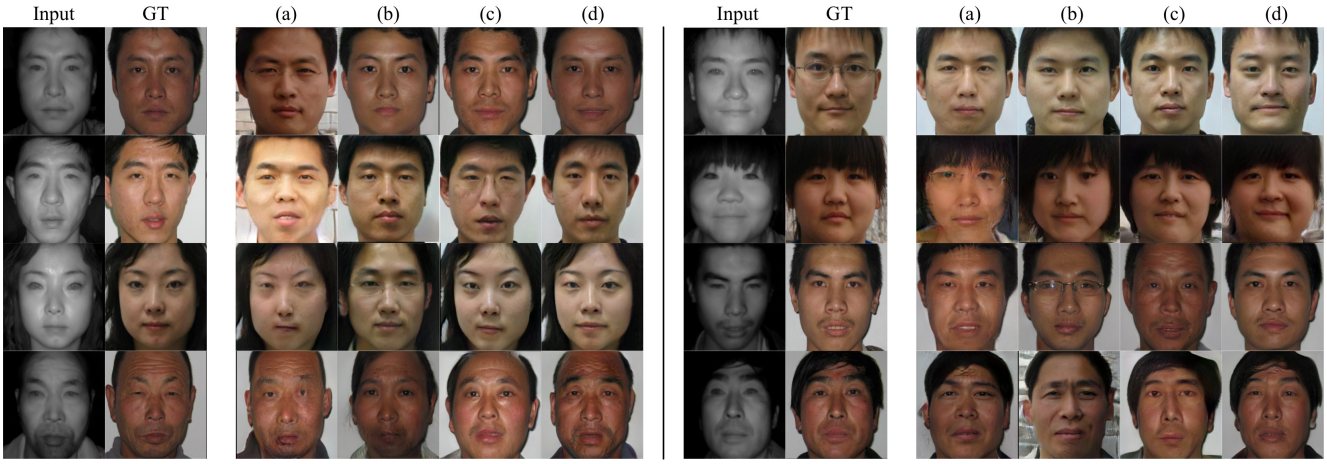


Figure 5. Face cross-spectrum translation results from the testing set of the CASIA NIR-VIS 2.0 dataset [13]. We train models with only 357 pairs of NIR-VIS images. The column of results are (a) the baseline (LightCNN [37] + StyleGAN [20]), (b) baseline + WM (Equation 13), (c) baseline + mixup [39], and (d) baseline + ReMix (ours). "GT" denotes the ground-truth VIS image (not strictly paired).

mation. Prior works evaluate image generation methods based on the "recognition via generation" protocol [13]. That is, given a NIR face image, we use the translated result for recognition. Using the same protocol, we use the 357 identities in the training set of the first fold [2] to train our model. The remaining images are used for testing. Table 2 shows Rank-1 accuracy and the verification rates of different methods. The competing data augmentation methods are the mixup [39] and WM (Equation 13) methods. Our method performs favorably against the other schemes in terms of both rank accuracy and verification rates. These experimental results show that our method reduces the domain gap between the NIR and VIS face images effectively.

We also consider an extreme scenario where only one NIR-VIS image pair of each identity is used for training. That is, the training set consists of only 357 pairs of images. We show the generated samples from the testing set in Figure 5. We observe that the ReMix method synthesizes plausible results even with limited data. The baseline model and WM cannot produce satisfactory results due to model overfitting. The appearances synthesized by the mixup method are realistic, but the identities look different from the input NIR images.

[2]There are 10 fold experiments on the CASIA NIR-VIS 2.0 Face Dataset. Image generation methods are usually evaluated on the first fold.

