Conditional Image Synthesis by Generative Adversarial Modeling

A Dissertation Presented for the

Doctor of Philosophy

Degree

The University of Tennessee, Knoxville

Zhifei Zhang

December 2018

© by Zhifei Zhang, 2018
All Rights Reserved.

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgments.



Zhifei Zhang

December 2018

Acknowledgments

I am forever grateful to everyone who has made my time in AICIP at the University of Tennessee the fulfilling and intensely meaningful experience it has been. I struggle to describe my advisor Prof. Hairong Qi's supervision without descending into a morass of superlatives, so I will be brief. Of course she is brilliant. Of course her piercing insight, intuitive clarity, droll wit, and laudable humility make being her student a delight. But even more exceptional are her integrity and her unwavering devotion to her students.

One of the great joys from my time in graduate school has been all of my group mates, past and present. I have been fortunate enough to be in the same group with them. I would like to offer my heartfelt thanks to all of the graduate students and alumnus who overlapped with me in the AICIP group. Without a doubt, they made the AICIP group an enjoyable and stimulating research environment: Dr. Wei Wang, Dr. Jiajia Luo, Dr. Rui Guo, Dr. Zhibo Wang, Dr. Ali Taalimi, Dr. Shuangjiang Li, Dr. Li He, Dr. Liu Liu, Dr. Ying Qu, Yang Song, Alireza Rahimpour, Chengcheng Li, Weisheng Tang, Elliot Davis Greenlee, Fanqi Wang, Nabati Ramin, and everyone I didn't list (who nonetheless have my gratitude). I thank them for their help and for being such inspiring role models. I would like to thank my wife Yang Song in particular for her indomitable spirit, her inexhaustible optimism, and for all the late night research discussions in our apartment.

Throughout my doctoral studies I have been fortunate to have an outstanding committee and I would like to thank my committee members Dr. Jens Gregor, Dr. Husheng Li, Dr. Russell Zaretzki, and my dear advisor Dr. Hairong Qi for their inspiration and guidance. Specially thank you to Dr. Gonzalez for involving me in your new book. Finally, I want to thank my parents for giving me every advantage a child can be given, for the love of learning they fostered, and for their unconditional support and encouragement.

Abstract

Recent years, image synthesis has attracted more interests. This work explores the recovery of details (low-level information) from high-level features. The generative adversarial nets (GAN) has led to the explosion of image synthesis. Moving away from those application-oriented alternatives, this work investigates its intrinsic drawbacks and derives corresponding improvements in a theoretical manner.

Based on GAN, this work further investigates the conditional image synthesis by incorporating an autoencoder (AE) to GAN. The GAN+AE structure has been demonstrated to be an effective framework for image manipulation. This work emphasizes the effectiveness of GAN+AE structure by proposing the conditional adversarial autoencoder (CAAE) for human facial age progression and regression. Instead of editing on the image level, i.e., explicitly changing the shape of face, adding wrinkle, etc., this work edits the high-level features which implicitly guide the recovery of images towards expected appearance.

While GAN+AE being prevalent in image manipulation, its drawbacks lack exploration. For example, GAN+AE requires a weight to balance the effects of GAN and AE. An inappropriate weight would generate unstable results. This work provides an insight to such instability, which is due to the interaction between GAN and AE. Therefore, this work proposes the decoupled learning (GAN//AE) to avoid the interaction between them and achieve a robust and effective framework for image synthesis. Most existing works used GAN+AE structure could be easily adapted to the proposed GAN//AE structure to boost their robustness. Experimental results demonstrate the correctness and effectiveness of the provided derivation and proposed methods, respectively.

In addition, this work extends the conditional image synthesis to the traditional area of image super-resolution, which recovers the high-resolution image according the low-resolution

counterpart. Diverting from such traditional routine, this work explores a new research direction — reference-conditioned super-resolution, in which a reference image containing desired high-resolution texture details is used besides the low-resolution image. We focus on transferring the high-resolution texture from reference images to the super-resolution process without the constraint of content similarity between reference and target images, which is a key difference from previous example-based methods.

Table of Contents

1 Introduction **1**

1.1 Generative Modeling . . . 2

1.1.1 Generative Models . . . 2

1.1.2 Learning of Generative Models . . . 3

1.1.3 Generative Adversarial Networks . . . 4

1.2 Image Generation and Manipulation . . . 6

1.2.1 Image Generation . . . 6

1.2.2 Image Manipulation . . . 7

1.3 Motivation and Contribution . . . 8

2 Instability of GAN and Stabilization **11**

2.1 Introduction . . . 12

2.2 Drawbacks of GAN . . . 13

2.2.1 Relation to Kullback–Leibler Divergence . . . 14

2.2.2 Mode missing . . . 15

2.2.3 Instability in Training . . . 16

2.3 GAN Incorporated with Autoencoder . . . 18

2.4 Adaptive Decay on Adversarial Effect . . . 20

2.5 Training Strategy . . . 22

2.5.1 Regularization on $\boldsymbol{z}$ . . . 23

2.5.2 Random Sampling of $\boldsymbol{z}$ . . . 27

2.6 Experimental Evaluation . . . 27

2.6.1 Performance Metrics . . . 28

2.6.2 Evaluation on the MNIST dataset . . . 29
2.6.3 Evaluation on the Compositional MNIST dataset . . . 30
2.6.4 Evaluation on Natural Images and Faces . . . 31
2.6.5 Evaluation on Multiple Domains . . . 32
2.6.6 Evaluation on Stability with Adaptive Decay . . . 33
2.7 Summary . . . 33

3 Conditional Image Synthesis **35**
3.1 A Task: Face Age Progression/Regression . . . 36
3.1.1 Related Work . . . 38
3.1.2 Traversing on the Manifold . . . 40
3.2 Conditional Adversarial Autoencoder . . . 41
3.2.1 Network Structure of CAAE . . . 42
3.2.2 Objective Function . . . 43
3.2.3 Discriminator on $\boldsymbol{z}$ . . . 44
3.2.4 Discriminator on Face Images . . . 46
3.2.5 Differences from Other Generative Networks . . . 47
3.3 Experimental Evaluation . . . 48
3.3.1 Data Collection . . . 49
3.3.2 Implementation of CAAE . . . 49
3.3.3 Qualitative and Quantitative Comparison . . . 50
3.3.4 Tolerance to Pose, Expression, and Occlusion . . . 53
3.4 Summary . . . 54

4 Decoupled Learning for Conditional Image Synthesis **55**
4.1 Background and Motivation . . . 56
4.2 Decoupled Learning . . . 58
4.2.1 Theoretic Reasoning . . . 58
4.2.2 Network Structure . . . 59
4.3 Normalized Relative Discriminative Score . . . 63
4.4 Experimental Evaluation . . . 66

4.4.1 Datasets . . . 67
4.4.2 Implementation . . . 67
4.4.3 Boosting Effect from Adversarial Learning . . . 68
4.4.4 Stabilizing the Training Process . . . 69
4.4.5 Adapting Existing Works into ED//GAN . . . 72
4.5 Summary . . . 74

5 Reference-Conditioned Image Super-Resolution 75
5.1 Introduction . . . 76
5.2 Related Works . . . 78
5.2.1 Deep Learning based Single Image SR . . . 78
5.2.2 Example-based SR Methods . . . 79
5.3 Approach . . . 80
5.3.1 Patch Matching and Texture Swapping . . . 82
5.3.2 Conditional Texture Transfer . . . 83
5.3.3 Content Loss $\mathcal{L}_c$ and Texture Loss $\mathcal{L}_t$ . . . 84
5.4 Dataset . . . 86
5.5 Experimental Results . . . 87
5.5.1 Training Details and Parameters . . . 87
5.5.2 Quantitative Evaluation . . . 88
5.5.3 Qualitative Evaluation by User Study . . . 90
5.5.4 Texture Transfer Results . . . 91
5.5.5 Investigation on Extreme Conditions . . . 91
5.6 Conclusion . . . 94

6 Conclusions and Future Works 95

Bibliography 97

Appendices 115
A The Network Structures in Decoupled Learning . . . 116
A.1 The Network Structure in Adversarial Effect (Sec. 4.4.3) . . . 116

A.2 The Network Structure in Image Transformation (Sec. 4.4.5) 117
A.3 The Network Structure in Image Manipulation (Sec. 4.4.5) 117
B Network Structure of SRNTT 120
C Comparison to Other SR Methods 123
D SRNTT Using References with Different Similarity 128

Vita **129**

List of Tables

2.1 Notation of methods in comparison 27
2.2 Details of convolution and deconvolution networks 28
2.3 Inception scores of each method tested with 10,000 samples on three datasets 30
2.4 Number of missed modes on the compositional MNIST dataset 31

3.1 Survey to compare generated faces and ground truth 51
3.2 Survey to compare our results and those from the best prior works. 53

4.1 Structure of the "Enc" and "Dec" networks. "$m^2 \times n$" denotes the m-by-m maps with n channels. The value 50 indicates a vector with the length of 50. 67
4.2 NRDS on the results partially illustrated in Fig. 4.7 71
4.3 NRDS on the results partially illustrated in Fig. 4.8 71
4.4 NRDS on the results partially illustrated in Fig. 4.10 (left) and Fig. 4.11 (right) 74

5.1 Comparison of different SR methods in PSNR and SSIM. 88
5.2 PSNR and SSIM of SRNTT with different reference levels. 89
5.3 Texture distance to the reference based on Gram matrix. 89
5.4 Gram distance of SRNTT with references of different similarity. 89
5.5 Results of user study: percentage of voting to our methods SRNTT- or SRNTT as compared to other SR methods. 92

A.1 Structure of the Enc and Dec networks as shown in Fig. A.1. The value 50 indicates a vector with the length of 50. Batch normalization is optional as indicated by (BN). The size of each layer is denoted by $h \times w \times c$, corresponding to width, and number of channels, respectively. 117

A.2 Structure of U-Net network as shown in Fig. A.2. Enc and Dec denote the encoding and decoding part in U-Net, respectively. The number of channels of the hidden layers in Dec are twice of those in Enc because of the direct passing. 118

A.3 Structure of the Enc, Dec, G, and D networks as shown in Fig. 4.11. The value 50 indicates a vector with the length of 50.The concatenated age vector y is 50. Structure of the "G" and "D" networks as shown in Fig. A.3. The value 50 indicates a vector with the length of 50. The concatenated age vector y is 50. If use WGAN or LSGAN, the BN is optional in G. 119

B.1 The network structure of content extractor, kernel size is 3×3. This structure is based on the generator of SRGAN [69], thus we could directly adopt the pre-trained model, speeding up the training process. 121

B.2 The network structure of conditional texture transfer, kernel size is 3×3. M^c is obtained from Table B.1, and M^t is obtained from texture swapping based on the relu3_1 layer of VGG19 model. 121

B.3 The network structure of upscaling, kernel size is 3×3. This network is based on the idea of sub-pixel [117]. The input is from the output of Table B.2. . . 121

B.4 The network structure of the discriminator, kernel size is 3×3. The output size is scaled down by stride 2, and the parameter of LReLU is 0.2. In the training dataset, the original image size is 160×160. 122

List of Figures

1.1 Comparison of network structures of the encoding-decoding networks and generative adversarial network (GAN). 7
1.2 Manipulation on the latent vectors [42] 9
1.3 Examples of conditional image manipulation. 9
1.4 Examples of image transformation. 9

2.1 A toy example of Kullback-Leibler divergence. 15
2.2 Comparison of GAN (top) and GAN + encoder (bottom). 20
2.3 Comparison of autoencoder (AE), GAN with autoencoder (GAN+AE) and additionally with $\alpha_{\mathcal{L}}$ (GAN+AE+$\alpha_{\mathcal{L}}$) on the MNIST dataset. 22
2.4 The behavior of $\mathcal{L}_z$ (Eq. 2.20) with different parameters β and γ. 25
2.5 Effect of the regularization on z. 26
2.6 Comparison of different methods on the MNIST dataset. 29
2.7 Comparison of different methods on compositional MNIST dataset. 30
2.8 Comparison of different methods on the CIFAR-10 and face datasets. 32
2.9 Comparison of different methods on the multi-domain dataset. 33
2.10 Comparison of GAN+AE with and without $\alpha_{\mathcal{L}}$. 34

3.1 Assuming the face images lie on a manifold ($\mathcal{M}$), and images are clustered according to their ages and personality by different directions. 38
3.2 Illustration of traversing on the face manifold $\mathcal{M}$. 41
3.3 Structure of the proposed CAAE network for age progression/regression. . . 42
3.4 Illustration of the learned face manifold $\mathcal{M}$. 44
3.5 Effect of D_z, which forces z to a uniform distribution. 45

3.6 Effect of the length of z. 46
3.7 Effect of D_{img}, which forces the generated faces to be more realistic in aspects of age and resolution. 47
3.8 Comparison of different generative models. 48
3.9 Comparison to the ground truth. 50
3.10 Comparison to prior works of face aging. 52
3.11 Comparison to prior work in rejuvenation. 52
3.12 Tolerance to occlusion and variation in pose and expression. 53

4.1 Artifacts introduced by the adversarial effect. Please zoom in for more details. The top raw shows the generated images from ED. The rest rows show the images from the coupled structure (ED plus GAN) with different weights (left column) on the adversarial effect. Assume the weight of reconstruction loss is 1. The generated images with the weight of 0.01 present similar (or a bit higher) fidelity compared to those with the weight of 0.001, but they tend to be unrealistic. The generated images with the weight of 0.1 only present random spots without any boosting on fidelity, indicating instability during the training. 57
4.2 The flow of proposed decoupled learning, i.e., ED//GAN. Enc and Dec are the encoder and decoder networks, and G and D are the generator and discriminator, respectively. L_1 indicates the pixel-level ℓ_1-norm. Solid black arrows denote feedforward path, and dashed arrows in red and blue indicate backpropagation from reconstruction loss (L_1) and adversarial loss (from D), respectively. The reconstruction image is generated from Dec, and residual is yielded from G. The two images are added up to generated the final output image. 59

4.3 Comparison between the ED+GAN and ED//GAN. Left: the existing ED+GAN. Right: the proposed ED//GAN, i.e., decoupled learning. Solid black arrows denote feedforward path, and dashed arrows in red and blue indicate backpropagation from reconstruction loss and adversarial loss, respectively. 62
4.4 Illustration of NRDS. G_n indicates the nth generative model, and its corresponding fake samples are Fake n, which are smapled randomly. The fake samples from n models, as well as the real samples, are used to train the binary classifier D (bottom left). Testing only uses fake samples and performs alternatively with the traing process. The bottom right shows an example of averaged output of D from fake samples of each model. 65
4.5 A toy example of computing NRDS. Left: the real and fake samples randomly sampled from 2-D normal distributions with different means but with the same (identity) covariance. The real samples (blue circle) is with zero mean. The red "x" and yellow "+" denote fake samples with the mean of $[0.5, 0]$ and $[1.5, 0]$, respectively. The notation fake-close/far indicates that the mean of correspondingly fake samples is close to or far from that of the real samples. Right: the curves of epoch vs. averaged output of discriminator on corresponding sets (colors) of samples. 66
4.6 Visualization of the adversarial effect — residual learned by the proposed decoupled learning (ED//GAN). From left to right in each triple: reconstruction, residual, and output images. 69
4.7 Comparison of ED//GAN (top) and ED+GAN (bottom) with batch normalization on ED on UTKFace Dataset. From left to right, the weights of adversarial loss are 0.001, 0.01, 0.1, and 1, respectively. Please zoom in for better view. 70
4.8 Comparison of ED//GAN (top) and ED+GAN (bottom) without batch normalization on ED on UTKFace dataset. From left to right, the weights of adversarial loss are 0.001, 0.01, 0.1, and 1, respectively. 70

4.9 Output of ED//GAN trained on the CUB-200 and Oxford Flower datasets. From left to right in each triple: reconstruction, residual, and output images. 71
4.10 Comparison between Pix2Pix [52] and the modified version using the proposed ED//GAN. Pix2Pix generates images at different weights as denoted by λ:1, where λ and 1 indicate the weights of reconstruction and adversarial losses, respectively. ED//GAN denotes the generated images from modified version. 72
4.11 Comparison between CAAE [148] and the modified version using the proposed ED//GAN. CAAE generates images at different weights as denoted by 1:λ, where 1 and λ indicate the weights of reconstruction and adversarial losses, respectively. ED//GAN denotes the generated images from the modified version. 73

5.1 From left to right: bicubic interpolation, SRCNN that minimizes MSR, SRGAN that further incorporates perceptual-related constraints, the proposed SRNTT conditioned on the reference shown in the upper-right corner, and the reference image for SRNTT. The upscaling factor is 4×. 77
5.2 Overview of the proposed SRNTT. I^{LR} and I^{Ref} are the inputs to SRNTT. I^{LR} is fed to the content extractor, obtaining the content feature map M^c. In parallel, I^{Ref} and I^{LR} are fed to the texture extractor after scale adjustment, yielding a series of feature maps for patch matching and texture swapping. The swapped texture map M^t carries rich texture from I^{Ref} while preserving the content structure of I^{LR}. Finally, M^c and M^t are adaptively fused through the conditional texture transfer, producing I^{SR} with enhanced texture. . . . 81
5.3 Patch matching and texture swapping. 82
5.4 Visualization of patch matching. The left image is the input, and right three images are the reference images. The two columns in the middle are matched patches (zoom-in) between the input and references. The colored boxes and arrows code the corresponding matches. 83
5.5 The pipeline of conditional texture transfer. The transfered feature map is fed to the sub-pixel layers for upscaling. The texture loss is is computed between M^t and the weighted feature map extracted from I^{SR}. 84

5.6 Examples from the CUFED5 dataset. The left column is the HR image for testing. The right columns are corresponding references in five similarity levels, i.e., extra-high (XH), high (H), medium (M), low (L), and extra-low (XL). 87
5.7 Visual comparison to other SR methods. 90
5.8 Percentage of votes to SRNTT as compared to each of other methods. "Voting percentage to SRNTT" means the percentage of users preferring the results of SRNTT over its competitor. 92
5.9 Effect of the texture loss $\mathcal{L}_t$. 92
5.10 The texture is enhanced conditioned on the reference that is shown in the bottom-right corner. Please zoom in for better view. 92
5.11 (a) The extreme references and the corresponding SR results from the proposed SRNTT and SRNTT/M^s which is a trimmed version of SRNTT by removing M^s. (b) Baseline results for comparison purpose. 93
5.12 Comparison with an existing work on the upscaling factor of 8×. Please zoom in for better visualization. 94

A.1 Left: the existing ED+GAN. Right: the adaption based on the proposed ED//GAN, i.e., decoupled learning. Solid black arrows denote the feedforward path, and dashed arrows in red and blue indicate backpropagation from the reconstruction loss and the adversarial loss, respectively. The network details are listed in Table A.1. 116
A.2 Left: the structure of Pix2Pix (ED+GAN). Right: the adaption to the proposed ED//GAN, i.e., decoupled learning. Solid black arrows denote the feedforward path, and dashed arrows in red and blue indicate backpropagation from the reconstruction loss and the adversarial loss, respectively. 118

A.3 Left: the ED+GAN structure used in CAAE [148]. Right: the adaption to the proposed ED//GAN, i.e., decoupled learning. Solid black arrows denote the feedforward path, and dashed arrows in red and blue indicate backpropagation from the reconstruction loss and the adversarial loss, respectively. The age label y is concatenated to z to control the age of generated faces. 118
B.1 Overview of the proposed SRNTT. D denotes the discriminator. 120
C.1 Comparison of different SR methods. 124
C.2 Comparison of different SR methods. 125
C.3 Comparison of different SR methods. 126
C.4 Comparison of different SR methods. 127
D.1 SRNTT results using reference images at different similarity levels. The results of SRGAN are shown as baseline. 128

Chapter 1

Introduction

Deep learning [116] has been widely adopted in many areas, e.g., speech recognition [43, 31, 21, 46], natural language processing [39, 5, 120, 89], image recognition [19, 63, 119, 125, 82, 79, 47], recommendation systems [131, 30, 135], bioinformatics [17, 114, 18], drug discovery [20, 130], etc. Especially, deep learning is deeply and widely explored in most computer vision tasks, e.g., object detection/recognition [63, 119, 79], semantic segmentation [82, 144, 14], target tracking [137, 48, 91], action/gesture recognition [136, 7, 132], etc. Since 2014, when Goodfellow et al. [42] proposed the generative adversarial network (GAN), image synthesis (i.e., image generation and manipulation) has been boosted through the years of 2016 and 2017 after a short fermentation during 2015. As the outpouring of applications on image synthesis, e.g., super resolution [145, 70], domain transformation [107, 52, 121], face aging [148, 1], and image editing [153], GAN has become an off-the-shelf and state-of-the-art approach to achieving higher reality and imaging quality as compared to its predecessors. Since the scope of this work is on image generation and manipulation, the rest of this chapter provides a more thorough background of generative models (Sec. 1.1), especially the GAN (Sec. 1.1.3), as well as image generation and manipulation works (Sec. 1.2) using deep learning.

1.1 Generative Modeling

Generative modeling aims to learn a model that could randomly generate observations with the same distribution to that of the given data. In probability and statistics, generative modeling refers to model the data by a set of parameters, i.e., modeling observations drawn from certain distribution. Mostly, the distribution of the data to be modeled is unknown or difficult to be explicitly determined, e.g., a set of images which distributes in a high-dimensional space with an arbitrarily unknown distribution. Therefore, the generative modeling will learn the mapping from a learned or well known distribution (e.g., Gaussian distribution or uniform distribution) to the data distribution. The learned mapping is referred to as the *generative model*. Generative models are typically probabilistic, specifying a probability distribution over observations, and thus could randomly generate samples similar to those given observations. This random generation process will be dominated by some latent variables, e.g., those drawn from Gaussian distribution.

1.1.1 Generative Models

In machine learning, the basic generative model should be the Naïve Bayes, which learns a parametric model, e.g., Gaussian model, to fit the observations. It requires a pre-defined model to fit the training dataset, which means one has to have prior knowledge on the training set distribution, so to choose an appropriate model. For image generation, however, it is difficult to obtain such a pre-defined parametric model because of the high dimensionality and unknown distribution of the images. More sophisticated generative models are Gaussian mixture model (GMM) and hidden Markov model (HMM), which have been widely adopted in machine learning and computer vision communities. However, they were mostly applied in classification-related applications (i.e., tend to perform as discriminative models), instead of image generation. For example, Reynolds and Rose [108] introduced and motivated the use of Gaussian mixture models (GMM) for speaker identification, and Permuter et al. [99] introduced Gaussian mixture models to classify colored textures in images, with a view to the retrieval of textured color images from databases. The hidden Markov model (HMM) can be represented as the simplest dynamic Bayesian network, which is especially known

for the applications in temporal pattern modeling, e.g., speech [36], handwriting [100, 6], action/gesture recognition [84, 40], etc.

The aforementioned traditional generative models are limited by their modeling ability on high dimension/complexity data, especially unsuitable for direct image modeling. The artificial neural network and its improvement — convolutional neural network (CNN) — has gained extreme momentum on image generation. A representative neural network related generative model is the restricted Boltzmann machine (RBM) and its alternatives, e.g., deep belief networks [71] and autoencoders [133]. The basic neural networks can only handle vectorized data, e.g., feature vectors and 1-D observations. Therefore, an intuitive adaption to image generation is to vectorize the 2-D images, which would destroy the intrinsic structure of the images. The convolutional neural network is specifically designed to deal with 2-D image data and thus widely explored in image generation. Usually, the training of convolutional neural networks is called *deep learning*. This work will discuss the image generation in the scope of deep learning, i.e., convolutional neural networks.

In 2014, Goodfellow et al. [42] proposed the generative adversarial networks (GAN) [42], and then GAN has experienced a tremendous boom since 2016, widely applied in image generation and manipulation. Compared to previous deep-learning-based methods, GAN significantly increased the reality and resolution of generated images. Goodfellow et al. [42] developed the mathematics behind GAN, and then Radford et al. [102] implemented GAN with deep convolutional neural network (DCGAN), leading huge adoption in computer vision applications. In recent two years, most state-of-the-art performance in image generation and manipulation was achieved by employing the GAN. For example, image super resolution [145, 70] and domain transformation [107, 52].

1.1.2 Learning of Generative Models

For simplicity, we divide the generative models into two categories: 1) the traditional models that do not use neural networks and 2) neural network based models, including convolutional neural networks.

The traditional generative models, e.g., Gaussian mixture model (GMM) and hidden Markov model (HMM), learn a set of parameters that determine a random process.

Therefore, the models could generate samples similar to those drawn from the training data distribution. Because of such random process, the same sampling process may yield different results. Usually, the traditional generative models are trained to achieve discriminative purposes, e.g., classification. The GMM mainly maximizes the likelihood between the generated and training samples. It requires to manually decide how many components to use in the learning process, and it will always use all the components it has access to, lacking of flexibility. By the same token, the HMM needs to decide the number of hidden states and also optimize the maximum likelihood. However, each state of HMM could be a GMM, so HMM is supposed to be more flexible than GMM from the perspective of generation distribution. Basically, HMM learns parameters (output and transition probabilities) from data through Baum-Welch a.k.a. forward-backward algorithm [101].

The deep learning based algorithms mostly optimize an objective function through the backpropagation algorithm [111], following the stochastic gradient descent, from which many solvers have been developed to adapt to the descending rate for better optimal solution, e.g., ProximalAdagrad [28], ADADELTA [147], ADAM [59], etc.

1.1.3 Generative Adversarial Networks

Generative adversarial networks (GAN) was first introduce by Goodfellow et al. [42] in 2014, while the similar idea of adversarial training dates back to 1992 by Schmidhuber [115]. In recent two years, GAN has demonstrated great potentials in a handful of generative tasks, e.g., random generation of natural images [22, 102] (DCGAN, LAPGAN) or 3-D objects [142], image generation from text [87, 107], super-resolution [145, 70], image manipulation (morphing) [139, 153], etc. In all these applications, GANs (or its variants) have shown appealing performance in generating realistic and sharp looking images. The GAN is generally implemented by a system of two neural networks contesting with each other in a minimax game, in which one player (i.e., a neural network) aims to discriminates real and fake (i.e., generated) samples, and the other player aims to generate samples that are fake but similar to real ones in order to fool the adversary. The former is called the *discriminator* and the latter is called the *generator*. Eventually, the two players will achieve the Nash equilibrium, the discriminator cannot better distinguish real samples from those generated

by the generator, and the generator cannot further fool the discriminator. Therefore, the generator could render samples similar to the real ones. Different from [115] and the other generative models, GAN is based on a minimax game that terminates at a saddle point rather than a traditional optimization problem. The objective of GANs [42] is

$$\min_G \max_D \mathbb{E}_{x\sim p_x} [\log D(x)] + \mathbb{E}_{z\sim p_z} [\log(1 - D(G(z)))] , \tag{1.1}$$

where p_x denotes the distribution of real data $x \in \Omega_x$, and p_z is a prior distribution of the latent variable $z \in \Omega_z$, which is generally assumed to be white noise. The functions $D : \Omega_x \to [0, 1]$ and $G : \Omega_z \to x$ denote the discriminator and generator, respectively. Given samples from a prior distribution $z \sim p_z$, they are mapped to the data space by $G(z) = x$. The discriminator $D(x)$ distinguishes the real data $x \sim p_x$ from generated data $G(z) = x \sim p_g$, where p_g denotes the distribution of the generated data. The goal is to learn p_g, achieving $p_g = p_x$ so that the discriminator cannot distinguish the generated data from real ones.

During the training of GAN, D and G are updated by alternatively maximizing $\mathcal{L}(D)$ and minimizing $\mathcal{L}(G)$ where

$$\mathcal{L}_D = \mathbb{E}_{x\sim p_x} [\log D(x)] + \mathbb{E}_{x\sim p_g} [\log(1 - D(x)] , \tag{1.2}$$

$$\mathcal{L}_G = \mathbb{E}_{z\sim p_z} [\log(1 - D(G(z))] . \tag{1.3}$$

Fixing the generator G, the optimal discriminator is

$$D^*(x) = \frac{p_x(x)}{p_x(x) + p_g(x)}. \tag{1.4}$$

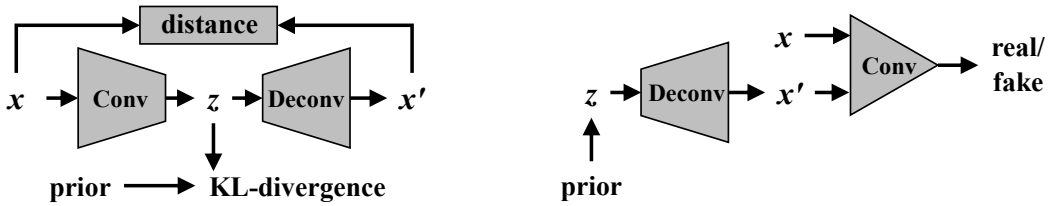
The proof can be found in [42]. Then, GAN tends to minimize the Jensen-Shannon (JS) divergence, which is a symmetric metric. In recent years, GAN is becoming an important algorithm widely adopted in image generation related works. This work will further discuss the properties, drawbacks, and improvements of GAN in Chapter 2.

1.2 Image Generation and Manipulation

1.2.1 Image Generation

Image generation based on deep learning was initially realized by encoding-decoding networks, e.g., autoencoder and variational autoencoder (VAE) [61]. They mostly generate images conditioned on certain input images. During the training stage, an objective function (e.g., Euclidean distance or ℓ_1-norm) is minimized straightforwardly. The main drawback of encoding-decoding networks is to generate blurry images, and improving the objective functions or network structure can hardly break the bottleneck. Therefore, image generation did not attract so much attention as after the invention of GAN, which significantly boosts the quality of generated images in aspects of reality and resolution. Because of the effectiveness of the adversarial training mechanism, GAN has become a hot research topic and widely applied in many areas and applications.

In encoding-decoding networks, the image generation process is usually implemented by a convolutional network and a deconvolutional network, which are concatenated to each other. As shown in Fig. 1.1a, where "Conv" and "Deconv" denote the convolutional network and deconvolutional network, respectively. x and x' are the input and generated images, respectively. z indicates a vector. The convolutional network yields high-level features z from input images x, and the deconvolutional network generates images x' from those high-level features z. Then, the distance between input and generated images is calculated to train the network to generate desired images. The GAN has also been implemented by the convolutional and deconvolutional networks but in a different cooperation between the two networks. The first implementation of GAN using deep convolutional networks was proposed by Radford et al. [102] in 2016, which has led to a boost of deep learning based image generation. Different from encoding-decoding networks, as shown in Fig. 1.1b, GAN generates images by a deconvolutional network from a latent vector z randomly drawn from certain distribution, e.g., Gaussian or uniform distribution. The latent vector is equivalent to the high-level feature in encoding-decoding networks. Then, a convolutional network is applied to distinguish input images x from those generated images x', rather than yielding high-level feature as in encoding-decoding networks.



(a) The encoding-decoding network **(b)** The network structure of GAN

Figure 1.1: Comparison of network structures of the encoding-decoding networks and generative adversarial network (GAN).

Since the invention of GAN, it has flooded image generation related areas in the computer vision community. Many alternatives of GAN have been proposed either to improve GAN in terms of the quality of generated images, or to modify the structure of GAN to achieve more tacks. For example, the works published in the years of 2016 and 2017 [113, 90, 152, 88, 4, 49] improved either training method or the objective function of GAN to achieve stable training or more photo-realistic images. In application-oriented works [87, 107, 145, 70, 139, 153], GAN is mostly incorporated with an encoder-decoding network to generate specific images instead of random generation in GAN.

1.2.2 Image Manipulation

Image manipulation is to alter or transform an image to achieve desired results. This work will only focuses on deep learning based image manipulation. Generally, deep learning based methods do not manipulated images in the pixel space, instead, they traverse in the latent vector space to achieve image morphing, or concatenating different conditions to the latent vector to yield conditioned images. Such latent vectors are mostly input to a deconvolutional network as shown in Fig. 1.1 (the latent vector z). The latent vectors could be high-level features obtained from a convolutional network like in encoding-decoding networks (Fig. 1.1a) or random draws as in GAN (Fig. 1.1b). A common way of manipulating the latent vector is to interpolate between two anchor points in the latent space and feedforward the interpolated points through a pre-trained deconvolutional network (e.g., a generator) to yield morphing between the images corresponding to the two anchor points. Therefore, images could be manipulated by traversing in the latent space as shown in Fig. 1.2.

A more empirical way of image manipulation is to concatenate conditions to the latent vectors. For example, Reed et al. [107] generated images conditioned on given text, Zhang et al. [148] rendered face images based on conditioned age and gender, and Zhu et al. [153] generated images based on manual edit on the original images. Fig. 1.3 shows some examples from existing works.

Another image manipulation is to directly learn a mapping from the original image to desired results. It may not involve manipulation of or conditional concatenation to the latent vectors. Usually, such manipulation is referred to as image transformation. Isola et al. [52] transfered image from one domain to another, e.g., from images to sketches, or from color images to gray scale images. Ledig et al. [70] generate high-resolution images from correspondingly low-resolution ones. Some examples are displayed in Fig. 1.4.

1.3 Motivation and Contribution

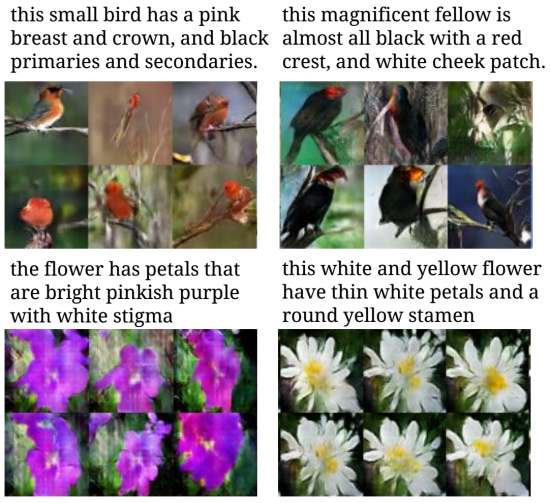
This work will focus on GAN-based image synthesis, which have been empirically explored in many areas and various of applications. A common framework is to incorporate GAN with autoencoding networks, and it has been demonstrated to be more effective and stable than the vanilla GAN. However, the lack of theoretical reasoning shadows the significance of those empirical achievements. Moving away from application-oriented derivations as in most existing works, Chapter 2 analyzes the intrinsic properties and drawbacks of GAN and discusses the empirical remedies in a more theoretical manner.

Based on GAN, which originally targets at image generation in a random fashion, Chapter 3 further investigates the conditional image generation that is better motivated by real-world applications like those aforementioned tasks. Specifically, this work develops a novel network structure for human face age progression and regression, which first achieves bidirectional face manipulation with respect to the age. More importantly, the generated faces from the proposed method are of higher reality and resolution compared to existing works.

