Continuous Conditional Generative Adversarial Networks: Novel Empirical Losses and Label Input Mechanisms

Xin Ding, Yongwei Wang, *Student Member, IEEE,* Zuheng Xu,
William J. Welch, and Z. Jane Wang, *Fellow, IEEE*

Abstract

This paper focuses on *conditional generative modeling* (CGM) for image data with continuous, scalar conditions (termed regression labels). We propose the first model for this task which is called *continuous conditional generative adversarial network* (CcGAN). Existing *conditional GANs* (cGANs) are mainly designed for categorical conditions (e.g., class labels). Conditioning on regression labels is mathematically distinct and raises two fundamental problems: (P1) since there may be very few (even zero) real images for some regression labels, minimizing existing empirical versions of cGAN losses (a.k.a. empirical cGAN losses) often fails in practice; and (P2) since regression labels are scalar and infinitely many, conventional label input mechanisms (e.g., combining a hidden map of the generator/discriminator with a one-hot encoded label) are not applicable. We solve these problems by: (S1) reformulating existing empirical cGAN losses to be appropriate for the continuous scenario; and (S2) proposing a *naive label input* (NLI) mechanism and an *improved label input* (ILI) mechanism to incorporate regression labels into the generator and the discriminator. The reformulation in (S1) leads to two novel empirical discriminator losses, termed the *hard vicinal discriminator loss* (HVDL) and the *soft vicinal discriminator loss* (SVDL) respectively, and a novel empirical generator loss. Hence, we propose four versions of CcGAN employing different proposed losses and label input mechanisms. The error bounds of the discriminator trained with HVDL and SVDL, respectively, are derived under mild assumptions. To evaluate the performance of CcGANs, two new benchmark datasets (RC-49 and Cell-200) are created. A novel evaluation metric (*Sliding Fréchet Inception Distance*) is also proposed to replace Intra-FID when Intra-FID is not applicable. Our extensive experiments on several benchmark datasets (i.e., RC-49, UTKFace, Cell-200, and Steering Angle with both low and high resolutions) support the following findings: the proposed CcGAN is able to generate diverse, high-quality samples from the image distribution conditional on a given regression label; and CcGAN substantially outperforms cGAN both visually and quantitatively.

Index Terms

CcGAN, conditional generative modeling, conditional generative adversarial networks, continuous and scalar conditions.

1 Introduction

Conditional generative adversarial networks (cGANs), first proposed in [1], aim to estimate the distribution of images conditioning on some auxiliary information (a.k.a. *conditional generative modeling* (CGM) for image data), especially class labels. Subsequent studies [2]–[5] confirm the feasibility of generating diverse, high-quality (even photo-realistic), and class-label consistent fake images from well-trained class-conditional GANs. Unfortunately, existing cGANs are not applicable for CGM with continuous, scalar conditions, termed *regression labels*, due to two problems:

(P1) cGANs are often trained to minimize the empirical versions of their losses (a.k.a. empirical cGAN losses) on some training data, a principle also known as *empirical risk minimization* (ERM) [6]–[8]. The success of ERM relies on a large sample size for each distinct condition. Unfortunately, we usually have only a few real images for some regression labels. Moreover, since regression labels are continuous, some values may not even appear in the training set. Consequently, a cGAN cannot accurately estimate the image distribution conditional on such missing labels.

(P2) In the class-conditional generative modeling, class labels are often encoded by one-hot vectors or label embedding and then fed into the generator and discriminator by hidden concatenation [1], an auxiliary classifier [2] or label projection [3]. A precondition for such label encoding is that the number of distinct labels (e.g., the number of classes) is finite and known. Unfortunately, in the continuous scenario, we may have infinitely many distinct regression labels.