4.3. Image Synthesis from Semantic Label Maps

Given a semantic layout, we train the translation models to synthesize a photorealistic image. The official training split of the Cityscapes dataset [8] consists of 3000 pairs of image and semantic label maps. We train the models under the 10% and 100% data settings for this task. We use FID to measure the distance between the distributions of the real images and the distribution of the synthesized results. In addition, we perform semantic segmentation on the synthesized images and then evaluate how well the predicted results match the input label maps. Similar to prior work [28], we use DRN-D-105 [38] to measure the segmentation accuracy. Table 3 reports the FID scores and the predicted segmentation accuracy of different methods. In

Table 3. Semantic segmentation scores (higher is better) and Fréchet Inception Distance (FID, lower is better) of different methods on the Cityscapes dataset [8]. "mIou" denotes the mean Intersection-Over-Union metric, and "accu" denotes the pixel-wise accuracy. "real data" denotes the results evaluated on the real images, which is the theoretical upper bound we can achieve.

	100% training data			10% training data		
Method	mIoU↑	accu↑	FID↓	mIoU↑	accu↑	FID↓
Real Data	75.6	84.8	-	-	-	-
Baseline: SPADE Net [28]	62.3	81.9	71.8	48.3	68.2	85.9
Baseline + WM	51.1	80.2	95.5	45.4	57.3	108.8
Baseline + mixup [39]	65.5	82.3	64.7	59.7	72.1	71.5
Baseline + ReMix (ours)	**70.3**	**82.7**	**50.1**	**62.1**	**74.4**	**68.0**

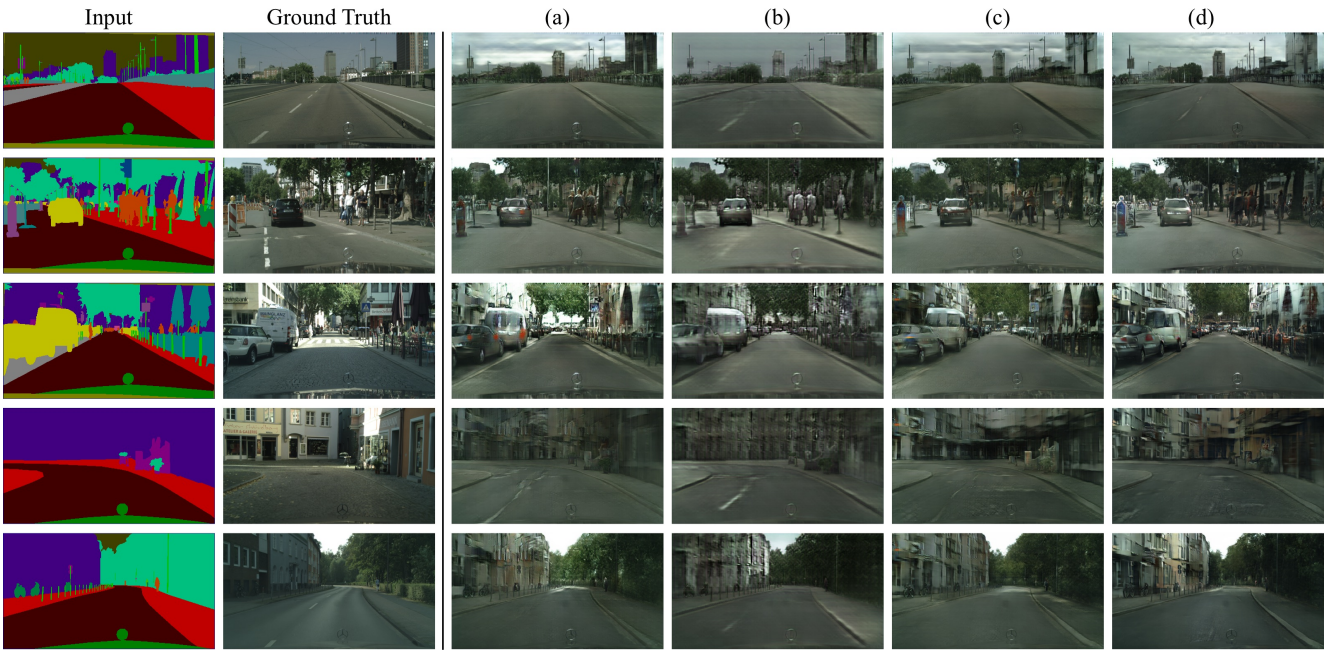


Figure 6. Visual examples synthesized by different methods with 10% training data on the Cityscapes dataset. The columns of results are (a) SPADE Net [6] (baseline), (b) baseline + WM (Equation 13), (c) baseline + mixup [39], and (d) baseline + ReMix (ours).