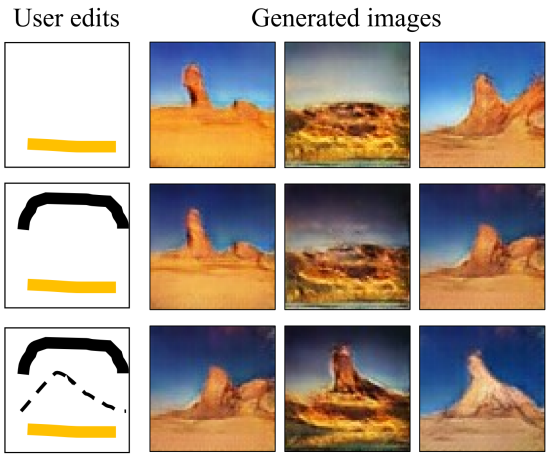
Observe that most application-oriented methods incorporate autoencoder (or conditional autoencoder) with GAN, including the face aging method described in Chapter 3, Chapter 4



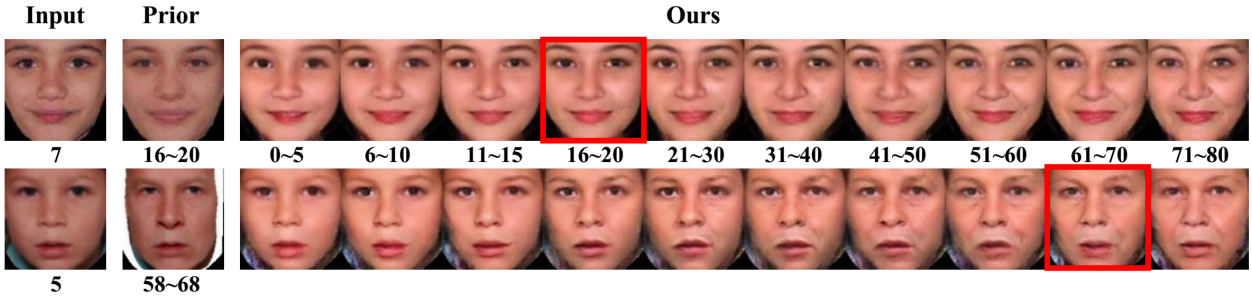
Figure 1.2: Manipulation on the latent vectors [42]



(a) Generating images from text [107]

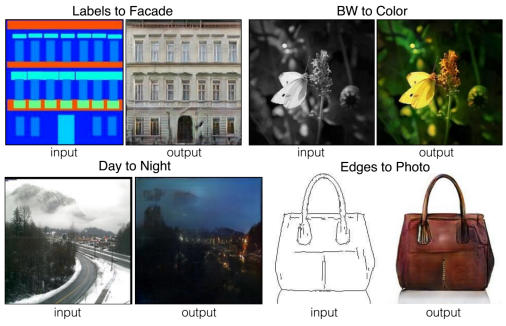


(b) Generating image based on edit [153]

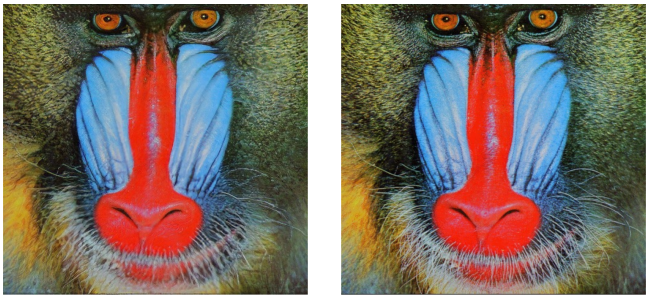


(c) Generating faces conditioned on age [148]

Figure 1.3: Examples of conditional image manipulation.



(a) Domain transfermation [52]



(b) Image super resolution [70]. Left: 4× super-resolved image. Right: original image.

Figure 1.4: Examples of image transformation.

gives the first attempt to investigating the drawback of such composite and proposes the decoupled learning to solve the potential instability issue.

Chapter 5 introduces the idea of conditional image synthesis in image super-resolution, which traditionally recovers higher-resolution image from its low-resolution counterpart. We further incorporate a reference image (high-resolution) as the condition to synthesize the high-resolution image by transferring rich and real texture from the reference. This method would break performance barrier of image super-resolution, i.e., recover finer and more realistic details.

Chapter 2

Instability of GAN and Stabilization

The GAN has become an important and widely adopted algorithm in image generation and manipulation. Many recent works cooperated GANs with autoencoding networks to generate photo-realistic images conditioned on input images. They were mostly driven by specific tasks — generating images from certain condition instead of noise. However, the merit of incorporating autoencoding nets is far more than introducing conditions. This chapter aims to demonstrate those hidden non-trivial merits of incorporating autoencoding nets to GAN, i.e., relaxing the mode missing problem where large volumes of probability mass tend to collapse onto a few major modes, and potentially stabilizing the training of generator. We provide theoretical reasonings on how GAN is powered by autoencoding to relax the two compelling issues. We further show that by adding an adaptive decay variable to the adversarial error, the instability issue caused by competition between the generator and discriminator is largely alleviated.

Several empirical methods (e.g., incorporating encoding-decoding structure or designing specific training strategies) have been proposed to resolve these issues. However, theoretical derivation and remedy to these drawbacks remains an opening topic. Experimental results on compositional digits, natural images, and faces all show superior performance of autoencoding-based GAN in handling the mode missing and instability problem.

2.1 Introduction

Despite the great potential, the vanilla (non-conditional) GANs are still generally considered very difficult to train with undamped oscillations often occur during the convergence process; and on top of that, they suffer from the so-called mode missing problem, where large volumes of probability mass tend to collapse onto a few modes [12]. In recent years, multiple works have been proposed to deal with the problem of mode missing and instability during training. Generally speaking, those methods can be divided into two categories: 1) autoencoding-based methods and 2) strategy-based methods.

In autoencoding-based methods, [113] proposed feature matching that utilized an intermediate layer of the discriminator (behaves like an encoder) to minimize the feature distance between the real and generated data. [67] combined GAN with VAE [61], thus utilizing reconstruction error between the real and generated data to regulate the training. Similarly, [152, 24, 12] incorporated the autoencoding structure into GANs to avoid mode missing as well as stabilize the training procedure. [72] proposed an alternative way of learning a generative model by maximum mean discrepancy (MMD). All of these works share the similar idea — directly matching the statistics of generated data to that of the real data by introducing an autoencoder-like architecture. In the original GANs, however, the statistics of generated data is matched to that specified by a discriminator (indirect estimate of the real data distribution), which has been demonstrated to cause mode missing [90, 12].

The strategy-based methods preserve the basic structure of GANs but improve the way how the generator and/or discriminator is updated. For example, [3] suggested to add continuous noise to the input of discriminator, which could fix the instability and vanishing gradients issues. [90] unrolled the iterative updating procedure of GANs and approximated the discriminator toward optimal when updating the generator. Thus, ideally, the GAN may minimize the Jason-Shannon divergence, solving the mode missing problem. [29] extended GANs to multiple discriminators and trained the generator against the best available discriminator. In this process, they also tried to obtain more optimal discriminator while updating the generator. [113] introduced an extra layer into the discriminator to

coordinate all samples with a mini batch, instead of considering those samples as independent like in the original GANs. This method avoided the collapse of the generator.

Although these solutions boosted the performance of GANs to various degrees, they tend to be heuristic-driven that lack theoretical reasoning. Therefore, the effectiveness of these solutions can only be validated through empirical study. From a more theoretical perspective, we ask ourselves the question why incorporating an autoencoder to GAN would solve the mode missing problem.

This chapter attempts to answer this question with more rigorous theoretical analysis [149]. The rest of the chapter is organized as follows. Section 2.2 provides general background on GANs and the theoretical reasoning on the cause of the mode missing problem based on study from [3]. Section 2.3 elaborates on that incorporating an autoencoder alike structure avoid mode missing. A simple but effective method of decaying the undamped oscillation as the generator approaching optimum is proposed in section 2.4. Experimental result are shown in section 2.6.

2.2 Drawbacks of GAN

We first review the basic objective of GANs and its relation to Kullback-Leibler (KL) divergence (Sec. 2.2.1). Then, we explain the cause of mode missing (Sec. 2.2.2) and instability (Sec. 2.2.3) from a theoretical perspective based on [3].

The objective of GAN [42] is (recall Sec. 1.1.3)

$$\min_G \max_D \mathbb{E}_{x\sim p_x}\left[\log D(x)\right] + \mathbb{E}_{z\sim p_z}\left[\log(1 - D(G(z)))\right], \tag{2.1}$$

where p_x denotes the distribution of real data $x \in \Omega_x$, and p_z is a prior distribution of the latent variable $z \in \Omega_z$, which is generally assumed to be white noise. The functions $D : \Omega_x \to [0, 1]$ and $G : \Omega_z \to x$ denote the discriminator and generator, respectively. Given samples from a prior distribution $z \sim p_z$, they are mapped to the data space by $G(z) = x$. The discriminator $D(x)$ distinguishes the real data $x \sim p_x$ from generated data $G(z) = x \sim p_g$, where p_g denotes the distribution of the generated data. The goal is to learn

p_g, achieving $p_g = p_x$ so that the discriminator cannot distinguish the generated data from real ones.

2.2.1 Relation to Kullback–Leibler Divergence

In GAN, D and G are updated by alternatively maximizing $\mathcal{L}(D)$ and minimizing $\mathcal{L}(G)$ where

$$\mathcal{L}_D = \mathbb{E}_{x\sim p_x}\left[\log D(x)\right] + \mathbb{E}_{x\sim p_g}\left[\log(1 - D(x)\right], \tag{2.2}$$

$$\mathcal{L}_G = \mathbb{E}_{z\sim p_z}\left[\log(1 - D(G(z))\right]. \tag{2.3}$$

Fixing G, the optimal discriminator has the form

$$D^*(x) = \frac{p_x(x)}{p_x(x) + p_g(x)}. \tag{2.4}$$

The proof can be found in [42]. Plugging Eq. 2.4 into Eq. 2.2, $\mathcal{L}_{D^*} = 2JS(p_x\|p_g) - 2\log 2$, where $JS(\cdot) \in [0, 1]$ computes the Jensen-Shannon (JS) divergence. When $p_x = p_g$, $\mathcal{L}_{D^*}$ yields the minimum, $-2\log 2$, indicating that the generator approaches its optimal. Therefore, the general strategy for generative modeling is to minimize JS divergence. Note that JS divergence is symmetric, which is constructed by the asymmetric KL divergence.

Ideally, the optimal G could be obtained by minimizing the JS divergence with respect to G when D is optimal. In the alternative updating manner, however, D is fixed when updating G, so D cannot be guaranteed optimal as G is updated. Assuming the optimal discriminator $D^*(x)$, $\mathcal{L}_G$ can be rewritten as

$$\begin{aligned}
\mathcal{L}_G =& \mathbb{E}_{x\sim p_g}\left[\log \frac{p_g(x)}{p_x(x) + p_g(x)}\right] \\
=& \int_{\Omega_x} p_g(x) \log \frac{p_g(x)}{p_x(x) + p_g(x)} dx \\
=& KL\left(p_g \middle\| \frac{p_x + p_g}{2}\right) - \log 2.
\end{aligned} \tag{2.5}$$

where $KL(\cdot)$ denotes the KL divergence. The generator aims to minimize $\mathcal{L}_G$, which is equivalent to minimizing the KL divergence between p_g and p_x when D is temporarily optimal. However, KL divergence is an asymmetric measurement, which is the intrinsic cause of the mode missing problem (detailed in Sec. 2.2.2).

2.2.2 Mode missing

As discussed in Sec. 2.2.1, the optimal G given D^* minimizes the JS divergence between p_x and p_g, which is the general objective of generative modeling. However, the loss function of G (shown in Eq. 2.5) only measures the asymmetric KL divergence instead of the symmetric JS divergence. Fig. 2.1 illustrates the asymmetric behavior of KL divergence. The dashed blue curve and dotted red curve represent $p_g(x)$ and $p_x(x)$, respectively. The data distribution p_x has two modes. The solid black curve displays the derivative of the Kullback-Leibler divergence at each x. If $p_g(x) > p_x(x)$, $\nabla_x KL\left(p_g \| (p_x + p_g)/2\right)$ is large; otherwise, if $p_g(x) < p_x(x)$, it is small. Because the generator tends to minimize Eq. 2.5, the cost $\mathcal{L}_G$ is much higher when $p_g(x) > p_x(x)$ than that when $p_g(x) < p_x(x)$.

Intuitively, $p_g(x) > p_x(x)$ indicates that a generated sample x does not look real, while $p_g(x) < p_x(x)$ implies that a real data sample is with low probability of being from the generator. In essence, Eq. 2.5 punishes more on generating unrealistic samples than on mode missing, which results in the problem of mode missing. This is a deficiency caused

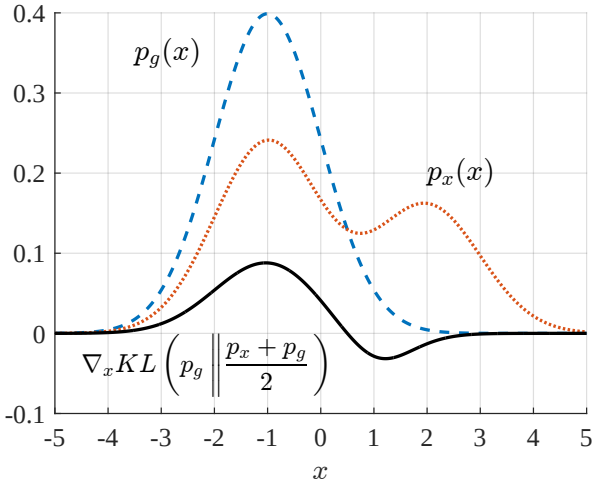


Figure 2.1: A toy example of Kullback-Leibler divergence.

intrinsically by the design of objective function of the generator. In practice, Eq. 2.6 is used instead of Eq. 2.5 since it is difficult to guarantee an optimal D and derive well formated loss like Eq. 2.5. In addition, p_x and p_g are intractable because we could only observe samples extracted from them, rather than the exact distribution. In Eq. 2.6, $p_d(y=0|x)$ denotes the probability of a given x is discriminated as fake ($y=0$). Minimizing $p_d(y=0|x\sim p_g)$ forces $x\sim p_g$ to be like real samples. Still, mode missing is not sufficiently punished in this loss function.

$$\mathcal{L}_G = \mathbb{E}_{x\sim p_g}[\log(p_d(y=0|x))], \tag{2.6}$$

2.2.3 Instability in Training

In this section, we discuss and tackle one component of the general instability issue of GAN, i.e., the unstable update of the generator during training, namely, gradient vanishing and exploring, and undamped oscillation as GAN approaches the saddle point, where the two losses in Eqs. 2.2 and 2.3 achieve the Nash equilibrium[1].

Fixing the discriminator $D(x)$ which is differentiable, the gradient of generator $G(z)$ with respect to its parameter θ is

$$\begin{aligned}\nabla_\theta \mathcal{L}_G =& \nabla_\theta \mathbb{E}_{z\sim p_z}\left[\log(1-D(G_\theta(z)))\right] \\ =& \mathbb{E}_{z\sim p_z}\left[\frac{1}{D(G_\theta(z))-1}\frac{\partial D(G_\theta(z))}{\partial G_\theta(z)}\frac{\partial G_\theta(z)}{\partial \theta}\right] \\ =& \mathbb{E}_{x\sim p_g}\left[\frac{\nabla_x D(x) J_\theta G_\theta}{D(x)-1}\right],\end{aligned} \tag{2.7}$$

where $J_\theta G_\theta$ denotes the Jacobian of G w.r.t. its parameter θ, and $\|\mathbb{E}_{x\sim p_g}[J_\theta G_\theta]\|_2$ is bounded for a differentiable generator. If D is a perfect discriminator $D^\dagger$, $D^\dagger(x)|_{x\sim p_x} = 1$ and $D^\dagger(x)|_{x\sim p_g} = 0$. Note that a perfect discriminator is an optimal discriminator, but an optimal discriminator is not necessarily a perfect one. When D is approaching $D^\dagger$ and

[1] In game theory, the Nash equilibrium is a solution concept of a non-cooperative game involving two or more players in which each player is assumed to know the equilibrium strategies of the other players, and no player has anything to gain by changing only his own strategy [97].

$x \sim p_g$,

$$\lim_{D \to D^\dagger} D(x) = 0, \tag{2.8}$$

$$\lim_{D \to D^\dagger} \nabla_x D(x) = 0. \tag{2.9}$$

Therefore,

$$\lim_{D \to D^\dagger} \mathbb{E}_{x \sim p_g} \left[\frac{\nabla_x D(x) J_\theta G_\theta}{D(x) - 1} \right] = 0, \tag{2.10}$$

which corresponds to the condition of *gradient vanishing*, where the generator is trapped into a bad state without update. On the contrary, if the generator achieves optimum G^* given $D^\dagger$, $D^\dagger(G^*(z)) = 1$. Then,

$$\lim_{G \to G^*} D(x) = 1, \text{ and } \|\mathbb{E}_{x \sim p_g}[\nabla_x D(x)]\|_2 > 0. \tag{2.11}$$

Because we assume G is optimal, making $D(x) > 0$ where $x \sim p_g$, $\nabla_x D(x)$ is non-zero. Thus,

$$\lim_{G \to G^*} \mathbb{E}_{x \sim p_g} \left[\frac{\nabla_x D(x) J_\theta G_\theta}{D(x) - 1} \right] = \infty, \tag{2.12}$$

which indicates the phenomenon of unstable update of G when it is approaching the optimum. Because G_θ is assumed to be differentiable on θ, the optimal G is achieved at certain θ, where the gradient w.r.t. θ is not necessary to be zero although the generated images from G_θ make D yield 1. In practice, D and G are updated alternatively based on mini-batch learning. Therefore, the discriminator is more likely optimal rather than perfect. For a mini-batch, however, the discriminator may be locally perfect, then the behavior of instability would deteriorate the generator. Therefore, starting to train the generator after training a perfect discriminator causes gradient explosion, which partly explains why alternative update of D and G is suggested in [42].

Now, assume $D = D^*$ and $G = G^*$ after epochs of training, then $p_g = p_x$ and $D(x) = 1/2$ (Eq. 2.4). Under this condition, we expect $\nabla_\theta \mathcal{L}_G = 0$. However, $\mathbb{E}_{x \sim p_g}[D(x)] = 1/2$ and $\|\mathbb{E}_{x \sim p_g}[\nabla_x D(x)]\|_2 > 0$. Therefore, $\nabla_\theta \mathcal{L}_G \neq 0$ that forces the generator to keep being updated towards $D(x)|_{x \sim p_g} = 1$. Simultaneously, the discriminator tries to make $D(x)|_{x \sim p_g} = 0$. In

this game, $\mathbb{E}_{x\sim p_g}[D(x)]$ will present undamped oscillation, which can be observed from the updated gradient of G.

2.3 GAN Incorporated with Autoencoder

Given the optimal discriminator $D^*(x)$, the original objective of G (Eq. 2.3) is equivalent to minimizing KL divergence. As discussed in Sec. 2.2.2, the mode missing is mainly caused by the weak penalty on the case of $p_g(x) < p_x(x)$. Let's first consider another KL divergence that emphasize the cost on mode missing, i.e., minimizing $KL(p_x\|p_g)$ which punishes more on mode missing than on unreality. From the original formulation of KL divergence,

$$\begin{aligned} KL(p_x\|p_g) &= \int_{\Omega_x} p_x(x)\log\frac{p_x(x)}{p_g(x)}dx \\ &= \int_{\Omega_x} p_x(x)\log p_x(x)dx - \int_{\Omega_x} p_x(x)\log p_g(x)dx \\ &= \mathbb{E}_{x\sim p_x}\left[\log p_x(x)\right] - \mathbb{E}_{x\sim p_x}\left[\log p_g(x)\right], \end{aligned} \tag{2.13}$$

where $\mathbb{E}_{x\sim p_x}\left[\log p_x(x)\right]$ could be considered as a constant because the unknown data distribution is fixed. We need to update G to maximize $\mathbb{E}_{x\sim p_x}\left[\log p_g(x)\right]$, which fits p_g to p_x without mode missing. However, we only have observations drawn from p_x and p_g instead of analytical expression, so direct comparison between p_x and p_g is intractable.

An intuitive way of comparing two unknown distributions would then be the Monte Carlo method, which estimates distance of two arbitrary distributions by repeated random sampling. A direct measurement is maximum mean discrepancy (MMD), which measures the kernel-based distance of every sample pair within and between p_x and p_g. In other word, MMD blindly computes the distance of all sample pairs. Yet a more effective measurement is to compare the sample pairs assigned by the Hungarian method, where average distance between non-repetitive pairs is minimized.

Assume two large but finite sample sets $\mathbb{X}_x$ and $\mathbb{X}_g$, of the same size, n, randomly drawn from p_x and p_g, respectively. Suppose the Hungarian assignment function is $\mathcal{H} : \mathbb{X}_x \to \mathbb{X}_g$ based on a distance metric $\mathcal{L}(x, \mathcal{H}(x))$, $x \in \mathbb{X}_x$ and $\mathcal{H}(x) \in \mathbb{X}_g$, such that $\mathbb{E}_{x\in\mathbb{X}_x}[\mathcal{L}(x, \mathcal{H}(x))]$

is minimized. Then the distance between p_x and p_g can be measured by

$$\mathbb{E}_{x \in \mathbb{X}_x}[\mathcal{L}(x, \mathcal{H}(x))]. \tag{2.14}$$

Ideally, if $p_x = p_g$, Eq. 2.14 achieves its minimum. In practice, the training dataset could be considered as $\mathbb{X}_x$, and $\mathbb{X}_g$ consists of the generated samples. In GANs, $G(z)$ generates $\mathbb{X}_g$ from random sampling, $z \sim p_z$. However, in mini-batch learning, the size of $\mathbb{X}_x$ is limited (e.g., 50 or 100), which cannot sufficiently represent the true distribution of p_x. In addition, both $x \in \mathbb{X}_x$ and $x_g \in \mathbb{X}_g$ are sampled through stochastic process, so the samples in batches may diverse significant although the they are drawn from the same distribution. Therefore, a limited number of random samples would result in larger average distance from Eq. 2.14. Furthermore, the computational complexity of Hungarian matching is $O(n^3)$. Therefore, it will be more efficient if the Hungarian assignment function $\mathcal{H} : \mathbb{X}_x \to \mathbb{X}_g$ is learned through a network.

In many existing works, $\mathcal{H}(x)$ is interpreted as an encoding-decoding structure where the input-output pairs would result in average minimum distance, and $\mathcal{L}(x, \mathcal{H}(x))$ may be interpreted as the reconstruction error. For example, [67, 12] concatenated an encoder to the generator, which together forms an autoencoder that is equivalent to the functionality of $\mathcal{H}(x)$. [113] reused the discriminator network as an encoder to compare higher level feature of the real and generated data, in which the discriminator also performed like $\mathcal{L}(x, \mathcal{H}(x))$. [152] replaced the discriminator by an autoencoder. Although many related works have verified the effectiveness of incorporating autoencoder-like structure to the GAN, the discussion is rather heuristic and empirical. Here, we provide theoretical reasoning of the effectiveness from the perspective of avoiding mode missing.

For the case of autoencoder, $\mathcal{H}(x) = G(E(x))$, where $E(x) = z$ denotes an encoder. Rewriting Eq. 2.14, the equivalent tractable objective is

$$\mathbb{E}_{x \sim p_x}\left[\mathcal{L}(x, G(E(x)))\right]. \tag{2.15}$$

Adding Eq. 2.15 to the original formulation of GANs (Eq. 2.1), we obtain the new objective function that penalizes both unrealisticness as well as mode missing,

$$\mathbb{E}_{x\sim p_x}[\log\left(D(x)(1-D(\mathcal{H}(x)))\right)+\lambda\mathcal{L}(x,\mathcal{H}(x))], \tag{2.16}$$

where λ balances the effect of reconstruction error, and $\mathcal{H}(x)=G(E(x))$. Fig. 2.2 illustrates the effect of adding an encoder to GAN. From left to right are results after training by 200, 400, 600, 800, and 1000 batches. The real data distribution (blue dotted) is mixed Gaussian with two modes. The red curve shows the distribution of generated data.

2.4 Adaptive Decay on Adversarial Effect

Compared to the objective of GANs, Eq. 2.16 appends an extra regularization $\mathbb{E}_{x\sim p_x}[\mathcal{L}(x,\mathcal{H}(x))]$, whose gradient with respect to the generator is non-zero if we do not consider overfitting. Therefore, the problem of gradient vanishing, as analyzed in Eq. 2.10, is naturally resolved because the gradient with respect to the generator in Eq. 2.16 is always non-zero even if $D\rightarrow D^{\dagger}$.

For the problem of instability, it tends to occur when G approaches G^* as shown in Eq. 2.12, which makes $\|\mathbb{E}_{x\sim p_g}[D(x)-1]\|_2$ approaching 0, resulting in gradient explosion. In addition, even though the discriminator and generator both achieve optimum, the loss of G is not zero, causing the undamped oscillation. To solve this problem, we introduce a dynamic

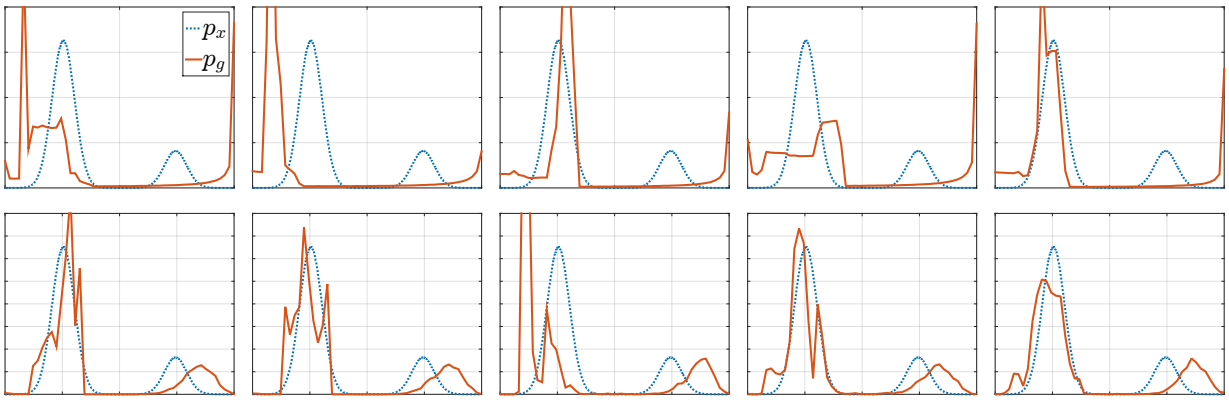


Figure 2.2: Comparison of GAN (top) and GAN + encoder (bottom).

variable $\alpha_{\mathcal{L}} \leq 1$ whose value will reduce as G approaches its optimum — $\mathbb{E}_{x\sim p_x}[\mathcal{L}(x, \mathcal{H}(x))]$ approach zero. Then, Eq. 2.16 can be rewritten as

$$\mathbb{E}_{x\sim p_x}[\log(D(x)(1 - \alpha_{\mathcal{L}} D(\mathcal{H}(x)))) + \lambda\mathcal{L}(x, \mathcal{H}(x))], \tag{2.17}$$

where $\alpha_{\mathcal{L}} = \min\{\alpha\mathbb{E}_{x\sim p_x}[\mathcal{L}(x, \mathcal{H}(x))], 1\}$, $\alpha \in [0, \infty)$. Intuitively, if $\alpha = 0$, the effect of discriminator is muted, then the network functions as an autoencoder. If α approaches infinity, it becomes a normal GAN+AE. When updating the generator, $\alpha_{\mathcal{L}}$ is considered a constant computed before the update. Then, the gradient of generator with parameter θ is

$$\Delta\theta = \mathbb{E}_{x\sim p_x}\left[\frac{\alpha_{\mathcal{L}}\nabla_\theta D(\mathcal{H}(x))}{\alpha_{\mathcal{L}} D(\mathcal{H}(x)) - 1} + \lambda\nabla_\theta\mathcal{L}(x, \mathcal{H}(x))\right].$$

For differentiable D and G, $\nabla_\theta D(\mathcal{H}(x)) = C_1$ is finite, and $D(\mathcal{H}(x)) = C_2 \in [0, 1]$. Because the extreme case $D(\mathcal{H}(x)) = 1$ yields infinite $\Delta\theta$ (Eq. 2.12), an $\alpha_{\mathcal{L}} < 1$ could suppress the unstable update driven by the discriminator.

The more general instability is the undamped oscillation when both D and G approach their optima, then $D(x) \approx D(\mathcal{H}(x)) \approx 1/2$. In addition, $\nabla_\theta\mathcal{L}(x, \mathcal{H}(x)) = \epsilon$, where ϵ is a small positive value. We thus have

$$\Delta\theta|_{\theta=\theta^*} = \mathbb{E}_{x\sim p_x}\left[\frac{\alpha_{\mathcal{L}} C_1}{\alpha_{\mathcal{L}} C_2 - 1} + \epsilon\right]. \tag{2.18}$$

In vanilla GAN or GAN+AE ($\alpha_{\mathcal{L}} = 1$), $\Delta\theta \neq 0$, although $p_x = p_g$. Then, the training process will arrive at the Nash equilibrium and present undamped oscillation on $\Delta\theta$. On the contrary, by incorporating $\alpha_{\mathcal{L}}$, which may decrease exponentially with $\mathcal{L}(x, \mathcal{H}(x))$, the gradient flows from the discriminator will be significantly suppressed towards zero.

For the update of discriminator with parameter ϕ,

$$\Delta\phi = \mathbb{E}_{x\sim p_x}\left[\frac{\nabla_\phi D(x)}{D(x)} + \frac{\alpha_{\mathcal{L}}\nabla_\phi D(\mathcal{H}(x))}{\alpha_{\mathcal{L}} D(\mathcal{H}(x)) - 1}\right],$$

which approaches infinite if $D(x)|_{x\sim p_x} \to 0$. This would effectively discourage the strategy of training the generator for a long time without updating the discriminator. To ensure symmetric objective, $\alpha_{\mathcal{L}} = 1$ ($\alpha \to \infty$) when updating the discriminator.

The effect of $\alpha_{\mathcal{L}}$ is shown in Fig. 2.3. They start with the same initial parameters. In GAN+AE+$\alpha_{\mathcal{L}}$, $\alpha_{\mathcal{L}}$ is activated when $\mathcal{L}(x, \mathcal{H}(x)) < 10$. The original AE converges slower than GAN+AE, and $\alpha_{\mathcal{L}}$ boosts the convergence. In addition, $\alpha_{\mathcal{L}}$ stabilizes the update of generator.

2.5 Training Strategy

Ideally, an optimal discriminator should be obtained first, and then a generator is trained based on it. In the original GAN, this idea is infeasible because the optimal discriminator cannot be achieved without sufficient high quality generated data. In addition, a perfect discriminator results in gradient vanishing. Therefore, the discriminator and generator have to be updated alternatively from the very beginning. However, with the auxiliary of autoencoder (reconstruction error), the generator can fit p_g to p_x without participation of the discriminator. From the empirical study in [67], p_g from autoencoder is a smoothed version of p_x, e.g., the image from an autoencoder looks like blurred. We refer the smoothed version of p_x as the *autoencoder mean*. At the same time, the discriminator is trained to

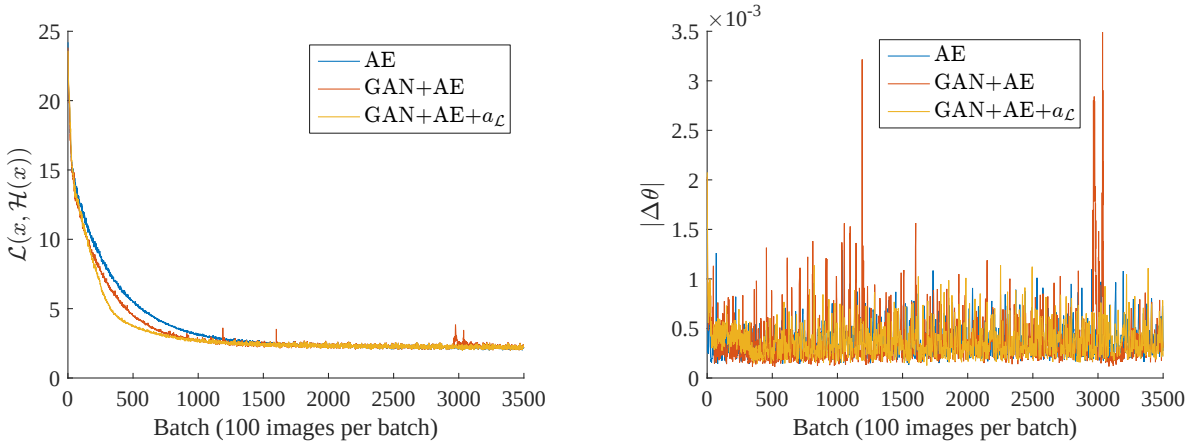


Figure 2.3: Comparison of autoencoder (AE), GAN with autoencoder (GAN+AE) and additionally with $\alpha_{\mathcal{L}}$ (GAN+AE+$\alpha_{\mathcal{L}}$) on the MNIST dataset.

distinguish the real data and generated data (autoencoder mean). Because the real data and autoencoder mean only differ in texture (sharpness), the discriminator will mainly learn this difference. The next stage is to alternatively update the generator and discriminator like GANs. In this stage, the texture learned by the discriminator is fed back to the generator, making it generate more realistic and high-fidelity samples. Our training strategy is detailed in Algorithm 2.1.

2.5.1 Regularization on z

The GAN draw z from a prior, e.g., Gaussian or uniform distribution. In the VAE and most adversarial generative models with autoencoder-like architecture, the latent variable z is also assumed to be Gaussian, mixed Gaussian, or uniform. However, z could be an arbitrary distribution [2], which could be decided in an unsupervised manner during the training process. In generative modeling, the distribution of z is supposed to fill the latent space of z, ideally evenly spread because it benefits the generating of unseen samples. However, the basic autoencoder (without regularization on z) aims to precisely reconstruct the input data instead of flexibly generating new data from random z. Therefore, z from a well-trained autoencoder may present a hole in its distribution. For example, assuming the tanh activation function is used to constrain z in values between -1 and 1, z may be with extremely high probability to be either -1 or 1, as shown in Fig. 2.5. (left). Another reason of presenting a "hole" is that the training process easily pushes z towards the saturation area of the activation functions like sigmoid and tanh. The "hole" should be avoid in generative modeling because squeezing the hight density area of z to the boundary of the latent space is equivalent to squeezing the capacity of generating unseen or interpolated samples. Therefore, more intuitively, we should punish the hole-like distribution, rather than directly assuming a specific one (e.g., Gaussian). Suppose an activation function $f : \mathbb{R} \to [a, b]$, $a, b \in \mathbb{R}$ and $a < b$ (e.g., sigmoid or hyperbolic tangent — tanh), we propose an novel regularization term

Algorithm 2.1 Training strategy of GAN+AE+$\alpha_{\mathcal{L}}$

Input: the training dataset $\mathbb{X}_x$, weight of reconstruction loss λ, and decay rate α.
Initialize the encoder E, generator G, and discriminator D, which are parameterized by θ, ϕ, and φ, respectively.