A naive approach denoted by **cGAN (K classes)** to solve **(P1)-(P2)** is to "bin" the regression labels into K disjoint intervals and still train a cGAN in the class-conditional manner (these intervals are treated as independent classes) [9]. Another naive approach denoted by **cGAN (concat)** for solving **(P2)** directly combines a regression label with the input or a hidden map of the generator and discriminator. However, the sampling results of both approaches in our empirical

- *Xin Ding, Zuheng Xu and William J. Welch are with the Department of Statistics, University of British Columbia, Vancouver, BC, V6T 1Z4 Canada (e-mail: xin.ding@stat.ubc.ca, zuheng.xu@stat.ubc.ca, will@stat.ubc.ca) .*
- *Yongwei Wang and Z. Jane Wang are with the Department of Electrical and Computer Engineering, University of British Columbia, Vancouver, BC, V6T 1Z4 Canada (e-mail: yongweiw@ece.ubc.ca, zjanew@ece.ubc.ca). (Corresponding author: Yongwei Wang)*

studies in Sections 5 and 6 show two types of failures of conventional cGANs in CGM with regression labels: (1) cGAN ($\boldsymbol{K}$ classes) cannot generate visually realistic and diverse images; (2) cGAN (concat) fails to generate images with respect to conditioning regression labels.

In machine learning, *vicinal risk minimization* (VRM) [6], [10] is an alternative rule to ERM. VRM assumes that a sample point shares the same label with other samples in its vicinity. Motivated by VRM, in generative modeling conditional on regression labels where we estimate a conditional distribution $p(\boldsymbol{x}|y)$ ($\boldsymbol{x}$ is an image and y is a regression label), it is natural to assume that a small perturbation to y results in a negligible change to $p(\boldsymbol{x}|y)$. This assumption is consistent with our perception of the world. For example, the image distribution of facial features for a population of 15-year-old teenagers should be close to that of 16-year olds.

We therefore introduce the *continuous conditional GAN* (CcGAN) to tackle **(P1)** and **(P2)**. To our best knowledge, this is the first generative model for image data conditional on regression labels. It is noted that [11] and [12] train GANs in an unsupervised manner and synthesize unlabeled fake images for a subsequent image regression task. [13] proposes a semi-supervised GAN for dense crowd counting. [14] uses a *convolutional neural network* (CNN) to generate images of objects in terms of some high-level parameters such as object style, viewpoint, color, brightness, etc. [15] proposes InfoGAN, which can control some continuous or discrete attributes of generated images. Some text-to-image generation methods [16]–[18] train generative models conditional on high-dimensional attribute vectors with continuous or discrete elements. The objectives of these works are entirely different from ours since they do not aim to estimate the image distribution conditional on regression labels. Moreover, some recent works [19]–[22] propose several novel schemes to train GANs when training data are limited, which seems to be relevant to **(P1)**. However, they are also fundamentally different from CcGAN, since they are designed for unconditional and class-conditional scenarios rather than continuous ones. Our contributions can be summarized as follows:

- We propose in Section 2.1 a solution to address **(P1)**, which consists of two novel empirical discriminator losses, termed the *hard vicinal discriminator loss* (HVDL) and the *soft vicinal discriminator loss* (SVDL), and a novel empirical generator loss. We take the vanilla cGAN loss as an example to show how to derive HVDL, SVDL, and the novel empirical generator loss by reformulating existing empirical cGAN losses.
- In Section 2.2, we propose two novel label input mechanisms, consisting of a *naive label input* (NLI) mechanism and an *improved label input* (ILI) mechanism, as solutions to address **(P2)**.
- We derive in Section 3 the error bounds of a discriminator trained with HVDL and SVDL. These error bounds not only help us understand how HVDL and SVDL influence the discriminator training but also guide our implementation in practice (especially the selection of hyper-parameters).
- We propose in Section 4 a novel evaluation metric, termed *Sliding Fréchet Inception Distance* (SFID), to evaluate the generative image modeling conditional on regression labels when there are insufficient real images to compute Intra-FID [3].
- In Sections 5 and 6, we propose two new benchmark datasets, RC-49 and Cell-200, for generative image modeling conditional on regression labels, since very few benchmark datasets are suitable for the studied continuous scenario. We conduct extensive experiments on four benchmark datasets with various resolutions (from 64×64 to 256×256) to demonstrate that CcGAN not only generates diverse, high-quality, and label consistent images, but also substantially outperforms cGAN both visually and quantitatively. The effectiveness of SFID is also studied on the RC-49 dataset at the end of Section 5.