Figures 6, we provide samples of the translation results under the 10% data setting. The competing methods in the table are also the mixup and WM methods. We observe that the ReMix method performs favorably against the state-of-the-art methods in terms of the quantitative metrics. Our method produces results with better visual quality and fewer artifacts. In contrast, the performances under the 10% data setting degrade significantly for the other approaches.

5. Conclusion

We introduce an interpolation-based data augmentation method to tackle the overfitting problem of GANs. In addition, we present to maintain the perceptual similarity among samples to reduce the ambiguity of generation. The proposed approach renders content-preserving results from the interpolated inputs, facilitating the model training in image-to-image translation. We demonstrate that our method vastly improves the image quality and quantitative metrics in numerous tasks, especially when the training data is limited.

6. Acknowledgement

This work is funded by National Natural Science Foundation of China (Grant No. U1836217) and Beijing Natural Science Foundation (Grant No. JQ18017). M.-H. Yang is partially supported by NSF CAREER 1149783.

References

[1] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. In *ICML*, 2017. 1

[2] Christopher Beckham, Sina Honari, Alex Lamb, Vikas Verma, Farnoosh Ghadiri, R Devon Hjelm, and Christopher Pal. Adversarial mixup resynthesizers. In *ICLRW*, 2019. 2

[3] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. Mixmatch: A holistic approach to semi-supervised learning. In *NeurIPS*, 2019. 2

[4] Konstantinos Bousmalis, Nathan Silberman, David Dohan, Dumitru Erhan, and Dilip Krishnan. Unsupervised pixel-level domain adaptation with generative adversarial networks. In *CVPR*, 2017. 2

[5] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *JAIR*, 2002. 2, 4

[6] Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. StarGAN: Unified generative adversarial networks for multi-domain image-to-image translation. In *CVPR*, 2018. 5, 8

[7] Yunjey Choi, Youngjung Uh, Jaejun Yoo, and Jung-Woo Ha. StarGAN v2: Diverse image synthesis for multiple domains. In *CVPR*, pages 8188–8197, 2020. 2, 4, 5, 6

[8] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016. 2, 5, 7, 8

[9] Terrance DeVries and Graham W Taylor. Dataset augmentation in feature space. In *ICLRW*, 2017. 2

[10] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017. 1

[11] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NeurIPS*, 2014. 1

[12] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *NeurIPS*, 2017. 1

[13] Ran He, Jie Cao, Lingxiao Song, Zhenan Sun, and Tieniu Tan. Adversarial cross-spectral face completion for nir-vis face recognition. *IEEE TPAMI*, 2019. 7

[14] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In *NeurIPS*, 2017. 6

[15] Xun Huang, Ming-Yu Liu, Serge Belongie, and Jan Kautz. Multimodal unsupervised image-to-image translation. In *ECCV*, 2018. 2

[16] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *CVPR*, 2017. 5

[17] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *ECCV*, 2016. 5

[18] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data. In *NeurIPS*, 2020. 1, 2

[19] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, 2019. 1, 7

[20] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *CVPR*, 2020. 1, 2, 5, 7

[21] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, 2012. 6

[22] Hsin-Ying Lee, Hung-Yu Tseng, Jia-Bin Huang, Maneesh Singh, and Ming-Hsuan Yang. Diverse image-to-image translation via disentangled representations. In *ECCV*, 2018. 2