Stage 1: pre-train E and G
repeat
  unrepeated samples $\{x_1, x_2, \cdots, x_m\} \subset \mathbb{X}_x$
  update the encoder E and generator G by SGD with gradient descent:

$$\Delta\theta, \Delta\varphi = \nabla_{\theta,\varphi} \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}(x_i - G(E(x_i)))$$

  update the discriminator D by SGD with gradient ascent:

$$\Delta\phi = \nabla_{\phi} \frac{1}{m} \sum_{i=1}^{m} [\log(D(x_i)(1 - D(G(E(x_i)))))]$$

until $\frac{1}{m} \sum_{i=1}^{m} \mathcal{L}(x_i, G(E(x_i))) < \frac{1}{\alpha}$

Stage 2: sharpen the generated data
repeat
  unrepeated samples $\{x_1, x_2, \cdots, x_m\} \subset \mathbb{X}_x$
  $\alpha_{\mathcal{L}} = \min \left\{ \frac{\alpha}{m} \sum_{i=1}^{m} \mathcal{L}(x_i, G(E(x_i))), 1 \right\}$
  update the encoder E and generator G by SGD with gradient descent:

$$\Delta\theta, \Delta\varphi = \nabla_{\theta,\varphi} \frac{\lambda}{m} \sum_{i=1}^{m} \mathcal{L}(x_i - G(E(x_i)))$$

  update the generator G by SGD with gradient descent:

$$\Delta\theta = \nabla_{\theta} \frac{1}{m} \sum_{i=1}^{m} [\log(1 - \alpha_{\mathcal{L}} D(G(E(x_i))) + \lambda \mathcal{L}(x_i - G(E(x_i)))]$$

  update the discriminator D by SGD with gradient ascend:

$$\Delta\phi = \nabla_{\phi} \frac{1}{m} \sum_{i=1}^{m} [\log(D(x_i)(1 - D(G(E(x_i)))))]$$

until terminating criterion is met

on z to avoid the "hole":

$$\mathcal{L}_z = \sum_i w_i \left| z_i - \frac{b+a}{2} \right| \tag{2.19}$$

$$w_i = \sigma\left(\gamma\left(\left| z_i - \frac{b+a}{2} \right| - \beta\frac{b-a}{2}\right)\right) \tag{2.20}$$

where $\sigma(\cdot)$ is the sigmoid function. $\mathcal{L}_z$ is maximized when z_i's are distributed at the boundary a and b, which indicates the extreme case of presenting a "hole". The parameter $\beta \in [0, 1]$ sets the threshold of how close to the boundary will be punished, and larger β pushes the threshold closer to the boundaries. The parameter $\gamma > 1$ controls steepness of the threshold. Fig. 2.4 illustrate the behavior of $\mathcal{L}_z$. Assume the activation function is tanh ($a = -1$ and $b = 1$). Larger $|z_i - (b-a)/2|$ indicates that z_i is more close to the boundary a or b. Comparing the solid blue and dashed red curves, which are only different in β, larger β punishes more on the condition that z is distributed more close to the boundaries. The solid blue and dotted yellow curves shows the difference in γ, which controls the steepness of thresholding.

When setting the parameters of Eq. 2.20, $\mathcal{L}_z$ is supposed to increase drastically only when z is distributed close to the saturation region of the activation function. In addition, we would like to avoid the saturation area of sigmoid function in Eq. 2.20 because it results in zero gradient and fails the regularization on z. Therefore, we set $\beta = 1$ to prevent zero gradient,

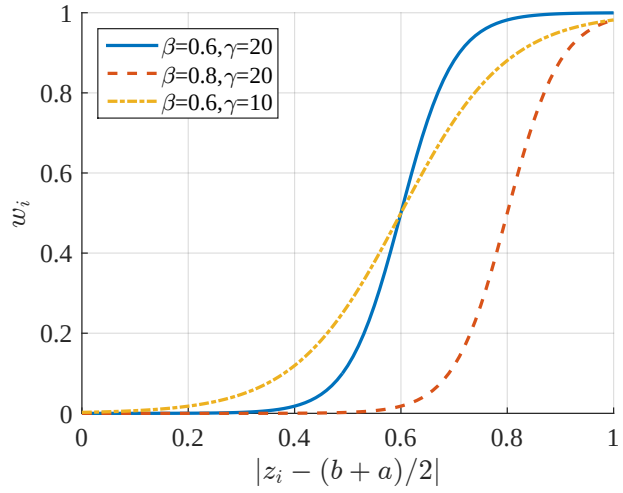


Figure 2.4: The behavior of $\mathcal{L}_z$ (Eq. 2.20) with different parameters β and γ.

and control the regularization only by γ. Fig. 2.5 illustrates the effect of regularization on z with different γ. Assume the activation function is tanh, and $\beta = 1$.The left figure shows the distribution of z from GAN+AE on MNIST. The rest show the distribution of z from GAN+AE with regularization on z but with different γ.

If the elements in z approaches ether a or b,

$$\begin{aligned} &\lim_{E(x)\to a,b} MAD(E(x)) \\ =&\lim_{E(x)\to a,b} \mathbb{E}[|E(x) - \mathbb{E}[E(x)]|] \\ =&\frac{2mn(b-a)}{(m+n)^2}. \end{aligned} \tag{2.21}$$

The maximum is achieved at $m = n$, where

$$\lim_{E(x)\to a,b} MAD(E(x)) = \frac{b-a}{2}. \tag{2.22}$$

The uniform distribution could be considered as the boundary of hole-like and Gaussian-like distribution, and its MAD is $(b-a)/4$. Ideally, $\mathcal{L}_z \approx 0$ for uniformly distributed $E(x)$. Therefore, we should set $\beta > 1/2$, which means it is seen as close to the boundaries when $MAD(E(x)) > (b-a)/4$, and $\mathcal{L}_z$ will increase quickly. Because of the non-binary thresholding, whose steepness is controlled by γ, β is related to γ in the way of

$$\gamma = -\frac{2\log\epsilon}{(\beta - 1/2)(b-a)}, \tag{2.23}$$

where $\epsilon \approx 0$ (e.g., $\epsilon = 0.001$) denotes the cost $\mathcal{L}_z$ when $E(x)$ presents a uniform distribution.

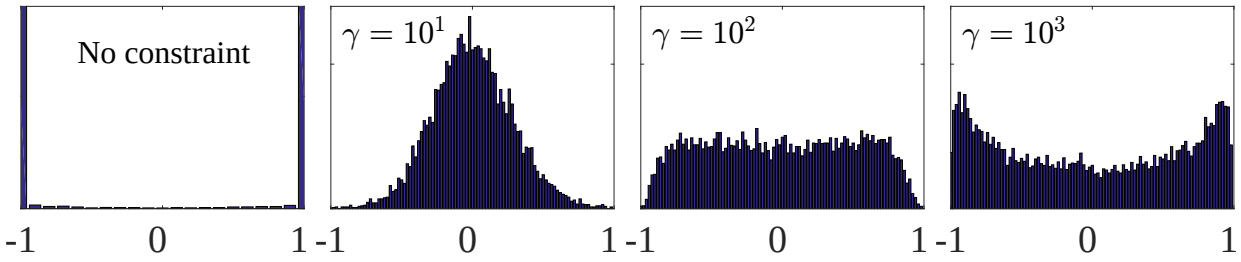


Figure 2.5: Effect of the regularization on z.

2.5.2 Random Sampling of z

Because we assume z is in an arbitrary distribution, a sampling mechanism is required. Markov chain Monte Carlo (MCMC) methods can learn to sample from a probability distribution, but it has to construct a Markov chain that has the desired distribution of its equilibrium distribution. Arici and Celikyilmaz [2] adopted a RBM, located in between the generator and discriminator, to learn the distribution of z. We use a simple method without learning any extra model — inverse transform sampling (ITS).

Definition 2.1. *Assume the distribution of z is p_z, and its cumulative distribution function is $U = \Phi(z)$, then U has a uniform distribution on $[0, 1]$. Therefore, $\Phi^{-1}(U) \sim p_z$.*

Give a pre-trained model, we can numerically approximate the distribution of z and obtain $\Phi(z)$. Then z from an arbitrary distribution p_z can be sampled from the uniformly distributed variable U through $\Phi^{-1}(U)$.

2.6 Experimental Evaluation

We claim that encoding-decoding structure is a more appealing way to stabilize the training of GAN, and at the same time avoiding mode missing. Therefore, we mainly compare two categories of methods that tend to solve the mode missing problem: 1) encoding-decoding-based GANs and 2) strategy-based alternatives that preserve the basic structure of GANs but improve the loss function (GMMN [72]), training procedure (mini-batch discriminator [113]), or update strategy (unrolled GAN [90]). Table 2.1 displays the methods in comparison.

Table 2.1: Notation of methods in comparison

Notation	Method
GAN	Vanilla GAN [42]
GMMN	Minimize MMD [72]
GAN+MMD	Incorporate MMD as loss function
GAN+MB	Mini-batch [113]
GAN+UR	Unrolled GAN [90]
GAN+AE	Incorporate encoder to GAN

We evaluate their performance in aspects of convergence speed, number of missing modes, and quality of generated images. For fair comparison, we implement each method with the same architecture of convolution and deconvolution network, with the same learning rate (0.002 with the Adam optimizer), kernel size (5×5), batch size (100), activation function (ReLU), and prior distribution of z (uniform). The proposed regularization on z is applied on GAN+AE to ensure uniform distribution. In addition, ℓ_2-norm is adopted to measure the reconstruction error, and the parameter λ in Eq. 2.17 is set to be 100 in all the following experiments. Table 2.2 shows details of the Conv and Deconv nets. Note that the network is not delicately designed to achieve the best performance, since the ultimate goal is to demonstrate the stability by adding autoencoder to GAN. For a fair comparison, we choose the same network structure for each method. Each method could be implemented by assembling one or two conv and deconv nets, e.g., GAN is constructed by a conv net (discriminator) and a deconv net (generator), and GAN+AE consists two conv nets (encoder and discriminator) and a deconv net (generator).

2.6.1 Performance Metrics

We first compare the convergence speed on MNIST [68] and use it as the base line, then on the compositional MNIST with 1000 modes to estimate the number of missing modes of each method. The Inception Scores [113] on MNIST, the compositional MNIST and CIFAR10 [62] are calculated to evaluate the quality of generated images. Note that we report the inception score results along with our own implementation on MNIST and the compositional MNIST datasets. For CIFAR10, the inception score is calculated based on the pre-trained Imagenet model. The specific experimental setting and analysis are discussed in the corresponding

Table 2.2: Details of convolution and deconvolution networks

Conv	Size	Deconv	Size
Input (gray/color image)	$a \times a \times 1$ or 3	Input	50
Conv, ReLU, BN	$a/2 \times a/2 \times 64$	FC, ReLU, BN	1024
Conv, ReLU, BN	$a/4 \times a/4 \times 128$	FC, ReLU, BN, reshape	$a/4 \times a/4 \times 128$
reshape, FC, ReLU, BN	1024	Deconv, ReLU, BN	$a/2 \times a/2 \times 64$
FC, Sigmoid	50 or 1 (for D)	Deconv, Sigmoid	$a \times a \times 1$ or 3

sections. In addition, CIFAR10 and three face datasets (i.e., Morph [109] and FGNET [65], CelebA [80]) are used to verify the performance on generating natural images. Finally, we verify the proposed training strategy in decaying the undamped oscillation problem.

2.6.2 Evaluation on the MNIST dataset

We apply all the methods in Table 2.1 on the MNIST dataset, and the interpolation results between two random digits at epochs 1 and 10 are shown in Fig. 2.6.

The corresponding inception score of each method at epoch 20 is listed in Table 2.3. Obviously, GMMN cannot generate clear digits, GAN+UR converges very slowly, and GAN+AE achieves the best visual quality and highest inception score, which also indicates the higher quality of generated images from GAN+AE.

Note that the inception score of GAN+AE on the CIFAR-10 dataset in Table 2.3 ranks lower than those of GAN, LSGAN, and GAN+UR. In addition, all standard deviation are significantly higher than those on MNIST and MNIST1000. This may be caused by the inherent design of inception score whose performance highly depends on that of the classifier. And we use a pre-trained model on ImageNet instead of a state-of-the-art model specifically trained on CIFAR-10.

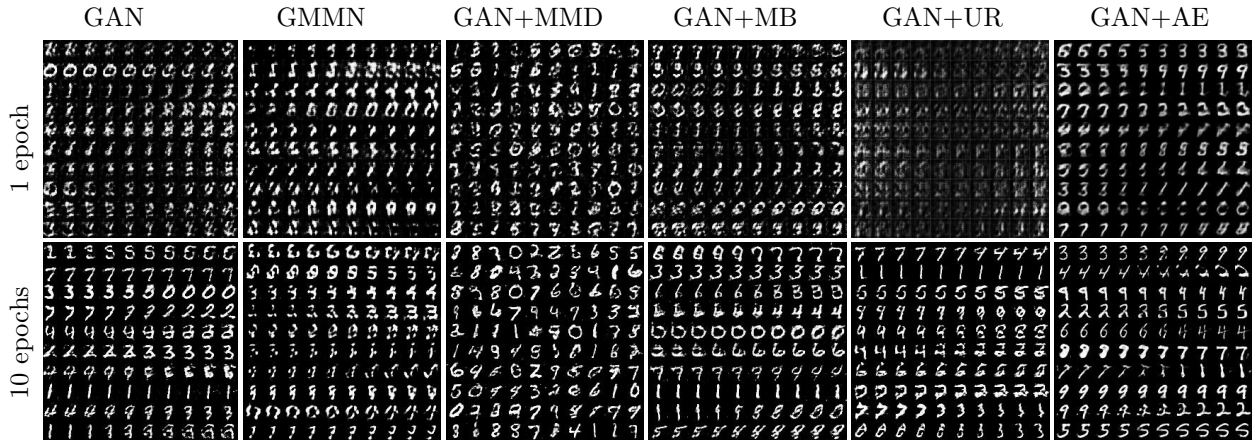


Figure 2.6: Comparison of different methods on the MNIST dataset.

Table 2.3: Inception scores of each method tested with 10,000 samples on three datasets

Method	MNIST	MNIST1000	CIFAR10
GAN	$2.22 \pm .009$	$2.18 \pm .017$	$\mathbf{2.53 \pm 2.34}$
GMMN	$2.00 \pm .014$	$2.05 \pm .028$	1.48 ± 1.44
GAN+MMD	$2.22 \pm .008$	$2.13 \pm .023$	2.37 ± 2.19
GAN+MB	$2.19 \pm .005$	$2.04 \pm .049$	1.41 ± 1.26
GAN+UR	$2.17 \pm .010$	$2.07 \pm .024$	2.39 ± 2.16
GAN+AE	$\mathbf{2.28 \pm .003}$	$\mathbf{2.29 \pm .028}$	1.89 ± 1.73

2.6.3 Evaluation on the Compositional MNIST dataset

The compositional MNIST dataset is constructed by randomly picking three digits from MNIST and form a three-digit number from 000 to 999. Specifically, we stitch three 28×28 images horizontally, and then remove the horizontal space between digits. Finally, the three-digit image is resized to the same size as MNIST. Fig. 2.7 shows the results of each method at epoch 20. Again, GAN+AE presents more appealing results.

To count the number of missed modes, we employ the method inspired by [12], i.e., testing the optimal discriminator by real data. If the expectation of discriminator outputs from certain mode is close to 1 (e.g., larger than 0.8), the corresponding input mode is considered as missed mode.

From Table 2.4, we observe that GAN+AE converges the fastest. After 50 epochs, GAN+MB, GAN+UR, and GAN+AE all perform similar. However, GMMN gains nothing

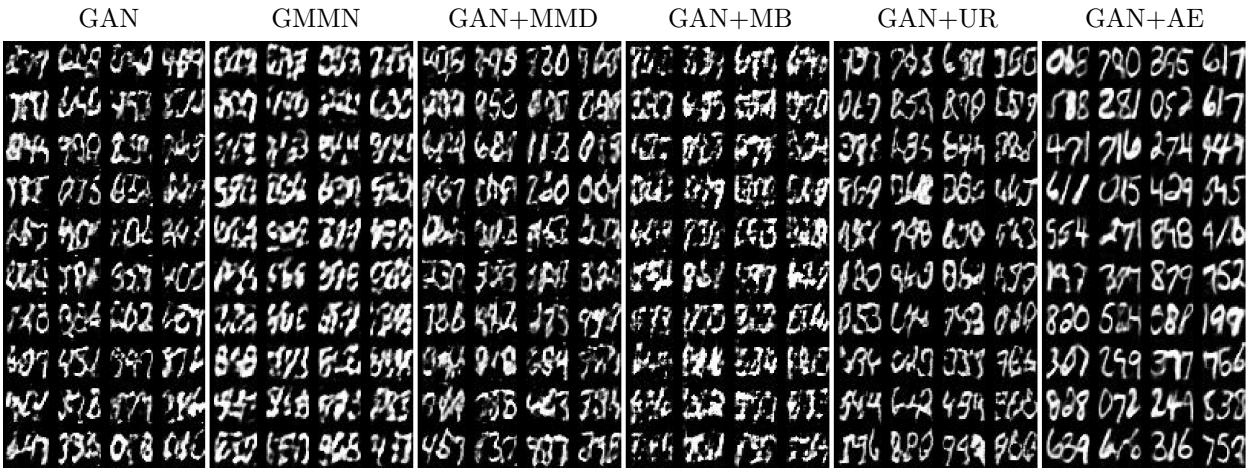


Figure 2.7: Comparison of different methods on compositional MNIST dataset.

Table 2.4: Number of missed modes on the compositional MNIST dataset

Method	#Miss at epoch 20	#Miss at epoch 50
GAN	154	82
GMMN	409	408
GAN+MMD	301	76
GAN+MB	516	79
GAN+UR	136	41
GAN+AE	**52**	**39**

from more training iterations. GAN+MMD eventually outperforms both GAN and GMMN. In this experiment, GAN+AE shows superior performance in aspects of convergence rate as well as the capability in avoiding missing mode.

2.6.4 Evaluation on Natural Images and Faces

In the experiment, we compare all the methods on natural images (CIFAR-10) and faces (CelebA [80]). The results are shown in Fig. 2.8. The samples are obtained after 20 epochs. On the CIFAR-10 dataset (Fig. 2.8 top), the generated images from GAN and GAN+UR look more plausible with lower noise and higher resolution. However, they are not realistic images, and they are mixture from multiple objects or backgrounds. Still, GMMN cannot compete with other GAN-based methods. GAN+MB shows unstable output at this stage. GAN+AE shows more noise (or details) as compared to GAN and GAN+UR, but it tends to prevent the mixing issue. We also provide the result of AE, which removes the discriminator from GAN+AE, to illustrate the effect of the encoding part. We can see that the result of AE is blurred but there is no mixing effect. This advantage is brought into GAN+AE, assisting the generation of more discriminative images. Same effect can be observed from the results on the face dataset (Fig. 2.8 bottom).

Generally, the methods without encoding-decoding structure, i.e., GAN, GMMN, GAN+MB, and GAN+UR, tend to generate images look like mixture from multiple objects, especially when learning on natural images. Compared to AE which generates images but with blurred objects, GAN+AE preserves more texture (or noise) and tends to prevent the mixing effect like other methods. Therefore, GAN+AE could be considered as intermediate



Figure 2.8: Comparison of different methods on the CIFAR-10 and face datasets.

between GANs and AE. On the face dataset, all methods perform better than on the CIFAR-10 dataset because the diversity of face is much lower than the natural objects. However, GANs without encoder still yield distorted faces as compared to GAN+AE and AE.

2.6.5 Evaluation on Multiple Domains

To test the performs on multi-domain dataset, we merge the MNIST, CIFAR-10, and UTKFace datasets. The result after 20 epochs is shown in Fig. 2.9. Note that the morphing between digits and natural images (or faces) shows relatively clear boundary for GAN, GAN+MB, and GAN+UR, while GMMN and GAN+AE present smoother morphing between any domains. Part of the reason is both GMMN and GAN+AE introduce direct comparison between real and generated data, rather than only fitting the generation distribution to a distribution indirectly specified by the discriminator. In addition, smooth morphing with variational inference (e.g., encoding) is more appealing for real applications,

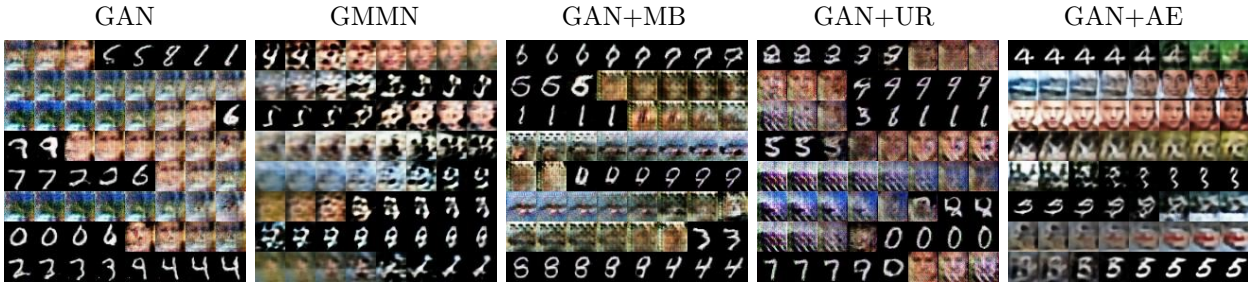


Figure 2.9: Comparison of different methods on the multi-domain dataset.

e.g., domain transformation and attribute manipulation, because the attributes of generated image can be specified by a input instead of uncontrollable random sampling.

2.6.6 Evaluation on Stability with Adaptive Decay

We claim that the variable $\alpha_{\mathcal{L}}$ (Eq. 2.17) will decay the undamped oscillation caused by competition between the generator and discriminator. In addition, α_L prevents the extreme cases of gradient vanishing or explosion. Comparing the results of GAN+AE and AE, the update of the generator propagated from the discriminator will significantly add texture and noise (or ghost) to the generated images. Therefore, we prefer a small α_L that suppresses the gradient from discriminator when the reconstruction error is lower than certain threshold. In this experiment, we set $\alpha = 0.1$ and compare GAN+AE and GAN+AE+$\alpha_{\mathcal{L}}$ on a composite dataset made with images from Morph and FGNET as they contain different ethnicity, age groups that would make the generative task even more challenging. Fig. 2.10 shows the comparison results using GAE+AE with and without $\alpha_{\mathcal{L}}$. GAN+AE and GAN+AE+$\alpha_{\mathcal{L}}$ are trained based on a pre-trained model on the face dataset. The top shows comparison on discriminator loss (average output of the discriminator). We can clearly observe the suppressed oscillation by adding the variable $\alpha_{\mathcal{L}}$. And as a result, the generated images are more photo-realistic and sharp.

2.7 Summary

In this chapter, we made the initial attempt to theoretically reasoning the effectiveness of autoencoder-based GANs. We showed that with the incorporation of an autoencoder,

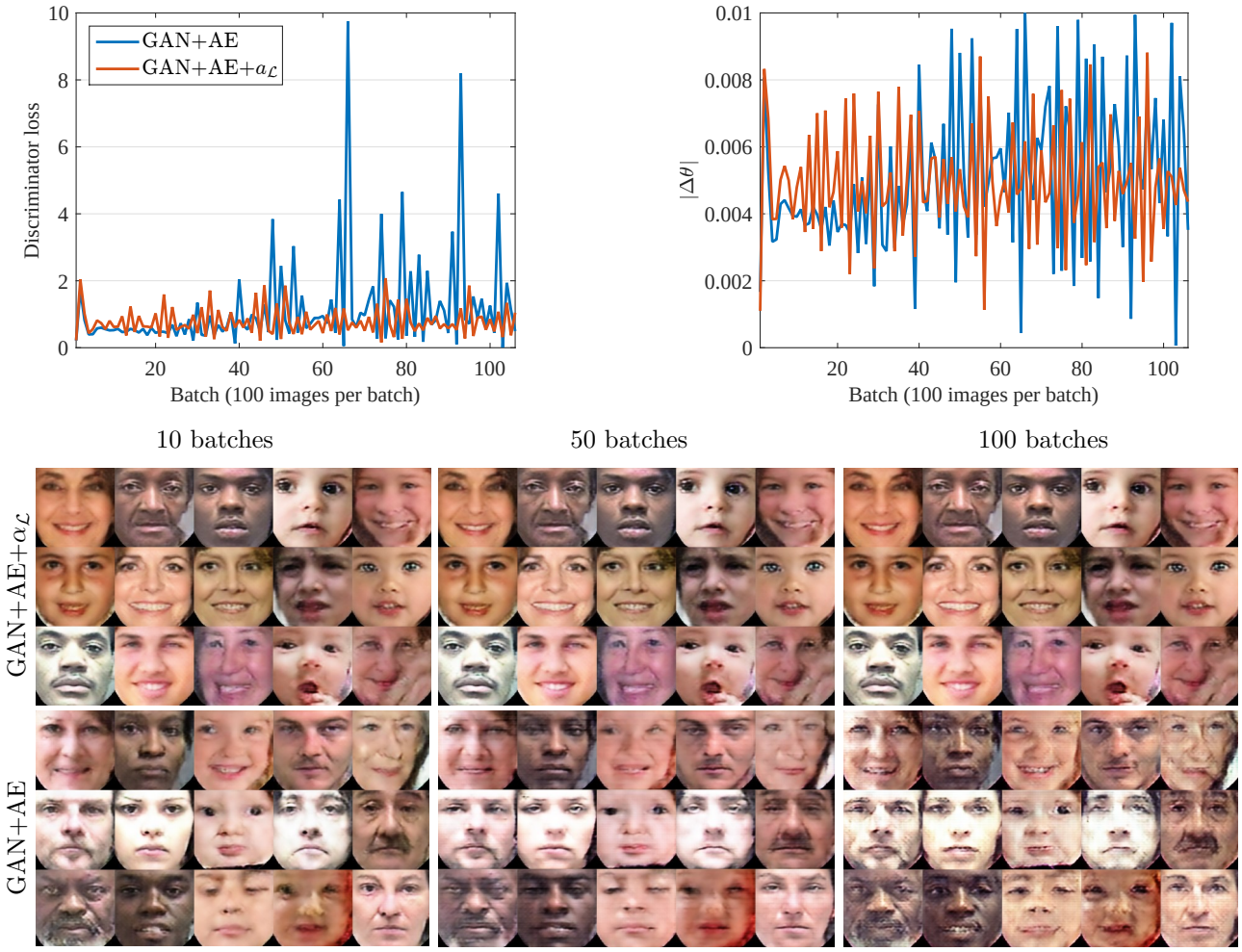


Figure 2.10: Comparison of GAN+AE with and without $\alpha_{\mathcal{L}}$.

the reconstruction error serves the purpose of penalizing the missing mode, thus effectively alleviating the mode missing problem in GAN. Based on this framework, we also elaborated the cause of instability in training generators and showed that by adding an adaptive decay variable in the objective function, the gradient vanishing and explosion, as well as the undamped oscillation problems can be effectively resolved.

Chapter 3

Conditional Image Synthesis

In deep learning based image manipulation, the encoding-decoding structure (i.e., autoencoder) is mostly adopted, which is a convolution-deconvolution network. Since the invention of GAN, the reality and resolution have been significantly enhanced compared to the lonely autoencoding network. Chapter 2 has discussed the advantages of incorporating an autoencoder with GAN, which stabilizes the training process, avoids mode missing, and more importantly spreads the practicality of GAN. Applications on image manipulation using deep learning (i.e., autoencoder and/or GAN) are too broad to be throughly discuss in limited pages. However, they share the similar idea that is to manipulate the latent variable (e.g., the high-level feature) or to concatenate conditions to the latent variable, and thus to indirectly and semantically manipulate the images. Instead of generally describing the large family of image manipulation, we focus on conditional image manipulation with a specific application — human face age progression and regression. To our best knowledge, manipulating images by conditions on the latent variable is the most popular way in various of tasks.

Manipulating the age appearance by GAN-based networks is still a novel and practically valuable task, and few of existing works could achieve perceptually good manipulation, i.e., generating realistic and high-resolution faces with required age. When asking the question "If I provide you a face image of mine (without telling you the actual age when I took the picture) and a large amount of face images that I crawled (containing labeled faces of different ages but not necessarily paired), can you show me what I would look like when I am 80 or what I was like when I was 5?" The answer is probably a "No." Most existing

face aging works attempt to learn the transformation between age groups and thus would require the paired samples as well as the labeled query image. In this chapter, we look at the problem from a generative modeling perspective such that no paired samples is required. In addition, given an unlabeled image, the generative model can directly produce the image with desired age attribute (detailed in Sec. 3.1). We propose a conditional adversarial autoencoder (CAAE in Sec. 3.2) that learns a face manifold, traversing on which smooth age progression and regression can be realized simultaneously. In CAAE, the face is first mapped to a latent vector through a convolutional encoder, and then the vector is projected to the face manifold conditional on age through a deconvolutional generator. The latent vector preserves personalized face features (i.e., personality) and the age condition controls progression vs. regression. Two adversarial networks are imposed on the encoder and generator, respectively, forcing to generate more photo-realistic faces. Experimental results (Sec. 3.3) demonstrate the appealing performance and flexibility of the proposed framework by comparing with the state-of-the-art and ground truth.

3.1 A Task: Face Age Progression/Regression

Face age progression (i.e., prediction of future looks) and regression (i.e., estimation of previous looks), also referred to as face aging and rejuvenation, aims to render face images with or without the "aging" effect but still preserve personalized features of the face (i.e., personality). It has tremendous impact to a wide-range of applications, e.g., face prediction of wanted/missing person, age-invariant verification, entertainment, etc. The area has been attracting a lot of research interests despite the extreme challenge in the problem itself. Most of the challenges come from the rigid requirement to the training and testing datasets, as well as the large variation presented in the face image in terms of expression, pose, resolution, illumination, and occlusion. The rigid requirement on the dataset refers to the fact that most existing works require the availability of paired samples, i.e., face images of the same person at different ages, and some even require paired samples over a long range of age span, which is very difficult to collect. For example, the largest aging dataset "Morph" [55] only captured images with an average time span of 164 days for each individual. In addition,

existing works also require the query image to be labeled with the true age, which can be inconvenient from time to time. Given the training data, existing works normally divide them into different age groups and learn a transformation between the groups, therefore, the query image has to be labeled in order to correctly position the image.

Although age progression and regression are equally important, most existing works focus on age progression. Very few works can achieve good performance of face rejuvenating, especially for rendering baby face from an adult because they are mainly surface-based modeling which simply remove the texture from a given image [81, 65, 35]. On the other hand, researchers have made great progress on age progression. For example, the physical model-based methods [127, 124, 65, 105] parametrically model biological facial change with age, e.g., muscle, wrinkle, skin, etc. However, they suffer from complex modeling, the requirement of sufficient dataset to cover long time span, and are computationally expensive; the prototype-based methods [128, 55, 118, 138] tend to divide training data into different age groups and learn a transformation between groups. However, some can preserve personality but induce severe ghosting artifacts, others smooth out the ghosting effect but lose personality, while most relaxed the requirement of paired images over long time span, and the aging pattern can be learned between two adjacent age groups. Nonetheless, they still need paired samples over short time span.

In this chapter, we investigate the age progression/regression problem from the perspective of generative modeling [148]. The rapid development of generative adversarial networks (GAN) has shown impressive results in face image generation [85, 152, 103, 78]. Here, we assume that the face images lie on a high-dimensional manifold as shown in Fig. 3.1. Given a query face, we could find the corresponding point (face) on the manifold. Stepping along the direction of age changing, we will obtain the face images of different ages while preserving personality. We propose a conditional adversarial autoencoder (CAAE)[1] network to learn the face manifold (detailed in Sec. 3.2). By controlling the age attribute, it will be flexible to achieve age progression and regression at the same time. Because it is difficult to directly manipulate on the high-dimensional manifold, the face is first mapped to a latent vector

[1]Bitbucket: https://bitbucket.org/aicip/face-aging-caae
Github: https://zzutk.github.io/Face-Aging-CAAE

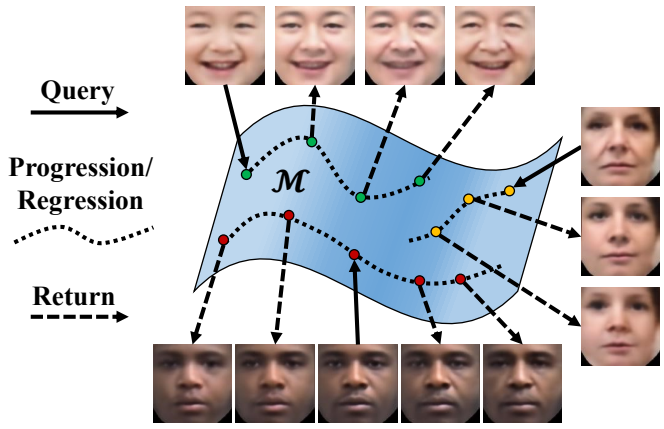


Figure 3.1: Assuming the face images lie on a manifold ($\mathcal{M}$), and images are clustered according to their ages and personality by different directions.

through a convolutional encoder, and then the vector is projected to the face manifold conditional on age through a deconvolutional generator. Two adversarial networks are imposed on the encoder and generator, respectively, forcing to generate more photo-realistic faces.

The benefit of the proposed CAAE can be summarized from four aspects. First, the novel network architecture achieves both age progression and regression while generating photo-realistic face images. Second, we deviate from the popular group-based learning, thus not requiring paired samples in the training data or labeled face in the test data, making the proposed framework much more flexible and general. Third, the disentanglement of age and personality in the latent vector space helps preserving personality while avoiding the ghosting artifacts. Finally, CAAE is robust against variations in pose, expression, and occlusion.

3.1.1 Related Work

Age Progression and Regression

In recent years, the study on face age progression has been very popular, with approaches mainly falling into two categories, physical model-based and prototype-based. Physical model-based methods model the biological pattern and physical mechanisms of aging, e.g., the muscles [123], wrinkle [106, 124], facial structure [105, 65] etc. through either parametric

or non-parametric learning. However, in order to better model the subtle aging mechanism, it will require a large face dataset with long age span (e.g., from 0 to 80 years old) of each individual, which is very difficult to collect. In addition, physical modeling-based approaches are computationally expensive.

On the other hand, prototype-based approaches [9, 55] often divide faces into groups by age, e.g., the average face of each group, as its prototype. Then, the difference between prototypes from two age groups is considered the aging pattern. However, the aged face generated from averaged prototype may lose the personality (e.g., wrinkles). To preserve the personality, [118] proposed a dictionary learning based method — age pattern of each age group is learned into the corresponding sub-dictionary. A given face will be decomposed into two parts: age pattern and personal pattern. The age pattern was transited to the target age pattern through the sub-dictionaries, and then the aged face is generated by synthesizing the personal pattern and target age pattern. However, this approach presents serious ghosting artifacts. The deep learning-based method [138] represents the state-of-the-art, where RNN is applied on the coefficients of eigenfaces for age pattern transition. All prototype-based approaches perform the group-based learning which requires the true age of testing faces to localize the transition state which might not be convenient. In addition, these approaches only provide age progression from younger face to older ones. To achieve flexible bidirectional age changes, it may need to retrain the model inversely.