A preliminary version of this paper has been presented at the International Conference on Learning Representations [23]. The current paper strengthens the initial version in several ways. (1) We propose an improved label input (ILI) mechanism to better incorporate regression labels into CcGAN. Our experiments in this paper demonstrate the superiority of ILI to the naive label input method in [23]. (2) We introduce Lemmas 1 and 2, which are used to derive the error bounds of HVDL and SVDL but are omitted in [23]. The motivation for deriving these error bounds is also better illustrated, and an improved proof for the derivation is provided in Appendix. (3) We propose SFID to replace Intra-FID when Intra-FID is not applicable. (4) We create a new benchmark dataset called Cell-200 for generative modeling conditional on regression labels. (5) We conduct a more extensive empirical study to demonstrate the effectiveness of CcGAN. This extensive study includes more datasets (e.g., Cell-200 and Steering Angle) and more complicated settings (e.g., various image resolutions). We also add a new baseline, cGAN (concat), to the comparison to better demonstrate that conventional cGANs are inapplicable to our task.

2 From cGAN to CcGAN

In this section, we introduce the *continuous conditional GAN* (CcGAN), consisting of solutions to **(P1)** and **(P2)**. The combinations of two vicinal discriminator losses (HVDL and SVDL) proposed in Section 2.1 and two novel label input mechanisms (NLI and ILI) proposed in Section 2.2 result in four CcGAN methods denoted by HVDL+NLI, SVDL+NLI, HVDL+ILI, and SVDL+ILI, respectively. The overall workflow of CcGAN is visualized in Fig. 1.

2.1 Solution to (P1): Reformulated Empirical Losses

Theoretically, cGAN losses (e.g., the vanilla cGAN loss [1], the Wasserstein loss [25], [26], and the hinge loss [24]) are suitable for both class labels and regression labels; however, their empirical versions fail in the continuous scenario (i.e., **(P1)**). Our

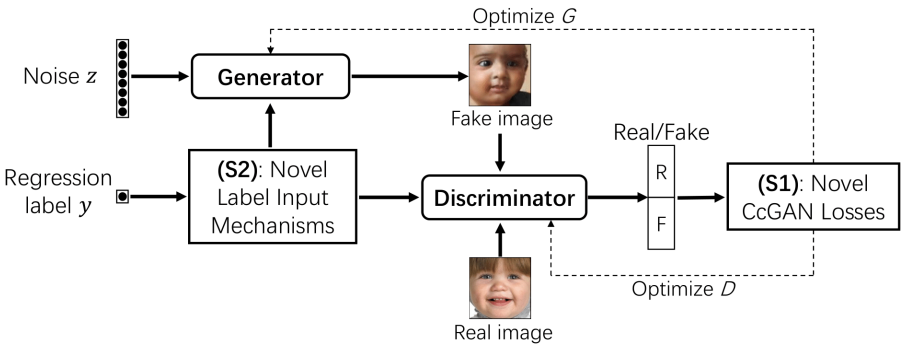

Fig. 1: **A typical workflow of the proposed CcGAN framework.** A regression label y is input into the generator (G) and the discriminator (D) by novel label input mechanisms proposed in Section 2.2. Novel empirical losses proposed in Section 2.1 are adopted to optimize G and D, respectively. The CcGAN framework is compatible with modern GAN architectures (e.g., SNGAN [24], SAGAN [5]) and advanced GAN training techniques (e.g., DiffAugment [19]).

first solution **(S1)** focuses on reformulating these empirical cGAN losses for continuous labels. Without loss of generality, we only take the vanilla cGAN loss as an example to show such reformulation (the empirical versions of the Wasserstein loss and the hinge loss can be reformulated similarly).

The vanilla discriminator loss and generator loss [1] are defined as:

$$\begin{aligned}\mathcal{L}(D) =& -\mathbb{E}_{y\sim p_r(y)}\left[\mathbb{E}_{\boldsymbol{x}\sim p_r(\boldsymbol{x}|y)}\left[\log\left(D(\boldsymbol{x},y)\right)\right]\right] \\ & -\mathbb{E}_{y\sim p_g(y)}\left[\mathbb{E}_{\boldsymbol{x}\sim p_g(\boldsymbol{x}|y)}\left[\log\left(1-D(\boldsymbol{x},y)\right)\right]\right] \\ =& -\int \log(D(\boldsymbol{x},y))p_r(\boldsymbol{x},y)d\boldsymbol{x}dy \\ & -\int \log(1-D(\boldsymbol{x},y))p_g(\boldsymbol{x},y)d\boldsymbol{x}dy, \end{aligned} \tag{1}$$