[23] Stan Li, Dong Yi, Zhen Lei, and Shengcai Liao. The casia nir-vis 2.0 face database. In *CVPRW*, 2013. 2, 5

[24] Ming-Yu Liu, Xun Huang, Arun Mallya, Tero Karras, Timo Aila, Jaakko Lehtinen, and Jan Kautz. Few-shot unsupervised image-to-image translation. In *CVPR*, 2019. 1

[25] Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley. Least squares generative adversarial networks. In *ICCV*, 2017. 1

[26] Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. Which training methods for gans do actually converge? *arXiv preprint arXiv:1801.04406*, 2018. 1

[27] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014. 1

[28] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *CVPR*, 2019. 2, 5, 7, 8

[29] Gabriel Pereyra, George Tucker, Jan Chorowski, Łukasz Kaiser, and Geoffrey Hinton. Regularizing neural networks by penalizing confident output distributions. In *ICLRW*, 2017. 2

[30] Kuniaki Saito, Kate Saenko, and Ming-Yu Liu. COCO-FUNIT: Few-shot unsupervised image translation with a content conditioned style encoder. In *ECCV*, 2020. 1

[31] Ashish Shrivastava, Tomas Pfister, Oncel Tuzel, Joshua Susskind, Wenda Wang, and Russell Webb. Learning from simulated and unsupervised images through adversarial training. In *CVPR*, 2017. 2

[32] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 5

[33] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, 2016. 2, 4, 6

[34] Yaniv Taigman, Adam Polyak, and Lior Wolf. Unsupervised cross-domain image generation. In *ICLR*, 2017. 2

[35] Ngoc-Trung Tran, Viet-Hung Tran, Ngoc-Bao Nguyen, Trung-Kien Nguyen, and Ngai-Man Cheung. Towards good practices for data augmentation in gan training. *arXiv preprint arXiv:2006.05338*, 2020. 1

[36] Ji Wan, Sheng Tang, Yongdong Zhang, Jintao Li, Pengcheng Wu, and Steven CH Hoi. Hdidx: High-dimensional indexing

for efficient approximate nearest neighbor search. *Neurocomputing*, 2017. 2, 4

[37] Xiang Wu, Ran He, Zhenan Sun, and Tieniu Tan. A light cnn for deep face representation with noisy labels. *IEEE TIFS*, 2018. 2, 5, 7

[38] Fisher Yu, Vladlen Koltun, and Thomas Funkhouser. Dilated residual networks. In *CVPR*, 2017. 7

[39] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *ICLR*, 2018. 2, 3, 4, 5, 6, 7, 8

[40] Han Zhang, Zizhao Zhang, Augustus Odena, and Honglak Lee. Consistency regularization for generative adversarial networks. In *ICLR*, 2019. 1, 2

[41] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018. 6

[42] Shengyu Zhao, Zhijian Liu, Ji Lin, Jun-Yan Zhu, and Song Han. Differentiable augmentation for data-efficient gan training. In *NeurIPS*, 2020. 1, 2

[43] Zhengli Zhao, Zizhao Zhang, Ting Chen, Sameer Singh, and Han Zhang. Image augmentations for gan training. *arXiv preprint arXiv:2006.02595*, 2020. 1

[44] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*, 2017. 2

ReMix Summary

Prepared By: Sushant Gautam
Part of 30 Day GAN Paper Reading
https://github.com/sushant097/30-Days-GANs-Paper-Reading

Problem: Image-to-image (I2I) translation methods based on generative adversarial networks (GANs) typically suffer from overfitting when limited training data is available

Solution: Propose a data augmentation ReMix method to augment training data for the GAN-based framework.

Main Idea: Interpolate Training Samples at feature level. Propose content loss which is based on perpetual relations of {e', e1, e2}. Try to reconstruct the input image with the helps of distance measure on intermediate features and similarity measure on input and output.

Data Augmentation Method Architecture:

The main goal is to reconstruct x1 and x2 and synthesize virtual input e' by interpolating e1 and e2.