Face age regression, which predicts the rejuvenating results, is comparatively more challenging. Most age regression works so far [81, 35] are physical model-based, where the textures are simply removed based on the learned transformation over facial surfaces. Therefore, they cannot achieve photo-realistic results for baby face predictions.

Generative Adversarial Network

Generating realistically appealing images is still challenging and has not achieved much success until the rapid advancement of the generative adversarial network (GAN). The original GAN work [42] introduced a novel framework for training generative models. It simultaneously trains two models: 1) the generative model G captures the distribution of training samples and learns to generate new samples imitating the training, and 2) the

discriminative model D discriminates the generated samples from the training. G and D compete with each other using a min-max game as Eq. 3.1, where z denotes a vector randomly sampled from certain distribution $p(\mathbf{z})$ (e.g., Gaussian or uniform), and the data distribution is $p_{data}(\mathbf{x})$, i.e., the training data $x \sim p_{data}(\mathbf{x})$.

$$\min_G \max_D \mathbb{E}_{x \sim p_{data}(\mathbf{x})}[\log D(x)] + \mathbb{E}_{z \sim p(\mathbf{z})}[\log(1 - D(G(z)))] \tag{3.1}$$

The two parts, G and D, are trained alternatively.

One of the biggest issues of GAN is that the training process is unstable, and the generated images are often noisy and incomprehensible. During the last two years, several approaches [103, 92, 44, 16, 22, 51, 85] have been proposed to improve the original GAN from different perspectives. For example, DCGAN [103] adopted deconvolutional and convolutional neural networks to implement G and D, respectively. It also provided empirical instruction on how to build a stable GAN, e.g., replacing the pooling by strides convolution and using batch normalization. CGAN [92] modified GAN from unsupervised learning into semi-supervised learning by feeding the conditional variable (e.g., the class label) into the data. The low resolution of the generated image is another common drawback of GAN. [22, 51] extended GAN into sequential or pyramid GAN to handle this problem, where the image is generated step by step, and each step utilizes the information from the previous step to further improve the image quality. Some GAN-related works have shown visually impressive results of randomly drawing face images [145, 85, 152, 103, 78]. However, GAN generates images from random noise, thus the output image cannot be controlled. This is undesirable in age progression and regression, where we have to ensure the output face looks like the same person as queried.

3.1.2 Traversing on the Manifold

We assume the face images lie on a high-dimensional manifold, on which traversing along certain direction could achieve age progression/regression while preserving the personality. This assumption will be demonstrated experimentally in Sec. 3.2.2. However, modeling the high-dimensional manifold is complicated, and it is difficult to directly manipulate

(traversing) on the manifold. Therefore, we will learn a mapping between the manifold and a lower-dimensional space, referred to as the *latent space*, which is easier to manipulate. As illustrated in Fig. 3.2, faces x_1 and x_2 are mapped to the latent space by E (i.e., an encoder), which extracts the personal features z_1 and z_2, respectively. Concatenating with the age labels l_1 and l_2, two points are generated in the latent space, namely $[z_1, l_1]$ and $[z_2, l_2]$. Note that the personality z and age l are disentangled in the latent space, thus we could simply modify age while preserving the personality. Starting from the red rectangular point $[z_2, l_2]$ (corresponding to x_2) and evenly stepping bidirectionally along the age axis (as shown by the solid red arrows), we could obtain a series of new points (red circle points). Through another mapping G (i.e.. a generator), those points are mapped to the manifold $\mathcal{M}$ — generating a series of face images, which will present the age progression/regression of x_2. By the same token, the green points and arrows demonstrate the age progressing/regression of x_1 based on the learned manifold and the mappings. If we move the point along the dotted arrow in the latent space, both personality and age will be changed as reflected on $\mathcal{M}$. We will learn the mappings E and G to ensure the generated faces lie on the manifold, which indicates that the generated faces are realistic and plausible for a given age.

3.2 Conditional Adversarial Autoencoder

In this section, we first present the pipeline of the proposed conditional adversarial autoencoder (CAAE) network (Sec. 3.2.1) that learns the face manifold (Sec. 3.2.2). The CAAE incorporates two discriminator networks, which are the key to generating more

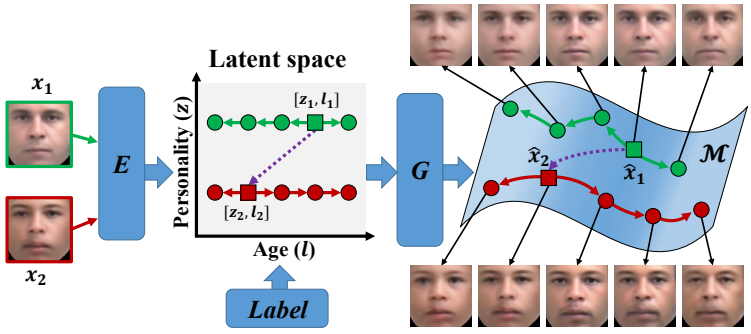


Figure 3.2: Illustration of traversing on the face manifold $\mathcal{M}$.

realistic faces. Sections 3.2.3 and 3.2.4 demonstrate their effectiveness, respectively. Finally, Section 3.2.5 discusses the difference of the proposed CAAE from other generative models.

3.2.1 Network Structure of CAAE

The detailed structure of the propose CAAE network is shown in Fig. 3.3. The encoder E maps the input face to a vector z (personality). Concatenating the label l (age) to z, the new latent vector $[z, l]$ is fed to the generator G. Both the encoder and the generator are updated based on the L_2 loss between the input and output faces. The discriminator D_z imposes the uniform distribution on z, and the discriminator D_{img} forces the output face to be photo-realistic and plausible for a given age label.

The input and output face images are 128×128 RGB images $x \in \mathbb{R}^{128\times128\times3}$. A convolutional neural network is adopted as the encoder. The convolution of stride 2 is employed instead of pooling (e.g., max pooling) because strided convolution is fully differentiable and allows the network to learn its own spacial downsampling [103]. The output of encoder $E(x) = z$ preserves the high-level personal feature of the input face x. The output face conditioned on certain age can be expressed by $G(z, l) = \hat{x}$, where l denotes the one-hot age label. Unlike existing GAN-related works, we incorporate an encoder to avoid random sampling of z because we need to generate a face with specific personality which is incorporated in z. In addition, two discriminator networks are imposed on E and G, respectively. The D_z regularizes z to be uniform distributed, smoothing the

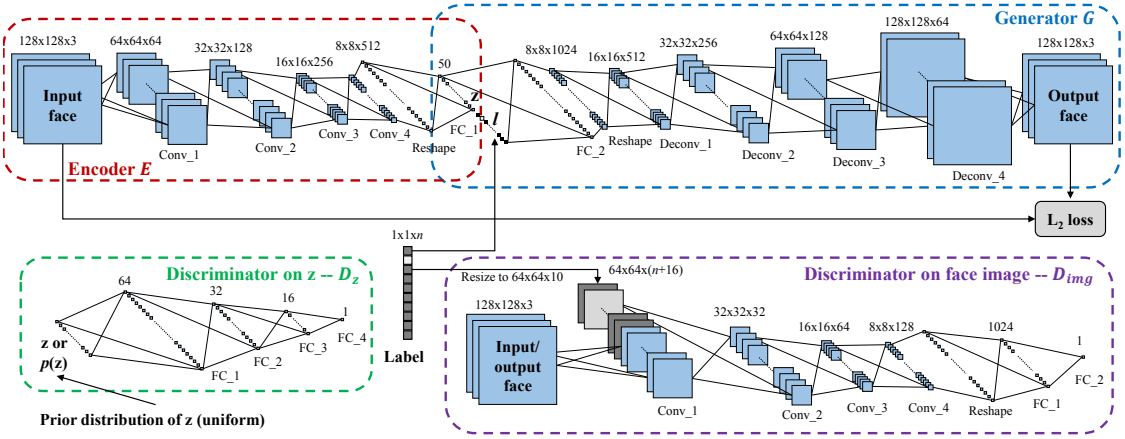


Figure 3.3: Structure of the proposed CAAE network for age progression/regression.

age transformation. The D_{img} forces G to generate photo-realistic and plausible faces for arbitrary z and l. The effectiveness of the two discriminators will be further discussed in Secs. 3.2.3 and 3.2.4, respectively.

3.2.2 Objective Function

The real face images are supposed to lie on the face manifold $\mathcal{M}$, so the input face image $x \in \mathcal{M}$. The encoder E maps the input face x to a feature vector, i.e., $E(x) = z \in \mathbb{R}^n$, where n is the dimension of the face feature. Given z and conditioned on certain age label l, the generator G generates the output face $\hat{x} = G(z, l) = G(E(x), l)$. Our goal is to ensure the output face $\hat{x}$ lies on the manifold while sharing the personality and age with the input face x (during training). Therefore, the input and output faces are expected to be similar as expressed in Eq. 3.2, where $\mathcal{L}(\cdot, \cdot)$ denotes L_2 norm.

$$\min_{E,G} \mathcal{L}\left(x, G(E(x), l)\right) \tag{3.2}$$

Simultaneously, the uniform distribution is imposed on z through D_z – the discriminator on z. We denote the distribution of the training data as $p_{data}(\mathbf{x})$, then the distribution of z is $q(\mathbf{z}|\mathbf{x})$. Assuming $p(\mathbf{z})$ is a prior distribution, and $z^* \sim p(\mathbf{z})$ denotes the random sampling process from $p(\mathbf{z})$. A min-max objective function can be used to train E and D_z,

$$\min_{E} \max_{D_z} \mathbb{E}_{z^* \sim p(\mathbf{z})}\left[\log D_z(z^*)\right] + \mathbb{E}_{x \sim p_{data}(\mathbf{x})}\left[\log(1 - D_z(E(x)))\right] \tag{3.3}$$

By the same token, the discriminator on face image, D_{img}, and G with condition l can be trained by

$$\min_{G} \max_{D_{img}} \mathbb{E}_{x,l \sim p_{data}(\mathbf{x},\mathbf{l})}\left[\log D_{img}(x, l)\right] + \mathbb{E}_{x,l \sim p_{data}(\mathbf{x},\mathbf{l})}\left[\log(1 - D_{img}(G(E(x), l)))\right] \tag{3.4}$$

Note that the age label is resized and concatenated to the first convolutional layer of D_{img} to make it discriminative on both age and human face. Sequentially updating the network by Eqs. 3.2, 3.3, and 3.4, we could finally learn the manifold $\mathcal{M}$ as illustrated in Fig. 3.4, where the horizontal axis indicates the traversing of age, and the vertical axis indicates different

personality. Finally the objective function becomes

$$\begin{aligned}\min_{E,G}\max_{D_z,D_{img}} \lambda\mathcal{L}\left(x, G(E(x), l)\right) + \gamma TV(G(E(x), l)) \\ + \mathbb{E}_{z^*\sim p(\mathbf{z})}\left[\log D_z(z^*)\right] \\ + \mathbb{E}_{x\sim p_{data}(\mathbf{x})}\left[\log(1 - D_z(E(x)))\right] \\ + \mathbb{E}_{x,l\sim p_{data}(\mathbf{x},\mathbf{l})}\left[\log D_{img}(x, l)\right] \\ + \mathbb{E}_{x,l\sim p_{data}(\mathbf{x},\mathbf{l})}\left[\log(1 - D_{img}(G(E(x), l)))\right],\end{aligned} \tag{3.5}$$

where $TV(\cdot)$ denotes the total variation which is effective in removing the ghosting artifacts. The coefficients λ and γ balance the smoothness and high resolution.

3.2.3 Discriminator on z

The discriminator on z, denoted by D_z, imposes a prior distribution (e.g., uniform distribution) on z. Specifically, D_z aims to discriminate the z generated by encoder E. Simultaneously, E will be trained to generate z that could fool D_z. Such adversarial process forces the distribution of the generated z to gradually approach the prior. We use uniform distribution as the prior, forcing z to evenly populate the latent space with no apparent

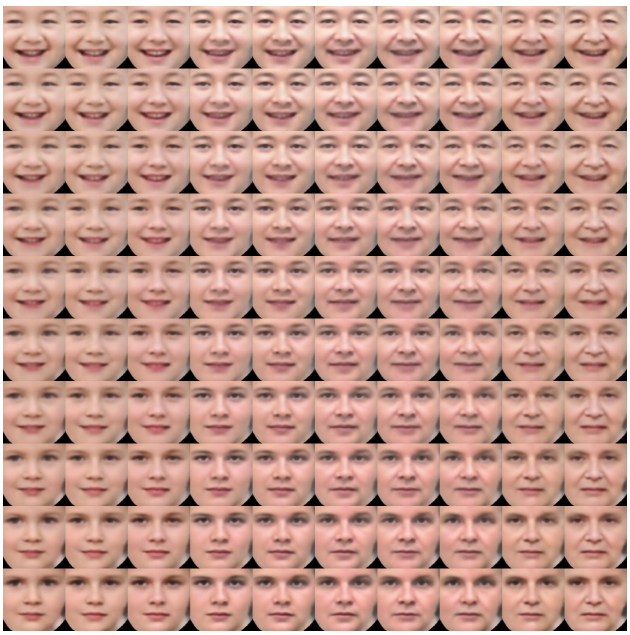


Figure 3.4: Illustration of the learned face manifold $\mathcal{M}$.

"holes". As shown in Fig. 3.5, the generated z's (depicted by blue dots in a 2-D space) present uniform distribution under the regularization of D_z, while the distribution of z exhibits a "hole" without the application of D_z. For simplicity, z is illustrated in a 2-D space. Blue dots indicate z's mapped from training faces through the encoder. With D_z, the distribution of z will approach uniform. Otherwise, z may present "holes". The rectangular points denote the corresponding z mapped from the input faces x_1 and x_2, and the dotted arrow indicates the traversing from z_1 to z_2. The intermediate points along the traversing are supposed to generate a series of plausible morphing faces from x_1 to x_2. Without D_z, the z presents a sparse distribution along the path, causing the generated face to look unreal. The series of figures at the bottom shows the traversing with and without D_z.

Exhibition of the "hole" indicates that face images generated by interpolating between arbitrary z's may not lie on the face manifold – generating unrealistic faces. For example, given two faces x_1 and x_2 as shown in Fig. 3.5, we obtain the corresponding z_1 and z_2 by E under the conditions with and without D_z, respectively. Interpolating between z_1 and z_2 (dotted arrows in Fig. 3.5), the generated faces are expected to show realistic and smooth morphing from x_1 to x_2 (bottom of Fig. 3.5). However, the morphing without D_z actually presents distorted (unrealistic) faces in the middle (indicated by dashed box), which corresponds to the interpolated z's passing through the "hole".

The length of vector z makes differences as well, but this difference may not be so obvious. Fig. 3.6 displays the age progressing results with different length of z. Each row shows the

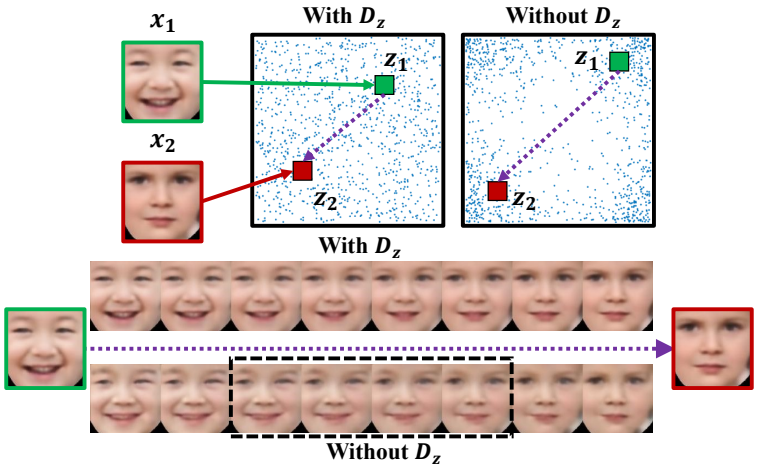


Figure 3.5: Effect of D_z, which forces z to a uniform distribution.



Figure 3.6: Effect of the length of z.

age progression with a specific length of z, i.e., 20, 30, 40, 50, 70, 100, 150. The faces generated from shorter z visually show wider span of age compared to those from longer z. Comparing the first and last rows, the first row shows relatively wider age span. However, the last row is more smooth in face morphing. To balance these two ends, we choose the length of z between 30 and 50.

3.2.4 Discriminator on Face Images

Inheriting the similar principle of GAN, the discriminator D_{img} on face images forces the generator to yield more realistic faces. In addition, the age label is imposed on D_{img} to make it discriminative against unnatural faces conditional on age. Although minimizing the distance between the input and output images as expressed in Eq. 3.2 forces the output face to be close to the real ones, Eq. 3.2 does not ensure the framework to generate plausible faces from those unsampled faces. For example, given a face that is unseen during training and a random age label, the pixel-wise loss could only make the framework generate a face close to the trained ones in a manner of interpolation, causing the generated face to be very blurred. The D_{img} will discriminate the generated faces from real ones in aspects of reality, age, resolution, etc. Fig. 3.7 demonstrates the effect of D_{img}. The first column shows the original faces, and their true ages are marked on the top. The right four columns are generated faces

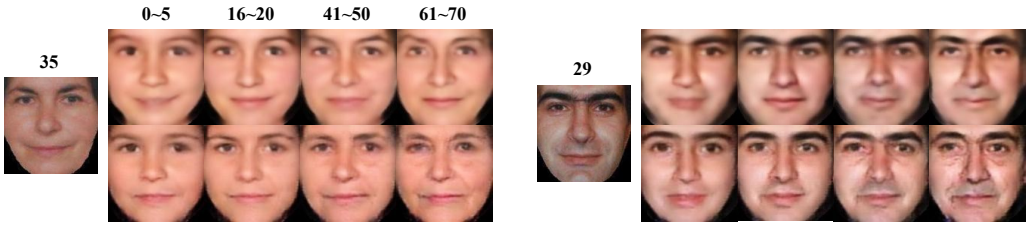


Figure 3.7: Effect of D_{img}, which forces the generated faces to be more realistic in aspects of age and resolution.

through the proposed framework, without (the upper row) or with (the lower row) D_{img}. The generated faces fall in four age groups as indicated at the top of each column.

Comparing the generated faces with and without D_{img}, it is obvious that D_{img} assists the framework to generate more realistic faces. The outputs without D_{img} could also present aging but the effect is not as obvious as that with D_{img} because D_{img} enhances the texture especially for older faces.

3.2.5 Differences from Other Generative Networks

In this section, we comment on the similarity and difference of the proposed CAAE with other generative networks, including GAN [42], variational autoencoder (VAE) [60], and adversarial autoencoder (AAE) [85]. Fig. 3.8 illustrates the fundamental differences between GAN, conditional GAN, VAE, AAE, and CAAE. G, D, and E denote generator (decoder), discriminator, and encoder. x, x', y, and z are input, output, label (condition), and latent/random variable.

VAE vs. GAN

VAE uses a recognition network to predict the posterior distribution over the latent variables, while GAN uses an adversarial training procedure to directly shape the output distribution of the network via back-propagation [85]. Because VAE follows an encoding-decoding scheme, we can directly compare the generated images to the inputs, which is not possible when using a GAN. A downside of VAE is that it uses mean squared error instead of an adversarial network in image generation, so it tends to produce more blurry images [66].

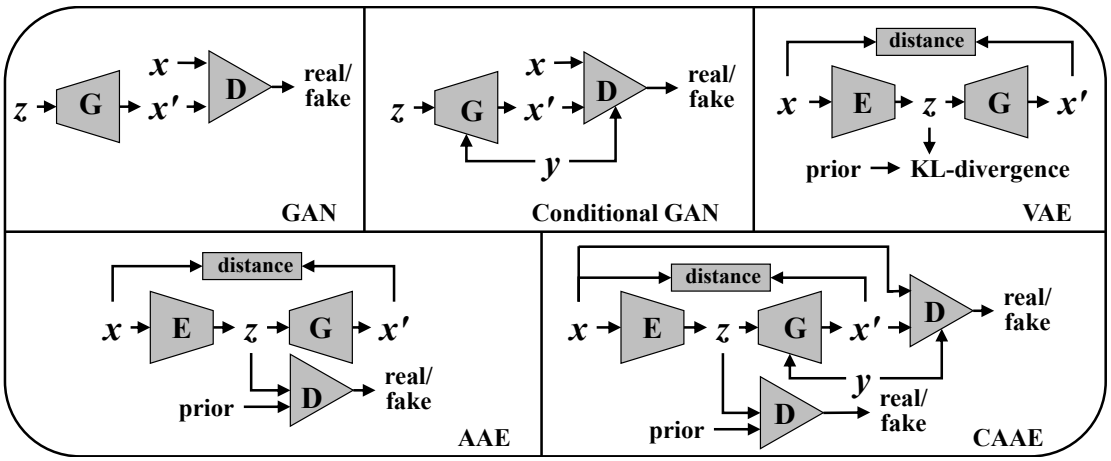


Figure 3.8: Comparison of different generative models.

AAE vs. GAN and VAE

AAE can be treated as the combination of GAN and VAE, which maintains the autoencoder network like VAE but replaces the KL-divergence loss with an adversarial network like in GAN. Instead of generating images from random noise as in GAN, AAE utilizes the encoder part to learn the latent variables approximated on certain prior, making the style of generated images controllable. In addition, AAE better captures the data manifold compared to VAE.

CAAE vs. AAE

The proposed CAAE is more similar to AAE. The main difference from AAE is that the proposed CAAE imposes discriminators on the encoder and generator, respectively. The discriminator on encoder guarantees smooth transition in the latent space, and the discriminator on generator assists to generate photo-realistic face images. Therefore, CAAE would generate higher quality images than AAE as discussed in Sec. 3.2.4.

3.3 Experimental Evaluation

In the section, we will first clarify the process of data collection (Sec. 3.3.1) and implementation of the proposed CAAE (Sec. 3.3.2). Then, both qualitative and quantitative comparisons with prior works and ground truth are performed in Sec. 3.3.3. Finally, the tolerance to occlusion and variation in pose and expression is illustrated in Sec. 3.3.4 .

3.3.1 Data Collection

We first collect face images from the Morph dataset [55] and the CACD [13] dataset. The Morph dataset [55] is the largest with multiple ages of each individual, including 55,000 images of 13,000 subjects from 16 to 77 years old. The CACD [13] dataset contains 13,446 images of 2,000 subjects. Because both datasets have limited images from newborn or very old faces, we crawl images from Bing and Google search engines based on the keywords, e.g., baby, boy, teenager, 15 years old, etc. Because the proposed approach does not require multiple faces from the same subject, we simply randomly choose around 3,000 images from the Morph and CACD dataset and crawl 7,670 images from the website. The age and gender of the crawled faces are estimated based on the image caption or the result from age estimator [110]. We divide the age into ten categories, i.e., 0–5, 6–10, 11–15, 16–20, 21–30, 31–40, 41–50, 51–60, 61–70, and 71–80. Therefore, we can use a one-hot vector of ten elements to indicate the age of each face during training. The final dataset consists of 10,670 face images with a uniform distribution on gender and age. We use the face detection algorithm with 68 landmarks [54, Dlib] to crop out and align the faces, making the training more attainable.

3.3.2 Implementation of CAAE

We construct the network according to Fig. 3.3 with kernel size of 5×5. The pixel values of the input images are normalized to $[-1, 1]$, and the output of E (i.e., z) is also restricted to $[-1, 1]$ by the hyperbolic tangent activation function. Then, the desired age label, the one-hot vector, is concatenated to z, constructing the input of G. To make fair concatenation, the elements of label is also confined to $[-1, 1]$, where -1 corresponds to 0. Finally, the output is also in range $[-1, 1]$ through the hyperbolic tangent function. Normalizing the input may make the training process converge faster. Note that we will not use the batch normalization for E and G because it blurs personal features and makes output faces drift far away from inputs in testing. However, the batch normalization will make the framework more stable if it is applied on D_{img}. All intermediate layers of each block (i.e., E, G, D_z, and D_{img}) use the ReLU activation function.

In training, $\lambda = 100$, $\gamma = 10$, and the four blocks are updated alternatively with a mini-batch size of 100 through the stochastic gradient descent solver, ADAM [59] ($\alpha = 0.0002$, $\beta_1 = 0.5$). Face and age pairs are fed to the network. After about 50 epochs, plausible generated faces can be obtained. During testing, only E and G are active. Given an input face without true age label, E maps the image to z. Concatenating an arbitrary age label to z, G will generate a photo-realistic face corresponding to the age and personality.

3.3.3 Qualitative and Quantitative Comparison

To evaluate that the proposed CAAE can generate more photo-realistic results, we compare ours with the ground truth and the best results from prior works [138, 55, 118, 124], respectively. We choose FGNET [65] as the testing dataset, which has 1002 images of 82 subjects aging from 0 to 69.

Comparison with ground truth

In order to verify whether the personality has been preserved by the proposed CAAE, we qualitatively and quantitatively compare the generated faces with the ground truth. The qualitative comparison is shown in Fig. 3.9, which shows appealing similarity.

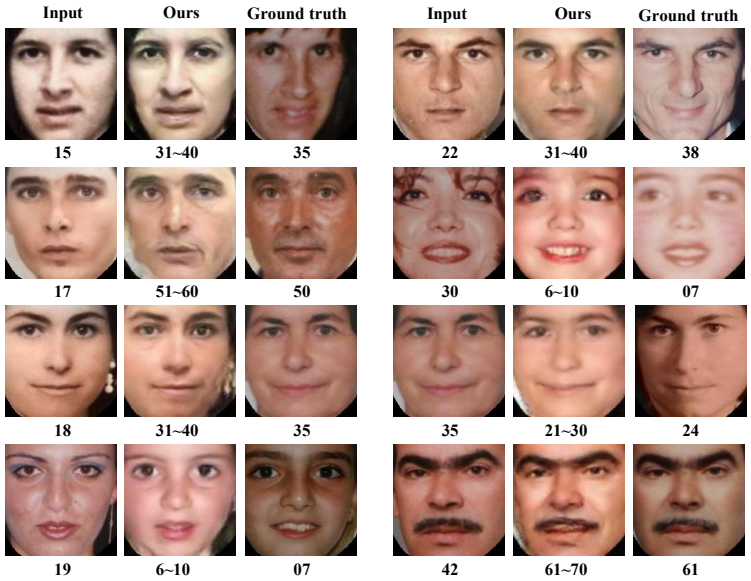


Figure 3.9: Comparison to the ground truth.

To quantitatively evaluate the performance, we pair the generated faces with the ground truth whose age gap is larger than 20 years. There are 856 pairs in total. We design a survey to compare the similarity where 63 volunteers participate. Each volunteer is presented with three images, an original image X, a generated image A, and the corresponding ground truth image B under the same group. They are asked whether the generated image A looks similar to the ground truth B; or not sure. We ask the volunteers to randomly choose 45 questions and leave the rest blank. We receive 3208 votes in total, with 48.38% indicating that the generated image A is the same person as the ground truth, 29.58% indicating they are not, and 22.04% not sure, as listed in Table 3.1. The voting results demonstrate that we can effectively generate photo-realistic faces under different ages while preserving their personality.

Comparison with prior work

We compare the performance of our method with some prior works [138, 55, 118, 124], for face age progression and Face Transformer (FT) [FT] for face age regression. To demonstrate the advantages of CAAE, we use the same input images collected from those prior works and perform long age span progression. To compare with prior works, we cite their results as shown in Fig. 3.10. The first column shows input faces, and second column are the best aged faces cited from prior works. The rest columns are our results from both age progression and regression. The red boxes indicate the comparable results to the prior works.

We also compare with age regression works using the FT demo [FT] as shown in Fig. 3.11, where the first row shows the input faces, the middle row shows the baby faces generated by FT [FT], and the last row shows our results. Our results obviously show higher fidelity, demonstrating the capability of CAAE in achieving smooth face aging and rejuvenation. CAAE better preserves the personality even with a long age span. In addition, our results provide richer texture (e.g., wrinkle for old faces), making old faces look more realistic.

Table 3.1: Survey to compare generated faces and ground truth

Same to ground truth	Different from ground truth	Not sure
48.38%	29.58%	22.04%



Figure 3.10: Comparison to prior works of face aging.

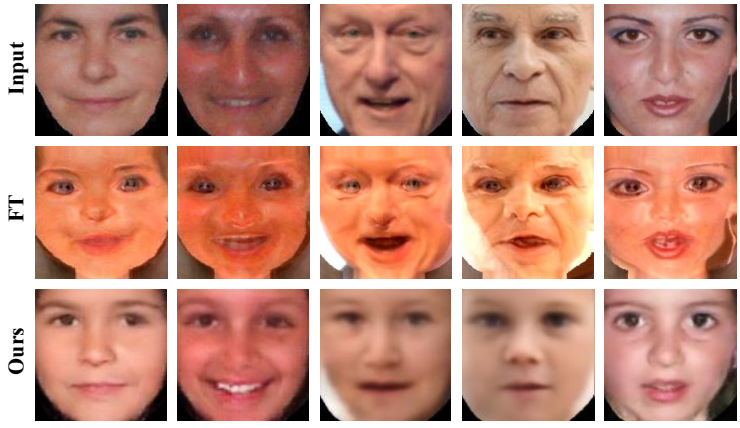


Figure 3.11: Comparison to prior work in rejuvenation.

Another survey is conducted to statistically evaluate the performance as compared with prior works, where for each testing image, the volunteer is to select the better result from CAAE or prior works, or hard to tell. We collect 235 paired images of 79 subjects from previous works [138, 55, 118, 124]. We receive 47 responses and 1508 votes in total with 52.77% indicating CAAE is better, 28.99% indicating the prior work is better, and 18.24% indicating they are equal, as listed in Table 3.2. This result further verifies the superior performance of the proposed CAAE.

3.3.4 Tolerance to Pose, Expression, and Occlusion

As mentioned above, the input images have large variation in pose, expression, and occlusion. To demonstrate the robustness of CAAE, we choose the faces with expression variation, non-frontal pose, and occlusion, respectively, as shown in Fig. 3.12. The very left column shows the input faces, and the right columns are generated faces by CAAE from younger to older ages. The first input face presents relatively more dramatic expression, the second input shows only the face profile, and the last one is partially occluded by facial marks. It is worth noting that the previous works [138, 55] often apply face normalization to alleviate from the variation of pose and expression but they may still suffer from the occlusion issue. In contrast, the proposed CAAE obtains the generated faces without the need to remove these variations, paving the way to robust performance in real applications.

Table 3.2: Survey to compare our results and those from the best prior works.

Better than prior	Worse than prior	Not sure
52.77%	28.99%	18.24%



Figure 3.12: Tolerance to occlusion and variation in pose and expression.

3.4 Summary

In this chapter, we proposed a novel conditional adversarial autoencoder (CAAE), which first achieves face age progression and regression in a holistic framework. We deviated from the conventional routine of group-based training by learning a manifold, making the aging progression/regression more flexible and manipulatable — from an arbitrary query face without knowing its true age, we can freely produce faces at different ages, while at the same time preserving the personality. We demonstrated that with two discriminators imposed on the generator and encoder, respectively, the framework generates more photo-realistic faces. Flexibility, effectiveness, and robustness of CAAE have been demonstrated through extensive evaluation.

The proposed framework has great potential to serve as a general framework for face-age related tasks. More specifically, we trained four sub-networks, i.e., E, G, D_z, and D_{img}, but only E and G are utilized in the testing stage. The D_{img} is trained conditional on age. Therefore, it is able to tell whether the given face corresponds to a certain age, which is exactly the task of age estimation. For the encoder E, it maps faces to a latent vector (face feature), which preserves the personality regardless of age. Therefore, E could be considered a candidate for cross-age recognition. The proposed framework could be easily applied to other image generation tasks, where the characteristics of the generated image can be controlled by the conditional label. In the future, we would extend current work to be a general framework, simultaneously achieving age progressing (E and G), cross-age recognition (E), face morphing (G), and age estimation (D_{img}).

Chapter 4

Decoupled Learning for Conditional Image Synthesis

The structure that incorporating encoding-decoding network with an adversarial network (i.e., discriminator) have been widely applied in many applications, e.g., image transformation, inpainting, manipulation, etc., to generate conditional results with high fidelity. Such composite network is considered as the conditional GAN-based network. Chapter 2 has proved the advantages of such composite network, and Chapter 3 verifies its effectiveness again with an specific application. While the composite structure is widely and largely adopted in image generation and manipulated related works in recent years, this chapter will further investigate its drawback and give the first attempt to improve it fundamentally.

During the training in existing composite framework, the gradient from encoding-decoding net (reconstruction effect) and discriminator (adversarial effect) are both imposed on a single generator/decoder. Although the encoding-decoding net is known to be stable, and the GANs [42] have been stabilized by many recent works, coupling the reconstruction and adversarial effects might still yield unstable results. In this work, we first illustrate the instability of the coupled structure. Stronger adversarial effect (i.e., higher weight on the discriminator loss) is supposed to achieve higher-fidelity images but may cause artifacts, unreality, etc., while weaker adversarial effect would be safe but weakening the boost on image quality. We propose a novel structure that decouples the two effects, avoiding the instability caused by their interaction [150]. A separate graph is learned to model the

adversarial effect, isolating the training of encoding-decoding network. A novel evaluation metric named *normalized relative discriminative score* (NRDS) is designed to numerically evaluated the performance compare with other works[1]. Experimental results demonstrate that the proposed decoupled learning effectively stabilizes the adversarial effect and achieve competitive imaging quality in multiple image generation tasks regardless of the trade-off issue.

4.1 Background and Motivation

The GAN [42] proposed an adversarial framework which generates images from noise while preserving high fidelity. However, generating "uncontrolled" images from noise doesn't meet the requirements in many real applications, e.g., image inpainting [98], image transformation [52], image manipulation [148, 153], etc. Therefore, some works [107, 94] concatenated additional features generated by an encoder or certain extractor to the random noise or replaced the noise by the features. Mostly, encoding-decoding networks (ED), e.g., VAE [60], AAE [86], autoencoder, etc., is incorporated with GANs. The encoder extracts features, which are fed to the decoder/generator to achieve target images. Although the encoding-decoding network can already stabilizes the generated images, it tends to yield blurry images under the common objectives, i.e., ℓ_1- or ℓ_2-norm. Incorporating a discriminator, as empirically demonstrated in many works [66, 52, 70, 148, 77, 154], effectively increases the fidelity/resolution of generated images from the encoding-decoding networks.