$$\begin{aligned}\mathcal{L}(G) =& -\mathbb{E}_{y\sim p_g(y)}\left[\mathbb{E}_{\boldsymbol{z}\sim q(\boldsymbol{z})}\left[\log\left(D(G(\boldsymbol{z},y),y)\right)\right]\right] \\ =& -\int \log(D(G(\boldsymbol{z},y),y))q(\boldsymbol{z})p_g(y)d\boldsymbol{z}dy, \end{aligned} \tag{2}$$

where $\boldsymbol{x}\in\mathcal{X}$ is an image, $y\in\mathcal{Y}$ is a label, $p_r(y)$ and $p_g(y)$ are respectively the actual and fake label marginal distributions, $p_r(\boldsymbol{x}|y)$ and $p_g(\boldsymbol{x}|y)$ are respectively the actual and fake image distributions conditional on y, $p_r(\boldsymbol{x},y)$ and $p_g(\boldsymbol{x},y)$ are respectively the actual and fake joint distributions of $\boldsymbol{x}$ and y, and $q(\boldsymbol{z})$ is the probability density function of $\mathcal{N}(\boldsymbol{0},\boldsymbol{I})$.

Since the distributions in the losses of Eqs. (1) and (2) are unknown, for class-conditional generative modeling, [1] follows ERM and minimizes the empirical losses:

$$\begin{aligned}\widehat{\mathcal{L}}^{\delta}(D) =& -\frac{1}{N^r}\sum_{c=1}^{C}\sum_{j=1}^{N_c^r}\log(D(\boldsymbol{x}_{c,j}^r,c)) \\ & -\frac{1}{N^g}\sum_{c=1}^{C}\sum_{j=1}^{N_c^g}\log(1-D(\boldsymbol{x}_{c,j}^g,c)),\end{aligned} \tag{3}$$

$$\widehat{\mathcal{L}}^{\delta}(G) = -\frac{1}{N^g}\sum_{c=1}^{C}\sum_{j=1}^{N_c^g}\log(D(G(\boldsymbol{z}_{c,j},c),c)), \tag{4}$$

where C is the number of classes, N^r and N^g are respectively the number of real and fake images, N_c^r and N_c^g are respectively the number of real and fake images with label c, $\boldsymbol{x}_{c,j}^r$ and $\boldsymbol{x}_{c,j}^g$ are respectively the j-th real image and the j-th fake image with label c, and the $\boldsymbol{z}_{c,j}$ are independently and identically sampled from $q(\boldsymbol{z})$. Eq. (3) implies we estimate $p_r(\boldsymbol{x},y)$ and $p_g(\boldsymbol{x},y)$ by their empirical probability density functions as follows:

$$\begin{aligned}\hat{p}_r^{\delta}(\boldsymbol{x},y) &= \frac{1}{N^r}\sum_{c=1}^{C}\sum_{j=1}^{N_c^r}\delta(\boldsymbol{x}-\boldsymbol{x}_{c,j}^r)\delta(y-c), \\ \hat{p}_g^{\delta}(\boldsymbol{x},y) &= \frac{1}{N^g}\sum_{c=1}^{C}\sum_{j=1}^{N_c^g}\delta(\boldsymbol{x}-\boldsymbol{x}_{c,j}^g)\delta(y-c),\end{aligned} \tag{5}$$

where $\delta(\cdot)$ is a Dirac delta function (Appendix A of [27]) centered at 0. However, $\hat{p}_r^{\delta}(\boldsymbol{x},y)$ and $\hat{p}_g^{\delta}(\boldsymbol{x},y)$ are not good estimates in the continuous scenario because of **(P1)**.