But the target to e' is unknown since it is new point interpolated from e1 and e2. Note that x1 corresponds to intermediate e1 point, x2 corresponds to intermediate e2 point.
So, y' needs additional constraints on image content and for that they use perceptual relationships among {x1, x2, y'} based on the relationships among {e1, e2, e'}. So, if e' is closer to e1 then enforce the output y' to be more similar to x1, else y' be more similar to x2.
By that approach, author mentioned that it can maintain constraint and accurately estimate tagets for the interpolate inputs.

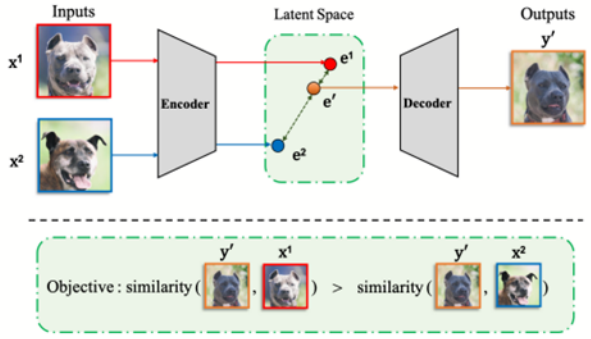


Figure 1. Overview of the proposed data augmentation method. We use the image reconstruction task as an example. The input $\mathbf{x}$ is first encoded into representation $\mathbf{e}$ and then decoded into the output $\mathbf{y}$, and superscript indicates the index of samples. The interpolated data $\mathbf{e}'$ is the convex combination of $\mathbf{e}^1$ and $\mathbf{e}^2$. In this case, we have $d(\mathbf{e}', \mathbf{e}^1) < d(\mathbf{e}', \mathbf{e}^2)$, where d denotes the disctance function. We propose to maintain $s(\mathbf{y}', \mathbf{x}^1) > s(\mathbf{y}', \mathbf{x}^2)$, where s is the similarity measure. Here we omit the outputs from $\mathbf{x}^1$ and $\mathbf{x}^2$ for clarity.

Note: This paper might contain more theory, but is straightforward when you go through paper.

Method:

Goal is to Learn the mapping function from the source domain X to the target domain Y.

X ⟶ Mapping Function G(X) ⟶ Y

Target is :
1) given x ∈ X , G (x) should be indistinguishable from the samples in Y , and
2) G (x) should preserve certain content information.

For that they uses two loss function: adversarial loss - Lgan and content loss- Lcon .

Content Level supervision through losses in eqn (1) and (2).

the generator G and the discriminator D as:

$$\mathcal{L}^G = \sum_{(\mathbf{x},\mathbf{t})\sim\mathbb{X},\ \mathbf{y}\sim\mathbb{Y}} \mathcal{L}_{\text{gan}}(G(\mathbf{x}),\mathbf{y}) + \mathcal{L}_{\text{con}}(\phi(G(\mathbf{x})),\phi(\mathbf{t})), \tag{1}$$

$$\mathcal{L}^D = \sum_{\mathbf{x}\sim\mathbb{X},\ \mathbf{y}\sim\mathbb{Y}} -\mathcal{L}_{\text{gan}}(G(\mathbf{x}),\mathbf{y}), \tag{2}$$

where ϕ denotes the function to extract content representations. The generator is trained to produce realistic samples that confuse the discriminator. In addition, we enforce the content of $G(\mathbf{x})$ to match the content of $\mathbf{t}$, as illustrated in