In existing works that incorporated encoding-decoding networks (ED) to GANs, the reconstruction effect (from the ED) and adversarial effect (from the discriminator) are both imposed on a single generator/decoder, making the two effects to be coupled and interact/compete with each other. Although the ED is known to be stable in training, and many GANs works, e.g., DCGAN [103], WGAN [4], LSGAN [88], etc., had stabilized the training of GANs, coupling between them may yield unstable results or introduce artifacts as shown in Fig.4.1. In Fig. 4.1, adding the discriminator loss (e.g. 0.001 or 0.01) increases the fidelity of generated images compared to the raw generation (the top raw in Fig. 4.1). But

[1]Github: https://github.com/ZZUTK/Decoupled-Learning-Conditional-GAN



Figure 4.1: Artifacts introduced by the adversarial effect. Please zoom in for more details. The top raw shows the generated images from ED. The rest rows show the images from the coupled structure (ED plus GAN) with different weights (left column) on the adversarial effect. Assume the weight of reconstruction loss is 1. The generated images with the weight of 0.01 present similar (or a bit higher) fidelity compared to those with the weight of 0.001, but they tend to be unrealistic. The generated images with the weight of 0.1 only present random spots without any boosting on fidelity, indicating instability during the training.

it may also bring obvious artifacts and tend to generate unrealistic results (e.g.,the 1st, 2nd faces with respect to 0.01 and 0.1). Generally, trade-off between the two effects is necessary, otherwise the generated images may present significant artifacts, e.g., stripe, overexposure, spots, or anything visually unreality. Existing works that balanced the two losses through exhaustive searching are time-consuming. However, how to set an appropriate weight or relax this dilemma is still unexplored.

In this chapter, we illustrate the sensitiveness of imaging quality to the adversarial effect, i.e., a higher weight on the adversarial loss preserves richer details in generated images but suffering higher risk of introducing significant artifacts or even causing instability, while a lower weight is safe but diluting the boosting effect from discriminator. We aim to solve the unstable issue induced by the coupled structures which is denoted as ED+GAN.[2]. A novel *decoupled learning* denoted as ED//GAN[3] that decouples the two effects, i.e. the reconstruction effect and adversarial effect is proposed to avoid the negative effect (i.e. blurry

[2]The coupled structures used in existing works are denoted as ED+GAN because they added the effects of ED and GAN together during the training.

[3] The proposed decoupled learning is denoted by ED//GAN, indicating that the effect from ED and GAN are learned independently.

results or induced artifacts) and generate stable results. The ED//GAN isolates GAN from ED, avoiding interaction between the two effects during training. Therefore, the weight of adversarial effect will not affect the training of ED, then the setting of weight becomes unnecessary. Based on existing works, ED with batch normalization is expected to be more stable compared to that without batch normalization. Under the framework of ED//GAN, stable training could be achieved regardless of batch normalization on ED to avoid a few unfortunate side [41]. Above all, the proposed ED//GAN stabilizes the adversarial effect and relaxes the trade-off issue. In addition, the ED//GAN provide a direct visualization of the adversarial effect as shown in Fig. 4.2 and Sec. 4.4.3 by introducing a single graph (generator) G. Through convincing visualization results and quantitative evaluations, we verify its effectiveness on many applications, i.e. image manipulation, image generation, image transformation.

4.2 Decoupled Learning

4.2.1 Theoretic Reasoning

In GAN-related applications, it is tricky and important to balance the effects of GAN and autoencoder. We give the first attempt to solve this problem. We focuses on stabilizing the widely used network structure ED+GAN, where ED appears to generate smooth/blurring results due to MSE in pixel space, while GAN drives results towards the natural image manifold producing more convincing solutions. However, incorporating the two parts as in existing works causes competition between the two forces, leading potentially to a bad solution. We aim to avoid such competition by training ED and GAN in an independent manner. Therefore, we preserve the structures of ED and GAN without sharing parameters, as compared to existing works where the parameters of decoder in ED and generator in GAN are shared. The independent network design explicitly decouples the interaction between ED and GAN, but still follows the classic objective functions — the reconstruction loss and minimax game for ED and GAN, respectively. Thus, any existing work based on ED+GAN can be easily adapt to the proposed structure without changing their objectives, meanwhile

gaining the benefit of not having to fine tune a balance between ED and GAN. In ED//GAN, the final output is obtained by adding the output from the generator to that from ED, thus the generator will render the residual between the blurry (from ED) and true samples, giving the essential insight on how GANs could boost the output of ED. We agree that rigid mathematical proof would have been ideal to support our claim. However, since the objective functions of ED+GAN and ED//GAN are similar, analytic proofs are not feasible here. This is a network design and thus training advantage, where through empirical study, we made a hypothesis on the potential cause of issues with ED+GAN, and proposed the decoupled structure design, which is validated through extensive experimental studies.

4.2.2 Network Structure

The proposed decoupled learning (ED//GAN), as shown in Fig. 4.2, incorporates the encoding-decoding network (ED) with GAN in a decoupled manner, i.e., G and Dec are trained separately corresponding to adversarial loss and reconstruction loss, respectively. Assuming the input image $I \in \mathbb{R}^{H \times W \times C}$, where H, W, and C denote the height, width, and number of channels, respectively. The reconstructed image from ED is I_{ED}, which is a blurred version of I. The generator G incorporates with the discriminator D to learn the

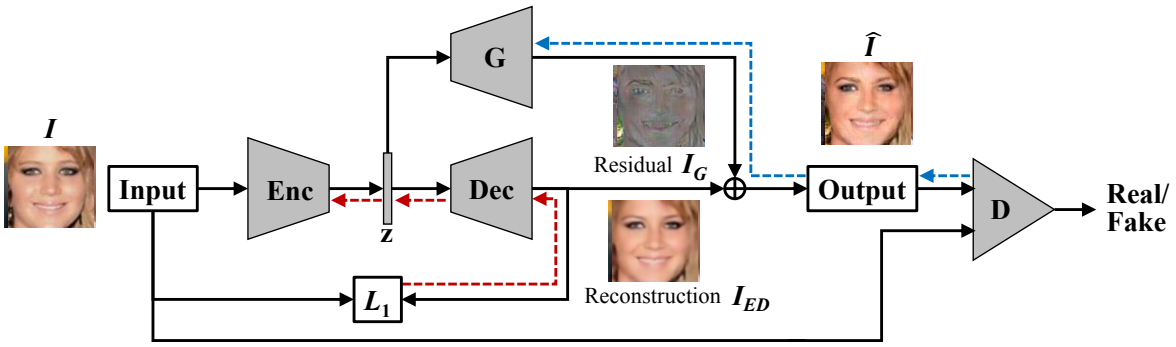


Figure 4.2: The flow of proposed decoupled learning, i.e., ED//GAN. Enc and Dec are the encoder and decoder networks, and G and D are the generator and discriminator, respectively. L_1 indicates the pixel-level ℓ_1-norm. Solid black arrows denote feedforward path, and dashed arrows in red and blue indicate backpropagation from reconstruction loss (L_1) and adversarial loss (from D), respectively. The reconstruction image is generated from Dec, and residual is yielded from G. The two images are added up to generated the final output image.

residual I_G which increase the resolution and reality . Ideally, $I \approx I_{ED} + I_G = \hat{I}$, which denotes the final output image. Unlike existing works that couple the learning of G and Dec (or together with Enc), which is equivalent to sharing the weight of G and Dec, we learn them separately. Therefore, the proposed decoupled learning can be graphically divided into two parts: 1) reconstruction learning of Enc and Dec (Sec. 4.2.2) and 2) adversarial learning of G and D (Sec. 4.2.2). Enc and Dec (i.e., ED) are trained independent of G and D (i.e., GAN), updated through the ℓ_1-norm in pixel level as shown by the red dashed arrow in Fig. 4.2. G and D are trained only based on the output of ED and updated by the adversarial loss as indicated by the blue dashed arrow. The final output image is obtained by pixel-wise summation of the outputs from G and Dec as indicated by the $\bigoplus$ marker.

In the proposed ED//GAN framework, the gradient derived from reconstruction and adversarial losses are directed in separated flows without any interaction, avoiding the competition between reconstruction and adversarial effects which may cause instability but widely present in existing works as discussed in section 4.1. The G performs like successive processing of ED, recovering details missed by the output from ED. The G and Dec share the latent variable because of the correspondence between blurry image I_{EG} and corresponding recoverable details I_G. For example, the details mainly contains the wrinkle, edges of eyes and mouth, etc, which vary conditioned on the feature of faces. Therefore, we assume that I_{EG} and I_G share the high-level features.

Reconstruction Learning

The encoding-decoding network (ED) aims to minimize the pixel-level error between the input image I and reconstructed image I_{ED}. The training of ED is well known to be stable, and ED could be any structures specifically for any applications, e.g., U-Net [52] or conditional network [148] with/without batch normalization. Most works that adopted batch normalization to enhance stability of the ED+GAN structure may bring a few unfortunate side effects [41] and hurt the diversity [148]. With the proposed ED//GAN, however, batch normalization becomes unnecessary because the training of ED is isolated from that of GAN, and ED itself could be stable without batch normalization. The reconstruction loss from the

ED part can be expressed by

$$\mathcal{L}_{const}(Enc, Dec) = \|I - Dec(Enc(I))\|_1 = \|I - Dec(z)\|_1 = \|I - I_{ED}\|_1, \tag{4.1}$$

where Enc and Dec indicate the functions of encoder and decoder, respectively. The latent variable derived from Enc is denoted by z. The reconstruction loss is computed by ℓ_1-norm in pixel level. The latent variable z could be constrained to certain prior distribution (e.g., Gaussian or uniform) to achieve generative ability like VAE [60] and AAE [86].

Adversarial Learning

In GANs, two competitive components, namely generator (G) and discriminator (D), are trained simultaneously through a minimax game. The D aims to discriminate the real samples (input images) and fake samples (generated by G), while G aims to fool D. In the proposed ED//GAN, GAN works different from the vanilla GAN in two aspects: 1) The inputs of G are features of input faces (sharing the latent variable with Dec) rather than random noise. 2) The fake samples fed to D are not directly generated by G. Instead, they are conditioned on the outputs from Dec. Therefore, the objective of GAN can be expressed by

$$\max_D \min_G \mathbb{E}\left[\log D(I)\right] + \mathbb{E}\left[\log\left(1 - D\left(I_{ED} + G(z)\right)\right)\right], \tag{4.2}$$

where $I_{ED} = Dec(z)$ and $z = Enc(I)$. For simplicity, we write the losses of G and D separately,

$$\mathcal{L}_{adv}(D) = \mathbb{E}\left[\log\left(1 - D(I)\right)\right] + \mathbb{E}\left[\log D\left(I_{ED} + I_G\right)\right], \tag{4.3}$$

$$\mathcal{L}_{adv}(G) = \mathbb{E}\left[\log\left(1 - D\left(I_{ED} + G(z)\right)\right)\right]. \tag{4.4}$$

Adversarial learning has shown great success in generating photo-realistic images in recent two years [12, 52, 154, 58, 126, 98]. However, the issues of GANs are instability and model missing. Some recent works, e.g., WGAN [4] and LSGAN [88], have stabilized the training of GANs but still cannot ensure the stability of ED+GAN because of the interaction with reconstruction effect. In ED//GAN, because the adversarial learning is decoupled from the

reconstruction learning, any improved GANs could be adopted to specifically ensure stable learning of the GAN part without affecting or be affected by the ED part. By the same token, it is unnecessary to trade off the reconstruction and adversarial effects like in ED+GAN. In addition, as demonstrated in [66, 12], incorporating GAN with ED can relax the mode missing issue because ED tends to imitate training samples (both major and minor models). The proposed ED//GAN inherits this property, making GAN generate residual based on the reconstruction (imitation) from ED. In other words, ED guides GAN to avoid mode missing.

Training of the Decoupled Learning

Compare to ED+GAN, the proposed ED//GAN has two distinguished differences in the training process: 1) the reconstruction and adversarial losses are backpropagated to separated networks, and 2) the trade-off problem between the two losses is relaxed. To generally distinguish the proposed ED//GAN from the existing ED+GAN, Fig. 4.3 illustrates their network structures and backpropagation path.

In ED+GAN, both reconstruction and adversarial losses are backpropagated to the Dec, and the general objective could be written as

$$\min_{Enc,Dec,D} \mathcal{L}_{const} + \lambda \mathcal{L}_{adv} \quad \text{or} \quad \min_{Enc,Dec,D} \lambda \mathcal{L}_{const} + \mathcal{L}_{adv}, \tag{4.5}$$

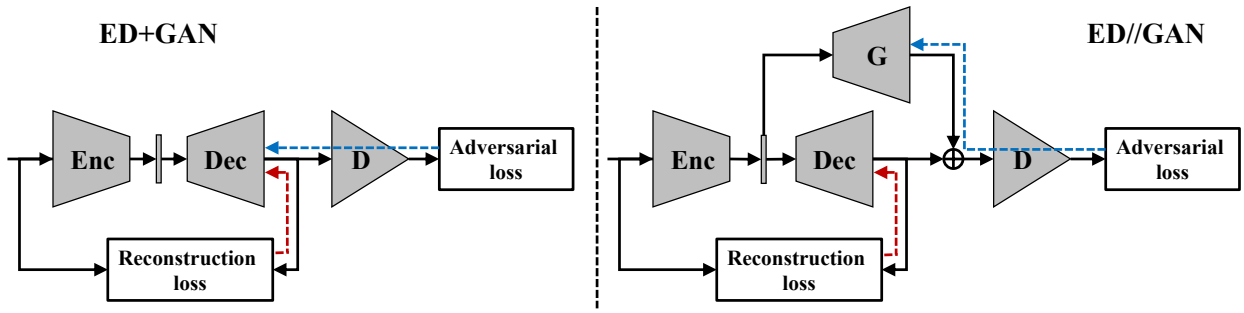


Figure 4.3: Comparison between the ED+GAN and ED//GAN. Left: the existing ED+GAN. Right: the proposed ED//GAN, i.e., decoupled learning. Solid black arrows denote feedforward path, and dashed arrows in red and blue indicate backpropagation from reconstruction loss and adversarial loss, respectively.

where $\mathcal{L}_{const}$ and $\mathcal{L}_{adv}$ denote the reconstruction and adversarial losses, respectively. The parameter λ is the weight to balance the two losses. During the training, $\mathcal{L}_{const}$ updates Enc and Dec,and $\mathcal{L}_{adv}$ updates D and Dec (or together with Enc). The Dec will be simultaneously updated by the two losses. In ED//GAN, we could relax the weight λ in Eq. 4.5, and the general objective for ED//GAN will be

$$\min_{Enc,Dec,G,D} \mathcal{L}_{const} + \mathcal{L}_{adv}. \tag{4.6}$$

Assembling Eqs. 4.1, 4.3, and 4.4, we obtain the objective of the proposed decoupled learning (ED//GAN),

$$\min_{Enc,Dec,G,D} \mathcal{L}_{const}(Enc, Dec) + \mathcal{L}_{adv}(G) + \mathcal{L}_{adv}(D) \tag{4.7}$$

$$\Rightarrow \min_{Enc,Dec} \mathcal{L}_{const}(Enc, Dec) + \min_{G} \mathcal{L}_{adv}(G) + \min_{D} \mathcal{L}_{adv}(D). \tag{4.8}$$

Note that there are no weighting parameters between the losses in the objective function, which relaxes the manual tuning that may require an expert with strong domain knowledge and rich experience. In the training, each part could be updated alternatively and separated as shown in Eq. 4.8 because the three parts do not overlap in backpropagation, i.e., the backpropagation paths are not overlapped. As illustrated in Fig. 4.3 (right), the reconstruction loss (L_1) updates Enc and Dec, and the adversarial loss updates G. The D is training by input and output images, i.e., I and $\hat{I}$. In addition the decoupled structure G visualize the mechanism about how the discriminator will increase the photo reality/fidelity, which remedy the limited work to investigate the effect of discriminator.

4.3 Normalized Relative Discriminative Score

In the evaluation of image quality (e.g., reality and resolution), how to design a reliable metric for generative models has been an open issue. Existing metrics (e.g., inception score [113] and related methods [96]), although successful in certain cases, have been demonstrated to be problematic in others [90]. If a perfect metric exists, the training of generative

models would be much easier because we could use such metric as loss directly without training a discriminator. The rationale behind our design is that if it is difficult to obtain the absolute score (perfect metric) of a model, we could at least compare which model generates better images than others. From this perspective, we propose to perform relative comparison rather than providing evaluation based on absolute score like existing works. More specifically, we train a single discriminator/classifier to separate real samples from generated samples, and those generated samples closer to real ones will be more difficult to be separated. For example, given two generative models G_1 and G_2, which define the distributions of generated samples p_{g1} and p_{g2}, respectively. Suppose the distribution of real data is p_{data}, if $JSD(p_{g1}|p_{data}) < JSD(p_{g2}|p_{data})$ where JSD denotes the Jensen-Shannon divergence and assume p_{g1} and p_{g2} intersect with p_{data}, a discriminator/classifier D trained to classify real samples as 1 and 0 otherwise would show the following inequality,

$$\mathbb{E}_{x\sim p_{data}}[D(x)] \geq \mathbb{E}_{x\sim p_{g1}}[D(x)] \geq \mathbb{E}_{x\sim p_{g2}}[D(x)]. \tag{4.9}$$

The main idea is that if the generated samples are closer to real ones, more epochs would be needed to distinguish them from real samples. The discriminator is a binary classifier to separate the real samples from fake ones generated by all the models in comparison. In each epoch, the discriminator output of each sample is recorded. The average discriminator output of real samples will increase with epoch (approaching 1), while that of generated samples from each model will decrease with epoch (approaching 0). However, the decrement rate of each model varies based on how close the generated samples to the real ones. Generally, the samples closer to real ones show slower decrement rate. Therefore, we compare the "decrement rate" of each model to relatively evaluate their generated images. The decrement rate is proportional to the area under the curve of average discriminator output versus epoch. Larger area indicates slower decrement rate, implying that the generated samples are closer to real ones. Fig. 4.4 illustrates the computation of normalized relative discriminative score (NRDS).

There are three steps to compute the proposed normalized relative discriminative score (NRDS): 1) Obtain the curve $\mathcal{C}_i$ $(i = 1, 2, \cdots, n)$ of discriminator output versus epoch (or

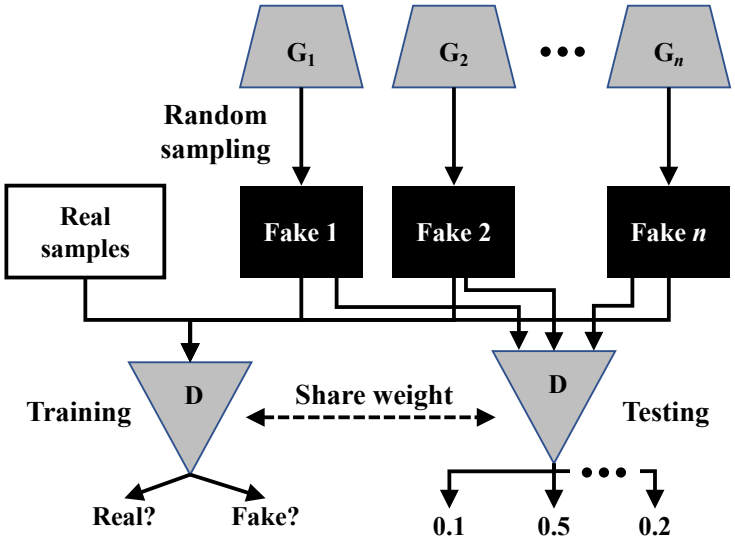


Figure 4.4: Illustration of NRDS. G_n indicates the nth generative model, and its corresponding fake samples are Fake n, which are smapled randomly. The fake samples from n models, as well as the real samples, are used to train the binary classifier D (bottom left). Testing only uses fake samples and performs alternatively with the traing process. The bottom right shows an example of averaged output of D from fake samples of each model.

mini-batch) for each model (assuming n models in comparison) during training; 2) Compute the area under each curve $A(\mathcal{C}_i)$; and 3) Compute NRDS of the ith model by

$$NRDS_i = \frac{A(\mathcal{C}_i)}{\sum_{j=1}^{n} A(\mathcal{C}_j)}. \tag{4.10}$$

To illustrate the computation of NRDS, Fig. 4.5 shows a toy example. Assume the samples named "fake-close" and "fake-far" are generated from two different models to simulate the real samples. We train a discriminator on the real and fake (i.e., fake-close and fake-far) samples. The structure of discriminator is a neural network with two hidden layers, both of which have 32 nodes, and ReLU is adopted as the activation function. After each epoch of training on the real and fake samples, the discriminator is tested on the same samples from real, fake-close, and fake-far, respectively. For example, all the real samples are fed to the discriminator, and then we compute the mean of the outputs from the discriminator. By the same token, we can obtain the average outputs of fake-close and fake-far, respectively. With 300 epochs, we plot the curves shown in Fig. 4.5 (right). Intuitively, the curve of fake-close approaches zero slower than that of fake-far because the samples in

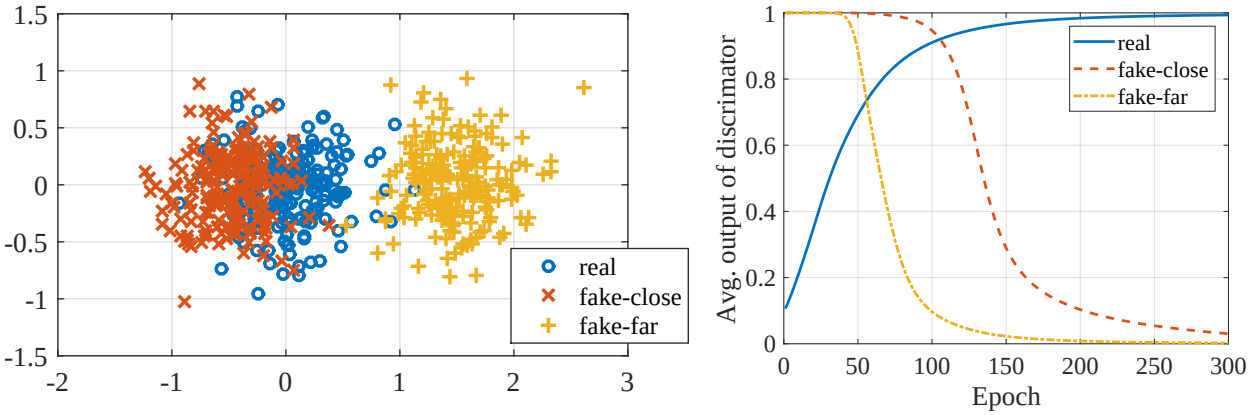


Figure 4.5: A toy example of computing NRDS. Left: the real and fake samples randomly sampled from 2-D normal distributions with different means but with the same (identity) covariance. The real samples (blue circle) is with zero mean. The red "x" and yellow "+" denote fake samples with the mean of $[0.5, 0]$ and $[1.5, 0]$, respectively. The notation fake-close/far indicates that the mean of correspondingly fake samples is close to or far from that of the real samples. Right: the curves of epoch vs. averaged output of discriminator on corresponding sets (colors) of samples.

fake-close are closer (similar) to the real samples. The area under the curves of fake-close ($\mathcal{C}_1$) and fake-far ($\mathcal{C}_2$) are $A(\mathcal{C}_1) = 145.4955$ and $A(\mathcal{C}_2) = 71.1057$, respectively. From Eq. 4.10,

$$NRDS_1 = \frac{A(\mathcal{C}_1)}{\sum_{i=1}^{2} A(\mathcal{C}_i)} = 0.6717 \tag{4.11}$$

$$NRDS_2 = \frac{A(\mathcal{C}_2)}{\sum_{i=1}^{2} A(\mathcal{C}_i)} = 0.3283. \tag{4.12}$$

Therefore, we can claim that the model generating fake-close is relatively better. Note that the actual value of NRDS for certain single model is meaningless. We can only conclude that the model with higher NRDS is better than those with lower NRDS in the same comparison, but a high NRDS does not necessarily indicate an absolutely good model.

4.4 Experimental Evaluation

This section will evaluate the proposed decoupled learning in two aspects corresponding to the two contributions claimed in Sec. 4.1: 1) visualizing the adversarial effect independently

from the effect of reconstruction loss and 2) demonstrating the advantage of decoupled learning in relaxing the parameter setting and stabilizing the training process. We mainly compare the proposed ED//GAN to the traditional ED+GAN.

4.4.1 Datasets

Four datasets are used to evaluate the effectiveness and stability of the proposed decoupled learning: 1) UTKFace [148], 2) Caltech-UCSD Birds 200 (CUB-200) [134], 3) Oxford Flower [95] and 4) CMP Facade Database [104]. The UTKFace dataset consists of about 20,000 aligned and cropped faces with large diversity in age and race. The decoupled learning applied on the UTKFace dataset aims to demonstrate the performance on image manipulation tasks. The CUB-200 dataset has 6,033 images of 200 birds species with large and diverse background, including ocean, trees, flowers, etc. The Oxford Flower dataset are images of flowers with diverse species and colors. The decoupled learning is applied on these two datasets to demonstrate the stability and generalization on image generation tasks. The CMP Facade dataset is utilized to illustrate the performance of the decoupled learning on image transformation tasks. These four datasets are in different domains (because of different objects and background). The experimental results validate the robustness and stability of the decoupled learning in parameter relaxation, i.e., guaranteeing stable training without parameter tuning on any datasets.

4.4.2 Implementation

In the experiment, we implement the "Enc" and "Dec" (Fig. 4.2) by convolutional and deconvolutional networks, respectively. The details of the networks are shown in Table 4.1.

In both Enc and Dec, the kernel size is 5×5, and the stride is 2. The activation function is ReLU for each hidden layer, the output layer adopts hyperbolic tangent function. Batch

Table 4.1: Structure of the "Enc" and "Dec" networks. "$m^2 \times n$" denotes the m-by-m maps with n channels. The value 50 indicates a vector with the length of 50.

Enc	$128^2 \times 3 \rightarrow 64^2 \times 64 \rightarrow 32^2 \times 128 \rightarrow 16^2 \times 256 \rightarrow 8^2 \times 512 \rightarrow 4^2 \times 1024 \rightarrow 50$
Dec	$50 \rightarrow 4^2 \times 1024 \rightarrow 8^2 \times 512 \rightarrow 16^2 \times 256 \rightarrow 32^2 \times 128 \rightarrow 64^2 \times 64 \rightarrow 128^2 \times 3$

normalization is optimally applied before ReLU. This network is neither specifically designed for any applications nor dedicatedly fine tuned to achieve the best result. The goal is to demonstrate the stability of the propose method (ED//GAN) compared to the traditional method (ED+GAN). Therefore, we only need to ensure that both methods are fairly compare on the same structure and hyperparameters.

The generator G uses the same structure as Dec, and they share the latent variable z. The discriminator D adopts the similar structure as Enc, but the length of output is 1 (indicating real and fake) instead of 50 (the latent variable z). In G and D networks, batch normalization is applied to ensure stable training of GAN as suggested in DCGAN [103]. Note that the Enc and Dec networks could be replaced by any encoding-decoding networks depending on different applications, e.g., UNet [52], ResNet [47], Conditional network [148], etc. Similarly, the G and D networks could also be any improved GANs, e.g., WGAN, LSGAN, SBGAN, etc., which are supposed to stabilize the GAN training or speed up convergence. The propose method tends to stabilize the cooperation between ED and GAN without caring about balancing the two parts.

Above all, ED+GAN and ED//GAN are using the same network structure in ED (i.e., Enc and Dec) and GAN (i.e., G and D). The only difference is that Dec and G are tied (sharing weight) in ED+GAN, while Dec and G are independent in ED//GAN. More details about the network structures used in this section is displayed in Appendix A.

4.4.3 Boosting Effect from Adversarial Learning

The learned residual in ED//GAN is considered as the boosting effect from the adversarial learning (discriminator) without coupling the reconstruction effect. In other words, the residual illustrates where the generated images from reconstruction of ED is boosted in pixel level. Generally, the images from ED tend to be blurry, and the residual from GAN carry the details or textures. Imposing the residual onto reconstructed images is supposed to yield higher-fidelity images compared to the reconstructed images. Fig. 4.6 illustrates the learning results from the proposed ED//GAN. The examples are randomly selected with different age, race, expression, and pose.

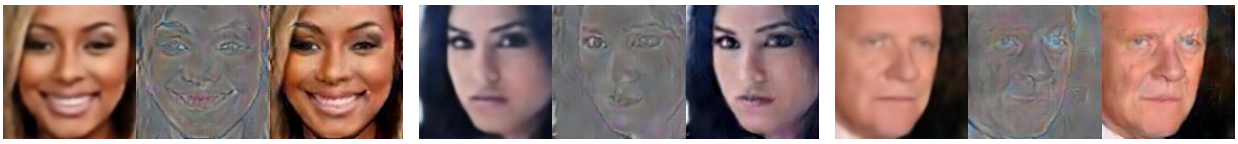


Figure 4.6: Visualization of the adversarial effect — residual learned by the proposed decoupled learning (ED//GAN). From left to right in each triple: reconstruction, residual, and output images.

In this experiment, Eq. 4.8 is optimized without setting the weight setting, and the results are reported after 200 epochs with batch size of 25. From the residual (Fig. 4.6 middle), we can observe that the adversarial effect mainly enhance the edges at eyebrow, eyes, mouth, teeth, etc. Adding the residual to the blurry images from ED (Fig. 4.6 left), the output images (Fig 4.6 right) present richer details.

4.4.4 Stabilizing the Training Process

To evaluate the stability of the proposed decoupled learning regardless of weight setting and batch normalization, we compare the proposed decoupled learning (ED//GAN) to the traditional method (ED+GAN). Two factors will be considered: 1) the weight of adversarial loss and 2) batch normalization which is common way to stabilize the training process. For example, comparing [52] and [148], the former adopted batch normalization in ED but the latter did not, so the former achieved stable training with higher weight of adversarial loss. We increase the weight of adversarial loss to compare the quality of generated images from the two methods. We fix the weight of reconstruction as 1 and increase the weight of adversarial loss from 0.001 to 1 with the step of 10x. After 200 epoch with the batch size of 25, Figs. 4.7 and 4.8 compares the output images with and without batch normalization, respectively.

The output images from ED//GAN stably yields relatively higher-fidelity images regardless of the weight changing. However, the outputs of ED+GAN are significantly affected by the weight. It obtains relatively better results when the weight of adversarial loss is 0.001. As the weight increases, the outputs becomes unstable, i.e., the images present significant artifacts and fall into a few modes.

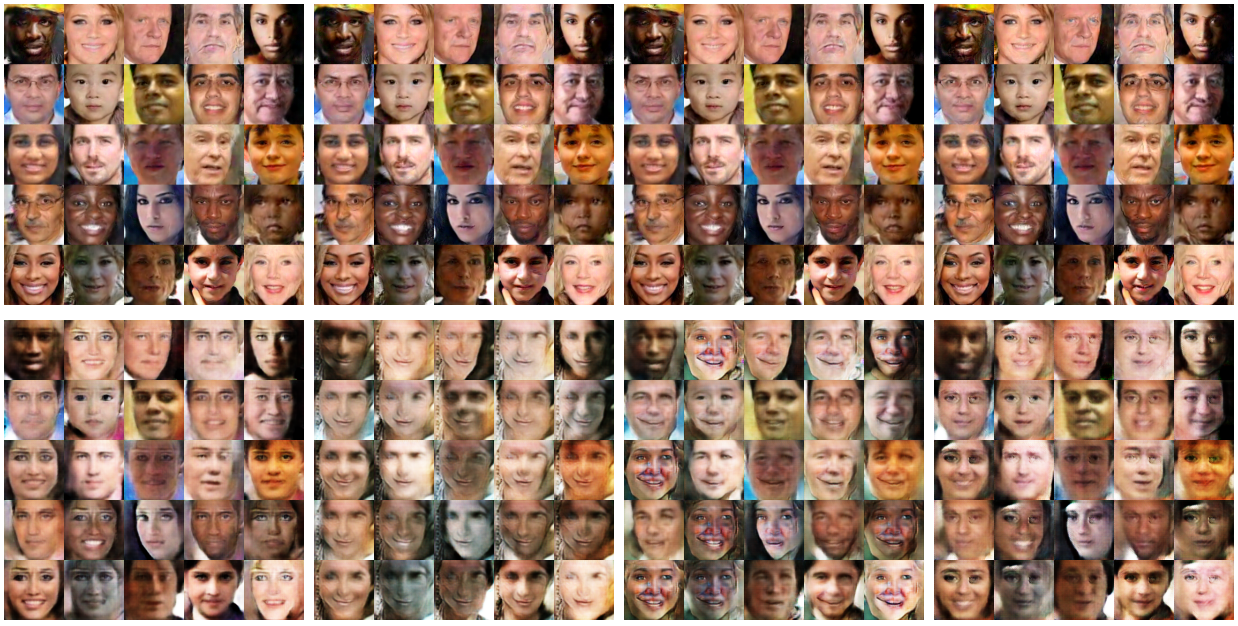


Figure 4.7: Comparison of ED//GAN (top) and ED+GAN (bottom) with batch normalization on ED on UTKFace Dataset. From left to right, the weights of adversarial loss are 0.001, 0.01, 0.1, and 1, respectively. Please zoom in for better view.

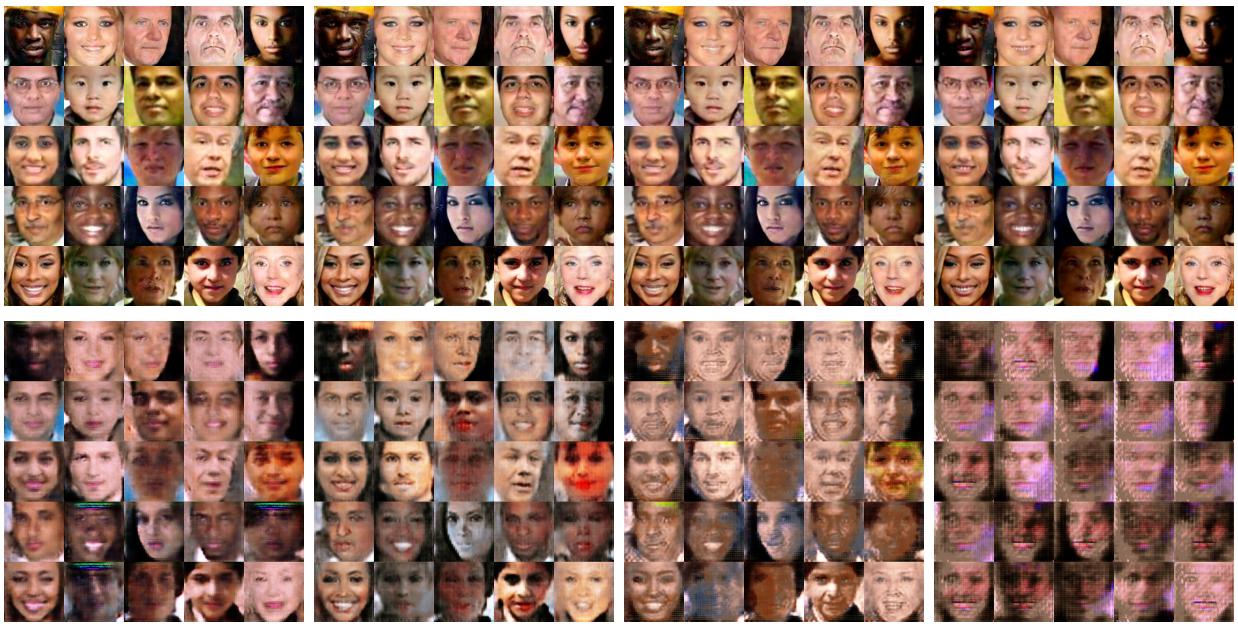


Figure 4.8: Comparison of ED//GAN (top) and ED+GAN (bottom) without batch normalization on ED on UTKFace dataset. From left to right, the weights of adversarial loss are 0.001, 0.01, 0.1, and 1, respectively.