To overcome **(P1)**, we propose a novel estimate for each of $p_r(\boldsymbol{x}, y)$ and $p_g(\boldsymbol{x}, y)$, termed the *hard vicinal estimate* (HVE). We also provide an intuitive alternative to HVE, named the *soft vicinal estimate* (SVE). The HVEs of $p_r(\boldsymbol{x}, y)$ and $p_g(\boldsymbol{x}, y)$ are:

$$\begin{aligned}\hat{p}_r^{\text{HVE}}(\boldsymbol{x}, y) =& C_1 \cdot \left[\frac{1}{N^r}\sum_{j=1}^{N^r}\exp\left(-\frac{(y-y_j^r)^2}{2\sigma^2}\right)\right] \\ & \cdot \left[\frac{1}{N_{y,\kappa}^r}\sum_{i=1}^{N^r}\mathbb{1}_{\{|y-y_i^r|\leq\kappa\}}\delta(\boldsymbol{x}-\boldsymbol{x}_i^r)\right], \end{aligned} \tag{6}$$

$$\begin{aligned}\hat{p}_g^{\text{HVE}}(\boldsymbol{x}, y) =& C_2 \cdot \left[\frac{1}{N^g}\sum_{j=1}^{N^g}\exp\left(-\frac{(y-y_j^g)^2}{2\sigma^2}\right)\right] \\ & \cdot \left[\frac{1}{N_{y,\kappa}^g}\sum_{i=1}^{N^g}\mathbb{1}_{\{|y-y_i^g|\leq\kappa\}}\delta(\boldsymbol{x}-\boldsymbol{x}_i^g)\right], \end{aligned} \tag{7}$$

where $\boldsymbol{x}_i^r$ and $\boldsymbol{x}_i^g$ are respectively real image i, fake image i, y_i^r and y_i^g are respectively the labels of $\boldsymbol{x}_i^r$ and $\boldsymbol{x}_i^g$, κ and σ are two positive hyper-parameters, C_1 and C_2 are two constants making these two estimates valid probability density functions, $N_{y,\kappa}^r$ is the number of the y_i^r satisfying $|y-y_i^r|\leq\kappa$, $N_{y,\kappa}^g$ is the number of the y_i^g satisfying $|y-y_i^g|\leq\kappa$, and $\mathbb{1}$ is an indicator function with support in the subscript. The terms in the first square brackets of $\hat{p}_r^{\text{HVE}}$ and $\hat{p}_g^{\text{HVE}}$ imply we estimate the marginal label distributions $p_r(y)$ and $p_g(y)$ by *kernel density estimates* (KDEs) [28]–[31]. The terms in the second square brackets are designed based on the assumption that a small perturbation to y results in negligible changes to $p_r(\boldsymbol{x}|y)$ and $p_g(\boldsymbol{x}|y)$. If this assumption holds, we can use images with labels in a small vicinity of y to estimate $p_r(\boldsymbol{x}|y)$ and $p_g(\boldsymbol{x}|y)$. The SVEs of $p_r(\boldsymbol{x}, y)$ and $p_g(\boldsymbol{x}, y)$ are:

$$\begin{aligned}\hat{p}_r^{\text{SVE}}(\boldsymbol{x}, y) =& C_3 \cdot \left[\frac{1}{N^r}\sum_{j=1}^{N^r}\exp\left(-\frac{(y-y_j^r)^2}{2\sigma^2}\right)\right] \\ & \cdot \left[\frac{\sum_{i=1}^{N^r} w^r(y_i^r, y)\delta(\boldsymbol{x}-\boldsymbol{x}_i^r)}{\sum_{i=1}^{N^r} w^r(y_i^r, y)}\right], \end{aligned} \tag{8}$$

$$\begin{aligned}\hat{p}_g^{\text{SVE}}(\boldsymbol{x}, y) =& C_4 \cdot \left[\frac{1}{N^g}\sum_{j=1}^{N^g}\exp\left(-\frac{(y-y_j^g)^2}{2\sigma^2}\right)\right] \\ & \cdot \left[\frac{\sum_{i=1}^{N^g} w^g(y_i^g, y)\delta(\boldsymbol{x}-\boldsymbol{x}_i^g)}{\sum_{i=1}^{N^g} w^g(y_i^g, y)}\right], \end{aligned} \tag{9}$$

where C_3 and C_4 are two constants making these two estimates valid probability density functions,

$$w^r(y_i^r, y) = e^{-\nu(y_i^r-y)^2} \quad \text{and} \quad w^g(y_i^g, y) = e^{-\nu(y_i^g-y)^2}, \tag{10}$$

and the hyper-parameter $\nu > 0$. In Eqs. (8) and (9), similar to the HVEs, we estimate $p_r(y)$ and $p_g(y)$ by KDEs. Instead of using samples in a hard vicinity, the SVEs use all respective samples to estimate $p_r(\boldsymbol{x}|y)$ and $p_g(\boldsymbol{x}|y)$ but each sample is assigned a weight based on the distance of its label from y. Two diagrams in Fig. 2 visualize the process of using hard/soft vicinal samples to estimate the Gaussian distribution $p(\boldsymbol{x}|y)$ conditional on y, for univariate $\boldsymbol{x}$.