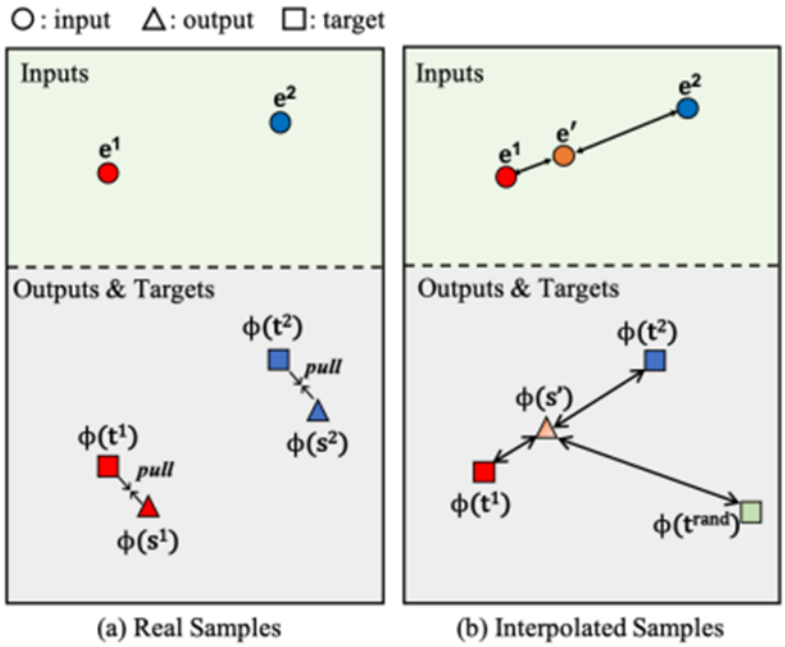
Relate Figure 1 and 2 to understand.
Relate e' to s' and ϕ is a function that extract the content 't' representations.

Idea is to minimize the distance d and similarity measure s between the feature maps such that reconstructed or predicted output is nearly same as input.

First, they mix the intermediate features extracted by G1 as shown in figure 1. The interpolated data e' is given by:

$$\mathbf{e}' = \lambda \cdot \mathbf{e}^1 + (1-\lambda)\cdot \mathbf{e}^2, \tag{3}$$

where e1 = G1(x1) and e2 = G1(x2), x1 and x2 are two random samples as input.
and λ ∈ [0 , 1]



(a) Real Samples (b) Interpolated Samples

Figure 2. Illustration of the proposed ReMix method. We use colors to indicate different samples. (a) For each feature $\mathbf{e}$ extracted from a real input $\mathbf{x}$, we minimize the distance between the output $\mathbf{s}$ and corresponding content target $\mathbf{t}$. (b) For the interpolated feature $\mathbf{e}' = \lambda\cdot\mathbf{e}^1 + (1-\lambda)\cdot\mathbf{e}^2$, we constrain the relative similarity of the output $\mathbf{s}'$. Concretely, if $\mathbf{e}'$ is closer/further to $\mathbf{e}^1$ than $\mathbf{e}^2$, we enforce $\phi(\mathbf{s}')$ to be closer/further to $\phi(\mathbf{t}^1)$ than $\phi(\mathbf{t}^2)$. We let ϕ denote the function to extract content representations. In addition, we enforce $\phi(\mathbf{s}')$ to be closer to $\phi(\mathbf{t}^2)$ than $\phi(\mathbf{t}^{\text{rand}})$, and $\mathbf{t}^{\text{rand}}$ is an arbitrary sample except $\mathbf{t}^1$ and $\mathbf{t}^2$.

As t1 and t2 are known targets of e1 and e2 but t' is unknown target of e'.
They suppose s' = G2(G1(e')) and constrain the perceptual relationships among {t1 , t2 , s'} in the metric space.