In Fig. 4.8, the batch normalization in ED is removed. The proposed ED//GAN can still yield stable output, while the ED+GAN generates images with stripe, spots, noise, etc.

Form the two experiments, ED//GAN vs. ED+GAN with/without batch normalization on ED, we can observe that the proposed decoupled learning performs stable and is insensitive to the weight setting. Because of the robustness to weight, ED//GAN can relax the weight setting. The NRDS results illustrated in Figs. 4.7 and 4.8 are listed in Tables 4.2 and 4.3, respectively. In each table, testing results from all methods (with different weight settings) are collected to train a single discriminator, i.e., $i = 8$ in Eq. 4.10. Tables 4.2 and 4.3 yield the same conclusion as the visual results — ED//GAN stabilize the training regardless of the trade-off issue because ED//GAN preserves relatively higher NRDS compared to ED+GAN. In addition, ED//GAN shows similar NRDS at different weights.

We apply the proposed ED//GAN structure on CUB-200 and Oxford Flower dataset without any parameters changing to further demonstrate the generality and stability. Fig. 4.9 displays the results after 200 epochs. The outputs gain more details compared to the reconstructed images. The residual illustrates that both details and colors are enhanced by the adversarial effect.

Table 4.2: NRDS on the results partially illustrated in Fig. 4.7

Method	ED+GAN				ED//GAN			
	0.001	0.01	0.1	1	0.001	0.01	0.1	1
NRDS	.1066	.1143	.1268	.1267	.1320	.1300	.1300	.1336

Table 4.3: NRDS on the results partially illustrated in Fig. 4.8

Method	ED+GAN				ED//GAN			
	0.001	0.01	0.1	1	0.001	0.01	0.1	1
NRDS	.1172	.1143	.1163	.0731	.1432	.1434	.1458	.1466



Figure 4.9: Output of ED//GAN trained on the CUB-200 and Oxford Flower datasets. From left to right in each triple: reconstruction, residual, and output images.

4.4.5 Adapting Existing Works into ED//GAN

To illustrate the effectiveness of ED//GAN, we modify on some existing works from ED//GAN to ED//GAN structure and compare the generated images before and after the modification. We will modify on two works: 1) Pix2Pix [52] for image transformation and 2) CAAE [148] for image manipulation.

Image transformation — Pix2Pix

The Pix2Pix [52] is a representative work in image transformation, which used the ED+GAN structure. We modify it by parallelizing an extra generator to the original one to learn the residual. The new generator copies the structure of the original one. Then, reconstruction loss is only applied on the original generator, and the adversarial loss is applied on the new generator. The final output is obtained by adding the outputs from the two generators. The weight of reconstruction and adversarial losses is set to be 1 and 100 (i.e., 1:100) in the original work. In Fig. 4.10, We try the weighs of 1:1, 1:100, and 1:1000 for the original structure and compare to the modified version.

From the comparison, generated images with the weight of 1:1 introduce significant artifacts (please zoom in for better view). With higher weight on the reconstruction loss,



Figure 4.10: Comparison between Pix2Pix [52] and the modified version using the proposed ED//GAN. Pix2Pix generates images at different weights as denoted by λ:1, where λ and 1 indicate the weights of reconstruction and adversarial losses, respectively. ED//GAN denotes the generated images from modified version.

100:1 and 10:1 yield more realistic images, whose quality is similar to that from the modified version that doesn't need weight setting.

Image manipulation — Face Aging

The CAAE [148] is also in the ED+GAN structure for face aging task by manipulate the latent variable z. Following the same way in Sec. 4.4.5 to modify CAAE to the ED//GAN version. Fig. 4.11 shows some random examples to compare the original and modified structures. The weights of reconstruction and adversarial losses are 1 and 10^{-4} (i.e., 1:10^{-4}) in the original work. We try 1:10^{-4}, 1:10^{-3}, 1:10^{-2}, and 1:1 for the original structure and compare to the modified version.

Evaluating based on the proposed NRDS, the results illustrated in 4.10 and 4.11 are listed in Table 4.4. In each subtable, testing results from all methods (with different weight settings) are collected to train a single discriminator, i.e., $i = 4$ in Eq. 4.10. The proposed ED//GAN does not necessarily achieve the best result as shown in Table 4.4 (left), where ED//GAN ranks two. However, ED//GAN could be close to the optimal results without parameter tuning.

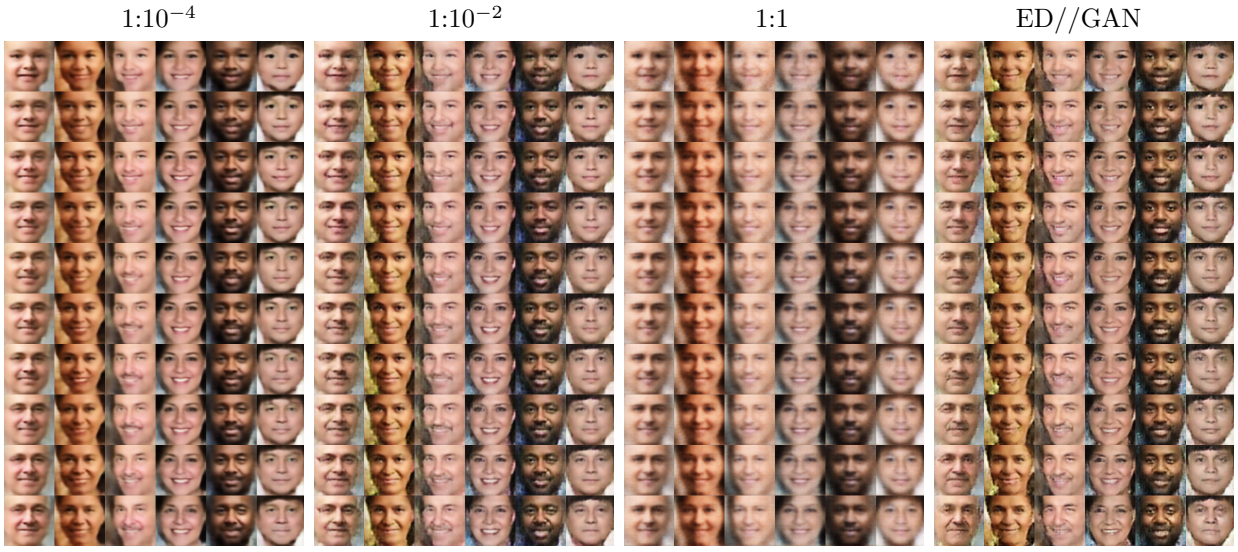


Figure 4.11: Comparison between CAAE [148] and the modified version using the proposed ED//GAN. CAAE generates images at different weights as denoted by 1:λ, where 1 and λ indicate the weights of reconstruction and adversarial losses, respectively. ED//GAN denotes the generated images from the modified version.

Table 4.4: NRDS on the results partially illustrated in Fig. 4.10 (left) and Fig. 4.11 (right)

Method	ED+GAN			ED//GAN
	1:1	100:1	1000:1	
NRDS	.2190	**.2641**	.2572	.2597

ED+GAN			ED//GAN
$1{:}10^{-4}$	$1{:}10^{-2}$	1:1	
.2527	.2496	.2430	**.2547**

4.5 Summary

This chapter proposed the decoupled learning (ED//GAN), improving the stability of the common structure of incorporating ED and GAN, i.e., ED+GAN. We illustrate that the instability of ED+GAN is introduced by the interaction between the reconstruction loss (from ED) and adversarial loss (from GAN). The ED//GAN decouples the two effects by training separated networks for the two effects, avoiding interaction between them and stabilizing the training process. This novel structure relaxes the trade-off issue that the ED+GAN suffers from because of the decoupling. The experimental results demonstrated the stability of the proposed decoupled learning. Existing works is easy to be adapted into ED//GAN by adding an extra generator that leans the residual. The ED//GAN could ensure stable and competitive results without parameter tuning. However, it is unnecessary to beat the best result from fine tuned ED+GAN.

Chapter 5

Reference-Conditioned Image Super-Resolution

With the recent advancement in deep learning, we have witnessed a great progress in single image super-resolution. However, due to the significant information loss of the image downscaling process, it has become extremely challenging to further advance the state-of-the-art, especially for large upscaling factors. This chapter explores a new research direction in super resolution, called reference-conditioned super-resolution, in which a reference image containing desired high-resolution texture details is provided besides the low-resolution image. We focus on transferring the high-resolution texture from reference images to the super-resolution process without the constraint of content similarity between reference and target images, which is a key difference from previous example-based methods. Inspired by recent work on image stylization, we address the problem via neural texture transfer. We design an end-to-end trainable deep model which generates detail enriched results by adaptively fusing the content from the low-resolution image with the texture patterns from the reference image[1] [151]. We create a benchmark dataset for the general research of reference-based super-resolution, which contains reference images paired with low-resolution inputs with varying degrees of similarity. Both objective and subjective evaluations demonstrate the great potential of using reference images as well as the superiority of our results over other state-of-the-art methods.

[1]Project page: http://web.eecs.utk.edu/~zzhang61/project_page/SRNTT/SRNTT.html

5.1 Introduction

The traditional single image super-resolution (SR) problem is defined as recovering a high-resolution (HR) image from its low-resolution (LR) observation [143], which has received substantial attention in the computer vision community. As in other fields of computer vision studies, the introduction of convolutional neural networks (CNNs) [25, 141, 57, 73] has greatly advanced the state-of-the-art performance of SR. However, due to the ill-posed nature of SR problems, most existing methods still suffer from blurry results at large upscaling factors, e.g., 4×, especially when it comes down to the recovery of fine texture details in the original HR image which are lost in the LR counterpart. In recent years, perceptual-related constraints, e.g., perception loss [53] and adversarial loss [42], have been introduced to the SR problem formulation, leading to major breakthroughs on visual quality under large upscaling factors [69, 112]. However, they tend to add fake textures and even artifacts to make the SR image of visually higher-resolution.

This work diverts from the traditional SR and explores a new research direction — reference-conditioned super-resolution, which utilizes the rich textures from HR references to compensate for the lost details in the LR images, relaxing the ill-posed issue and producing more detailed and realistic textures with the help of reference images. The reference images may come from photo albums, video frames, or web image search. There are existing example-based SR approaches [33, 11, 32, 146, 129, 76] that adopted external high-frequency information to enhance textures. However, they assume the reference images could be well aligned or present similar texture to the LR images. By contrast, the reference image plays a different role in our setting: it does not need to have similar content with target HR image. Instead, we only intend to transfer the relevant texture from reference image to target image.

Inspired by recent work on image stylization [38, 53, 15], we propose the Super-Resolution by Neural Texture Transfer (SRNTT), which *adaptively* transfers textures to the SR image conditioned on the reference image. More specifically, SRNTT conducts local texture matching in the high-level feature space and adaptively fuses matched textures with a deep model. Fig. 5.1 illustrates the advantage of the proposed SRNTT compared with two representative works, SRCNN [25, 26] and SRGAN [69], which are without and with

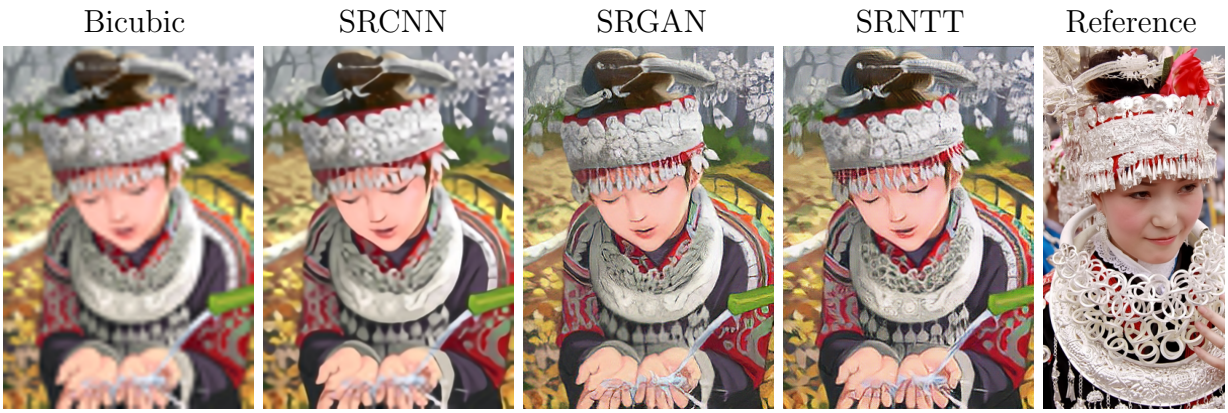


Figure 5.1: From left to right: bicubic interpolation, SRCNN that minimizes MSR, SRGAN that further incorporates perceptual-related constraints, the proposed SRNTT conditioned on the reference shown in the upper-right corner, and the reference image for SRNTT. The upscaling factor is 4×.

perceptual-related constraints, respectively. SRNTT shows improved texture transferred from the reference. Note that the texture in reference does not have to match the one in HR ground truth. We emphasize the capacity of SRNTT in handling arbitrary reference images, i.e., section 5.5.5 demonstrates the extreme case of the reference image being simply random noise, where the proposed SRNTT is still able to generate the SR image with comparable visual quality as that from SRGAN.

That being said, similarity between the reference and LR image is still the key factor that affects the performance of reference-conditioned SR. However, there is no existing benchmark dataset that could provide different similarity levels of references for the investigation of adaptiveness and robustness. To facilitate fair comparison and further research on the reference-conditioned SR problem, we propose a new dataset, named CUFED5, which provides training and testing sets accompanied with references of five similarity levels that vary in content, texture, color, illumination, and view point (section 5.4).

The main contributions of this work are:

- We explore a new research direction of SR, i.e., reference-conditioned super-resolution, as opposed to SISR which only relies on the LR input image, and example-based SR which makes rigid assumptions on the external example image used. Reference-conditioned SR aims to generate HR texture information for LR input image by referring to an arbitrary external image, and thus enables the generation of SR images

with plausible texture details even at large upscale factor, further advancing the state-of-the-art in SR research.

- We propose an end-to-end deep model, SRNTT, to recover the LR image conditioned on any given reference. We demonstrate the adaptiveness, effectiveness, and visual improvement of the proposed SRNTT by extensive empirical studies.

- We create a benchmark dataset, CUFED5, to facilitate the performance evaluation of SR methods in handling references with different levels of similarity to the LR input image.

In the rest of this chapter, we review the related works in Section 5.2. The network architecture and training criteria are discussed in Section 5.3. In Section 5.4, the proposed dataset CUFED5 is described in detail. The results of quantitative and qualitative evaluations are presented in Section 5.5. Finally, Section 5.6 concludes this chapter.

5.2 Related Works

5.2.1 Deep Learning based Single Image SR

In recent years, deep learning based SISR has shown superior performance in either PSNR or visual quality compared to those non-deep-learning based methods [25, 141, 69]. The reader could refer to [93, 143] for more comprehensive review of SR. Here we only cover deep learning based methods.

A milestone work that introduced CNN into SR was proposed by Dong et al. [25], where a three-layer fully convolutional network was trained to minimize MSE between the SR image and original HR image. It demonstrated the effectiveness of deep learning in SR and achieved the state-of-the-art performance. Wang et al. [141] combined the strengths of sparse coding and deep network and made considerable improvement over previous models. To speed up the SR process, Dong et al. [27] and Shi et al. [117] extracted features directly from the LR image, that also achieved better performance compared to processing the upscaled LR image through bicubic. To further reduce the number of parameters, Lai et al. [64] progressively

reconstructed the sub-band residuals of HR images at multiple pyramid levels. In recent two years, the state-of-the-art performance (in PSNR) were all achieved by deep learning based models [57, 56, 73].

The above mentioned methods, in general, aim at minimizing the mean squared error (MSE) between the SR and HR images, which might not always be consistent with the human evaluation results (i.e., perceptual quality) [69, 112]. As a result, perceptual-related constraints were incorporated to achieve better visual quality. Johnson et al. [53] demonstrated the effectiveness of adding perception loss using VGG [119] in SR. Ledig et al. [69] introduced adversarial loss from the generative adversarial nets (GANs) [42] to minimize the perceptually relevant distance between the SR and HR images. Sajjadi et al. [112] further incorporated the texture matching loss based on the idea of style transfer [37, 38], to enhance the texture in the SR image. The proposed SRNTT is more related to [69, 112], where perceptual-related constraints (i.e., perceptual loss and adversarial loss) are incorporated to recover more visually plausible SR images.

5.2.2 Example-based SR Methods

In contrast to SISR where only the single LR image is used as input, example-based SR methods introduce additional images to assist the SR process. In general, the example images need to possess very similar texture or content structure with the LR image. The examples could be selected from adjacent frames in a video [75, 10], images from web retrieval [146], or self patches [33, 32]. We will not discuss video (multi-frame) super-resolution which are specifically designed taking advantage of the similarity nature of adjacent frames. The proposed reference-conditioned SR allows a more relaxed scenario — the reference could be an arbitrary image.

Those early works [33, 11] not using deep learning mostly built the mapping from LR to HR patches and fused the HR patches at the pixel level and by a shallow model, which is insufficient to model the complicated dependency between the LR image and extracted details from the HR patches, i.e., examples. In addition, they implied that each LR patch could be matched to an appropriate HR patch (similar textures always present in the example). Freedman and Fattal [32] and Huang et al. [50] referred to self examples for

similar textures and used a shallow model for texture transfer. A more generic scenario of utilizing the examples was proposed by Yue et al. [146], which instantly retrieved similar images from web and conducted global registration and local matching. However, they made a strong assumption on the example – the example has to be well aligned to the LR image. In addition, the shallow model for patch blending made its performance highly dependent on how well the example could be aligned, limiting its adaptiveness to a more generic setting. The proposed SRNTT adopts the ideas of local texture matching and texture fusion like existing works, but we perform with high-level features and deep models targeting at the most generic scenario where the reference images can be arbitrary.

5.3 Approach

The reference-conditioned image super-resolution aims to estimate the SR image I^{SR} from its LR counterpart I^{LR} and the given reference image I^{Ref}, increasing plausible textures conditioned on the reference, while preserving the consistency to the LR image in color and content. In the proposed SRNTT, beyond minimizing the distance between I^{SR} and the original HR image I^{HR} as most existing SR methods do, we further regularize on the texture consistency between I^{SR} and I^{Ref}. The general objective could be expressed by

$$\hat{\theta} = \arg\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} \left\{ \mathcal{L}_c \left(G_\theta(I_i^{LR}), I_i^{HR} \right) + \lambda \mathcal{L}_t \left(G_\theta(I_i^{LR}), I_i^{Ref} \right) \right\}, \tag{5.1}$$

where G denotes the SR network with parameter θ. $\mathcal{L}_c(\cdot,\cdot)$ and $\mathcal{L}_t(\cdot,\cdot)$ indicate the content loss and texture loss, respectively. For simplicity, assume each LR image corresponds to one reference, and there are n LR images. Section 5.3.3 will discuss the loss functions in detail.

An overall structure of the proposed SRNTT is shown in Fig. 5.2. The main idea is to search for locally matched textures from the reference and adaptively transfer these textures to the SR image. We design the structure as fusing two parallel streams, i.e., content and texture, which is consistent to the intuition of fusing texture to content. The content and texture are represented as high-level features extracted through deep models (i.e., content extractor and texture extractor, respectively), facilitating deep texture fusion which is more

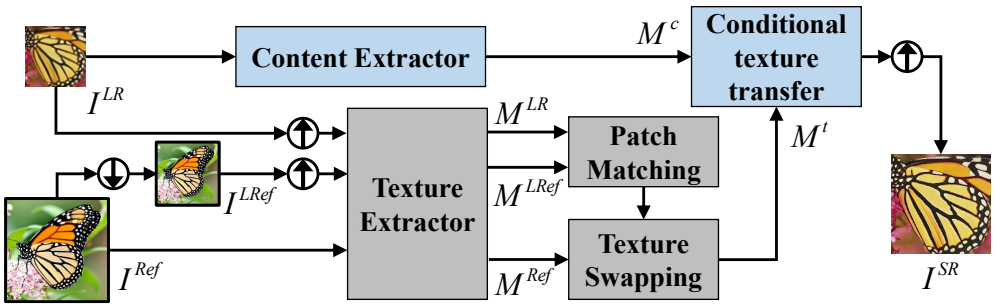


Figure 5.2: Overview of the proposed SRNTT. I^{LR} and I^{Ref} are the inputs to SRNTT. I^{LR} is fed to the content extractor, obtaining the content feature map M^c. In parallel, I^{Ref} and I^{LR} are fed to the texture extractor after scale adjustment, yielding a series of feature maps for patch matching and texture swapping. The swapped texture map M^t carries rich texture from I^{Ref} while preserving the content structure of I^{LR}. Finally, M^c and M^t are adaptively fused through the conditional texture transfer, producing I^{SR} with enhanced texture.

adaptive than using a shallow model. The content feature M^c extracted from I^{LR} and the texture feature M^t extracted from I^{Ref} are fused by the conditional texture transfer, which could learn to adaptively transfer perceptually consistent texture from M^t to M^c. More details on conditional texture transfer will be discussed in Section 5.3.2.

The content feature map M^c is directly extracted from I^{LR} through the content extractor. The corresponding texture feature map M^t is obtained by local matching between the LR image I^{LR} and reference I^{Ref}. Because I^{LR} and I^{Ref} may differ from each other in color and illumination, affecting texture matching and transfer, we choose to perform in the high-level feature space where most content structure as well as texture-related information are preserved. Typically, the VGG19 [119] model is adopted as the texture extractor, whose effectiveness on texture representation has been demonstrated by many empirical studies [37, 38, 74]. To offset the bias from scale/resolution in patch matching, I^{LR} and I^{Ref} should be matched at similar scale. Intuitively, I^{Ref} could be downsampled to the scale of I^{LR}, or I^{LR} upsampled to the scale of I^{Ref}. The former is easier but the latter achieves more accurate matching with respect to location since matching is done at a larger scale. In addition, upscaling I^{LR} preserves scale consistency with M^{Ref}, facilitating texture swapping from M^{Ref} that carries richer texture information. Therefore, we upscale I^{LR} before feeding it to the texture extractor. However, we do not feed I^{Ref} directly for patch matching because the upscaling process of I^{LR} may introduce artifacts and blurry effects that would negatively

affect the matching result. Hence, we downscale I^{Ref} and followed by the same upscaling process as that of I^{LR} to achieve more accurate patch matching. Typically, the downscaling uses bicubic interpolation, while the upscaling could be an existing SR method or even bicubic interpolation. Section 5.3.1 will further detail the patching matching and texture swapping.

5.3.1 Patch Matching and Texture Swapping

Patch matching and texture swapping generate the texture map M^t that carries rich texture information from I^{Ref} while preserving the content structure of I^{LR}. The details are shown in Fig. 5.3. For simplicity, all feature maps are shown as single channel.

Patch matching is first performed between M^{LR} and M^{LRef}. The patch-wise similarity is measured by inner product,

$$s_{i,j} = \left\langle \frac{p_i^{LR}}{\|p_i^{LR}\|}, \frac{p_j^{LRef}}{\|p_j^{LRef}\|} \right\rangle \tag{5.2}$$

where $s_{i,j}$ denotes the similarity score between the ith patch from M^{LR} (i.e., p_i^{LR}) and the jth patch from M^{LRef} (i.e., p_j^{LRef}). As shown in Fig. 5.3, M^{LRef} is reshaped to a sequence of patches by dense sampling. The patches can be considered as kernels, hence the inner product can be approximated by performing convolution between M^{LR} and the patch kernels, to yield a sequence of similarity score maps. The maximum score at each pixel location across the similarity score maps indicates the best matched M^{LRef} patch. The similarity map M^s records those maximum scores with structural identity to M^{LR}. Since M^{Ref} is structurally identical to M^{LRef}, the patch matching correspondence between M^{LR} and M^{LRef} is identical

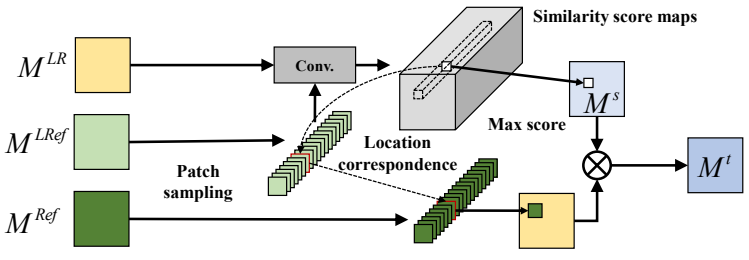


Figure 5.3: Patch matching and texture swapping.

to that between M^{LR} and M^{Ref}. Therefore, texture-richer patches from M^{Ref} is swapped to M^{LR} according to the correspondence. The overlaps between swapped patches are averaged. Considering the uncorrelated texture that may degrade the SR performance, the swapped texture map is multiplied by M^s, which weights down those uncorrelated texture because of its low similarity score. This significantly boost the adaptiveness of texture transfer, which will be demonstrated in Section 5.5.5. Finally, we obtain a weighted texture map M^t.

Patch matching is visualized in Fig. 5.4. The matching is performed in the high-level features space. For visualization purpose, the matches are back projected to the pixel space. It can be observed that the edge, contour, and texture are matched regardless of color. Note that the reference images are unnecessarily consistent to the input image in content.

5.3.2 Conditional Texture Transfer

Based on the content feature map M^c and weighted texture map M^t, the conditional texture transfer would adaptively fuse textures from M^t to M^c by a deep residual network. The pipeline of conditional texture transfer is shown in Fig. 5.5, where $\oplus$ and $\otimes$ represent element-wise summation and multiplication, respectively. Since the texture transferred to I^{SR} is supposed to be visually consistent with I^{LR}, it is necessary to transfer M^t conditioned on M^c. As shown in the blue block of Fig. 5.5, M^c is concatenated to M^t as the condition, and they are fed to the deep residual blocks [47], which learn to adaptively extract consistent texture from M^t conditioned on M^c. The extracted texture is then added to M^c. Finally,

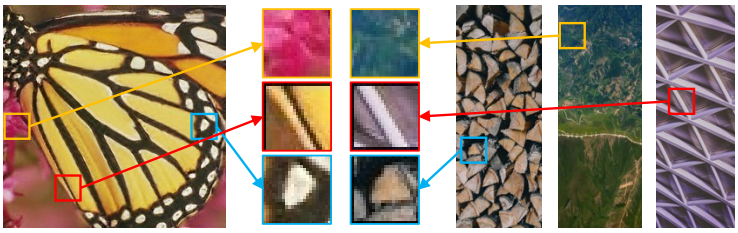


Figure 5.4: Visualization of patch matching. The left image is the input, and right three images are the reference images. The two columns in the middle are matched patches (zoom-in) between the input and references. The colored boxes and arrows code the corresponding matches.

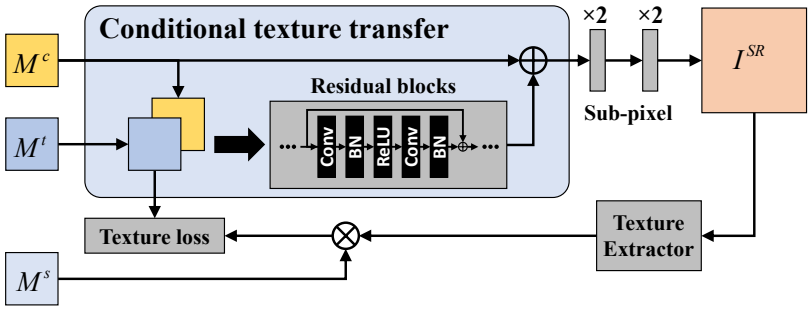


Figure 5.5: The pipeline of conditional texture transfer. The transfered feature map is fed to the sub-pixel layers for upscaling. The texture loss is is computed between M^t and the weighted feature map extracted from I^{SR}.

sub-pixel convolution [117] is employed as the upscaling process, which is beneficial in both accuracy and speed.

Different from traditional SISR methods that only focus on losses between I^{SR} and the ground truth, the reference-conditioned SR also takes into account the loss between I^{SR} and I^{Ref}, which we refer to as the texture loss. Specifically, the texture loss is calculated between M^t derived from I^{Ref} and the weighted feature map extracted from I^{SR} with M^s being the weight. More details on loss functions will be discussed in Section 5.3.3.

5.3.3 Content Loss $\mathcal{L}_c$ and Texture Loss $\mathcal{L}_t$

As briefly discussed at the beginning of Section 5.3, in order to preserve the structural information of the LR image, improve the visual quality of the SR image, as well as taking advantage of the rich texture details from the reference image, the objective function we develop involves both content loss $\mathcal{L}_c$ and texture loss $\mathcal{L}_t$. The content loss is three-fold, including the reconstruction loss, $\mathcal{L}_{rec}$, to preserve the structural information, the perceptual loss, $\mathcal{L}_{per}$, and the adversarial loss, $\mathcal{L}_{adv}$, to boost the visual quality. The texture loss is added for the network to adaptively enhance the texture transferred from the reference image.

Reconstruction loss is widely adopted in most SR works. To achieve the objective of obtaining higher PSNR, MSE is usually used to measure the reconstruction loss. In this work, we adopt the ℓ_1-norm (Eq. 5.3), where H and W denote the height and width of the HR/SR image, respectively. The ℓ_1-norm would further sharpen I^{SR} as compared to MSE. In addition, it is consistent to the objective of WGAN-GP, which will be discussed later in

the adversarial loss.

$$\mathcal{L}_{rec} = \frac{1}{HW}\sum_{x=1}^{H}\sum_{y=1}^{W}|I_{x,y}^{HR} - I_{x,y}^{SR}|, \tag{5.3}$$

Perceptual loss has been investigated in recent SR works [8, 53, 69, 112] for better visual quality. We calculate the perceptual loss based on the relu5_1 layer of VGG19 [119],

$$\mathcal{L}_{per} = \frac{1}{V}\sum_{i=1}^{C}\left\|\phi_i(I^{HR}) - \phi_i(I^{SR})\right\|_F, \tag{5.4}$$

where V and C indicate the tensor volume and channel number of the feature maps, respectively, and ϕ_i denotes the ith channel of the feature maps extracted from the hidden layer of VGG19 model. $\|\cdot\|_F$ denotes the Frobenius norm.

Adversarial loss could significantly enhance the sharpness/visual quality of I^{SR}. Here, we adopt WGAN-GP [45], which improves upon WGAN by penalizing the gradient, achieving more stable results. Because the Wasserstein distance in WGAN is based on ℓ_1-norm, we also use ℓ_1-norm as the reconstruction loss (Eq. 5.3). Intuitively, consistent objectives in optimization would help yield more stable results. The adversarial loss and objective of WGAN are expressed as

$$\mathcal{L}_{adv} = -\mathbb{E}_{\tilde{x}\sim\mathbb{P}_g}[D(\tilde{x})], \quad \min_G \max_{D\in\mathcal{D}} \mathbb{E}_{x\sim\mathbb{P}_r}[D(x)] - \mathbb{E}_{\tilde{x}\sim\mathbb{P}_g}[D(\tilde{x})], \tag{5.5}$$

where $\mathcal{D}$ is the set of 1-Lipschitz functions, and $\mathbb{P}_r$ and $\mathbb{P}_g$ are the model distribution and real data distribution, respectively.

Texture loss is built on the idea of Gatys et al. [37, 38], where Gram matrix was applied to statistically preserve the texture from the style image. To preserve the consistency between the content and transferred texture, the similarity map M^s is utilized, as illustrated in Fig. 5.5, to suppress uncorrelated texture. The texture loss is written as

$$\mathcal{L}_t = \frac{1}{4V^2}\left\|Gr(\phi(I^{SR}) \otimes M^s) - Gr(M^t)\right\|_F, \tag{5.6}$$

where $Gr(\cdot)$ denotes the operator that computes the Gram matrix, and $\phi(\cdot)$ indicates the feature maps from the relu3_1 layer of VGG19 model, whose scale is the same as that of the

texture feature map M^t. V is the volume of the feature map tensor from VGG19, and $\otimes$ denotes the element-wise multiplication.

5.4 Dataset

For reference-based SR problems, the similarity between the LR image and reference affects SR results significantly. In general, references with various similarity levels to the corresponding LR image should be provided for the purpose of learning and evaluating the adaptiveness to similar and dissimilar textures in references. We refer to pairs of LR and reference images as LR-Ref pairs.

We use SIFT [83] features to measure the similarity between two images because SIFT features characterize local texture information that is in line with the objective of local texture matching. In addition, SIFT feature matching is conducted in the pixel space which is more rigorous than high-level feature matching, providing more visually correlated pairs.

We build the training and validation datasets based on CUFED [140] that contains 1,883 albums. We choose to use album images since images collected from the same event are supposed to be taken in similar environment. Each album describes one of the 23 most common events in our daily life, ranging from Wedding to Nature Trip. The size of albums varies between 30 and 100 images. Within each album, we collect image pairs in different similarity levels based on SIFT feature matching. We quantitatively collect five similarity levels denoted as XH (extra-high), H (high), M (medium), L (low), and XL (extra-low). From each paired images, we randomly crop 160×160 patches from one image as the original HR images (LR images are obtained by downscaling), and the corresponding references are cropped from the other image. In this way, we collect 13,761 LR-Ref pairs for training purpose. It is worth noting that in building the training dataset, it is not necessary to present references at all five similarity levels for 'each' LR image although the training set as a whole should contain LR-Ref pairs at different similarity levels.

On the other hand, the validation dataset does need each LR image to have references at all five similarity levels in order to extensively evaluate the adaptiveness of the network to references with different similarity levels. We use the same way to collect image pairs

as in building the training dataset. In total, the validation set contains 126 groups of testing samples with each group consisting of one HR image and five references at five different similarity levels. Some randomly selected testing groups are shown in Fig. 5.6. We refer to the collected training and validation sets as CUFED5. The construction of CUFED5 largely facilitates performance evaluation of the proposed reference-conditioned SR research, providing a benchmark for the study of reference-based SR in general.

5.5 Experimental Results

In this section, both quantitative and qualitative comparisons are conducted to demonstrate the effectiveness of the proposed SRNTT in boosting SR performance in aspects of visual quality, texture richness, and content consistency.