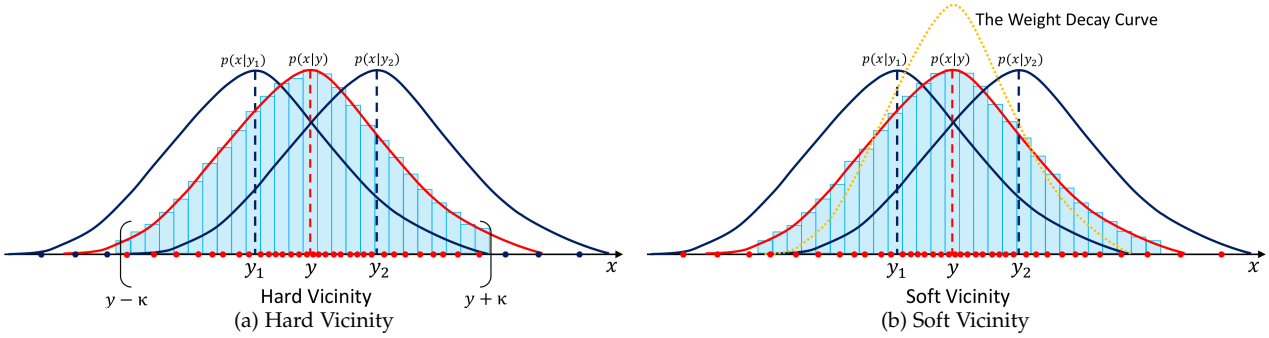

Fig. 2: HVE (Eqs. (6) and (7)) and SVE (Eqs. (8) and (9)) estimate $p(\boldsymbol{x}|y)$ (a univariate Gaussian conditional on y) by using two samples in hard and soft vicinities, respectively, of y. To estimate $p(\boldsymbol{x}|y)$ (the red Gaussian curve) only from samples drawn from $p(\boldsymbol{x}|y_1)$ and $p(\boldsymbol{x}|y_2)$ (the blue Gaussian curves), estimation is based on the samples (red dots) in a hard vicinity (defined by $y \pm \kappa$) or a soft vicinity (defined by the weight decay curve) around y. The histograms in blue are samples in the hard or soft vicinity. The labels y_1, y, and y_2 on the x-axis denote the means of $\boldsymbol{x}$ conditional on y_1, y, and y_2, respectively.

By plugging Eq. (6), (7), (8), and (9) into Eq. (1), we derive the *hard vicinal discriminator loss* (HVDL) and the *soft vicinal discriminator loss* (SVDL) as follows:

$$\begin{aligned}&\widehat{\mathcal{L}}^{\text{HVDL}}(D)\\ =&-\frac{C_5}{N^r}\sum_{j=1}^{N^r}\sum_{i=1}^{N^r}\mathbb{E}_{\epsilon^r\sim\mathcal{N}(0,\sigma^2)}\left[W_1\log(D(\boldsymbol{x}_i^r,y_j^r+\epsilon^r))\right]\\ &-\frac{C_6}{N^g}\sum_{j=1}^{N^g}\sum_{i=1}^{N^g}\mathbb{E}_{\epsilon^g\sim\mathcal{N}(0,\sigma^2)}\left[W_2\log(1-D(\boldsymbol{x}_i^g,y_j^g+\epsilon^g))\right],\end{aligned}\tag{11}$$

$$\begin{aligned}&\widehat{\mathcal{L}}^{\text{SVDL}}(D)\\ =&-\frac{C_7}{N^r}\sum_{j=1}^{N^r}\sum_{i=1}^{N^r}\mathbb{E}_{\epsilon^r\sim\mathcal{N}(0,\sigma^2)}\left[W_3\log(D(\boldsymbol{x}_i^r,y_j^r+\epsilon^r))\right]\\ &-\frac{C_8}{N^g}\sum_{j=1}^{N^g}\sum_{i=1}^{N^g}\mathbb{E}_{\epsilon^g\sim\mathcal{N}(0,\sigma^2)}\left[W_4\log(1-D(\boldsymbol{x}_i^g,y_j^g+\epsilon^g))\right],\end{aligned}\tag{12}$$