From Fig 2, e1 is more closer to e' than e2 => corresponding output s' should be closer to t1 than t2.
So, content loss of s' and t1 is less than s1 and t2 inequality given as:

$$\mathcal{L}_{\text{con}}\big(\phi(\mathbf{s}'),\phi(\mathbf{t}^1)\big) < \mathcal{L}_{\text{con}}\big(\phi(\mathbf{s}'),\phi(\mathbf{t}^2)\big). \tag{4}$$

This have no any upper bound and author suppor Xrand, different sample other than x1 and x2 and e_rand = G(Xrand).

clear from fig 2

$$\mathcal{L}_{\text{con}}\big(\phi(\mathbf{s}'),\phi(\mathbf{t}^2)\big) < \mathcal{L}_{\text{con}}\big(\phi(\mathbf{s}'),\phi(\mathbf{t}^{\text{rand}})\big), \tag{5}$$

where $\mathbf{t}^{\text{rand}}$ is the content target of $\mathbf{x}^{\text{rand}}$. Since $\mathbf{x}^{\text{rand}}$ does not contribute to the interpolation, $\mathbf{t}^{\text{rand}}$ should be irrelevant to the output $\mathbf{s}'$. Therefore, we enforce $\mathbf{s}'$ to be closer to $\mathbf{t}^2$ than $\mathbf{t}^{\text{rand}}$.

Learning Gan Models from Limited Data:

Finally they apply the proposed ReMix method for GAN training.

Steps:

1. Sample μ from Beta distribution parameterized by α.

$$\mu = \text{Beta}(\alpha, \alpha), \tag{6}$$

$$\lambda = \max(\mu, 1-\mu), \tag{7}$$

2. Obtain the augmented inputs {ei'}(i=1 to n)
by the interpolation scheme formulated in Equation 3.

3. We compute the adversarial loss Lgan using the augmented batch for domain-level supervision

Domain Level Supervision

$$\mathcal{L}_p = \sum_i \max\{0,\ \mathcal{L}_{\text{con}}(\phi(\mathbf{s}'_i), \phi(\mathbf{t}_i^1)) - \mathcal{L}_{\text{con}}(\phi(\mathbf{s}'_i), \phi(\mathbf{t}_i^2))\} \tag{8}$$

$$\mathcal{L}_n = \sum_i \max\{0,\ \mathcal{L}_{\text{con}}(\phi(\mathbf{s}'_i), \phi(\mathbf{t}_i^2)) - \bar{a}\}. \tag{9}$$

4. Minimize Lcon' for the interpolated inputs.

We minimize $\mathcal{L}'_{con} = \mathcal{L}_p + \mathcal{L}_n$ for the interpolated inputs, which is referred to as the relative form of $\mathcal{L}_{con}$. We initialize $\bar{a}$ to 0 and update it dynamically during training. Concretely, we first compute:

$$a = \sum_i \mathcal{L}_{\text{con}}(\phi(\mathbf{s}'_i), \phi(\mathbf{t}_j^2)), \tag{10}$$

5. Compute the momentum a' as:

$$\bar{a} \leftarrow m \cdot \bar{a} + (1-m) \cdot (a - \bar{a}),$$

where we set the momentum coefficient m to 0.99.

Complete Algorithm shown as below:

Algorithm 1 Pseudocode of the ReMix method.

```
# D : Discriminator, (N * C * H * W) -> N
# G: Generator, which consists of G1 and G2
# G1: (N * C * H * W) -> (N * C' * H' * W')
# G2: (N * C' * H' * W') -> (N * C * H * W)
# gan : the adversarial loss, N -> N
# phi: extracting content, (N * C * H * W) -> (N * E)
# con: the content loss, (N * E) -> N

for batch1, batch2 in data_loader:
   # the probability of augmentation is p
   if p > rand(0, 1):
      # x, t : input and target, (N * C * H * W)
      x1, t1 = batch1
      x2, t2 = batch2
      e1, e2 = G1.forward(x1), G1.forward(x2)

      # interpolating the input
      mu = beta.draw() # the beta distribution
      lambda = max(mu, 1-mu)
      e_prime = lambda * e1 + (1 - lambda) * e2

      # calculating the adversarial loss
      s_prime = G2.forward(e_prime)
      prediction = D.forward(s_prime)
      loss_gan = gan(prediction).mean()

      # calculating the relative content loss
      d1 = con(phi(s_prime), phi(t1))
      d2 = con(phi(s_prime), phi(t2))
      # clamp : clamp all elements into [0, Infinity]
      l_p = clamp(d1 - d2).mean()
      l_n = clamp(d2 - a_mean).mean()
      loss_con = l_p + l_n

      # update of Generator
      loss = loss_gan + loss_con
      loss.backward()
      update(G.parameters)

      # momentum update of a_mean
      # shuf : shuffle data along the batch axis
      a = con(phi(s_prime), phi(shuf(t2))).mean()
      a_mean = m * a_mean + (1 - m) * (a - a_mean)
```