5.5.1 Training Details and Parameters

In training, the LR images are obtained by downscaling ($4\times$) the HR images through bicubic interpolation, thus the LR images are of the size 40×40. The corresponding reference is fed with the original size, 160×160. All feature maps keep the same size of 40×40, to facilitate patch matching, texture swapping, and concatenation of content and texture maps. The weights parameters for $\mathcal{L}_{per}$, $\mathcal{L}_{adv}$, and $\mathcal{L}_t$ are α=1e-4, β=1e-6, and λ=1e-4, respectively. Adam optimizer is used with the initial learning rate of 1e-4. The network is pre-trained for 5 epochs, where only $\mathcal{L}_{rec}$ is applied. Then, all losses are involved to train another 100

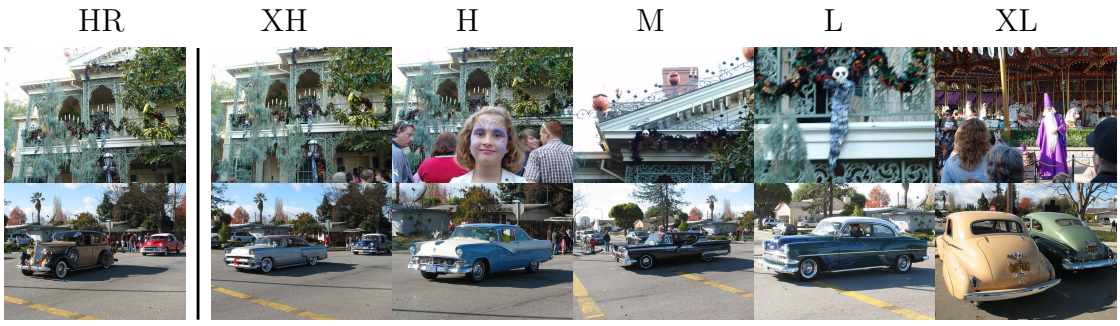


Figure 5.6: Examples from the CUFED5 dataset. The left column is the HR image for testing. The right columns are corresponding references in five similarity levels, i.e., extra-high (XH), high (H), medium (M), low (L), and extra-low (XL).

epochs, during which the learning rate is decayed by 0.1 for each 50 epochs. Note that $\mathcal{L}_t$ is only applied on the conditional texture transfer network. The whole framework could be trained end-to-end. However, the patch matching is time-consuming, occupying over 90% run time during training. Hence we calculate M^t offline for each training pair because the process of generating M^t only involves the pre-trained VGG19 model. To further speed up the training process, we could use a pre-trained SR network, e.g., MSE-based EDSR or GAN-based SRGAN, as the content extractor to obtain M^c. The network structures we used in the experiment are detailed in Appendix B.

5.5.2 Quantitative Evaluation

We compare the PSNR and SSIM with other SR methods, as shown in Table 5.1. SelfEx [50] is a non-learning-based method using the LR input itself as reference. SRCNN [25], SCN [141], DRCN [57], LapSRN [155], and EDSR [73] are learning-based methods by minimizing MSE. ENet [112] and SRGAN [69] are also learning-based but further utilize the perceptual-related constraints to enhance the visual quality. Landmark [146] is a reference-based SR method, retrieving references that could be well-aligned to I^{LR}. The "SRNTT-" denotes a simplified version of SRNTT by removing the adversarial loss, which is supposed to achieve comparable PSNR as the MSE-based methods. All methods are tested on the CUFED5 dataset. The Landmark method is tested with the reference of similarity level M. Our methods are tested with each of the five references and results averaged. This individual performance is listed in Table 5.2.

Table 5.1: Comparison of different SR methods in PSNR and SSIM.

	Bicubic	SRCNN	SCN	DRCN	LapSRN	EDSR
PSNR	24.18	25.33	25.45	25.26	24.92	**26.81**
SSIM	.6837	.7451	.7426	.7337	.7299	**.7923**
	SelfEx	Land-mark	ENet	SRGAN	SRNTT-	SRNTT
PSNR	23.22	24.91	24.24	24.40	26.23	24.60
SSIM	.6799	.7176	.6948	.7021	.7737	.7086

Table 5.2: PSNR and SSIM of SRNTT with different reference levels.

	XH	H	M	L	XL	Average
PSNR	24.57	24.58	24.56	24.63	**24.69**	24.60
SSIM	.7099	.7082	.7075	.7079	**.7094**	.7086

From PSNR and SSIM, we observe that those methods minimizing MSE perform better than GAN-based methods. However, PSNR and SSIM cannot adequately reflect visual quality of the image, especially when the upscaling factor is relatively large (e.g., 4×) and/or the image is originally with rich texture. The GAN-based methods would present more details that may deviate from the content of the original image, or even introduce artifacts to make the generated images look sharper. Note that SRNTT- achieves comparable PSNR and SSIM with the state-of-the-art, i.e., EDSR. In the individual performance of SRNTT with each reference level, the highest PSNR/SSIM is achieved at XL because the texture from reference is mostly suppressed, yielding relatively smooth results. By contrast, the score of XH is lower because correlated texture are transferred to the SR results which may deviate from the original HR image.

To illustrate the conditioned texture transfer, we measure the textural similarity between I^{SR} and I^{Ref} (level M) based on the Gram matrix as shown in Table 5.3. SRNTT presents the smallest distance since it transfers texture from the reference. ENet also achieves a relatively small distance because it also regularizes on texture but between I^{SR} and the original HR image. Some typical testing results are shown in Fig. 5.7, where SRNTT and SRNTT- present more textures that are transferred from the references. We observe that Landmark could well utilize the texture from references only when the reference can be well aligned to the input. For example, Landmark can better recover the flag since it can find well

Table 5.3: Texture distance to the reference based on Gram matrix.

	Landmark	EDSR	ENet	SRGAN	SRNTT-	SRNTT
Gram	55.25	37.00	27.26	32.21	33.72	**22.77**

Table 5.4: Gram distance of SRNTT with references of different similarity.

Reference similarity	XH	H	M	L	XL
Gram	15.59	21.54	22.94	24.77	29.01

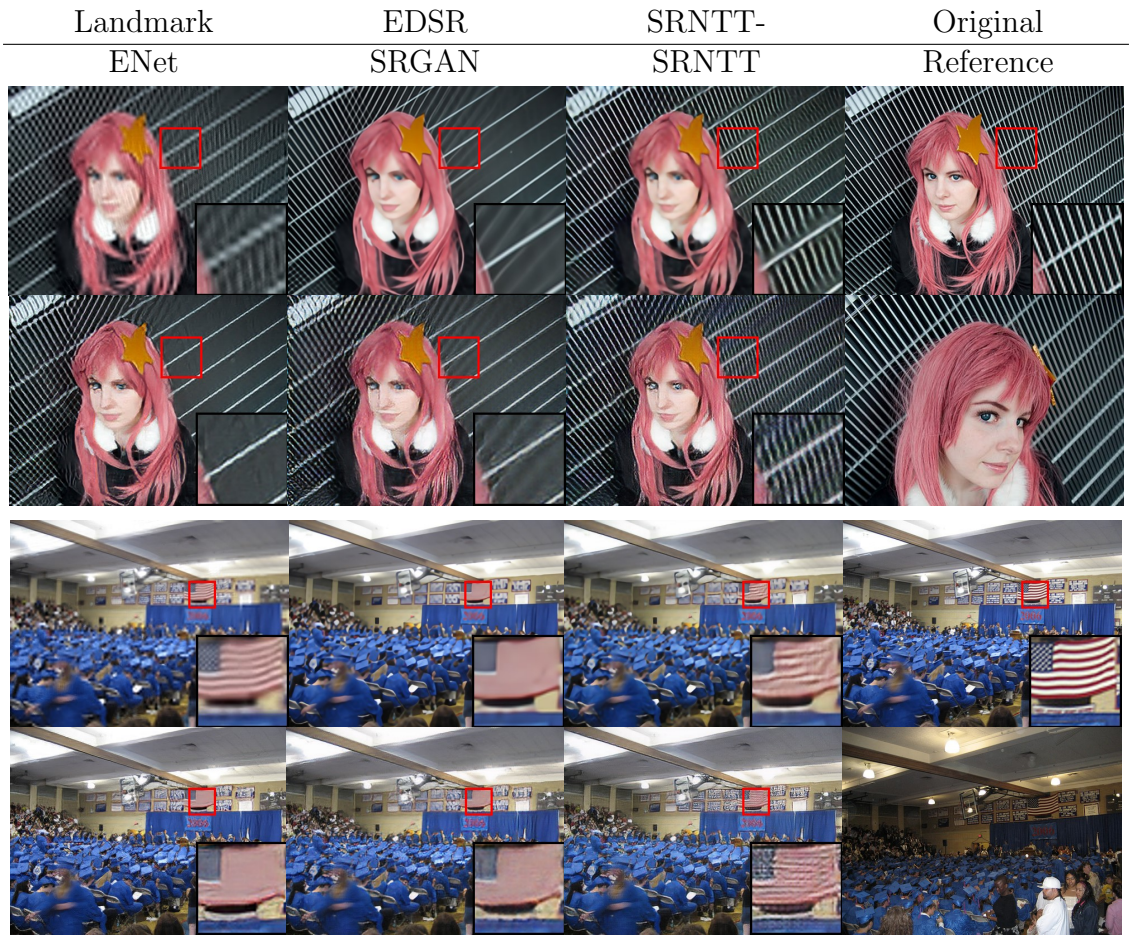


Figure 5.7: Visual comparison to other SR methods.

aligned patch in the reference. For the other examples, Landmark fails to recover details. Our method could better tolerate misaligned or unrelated texture. The MSE-based methods tend to generate clean but blurry images. By contrast, SRNTT- gains rich textures from references. The GAN-based methods present better visual quality, but still cannot recover plausible textures like SRNTT. For more comparison, please refer to Appendix C.

5.5.3 Qualitative Evaluation by User Study

To evaluate the visual quality of the SR images, we conduct user study which is widely adopted in many GAN-related works [69, 112]. SRNTT is compared to other six methods, i.e., Landmark, SCN, DRCN, EDSR, ENet, and SRGAN. We present the users with pair-wise comparisons, i.e., SRNTT vs. other, and ask the users to select the one with better

visual quality and more natural looking. For each reference level, 1,890 votes are collected on the testing results randomly selected from the CUFED5 dataset. Fig. 5.8 shows the voting results, i.e., the percentages of votes to SRNTT as compared to other methods, which demonstrate a roughly descending trend as the references become less similar to the LR image. Table 5.5 lists the voting results. More results of SRNTT using references with different similarity levels are visualized in Appendix D

5.5.4 Texture Transfer Results

Fig. 5.9 demonstrates the effect of texture loss $\mathcal{L}_t$. The experimental setup is the same as the first example in Fig. 5.7. With $\mathcal{L}_t$, the lost texture (fence) in the input is recovered by transferring similar texture from the reference. Please note that the orientation of fence in the reference differs from that in the ground truth, while the result with $\mathcal{L}_t$ generate texture similar to the ground truth. Although the weight of $\mathcal{L}_t$ is relatively low, it improves the visual quality of the texture significantly. A larger weight will make the result preserving less image structures, which deviates the problem away from super-resolution.

More general but still extreme for existing reference-based SR methods, the reference could be an arbitrary image, which may significantly deviate from I^{LR} in aspect of content and texture. An example is shown in Fig. 5.10, where the SR results are conditioned on the references shown in the bottom-right corner.

5.5.5 Investigation on Extreme Conditions

SRNTT could adaptively transfer correlated textures to I^{SR}, thus presenting enhanced texture conditioned on the reference. Comparing the first and last results, the latter is visually sharper because its reference carries richer texture and with higher resolution. The third result shows strong edge as the reference.

This section investigates extreme cases in reference-conditioned SR where the reference is simply image with homogeneous intensity levels or even random noise. The proposed SRNTT can cope with these extreme cases by introducing the similarity map M^s, which could effectively suppress unrelated textures. We also train SRNTT without M^s, which is referred

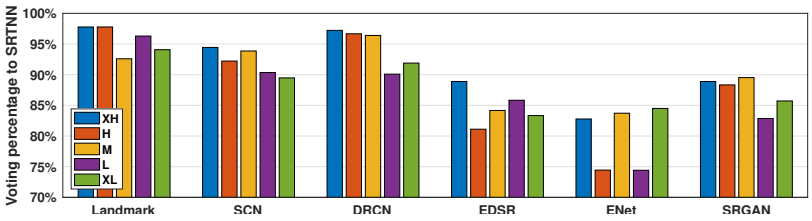


Figure 5.8: Percentage of votes to SRNTT as compared to each of other methods. "Voting percentage to SRNTT" means the percentage of users preferring the results of SRNTT over its competitor.

Table 5.5: Results of user study: percentage of voting to our methods SRNTT- or SRNTT as compared to other SR methods.

Level XH	Landmark	SCN	DRCN	EDSR	ENet	SRGAN	Average
SRNTT-	96.7%	90.0%	96.7%	65.6%	52.2%	46.7%	74.6%
SRNTT	97.8%	94.4%	97.2%	88.9%	82.8%	88.9%	91.7%
Level H							
SRNTT-	87.8%	88.9%	92.2%	48.9%	27.8%	26.7%	62.0%
SRNTT	97.8%	92.2%	96.7%	81.1%	74.4%	88.3%	88.4%
Level M							
SRNTT-	98.3%	90.2%	95.1%	63.3%	57.6%	51.7%	76.0%
SRNTT	92.6%	93.9%	96.4%	84.2%	83.7%	89.5%	90.0%
Level L							
SRNTT-	90.5%	94.1%	91.4%	56.7%	47.0%	54.1%	73.0%
SRNTT	96.3%	90.4%	90.1%	85.8%	74.4%	82.9%	86.6%
Level XL							
SRNTT-	88.3%	82.4%	90.1%	60.0%	54.5%	56.7%	72.0%
SRNTT	94.1%	89.5%	91.9%	83.3%	84.5%	85.7%	88.2%

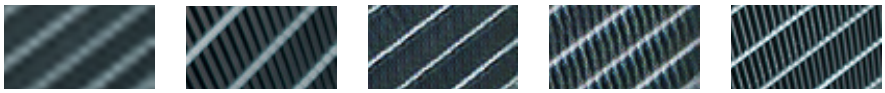


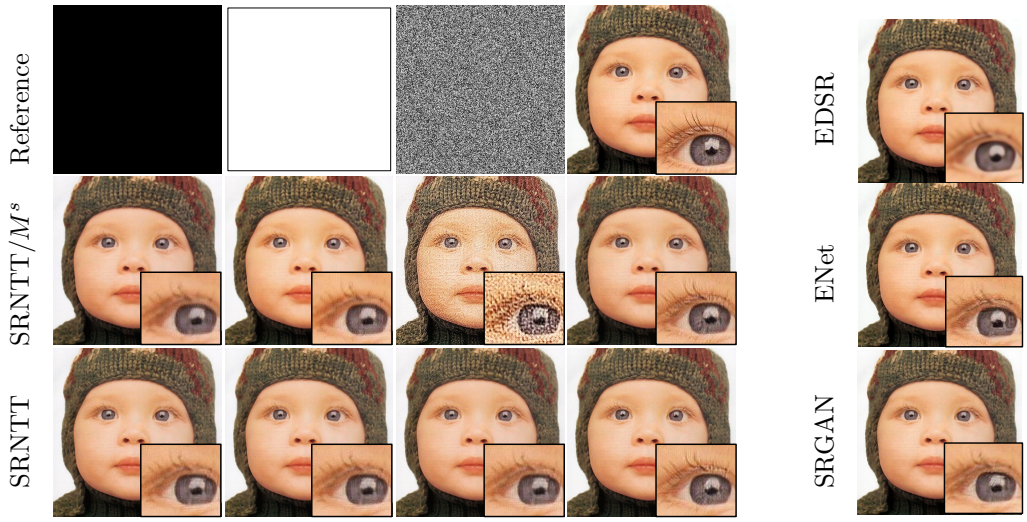
Figure 5.9: Effect of the texture loss $\mathcal{L}_t$.



Figure 5.10: The texture is enhanced conditioned on the reference that is shown in the bottom-right corner. Please zoom in for better view.

to as SRNTT/M^s. Without considering the extreme references, SRNTT and SRNTT/M^s show similar performance. However, when given extreme references, SRNTT/M^s may introduce negative effects from the reference as shown in Fig. 5.11. Given all-dark or all-bright references, which do not provide any extra texture information to the conditional texture transfer network, the results of SRNTT and SRNTT/M^s are close to the state-of-the-art GAN-based methods, i.e., ENet and SRGAN. However, given the random noise as reference, SRNTT/M^s transfers the texture of noise to the SR image by mistake, while such uncorrelated texture is successfully suppressed by SRNTT, demonstrating the adaptiveness gained from M^s. When the reference is perfect, i.e., the original image, the results from both SRNTT and SRNTT/M^s show much finer details. Therefore, M^s plays a critical role in suppressing uncorrelated textures while encouraging correlated ones.

Example-based methods without using deep learning cannot beat the deep learning based methods even with highly similar references. So, they are not considered as major competitors in our experiments. Nevertheless, we compared with Landmark [146] which is outperformed by most deep methods as shown in previous experiments. Here, we compare to another example-based method [122] as shown in Fig. 5.12, where the same references



(a) Our results with extreme references **(b)** Baselines

Figure 5.11: (a) The extreme references and the corresponding SR results from the proposed SRNTT and SRNTT/M^s which is a trimmed version of SRNTT by removing M^s. (b) Baseline results for comparison purpose.



Truth | Sun and Hays [122] (8×) | SRNTT (8×)

Figure 5.12: Comparison with an existing work on the upscaling factor of 8×. Please zoom in for better visualization.

are used as provided by [122]. Those references are retrieved by web searching and include highly similar references. We achieve better result with the upscaling factor of 8.

5.6 Conclusion

This chapter exploited the reference-conditioned solution for solving SR problems where the reference can be an arbitrary image. We proposed SRNTT, an end-to-end network structure that performs adaptive texture transfer from the reference to recover more plausible texture in the SR image. Both quantitative and qualitative experiments were conducted to demonstrate the effectiveness and adaptiveness of SRNTT, even with extreme cases of references. In addition, a new dataset CUFED5 was constructed to facilitate the evaluation of reference-conditioned SR methods. It also provides a benchmark for future reference-based SR research in general.

Chapter 6

Conclusions and Future Works

This work focuses on image synthesis by generative adversarial modeling, in aspects of theocratic reasoning, improvement, and novel applications. From the vanilla GAN, Chapter 2 attempted to theoretically reasoning the effectiveness of conditional GAN, showing that with the incorporation of an autoencoder, the reconstruction error serves the purpose of penalizing the missing mode, thus effectively alleviating the mode missing problem in GAN. Chapter 2 also elaborated the cause of instability in training generators and showed that by adding an adaptive decay variable in the objective function, the gradient vanishing and explosion, as well as the undamped oscillation problems can be effectively resolved.

Taking advantage of the composite structure discussed in Chapter 2, Chapter 3 proposed a novel conditional adversarial autoencoder (CAAE), which, to the best of our knowledge, first achieves face age progression and regression in a holistic framework. We deviated from the conventional routine of group-based training by learning a manifold, making the aging progression/regression more flexible and manipulatable — from an arbitrary query face without knowing its true age, we can freely produce faces at different ages, while at the same time preserving the personality. We demonstrated that with two discriminators imposed on the generator and encoder, respectively, the framework generates more photo-realistic faces. Flexibility, effectiveness, and robustness of CAAE have been demonstrated through extensive evaluation.

In order to deal with the drawback in existing works on image synthesis using GAN plus encoding-decoding networks, Chapter 4 proposed the decoupled learning (ED//GAN),

improving the stability of the common structure of incorporating ED and GAN, i.e., ED+GAN. We illustrate that the instability of ED+GAN is introduced by the interaction between the reconstruction loss (from ED) and adversarial loss (from GAN). The ED//GAN decouples the two effects by training separate networks for the two purposes of avoiding interaction between them and stabilizing the training process. This novel structure relaxes the trade-off issue that the ED+GAN suffers from because of the decoupling. The experimental results demonstrated the stability of the proposed decoupled learning. Existing works can be easily adapted into ED//GAN structure by adding an extra generator that learns the residual. The ED//GAN could ensure stable and competitive results without parameter tuning. However, it is unnecessary to beat the best result from fine tuned ED+GAN.

To further explore conditional image synthesis in traditional areas, Chapter 5 exploited the reference-conditioned solution for solving SR problems where the reference can be an arbitrary image, i.e., a generic scenario without drawing any assumptions on the reference. Correspondingly, we proposed SRNTT, an end-to-end network structure that performs adaptive texture transfer from the reference to recover more plausible texture in the SR image. Both quantitative and qualitative experiments were conducted to demonstrate the effectiveness and adaptiveness of SRNTT, even with extreme cases of references. In addition, a new dataset CUFED5 was constructed to facilitate the evaluation of reference-conditioned SR methods. It also provides a benchmark for future reference-based SR research in general. Note that from recent SR works, either competed on PSNR or focused on visual quality, they are all limited by the lack of plausible textures.

In the future, we will further extend the potential of conditional image synthesis in generating larger scale and more visually plausible images, which is appealing to practical applications, e.g., image editing and composition, artistic rendering, special effects in movies, etc. The ultimate goal would be aligning pixels to human vision and even aesthetics. As we are still in the preliminary stage, there are still some fundamentally critical problems left opening, e.g., artifacts and distortion in synthesized images. We will move our current works on to achieve higher visual quality, e.g., $8\times$ or even $16\times$ SR, and more effective utilization of conditions. It is still too far away from the end to stop our steps.

Bibliography

[1] Antipov, G., Baccouche, M., and Dugelay, J.-L. (2017). Face aging with conditional generative adversarial networks. *International Conference on Image Processing (ICIP)*. 1

[2] Arici, T. and Celikyilmaz, A. (2016). Associative adversarial networks. *Advances in Neural Information Processing Systems (NIPS) Workshop*. 23, 27

[3] Arjovsky, M. and Bottou, L. (2017). Towards principled methods for training generative adversarial networks. In *International Conference on Learning Representation (ICLR)*. 12, 13

[4] Arjovsky, M., Chintala, S., and Bottou, L. (2017). Wasserstein gan. *arXiv preprint arXiv:1701.07875*. 7, 56, 61

[5] Bengio, Y., Ducharme, R., Vincent, P., and Jauvin, C. (2003). A neural probabilistic language model. *Journal of Machine Learning Research*, 3(Feb):1137–1155. 1

[6] Biadsy, F., El-Sana, J., and Habash, N. (2006). Online arabic handwriting recognition using hidden markov models. In *Tenth International Workshop on Frontiers in Handwriting Recognition*. Suvisoft. 3

[7] Bilen, H., Fernando, B., Gavves, E., Vedaldi, A., and Gould, S. (2016). Dynamic image networks for action recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3034–3042. 1

[8] Bruna, J., Sprechmann, P., and LeCun, Y. (2016). Super-resolution with deep convolutional sufficient statistics. *In International Conference on Learning Representations (ICLR)*. 85

[9] Burt, D. M. and Perrett, D. I. (1995). Perception of age in adult caucasian male faces: Computer graphic manipulation of shape and colour information. *Proceedings of the Royal Society of London B: Biological Sciences*, 259(1355):137–143. 39

[10] Caballero, J., Ledig, C., Aitken, A., Acosta, A., Totz, J., Wang, Z., and Shi, W. (2017). Real-time video super-resolution with spatio-temporal networks and motion compensation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4778–4787. 79

[11] Chang, H., Yeung, D.-Y., and Xiong, Y. (2004). Super-resolution through neighbor embedding. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages I–I. IEEE. 76, 79

[12] Che, T., Li, Y., Jacob, A. P., Bengio, Y., and Li, W. (2016). Mode regularized generative adversarial networks. *arXiv preprint arXiv:1612.02136*. 12, 19, 30, 61, 62

[13] Chen, B.-C., Chen, C.-S., and Hsu, W. H. (2014). Cross-age reference coding for age-invariant face recognition and retrieval. In *Proceedings of the European Conference on Computer Vision*. 49

[14] Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., and Yuille, A. L. (2017). Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*. 1

[15] Chen, T. Q. and Schmidt, M. (2016). Fast patch-based style transfer of arbitrary style. *arXiv preprint arXiv:1612.04337*. 76

[16] Chen, X., Duan, Y., Houthooft, R., Schulman, J., Sutskever, I., and Abbeel, P. (2016). InfoGAN: Interpretable representation learning by information maximizing generative adversarial nets. In *Advances in Neural Information Processing Systems (NIPS)*. 40

[17] Chicco, D., Sadowski, P., and Baldi, P. (2014). Deep autoencoder neural networks for gene ontology annotation predictions. In *ACM Conference on Bioinformatics, Computational Biology, and Health Informatics*, pages 533–540. ACM. 1

[18] Choi, E., Schuetz, A., Stewart, W. F., and Sun, J. (2016). Using recurrent neural network models for early detection of heart failure onset. *Journal of the American Medical Informatics Association*, 24(2):361–370. 1

[19] Ciregan, D., Meier, U., and Schmidhuber, J. (2012). Multi-column deep neural networks for image classification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3642–3649. IEEE. 1

[20] Dahl, G. E., Jaitly, N., and Salakhutdinov, R. (2014). Multi-task neural networks for qsar predictions. *arXiv preprint arXiv:1406.1231*. 1

[21] Deng, L., Hinton, G., and Kingsbury, B. (2013). New types of deep neural network learning for speech recognition and related applications: An overview. In *International Conference on Acoustics, Speech and Signal Processing*, pages 8599–8603. IEEE. 1

[22] Denton, E. L., Chintala, S., Szlam, A., and Fergus, R. (2015). Deep generative image models using a Laplacian pyramid of adversarial networks. In *Advances in neural information processing systems (NIPS)*, pages 1486–1494. 4, 40

[Dlib] Dlib. Dlib C++ Library. http://dlib.net/. [Online]. 49

[24] Donahue, J., Krähenbühl, P., and Darrell, T. (2016). Adversarial feature learning. *arXiv preprint arXiv:1605.09782*. 12

[25] Dong, C., Loy, C. C., He, K., and Tang, X. (2014). Learning a deep convolutional network for image super-resolution. In *European Conference on Computer Vision*, pages 184–199. Springer. 76, 78, 88, 123

[26] Dong, C., Loy, C. C., He, K., and Tang, X. (2016a). Image super-resolution using deep convolutional networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(2):295–307. 76

[27] Dong, C., Loy, C. C., and Tang, X. (2016b). Accelerating the super-resolution convolutional neural network. In *European Conference on Computer Vision*, pages 391–407. Springer. 78

[28] Duchi, J. and Singer, Y. (2009). Efficient online and batch learning using forward backward splitting. *Journal of Machine Learning Research*, 10(Dec):2899–2934. 4

[29] Durugkar, I., Gemp, I., and Mahadevan, S. (2016). Generative multi-adversarial networks. *arXiv preprint arXiv:1611.01673*. 12

[30] Elkahky, A. M., Song, Y., and He, X. (2015). A multi-view deep learning approach for cross domain user modeling in recommendation systems. In *International Conference on*

World Wide Web, pages 278–288. International World Wide Web Conferences Steering Committee. 1

[31] Fernández, S., Graves, A., and Schmidhuber, J. (2007). An application of recurrent neural networks to discriminative keyword spotting. *International Conference on Artificial Neural Networks*, pages 220–229. 1

[32] Freedman, G. and Fattal, R. (2011). Image and video upscaling from local self-examples. *ACM Transactions on Graphics (TOG)*, 30(2):12. 76, 79

[33] Freeman, W. T., Jones, T. R., and Pasztor, E. C. (2002). Example-based super-resolution. *IEEE Computer graphics and Applications*, 22(2):56–65. 76, 79

[FT] FT. Face Transformer (FT) demo. http://cherry.dcs.aber.ac.uk/transformer/. [Online]. 51

[35] Fu, Y. and Zheng, N. (2006). M-face: An appearance-based photorealistic model for multiple facial attributes rendering. *IEEE Transactions on Circuits and Systems for Video Technology*, 16(7):830–842. 37, 39

[36] Gales, M. and Young, S. (2008). The application of hidden markov models in speech recognition. *Foundations and trends in signal processing*, 1(3):195–304. 3

[37] Gatys, L., Ecker, A. S., and Bethge, M. (2015). Texture synthesis using convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 262–270. 79, 81, 85

[38] Gatys, L. A., Ecker, A. S., and Bethge, M. (2016). Image style transfer using convolutional neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2414–2423. IEEE. 76, 79, 81, 85

[39] Gers, F. A. and Schmidhuber, E. (2001). LSTM recurrent networks learn simple context-free and context-sensitive languages. *IEEE Transactions on Neural Networks*, 12(6):1333–1340. 1

[40] Gharasuie, M. and Seyedarabi, H. (2013). Real-time dynamic hand gesture recognition using hidden markov models. In *Iranian Conference on Machine Vision and Image Processing*, pages 194–199. IEEE. 3

[41] Goodfellow, I. (2016). Nips 2016 tutorial: Generative adversarial networks. *arXiv preprint arXiv:1701.00160*. 58, 60

[42] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2672–2680. xiii, 1, 3, 4, 5, 9, 13, 14, 17, 27, 39, 47, 55, 56, 76, 79

[43] Graves, A., Eck, D., Beringer, N., and Schmidhuber, J. (2004). Biologically plausible speech recognition with lstm neural nets. In *International Workshop on Biologically Inspired Approaches to Advanced Information Technology*, pages 127–136. Springer. 1

[44] Gregor, K., Danihelka, I., Graves, A., Rezende, D. J., and Wierstra, D. (2015). Draw: A recurrent neural network for image generation. *arXiv preprint arXiv:1502.04623*. 40

[45] Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. C. (2017). Improved training of wasserstein gans. In *Advances in Neural Information Processing Systems*, pages 5769–5779. 85

[46] Hannun, A., Case, C., Casper, J., Catanzaro, B., Diamos, G., Elsen, E., Prenger, R., Satheesh, S., Sengupta, S., Coates, A., et al. (2014). Deep speech: Scaling up end-to-end speech recognition. *arXiv preprint arXiv:1412.5567*. 1

[47] He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778. 1, 68, 83

[48] Held, D., Thrun, S., and Savarese, S. (2016). Learning to track at 100 fps with deep regression networks. In *European Conference on Computer Vision (ECCV)*, pages 749–765. Springer. 1

[49] Hjelm, R. D., Jacob, A. P., Che, T., Cho, K., and Bengio, Y. (2017). Boundary-seeking generative adversarial networks. *arXiv preprint arXiv:1702.08431*. 7

[50] Huang, J.-B., Singh, A., and Ahuja, N. (2015). Single image super-resolution from transformed self-exemplars. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5197–5206. 79, 88, 123

[51] Im, D. J., Kim, C. D., Jiang, H., and Memisevic, R. (2016). Generating images with recurrent adversarial networks. *arXiv preprint arXiv:1602.05110*. 40

[52] Isola, P., Zhu, J.-Y., Zhou, T., and Efros, A. A. (2017). Image-to-image translation with conditional adversarial networks. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. xvi, 1, 3, 8, 9, 56, 60, 61, 68, 69, 72, 117

[53] Johnson, J., Alahi, A., and Fei-Fei, L. (2016). Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision*, pages 694–711. Springer. 76, 79, 85

[54] Kazemi, V. and Sullivan, J. (2014). One millisecond face alignment with an ensemble of regression trees. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1867–1874. 49

[55] Kemelmacher-Shlizerman, I., Suwajanakorn, S., and Seitz, S. M. (2014). Illumination-aware age progression. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3334–3341. IEEE. 36, 37, 39, 49, 50, 51, 53

[56] Kim, J., Kwon Lee, J., and Mu Lee, K. (2016a). Accurate image super-resolution using very deep convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1646–1654. 79

[57] Kim, J., Kwon Lee, J., and Mu Lee, K. (2016b). Deeply-recursive convolutional network for image super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and pattern recognition*, pages 1637–1645. 76, 79, 88, 123

[58] Kim, T., Cha, M., Kim, H., Lee, J., and Kim, J. (2017). Learning to discover cross-domain relations with generative adversarial networks. *arXiv preprint arXiv:1703.05192*. 61

[59] Kingma, D. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*. 4, 50

[60] Kingma, D. P. and Welling, M. (2013). Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*. 47, 56, 61

[61] Kingma, D. P. and Welling, M. (2014). Auto-encoding variational bayes. In *The International Conference on Learning Representations (ICLR)*. 6, 12

[62] Krizhevsky, A. and Hinton, G. (2009). Learning multiple layers of features from tiny images. 28

[63] Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1097–1105. 1

[64] Lai, W.-S., Huang, J.-B., Ahuja, N., and Yang, M.-H. (2017). Fast and accurate image super-resolution with deep laplacian pyramid networks. *arXiv preprint arXiv:1710.01992*. 78

[65] Lanitis, A., Taylor, C. J., and Cootes, T. F. (2002). Toward automatic simulation of aging effects on face images. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 24(4):442–455. 29, 37, 38, 50

[66] Larsen, A. B. L., Sønderby, S. K., and Winther, O. (2015). Autoencoding beyond pixels using a learned similarity metric. *arXiv preprint arXiv:1512.09300*. 47, 56, 62

[67] Larsen, A. B. L., Sønderby, S. K., and Winther, O. (2016). Autoencoding beyond pixels using a learned similarity metric. *International Conference on Machine Learning (ICML)*. 12, 19, 22

[68] LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324. 28

[69] Ledig, C., Theis, L., Huszar, F., Caballero, J., Cunningham, A., Acosta, A., Aitken, A., Tejani, A., Totz, J., Wang, Z., et al. (2017a). Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4681–4690. xii, 76, 78, 79, 85, 88, 90, 121, 123

[70] Ledig, C., Theis, L., Huszár, F., Caballero, J., Cunningham, A., Acosta, A., Aitken, A., Tejani, A., Totz, J., Wang, Z., et al. (2017b). Photo-realistic single image super-resolution using a generative adversarial network. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 1, 3, 4, 7, 8, 9, 56

[71] Lee, H., Grosse, R., Ranganath, R., and Ng, A. Y. (2009). Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *International Conference on Machine Learning (ICML)*, pages 609–616. ACM. 3

[72] Li, Y., Swersky, K., and Zemel, R. (2015). Generative moment matching networks. In *International Conference on Machine Learning (ICML)*, pages 1718–1727. 12, 27

[73] Lim, B., Son, S., Kim, H., Nah, S., and Lee, K. M. (2017). Enhanced deep residual networks for single image super-resolution. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. 76, 79, 88, 123

[74] Lin, T.-Y. and Maji, S. (2016). Visualizing and understanding deep texture representations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2791–2799. 81

[75] Liu, C. and Sun, D. (2011). A bayesian approach to adaptive video super resolution. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 209–216. IEEE. 79

[76] Liu, J., Yang, W., Zhang, X., and Guo, Z. (2017a). Retrieval compensated group structured sparsity for image super-resolution. *IEEE Transactions on Multimedia*, 19(2):302–316. 76

[77] Liu, M.-Y., Breuel, T., and Kautz, J. (2017b). Unsupervised image-to-image translation networks. *arXiv preprint arXiv:1703.00848*. 56

[78] Liu, M. Y. and Tuzel, O. (2016). Coupled generative adversarial networks. In *Advances in Neural Information Processing Systems (NIPS)*. 37, 40

[79] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., and Berg, A. C. (2016). SSD: Single shot multibox detector. In *European Conference on Computer Vision (ECCV)*, pages 21–37. Springer. 1

[80] Liu, Z., Luo, P., Wang, X., and Tang, X. (2015). Deep learning face attributes in the wild. In *International Conference on Computer Vision (ICCV)*. 29, 31

[81] Liu, Z., Zhang, Z., and Shan, Y. (2004). Image-based surface detail transfer. *IEEE Computer Graphics and Applications*, 24(3):30–35. 37, 39

[82] Long, J., Shelhamer, E., and Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3431–3440. 1

[83] Lowe, D. G. (1999). Object recognition from local scale-invariant features. In *IEEE International Conference on Computer Vision*, volume 2, pages 1150–1157. IEEE. 86

[84] Lv, F. and Nevatia, R. (2006). Recognition and segmentation of 3-d human action using hmm and multi-class adaboost. *European Conference on Computer Vision (ECCV)*, pages 359–372. 3

[85] Makhzani, A., Shlens, J., Jaitly, N., and Goodfellow, I. (2016). Adversarial autoencoders. In *International Conference on Learning Representations*. 37, 40, 47

[86] Makhzani, A., Shlens, J., Jaitly, N., Goodfellow, I., and Frey, B. (2015). Adversarial autoencoders. *arXiv preprint arXiv:1511.05644*. 56, 61

[87] Mansimov, E., Parisotto, E., Ba, J. L., and Salakhutdinov, R. (2015). Generating images from captions with attention. *The International Conference on Learning Representations (ICLR)*. 4, 7

[88] Mao, X., Li, Q., Xie, H., Lau, R. Y., Wang, Z., and Smolley, S. P. (2016). Least squares generative adversarial networks. *arXiv preprint ArXiv:1611.04076*. 7, 56, 61

[89] Mesnil, G., Dauphin, Y., Yao, K., Bengio, Y., Deng, L., Hakkani-Tur, D., He, X., Heck, L., Tur, G., Yu, D., et al. (2015). Using recurrent neural networks for slot filling in spoken language understanding. *IEEE/ACM Transactions on Audio, Speech and Language Processing*, 23(3):530–539. 1

[90] Metz, L., Poole, B., Pfau, D., and Sohl-Dickstein, J. (2017). Unrolled generative adversarial networks. *The International Conference on Learning Representations (ICLR)*. 7, 12, 27, 63

[91] Milan, A., Rezatofighi, S. H., Dick, A. R., Reid, I. D., and Schindler, K. (2017). Online multi-target tracking using recurrent neural networks. In *Association for the Advancement of Artificial Intelligence (AAAI)*, pages 4225–4232. 1

[92] Mirza, M. and Osindero, S. (2014). Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*. 40

[93] Nasrollahi, K. and Moeslund, T. B. (2014). Super-resolution: a comprehensive survey. *Machine Vision and Applications*, 25(6):1423–1468. 78

[94] Nguyen, A., Dosovitskiy, A., Yosinski, J., Brox, T., and Clune, J. (2016). Synthesizing the preferred inputs for neurons in neural networks via deep generator networks. In *Advances in Neural Information Processing Systems (NIPS)*, pages 3387–3395. 56

[95] Nilsback, M.-E. and Zisserman, A. (2008). Automated flower classification over a large number of classes. In *Proceedings of the Indian Conference on Computer Vision, Graphics and Image Processing*. 67

[96] Odena, A., Olah, C., and Shlens, J. (2016). Conditional image synthesis with auxiliary classifier gans. *arXiv preprint arXiv:1610.09585*. 63

[97] Osborne, M. J. and Rubinstein, A. (1994). *A course in game theory*. MIT press. 16

[98] Pathak, D., Krahenbuhl, P., Donahue, J., Darrell, T., and Efros, A. A. (2016). Context encoders: Feature learning by inpainting. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2536–2544. 56, 61

[99] Permuter, H., Francos, J., and Jermyn, I. H. (2003). Gaussian mixture models of texture and colour for image database retrieval. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 3, pages III–569. IEEE. 2

[100] Plötz, T. and Fink, G. A. (2009). Markov models for offline handwriting recognition: a survey. *International Journal on Document Analysis and Recognition*, 12(4):269. 3

[101] Rabiner, L. R. (1989). A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286. 4

[102] Radford, A., Metz, L., and Chintala, S. (2016a). Unsupervised representation learning with deep convolutional generative adversarial networks. *International Conference on Learning Representation (ICLR)*. 3, 4, 6

[103] Radford, A., Metz, L., and Chintala, S. (2016b). Unsupervised representation learning with deep convolutional generative adversarial networks. In *International Conference on Learning Representations*. 37, 40, 42, 56, 68, 116

[104] Radim Tyleček, R. Š. (2013). Spatial pattern templates for recognition of objects with regular structure. In *Proc. GCPR*, Saarbrucken, Germany. 67

[105] Ramanathan, N. and Chellappa, R. (2006). Modeling age progression in young faces. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 387–394. IEEE. 37, 38

[106] Ramanathan, N. and Chellappa, R. (2008). Modeling shape and textural variations in aging faces. In *IEEE International Conference on Automatic Face & Gesture Recognition*, pages 1–8. IEEE. 38

[107] Reed, S., Akata, Z., Yan, X., Logeswaran, L., Schiele, B., and Lee, H. (2016). Generative adversarial text to image synthesis. *International Conference on Machine Learning*. 1, 3, 4, 7, 8, 9, 56

[108] Reynolds, D. A. and Rose, R. C. (1995). Robust text-independent speaker identification using gaussian mixture speaker models. *IEEE transactions on Speech and Audio Processing*, 3(1):72–83. 2

[109] Ricanek, K. and Tesafaye, T. (2006). Morph: A longitudinal image database of normal adult age-progression. In *International Conference on Automatic Face and Gesture Recognition*, pages 341–345. IEEE. 29

[110] Rothe, R., Timofte, R., and Gool, L. V. (2015). DEX: Deep expectation of apparent age from a single image. In *IEEE International Conference on Computer Vision (ICCV) Workshops*. 49

[111] Rumelhart, D. E., Hinton, G. E., Williams, R. J., et al. (1988). Learning representations by back-propagating errors. *Cognitive Modeling*, 5(3):1. 4

[112] Sajjadi, M. S., Scholkopf, B., and Hirsch, M. (2017). Enhancenet: Single image super-resolution through automated texture synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4491–4500. 76, 79, 85, 88, 90, 123

[113] Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., and Chen, X. (2016). Improved techniques for training GANs. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2226–2234. 7, 12, 19, 27, 28, 63

[114] Sathyanarayana, A., Joty, S., Fernandez-Luque, L., Ofli, F., Srivastava, J., Elmagarmid, A., Arora, T., and Taheri, S. (2016). Sleep quality prediction from wearable data using deep learning. *JMIR mHealth and uHealth*, 4(4). 1

[115] Schmidhuber, J. (1992). Learning factorial codes by predictability minimization. *Neural Computation*, 4(6):863–879. 4, 5

[116] Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117. 1

[117] Shi, W., Caballero, J., Huszár, F., Totz, J., Aitken, A. P., Bishop, R., Rueckert, D., and Wang, Z. (2016). Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1874–1883. xii, 78, 84, 121

[118] Shu, X., Tang, J., Lai, H., Liu, L., and Yan, S. (2015). Personalized age progression with aging dictionary. In *IEEE International Conference on Computer Vision (ICCV)*, pages 3970–3978. 37, 39, 50, 51, 53

[119] Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*. 1, 79, 81, 85, 120

[120] Socher, R., Bengio, Y., and Manning, C. (2013). Deep learning for NLP. *Tutorial at Association of Computational Logistics, 2012, and North American Chapter of the Association of Computational Linguistics*. 1

[121] Song, Y., Zhang, Z., and Qi, H. (2018). r-btn: Cross-domain face composite and synthesis from limited facial patches. *The Thirty-Second AAAI Conference on Artificial Intelligence (AAAI)*. 1

[122] Sun, L. and Hays, J. (2012). Super-resolution from internet-scale scene matching. In *Computational Photography (ICCP), 2012 IEEE International Conference on*, pages 1–12. IEEE. 93, 94

[123] Suo, J., Chen, X., Shan, S., Gao, W., and Dai, Q. (2012). A concatenational graph evolution aging model. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 34(11):2083–2096. 38

[124] Suo, J., Zhu, S.-C., Shan, S., and Chen, X. (2010). A compositional and dynamic model for face aging. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 32(3):385–401. 37, 38, 50, 51, 53

[125] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015). Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9. 1

[126] Taigman, Y., Polyak, A., and Wolf, L. (2016). Unsupervised cross-domain image generation. *arXiv preprint arXiv:1611.02200*. 61

[127] Tazoe, Y., Gohara, H., Maejima, A., and Morishima, S. (2012). Facial aging simulator considering geometry and patch-tiled texture. In *ACM SIGGRAPH 2012 Posters*, page 90. ACM. 37

[128] Tiddeman, B., Burt, M., and Perrett, D. (2001). Prototyping and transforming facial textures for perception research. *IEEE Computer Graphics and Applications*, 21(5):42–50. 37

[129] Timofte, R., De, V., and Van Gool, L. (2013). Anchored neighborhood regression for fast example-based super-resolution. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 1920–1927. IEEE. 76

[130] Unterthiner, T., Mayr, A., Klambauer, G., Steijaert, M., Wegner, J. K., Ceulemans, H., and Hochreiter, S. (2014). Deep learning as an opportunity in virtual screening. In *Proceedings of the Deep Learning Workshop at NIPS*. 1

[131] Van den Oord, A., Dieleman, S., and Schrauwen, B. (2013). Deep content-based music recommendation. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2643–2651. 1

[132] Varol, G., Laptev, I., and Schmid, C. (2017). Long-term temporal convolutions for action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*. 1

[133] Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., and Manzagol, P.-A. (2010). Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, 11(Dec):3371–3408. 3

[134] Wah, C., Branson, S., Welinder, P., Perona, P., and Belongie, S. (2011). The caltech-ucsd birds-200-2011 dataset. 67

[135] Wang, H., Wang, N., and Yeung, D.-Y. (2015a). Collaborative deep learning for recommender systems. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1235–1244. ACM. 1

[136] Wang, L., Xiong, Y., Wang, Z., Qiao, Y., Lin, D., Tang, X., and Van Gool, L. (2016a). Temporal segment networks: Towards good practices for deep action recognition. In *European Conference on Computer Vision (ECCV)*, pages 20–36. Springer. 1

[137] Wang, N. and Yeung, D.-Y. (2013). Learning a deep compact image representation for visual tracking. In *Advances in Neural Information Processing Systems (NIPS)*, pages 809–817. 1

[138] Wang, W., Cui, Z., Yan, Y., Feng, J., Yan, S., Shu, X., and Sebe, N. (2016b). Recurrent face aging. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2378–2386. IEEE. 37, 39, 50, 51, 53

[139] Wang, X. and Gupta, A. (2016). Generative image modeling using style and structure adversarial networks. *European Conference on Computer Vision (ECCV)*. 4, 7

[140] Wang, Y., Lin, Z., Shen, X., Mech, R., Miller, G., and Cottrell, G. W. (2016c). Event-specific image importance. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 86

[141] Wang, Z., Liu, D., Yang, J., Han, W., and Huang, T. (2015b). Deep networks for image super-resolution with sparse prior. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 370–378. 76, 78, 88, 123

[142] Wu, J., Zhang, C., Xue, T., Freeman, B., and Tenenbaum, J. (2016). Learning a probabilistic latent space of object shapes via 3D generative-adversarial modeling. In *Advances in Neural Information Processing Systems (NIPS)*, pages 82–90. 4

[143] Yang, C.-Y., Ma, C., and Yang, M.-H. (2014). Single-image super-resolution: A benchmark. In *European Conference on Computer Vision*, pages 372–386. Springer. 76, 78

[144] Yu, F. and Koltun, V. (2016). Multi-scale context aggregation by dilated convolutions. *International Conference on Learning Representation (ICLR)*. 1

[145] Yu, X. and Porikli, F. (2016). Ultra-resolving face images by discriminative generative networks. In *European Conference on Computer Vision (ECCV)*, pages 318–333. Springer. 1, 3, 4, 7, 40

[146] Yue, H., Sun, X., Yang, J., and Wu, F. (2013). Landmark image super-resolution by retrieving web images. *IEEE Transactions on Image Processing*, 22(12):4865–4878. 76, 79, 80, 88, 93, 123

[147] Zeiler, M. D. (2012). ADADELTA: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*. 4

[148] Zhang, Z., Song, Y., and Qi, H. (2017a). Age progression/regression by conditional adversarial autoencoder. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2. xvi, xviii, 1, 8, 9, 37, 56, 60, 67, 68, 69, 72, 73, 117, 118

[149] Zhang, Z., Song, Y., and Qi, H. (2017b). Gans powered by autoencodinga theoretic reasoning. *ICML Workshop on Implicit Models*. 13

[150] Zhang, Z., Song, Y., and Qi, H. (2018a). Decoupled learning for conditional adversarial networks. *IEEE Winter Conference on Applications of Computer Vision (WACV)*. 55

[151] Zhang, Z., Wang, Z., Lin, Z., and Qi, H. (2018b). Reference-conditioned super-resolution by neural texture transfer. *arXiv preprint arXiv:1804.03360*. 75

[152] Zhao, J., Mathieu, M., and LeCun, Y. (2016). Energy-based generative adversarial network. *arXiv preprint arXiv:1609.03126*. 7, 12, 19, 37, 40

[153] Zhu, J.-Y., Krähenbühl, P., Shechtman, E., and Efros, A. A. (2016). Generative visual manipulation on the natural image manifold. In *European Conference on Computer Vision (ECCV)*, pages 597–613. Springer. 1, 4, 7, 8, 9, 56

[154] Zhu, J.-Y., Park, T., Isola, P., and Efros, A. A. (2017). Unpaired image-to-image translation using cycle-consistent adversarial networks. *arXiv preprint arXiv:1703.10593*. 56, 61

[155] Zhu, Z., Guo, F., Yu, H., and Chen, C. (2014). Fast single image super-resolution via self-example learning and sparse representation. *IEEE Transactions on Multimedia*, 16(8):2178–2190. 88, 123

Appendices

A The Network Structures in Decoupled Learning

In Enc and Dec, the kernel size is 5×5, and the stride is 2. The activation function is ReLU or Leaky ReLU (LReLU) for each hidden layer. The output layer adopts the hyperbolic tangent (tanh) function. Batch normalization is optionally applied before activation functions. ADAM optimizer is adopted with the learning rate of 0.0002 and $\beta_1 = 0.5$.

A.1 The Network Structure in Adversarial Effect (Sec. 4.4.3)

This network is neither specifically designed for any applications nor delicately fine-tuned to achieve the best result. The goal is to demonstrate the stability of the propose method (ED//GAN) as compared to the traditional method (ED+GAN). Therefore, we only need to ensure that both methods are compared fairly on the same structure and using the same hyperparameters. Fig. A.1 compares the structures of ED+GAN and ED//GAN, and the correspondingly detailed network structure is shown in Table A.1.

The generator G uses the same structure as Dec, and they share the latent variable z. The discriminator D adopts the similar structure as Enc, but the length of output is 1 (indicating real and fake) instead of 50 (the latent variable z). In G and D networks, batch normalization is applied to ensure stable training of GAN as suggested in DCGAN [103].

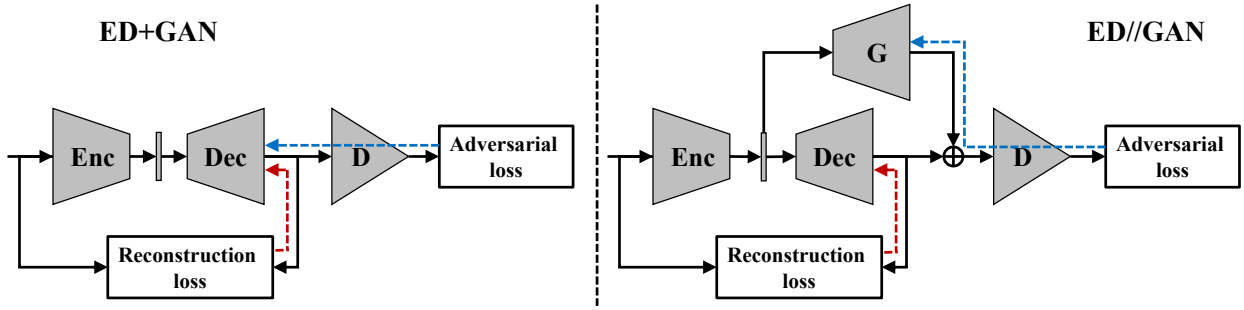


Figure A.1: Left: the existing ED+GAN. Right: the adaption based on the proposed ED//GAN, i.e., decoupled learning. Solid black arrows denote the feedforward path, and dashed arrows in red and blue indicate backpropagation from the reconstruction loss and the adversarial loss, respectively. The network details are listed in Table A.1.

Table A.1: Structure of the Enc and Dec networks as shown in Fig. A.1. The value 50 indicates a vector with the length of 50. Batch normalization is optional as indicated by (BN). The size of each layer is denoted by $h \times w \times c$, corresponding to width, and number of channels, respectively.

Enc	Size	Dec	Size
Input	$128 \times 128 \times 3$	Input	50
Conv, (BN), ReLU	$64 \times 64 \times 64$	FC, ReLU, (BN), Reshape	$4 \times 4 \times 1024$
Conv, (BN), ReLU	$32 \times 32 \times 128$	Deconv, (BN), ReLU	$8 \times 8 \times 512$
Conv, (BN), ReLU	$16 \times 16 \times 256$	Deconv, (BN), ReLU	$16 \times 16 \times 256$
Conv, (BN), ReLU	$8 \times 8 \times 512$	Deconv, (BN), ReLU	$32 \times 32 \times 128$
Conv, (BN), ReLU	$4 \times 4 \times 1024$	Deconv, (BN), ReLU	$64 \times 64 \times 64$
Reshape, FC, tanh	50	Deconv, tanh	$128 \times 128 \times 3$

A.2 The Network Structure in Image Transformation (Sec. 4.4.5)

We adapt the network in Pix2Pix [52], which is ED+GAN structure, to the proposed ED//GAN structure as shown in Fig. A.2.

In Pix2Pix, the ED is implemented by the U-Net, which directly passes feature maps from encoder to decoder, preserving more details. For simplicity and not breaking the structure of U-Net, we apply another U-Net as the generator G in correspondingly adaption to ED//GAN. The discriminator adopts the same network structure as in Pix2Pix. The network details are listed in Table A.2.

A.3 The Network Structure in Image Manipulation (Sec. 4.4.5)

We adapt the face aging work [148] (CAAE), which proposed a conditional ED+GAN structure, to the proposed ED//GAN structure as shown in Fig. A.3. CAAE generated aged face by manipulating the label concatenated to the latent variable z from Enc.

The original network used in CAAE has an extra discriminator on z to fore z uniformly distributed. We do not show this discriminator in Fig. A.3 because it does not affect the adaptation. The network details are listed in Table A.3.

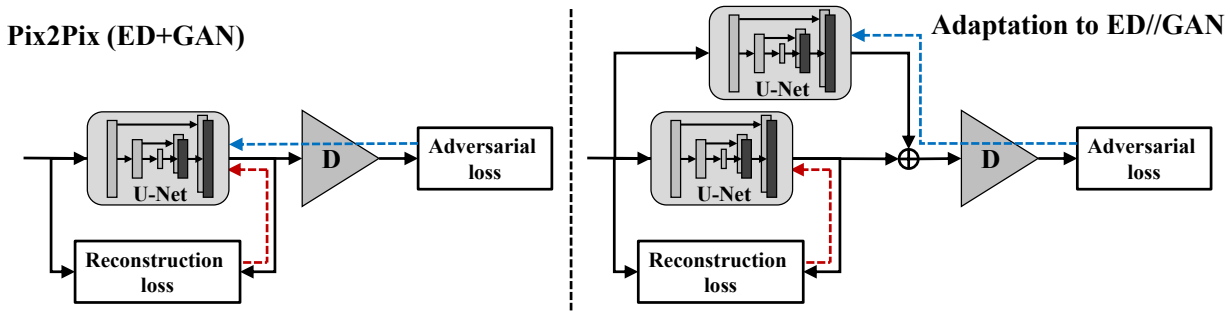


Figure A.2: Left: the structure of Pix2Pix (ED+GAN). Right: the adaption to the proposed ED//GAN, i.e., decoupled learning. Solid black arrows denote the feedforward path, and dashed arrows in red and blue indicate backpropagation from the reconstruction loss and the adversarial loss, respectively.

Table A.2: Structure of U-Net network as shown in Fig. A.2. Enc and Dec denote the encoding and decoding part in U-Net, respectively. The number of channels of the hidden layers in Dec are twice of those in Enc because of the direct passing.

Enc	Size	Dec	Size
Input	$256 \times 256 \times 3$	Input	$1 \times 1 \times 512$
Conv, BN, LReLU	$128 \times 128 \times 64$	Deconv, BN, ReLU, Dropout	$2 \times 2 \times 1024$
Conv, BN, LReLU	$64 \times 64 \times 128$	Deconv, BN, ReLU, Dropout	$4 \times 4 \times 1024$
Conv, BN, LReLU	$32 \times 32 \times 256$	Deconv, BN, ReLU, Dropout	$8 \times 8 \times 1024$
Conv, BN, LReLU	$16 \times 16 \times 512$	Deconv, BN, ReLU	$16 \times 16 \times 1024$
Conv, BN, LReLU	$8 \times 8 \times 512$	Deconv, BN, ReLU	$32 \times 32 \times 512$
Conv, BN, LReLU	$4 \times 4 \times 512$	Deconv, BN, ReLU	$64 \times 64 \times 256$
Conv, BN, LReLU	$2 \times 2 \times 512$	Deconv, BN, ReLU	$128 \times 128 \times 128$
Conv, BN, LReLU	$1 \times 1 \times 512$	Deconv, Tanh	$256 \times 256 \times 3$

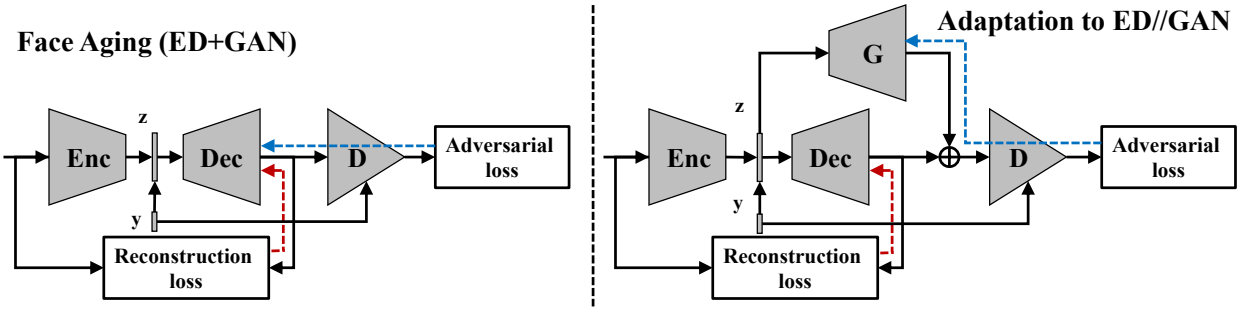


Figure A.3: Left: the ED+GAN structure used in CAAE [148]. Right: the adaption to the proposed ED//GAN, i.e., decoupled learning. Solid black arrows denote the feedforward path, and dashed arrows in red and blue indicate backpropagation from the reconstruction loss and the adversarial loss, respectively. The age label y is concatenated to z to control the age of generated faces.

Table A.3: Structure of the Enc, Dec, G, and D networks as shown in Fig. 4.11. The value 50 indicates a vector with the length of 50.The concatenated age vector y is 50. Structure of the "G" and "D" networks as shown in Fig. A.3. The value 50 indicates a vector with the length of 50. The concatenated age vector y is 50. If use WGAN or LSGAN, the BN is optional in G.

Enc	Size	Dec	Size
Input	$128 \times 128 \times 3$	Input	$50 + 50$ (length of the label)
Conv, ReLU	$64 \times 64 \times 64$	FC, ReLU, Reshape	$8 \times 8 \times 1024$
Conv, ReLU	$32 \times 32 \times 128$	Deconv, ReLU	$16 \times 16 \times 512$
Conv, ReLU	$16 \times 16 \times 256$	Deconv, ReLU	$32 \times 32 \times 256$
Conv, ReLU	$8 \times 8 \times 512$	Deconv, ReLU	$64 \times 64 \times 128$
Conv, ReLU	$4 \times 4 \times 1024$	Deconv, ReLU	$128 \times 128 \times 64$
Reshape, FC, tanh	50	Deconv, tanh	$128 \times 128 \times 3$

G	Size	D	Size
Input	$50 + 50$	Input	$128 \times 128 \times 3$
FC, BN, ReLU, Reshape	$8 \times 8 \times 1024$	Conv, BN, ReLU	$64 \times 64 \times (16 + 50)$
Deconv, BN, ReLU	$16 \times 16 \times 512$	Conv, BN, ReLU	$32 \times 32 \times 32$
Deconv, BN, ReLU	$32 \times 32 \times 256$	Conv, BN, ReLU	$16 \times 16 \times 64$
Deconv, BN, ReLU	$64 \times 64 \times 128$	Conv, BN, ReLU	$8 \times 8 \times 128$
Deconv, BN, ReLU	$128 \times 128 \times 64$	Reshape, FC, ReLU	1024
Deconv, tanh	$128 \times 128 \times 3$	FC, sigmoid	1

B Network Structure of SRNTT

Fig. B.1 illustrates the network structure of the proposed SRNTT. The blocks in gray indicate the non-trainable components that do not involve any learning process. For example, the texture extractor uses the pre-trained VGG19 model [119]. The algorithms involved in patch matching and texture swapping are detailed in Chapter 5 (Sec. 5.3). The elaboration in this section mainly focuses on the network structure of the blocks in blue, i.e., the content extractor, conditional texture transfer, upscaling, and the discriminator, that are trainable components. Note that the upscaling in the blue block is different from those before the texture extractor, which are not involved in the learning process, and they could be either pre-trained SR models or even bicubic interpolation.

As discussed in the subsection of "Training details and parameters" (Sec. 5.5.1) in Chapter 5, the content extractor is flexible to be a pre-trained model or trained from scratch. Its structure may vary with the pre-trained model. By contrast, the conditional texture transfer, upscaling, and discriminator are fixed in the framework. Tables B.1, B.2, B.3, and B.4 list the layer details of the content extractor, conditional texture transfer, upscaling, and the discriminator.

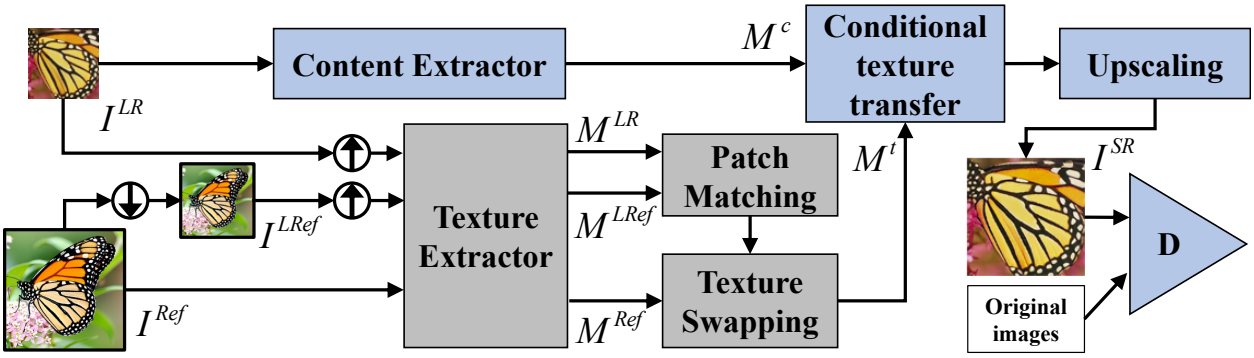


Figure B.1: Overview of the proposed SRNTT. D denotes the discriminator.

Table B.1: The network structure of content extractor, kernel size is 3×3. This structure is based on the generator of SRGAN [69], thus we could directly adopt the pre-trained model, speeding up the training process.

#	**Layer name(s)**	**Output size**
0	Input	$H \times W \times 3$
1	Conv, ReLU	$H \times W \times 64$
2∼17	Residual blocks (Conv, BN, ReLU, Conv, BN)	$H \times W \times 64$
18	Conv, BN	$H \times W \times 64$
19	#1 + #18	M^c : $H \times W \times 64$

Table B.2: The network structure of conditional texture transfer, kernel size is 3×3. M^c is obtained from Table B.1, and M^t is obtained from texture swapping based on the relu3_1 layer of VGG19 model.

#	**Layer name(s)**	**Output size**
0	Input	M^c : $H \times W \times 64$ M^t : $H \times W \times 256$
1	Concatenate	$H \times W \times 320$
2	Conv, ReLU	$H \times W \times 64$
3∼18	Residual blocks (Conv, BN, ReLU, Conv, BN)	$H \times W \times 64$
19	Conv, BN	$H \times W \times 64$
20	M^c + #19	$H \times W \times 64$

Table B.3: The network structure of upscaling, kernel size is 3×3. This network is based on the idea of sub-pixel [117]. The input is from the output of Table B.2.

#	**Layer name(s)**	**Output size**
0	Input	$H \times W \times 64$
1	Conv	$H \times W \times 256$
2	Sub-pixcel, ReLU	$2H \times 2W \times 64$
3	Conv	$2H \times 2W \times 256$
4	Sub-pixcel, ReLU	$4H \times 4W \times 64$
5	Conv, tanh	I^{SR} : $4H \times 4W \times 3$

Table B.4: The network structure of the discriminator, kernel size is 3×3. The output size is scaled down by stride 2, and the parameter of LReLU is 0.2. In the training dataset, the original image size is 160×160.

#	**Layer name(s)**	**Output size**
0	Input	I^{HR} or I^{SR}: $160 \times 160 \times 3$
1	Conv, BN, LReLU	$160 \times 160 \times 32$
2	Conv, BN, LReLU	$80 \times 80 \times 32$
3	Conv, BN, LReLU	$80 \times 80 \times 64$
4	Conv, BN, LReLU	$40 \times 40 \times 64$
5	Conv, BN, LReLU	$40 \times 40 \times 128$
6	Conv, BN, LReLU	$20 \times 20 \times 128$
7	Conv, BN, LReLU	$20 \times 20 \times 256$
8	Conv, BN, LReLU	$10 \times 10 \times 256$
9	Conv, BN, LReLU	$10 \times 10 \times 512$
10	Conv, BN, LReLU	$5 \times 5 \times 512$
11	Flatten	12800
12	FC, LReLU	1024
13	FC	1

C Comparison to Other SR Methods

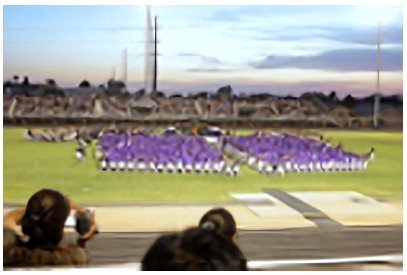
This section visually compares all the methods cited in the experimental part of Chapter 5, namely, SelfEx [50], SRCNN [25], LapSRN [155], SCN [141], DRCN [57], EDSR [73], ENet [112], SRGAN [69], Landmark [146], and the proposed SRNTT. Figs. C.1–C.4 show examples from the CUFED5 dataset, using the reference images of level M. We observe that, in general, GAN-based methods, i.e., ENet, SRGAN, and the proposed SRNTT, show more clear or less blurry SR results than MSE-based methods. In addition, compared to all the other methods, SRNTT presents more rich texture that is transferred from the reference.



Original



Reference



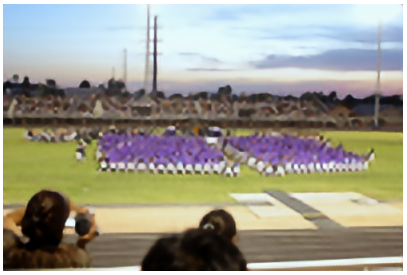
SelfEx



SRCNN



DRCN



LapSRN



SCN



EDSR



Landmark



ENet



SRGAN



SRNTT

Figure C.1: Comparison of different SR methods.



Original | Reference | SelfEx | SRCNN



DRCN | LapSRN | SCN | EDSR



Landmark | ENet | SRGAN | SRNTT

Figure C.2: Comparison of different SR methods.


Original


Reference


SelfEx


SRCNN

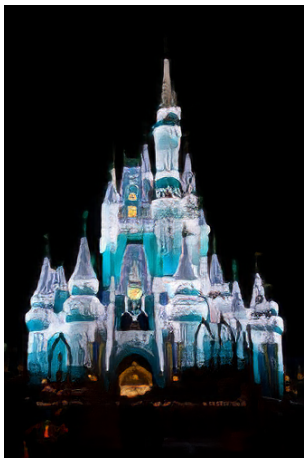

DRCN


LapSRN


SCN


EDSR


Landmark


ENet


SRGAN


SRNTT

Figure C.3: Comparison of different SR methods.



Original



Reference



SelfEx



SRCNN



DRCN



LapSRN



SCN



EDSR



Landmark



ENet



SRGAN



SRNTT

Figure C.4: Comparison of different SR methods.

D SRNTT Using References with Different Similarity

This section visually compares the results of SRNTT using reference images at different similarity levels, as shown in Fig. D.1. Specifically, the references of similarity level XH (extra-high), M (medium), XL (extra-low) are adopted to illustrate the effect of reference on SRNTT. The results of SRGAN are shown as the baseline. The SR performance of the proposed SRNTT is boosted if using a correlated reference, presenting more plausible texture. Otherwise, SRNTT performs similar to the state-of-the-art GAN-based methods, e.g., SRGAN and ENet.



Figure D.1: SRNTT results using reference images at different similarity levels. The results of SRGAN are shown as baseline.

Vita

In 2006, Zhifei Zhang received the Excellent New Student Scholarship from the Northeastern University (NEU) in Shenyang, China, where he became an undergrad student in the Department of Electrical Engineering. In 2010, he received the Bachelor's degree and then entered the Master's program in Zhejiang University (ZJU) in Hangzhou, China.

After graduating from ZJU in 2013, he began to pursue the doctoral degree at Oregon State University in Corvallis, OR. One year later, he transferred to the University of Tennessee in Knoxville, supervised by Dr. Hairiong Qi in the Department of Electrical Engineering and Computer Science. His research mainly focuses on deep learning based image synthesis. During the doctoral program, he had research internship in GE Global Research and Adobe Research, respectively. In September 2018, he successfully passed the Ph.D. defense and will return to Adobe as a full-time researcher. Currently, Zhifei Zhang lives in Knoxville with his wife (Ph.D. candidate) Yang Song, and they are waiting for the birth of their first princess.