Outputs:

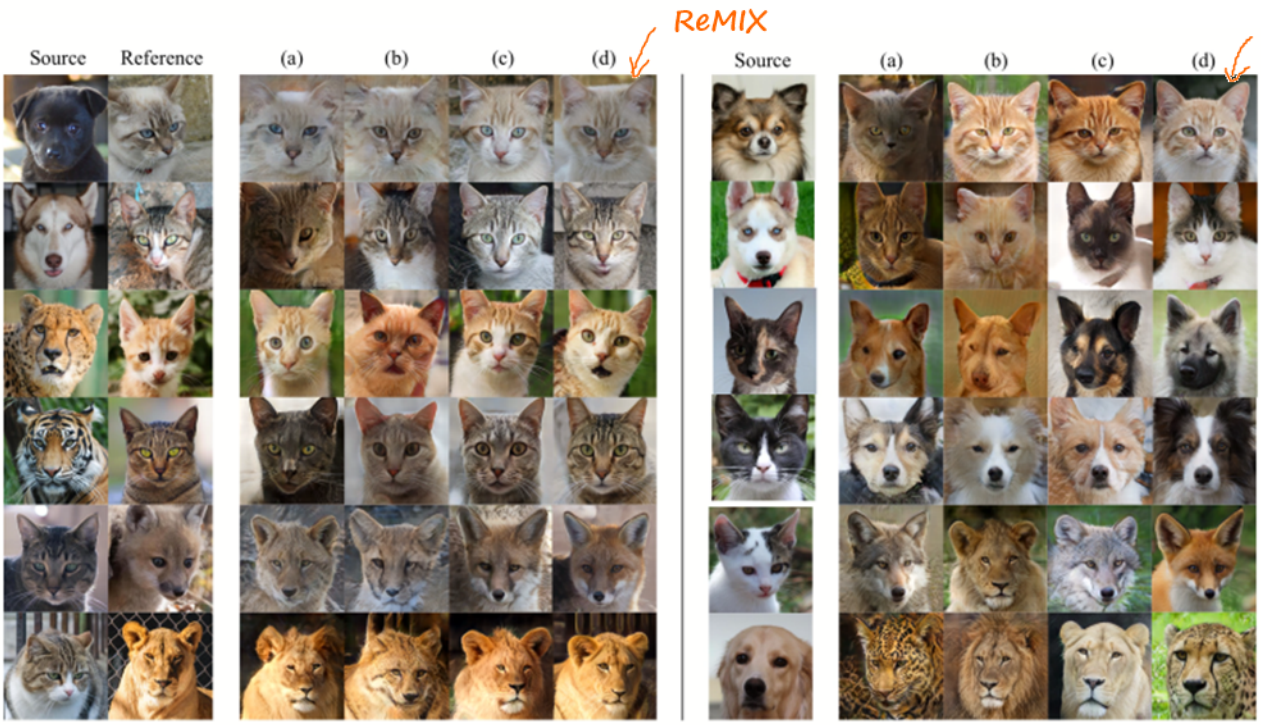


Figure 3. Visual examples synthesized by different methods with 10 % training data on the AFHQ dataset [7]. The left part is the results of reference-guided translation, and the right part is the results of latent-guided translation. The column of results are (a) StarGAN v2 [6] (baseline), (b) baseline + WM (Equation 13), (c) baseline + mixup [39], and (d) baseline + ReMix (ours).