where $\epsilon^r \triangleq y - y_j^r$, $\epsilon^g \triangleq y - y_j^g$,

$$W_1 = \frac{\mathbb{1}_{\{|y_j^r+\epsilon^r-y_i^r|\leq\kappa\}}}{N^r_{y_j^r+\epsilon^r,\kappa}}, \quad W_2 = \frac{\mathbb{1}_{\{|y_j^g+\epsilon^g-y_i^g|\leq\kappa\}}}{N^g_{y_j^g+\epsilon^g,\kappa}}$$
$$W_3 = \frac{w^r(y_i^r,y_j^r+\epsilon^r)}{\sum_{i=1}^{N^r}w^r(y_i^r,y_j^r+\epsilon^r)}, \quad W_4 = \frac{w^g(y_i^g,y_j^g+\epsilon^g)}{\sum_{i=1}^{N^g}w^g(y_i^g,y_j^g+\epsilon^g)},$$

and C_5, C_6, C_7, and C_8 are some constants.

Generator training: The generator of CcGAN is trained by minimizing Eq. (13),

$$\widehat{\mathcal{L}}^{\epsilon}(G) = -\frac{1}{N^g}\sum_{i=1}^{N^g}\mathbb{E}_{\epsilon^g\sim\mathcal{N}(0,\sigma^2)}\log(D(G(\boldsymbol{z}_i,y_i^g+\epsilon^g),y_i^g+\epsilon^g)).\tag{13}$$

How do HVDL, SVDL, and Eq. (13) **overcome (P1)?**
(i) Given a label y as the condition, we use images in a hard/soft vicinity of y to train the discriminator instead of just using images with label y. It enables us to estimate $p_r(\boldsymbol{x}|y)$ when there are not enough real images with label y.
(ii) From Eqs. (11) and (12), we can see that the KDEs in Eqs. (6), (7), (8), and (9) are adjusted by adding Gaussian noise to the labels. Moreover, in Eq. (13), we add Gaussian noise to seen labels (assume the y_i^g are seen) to train the generator to generate images at unseen labels. This enables estimation of $p_r(\boldsymbol{x}|y')$ when y' is not in the training set.

Remark 1. Based on the kernel density estimation [28]–[31] and the property of the Dirac delta function (Appendix A of [27]), $\int\delta(\boldsymbol{x}-\boldsymbol{x}_i^r)d\boldsymbol{x} = \int\delta(\boldsymbol{x}-\boldsymbol{x}_i^g)d\boldsymbol{x} = 1$ and $C_1 = C_2 = C_3 = C_4 = 1/\sigma$. Therefore, $C_5 = C_6$ and $C_7 = C_8$, which implies these constants $C_1,\ldots,C_8$ can be ignored when minimizing $\widehat{\mathcal{L}}^{\text{HVDL}}(D)$ and $\widehat{\mathcal{L}}^{\text{SVDL}}(D)$.

Remark 2. An algorithm is proposed in Supp. S.9 for training CcGAN in practice. Moreover, CcGAN does not require any specific network architecture, so it can use modern GAN architectures such as SNGAN [24], SAGAN [5] and BigGAN [4]. CcGAN is also compatible with modern GAN training techniques such as DiffAugment [19].

Remark 3 (A rule of thumb for hyper-parameter selection)**.** In our experiments, we normalize labels to real numbers in $[0,1]$ and the hyper-parameter selection is conducted based on the normalized labels. To be more specific, the hyper-parameter σ is computed based on a rule of thumb formula for the bandwidth selection of KDE [30], i.e., $\sigma = \left(4\hat{\sigma}_{y^r}^5/3N^r\right)^{1/5}$, where $\hat{\sigma}_{y^r}$ is the sample standard deviation of normalized labels in the training set. Let $\kappa_{\text{base}} = \max\left(y_{[2]}^r - y_{[1]}^r, y_{[3]}^r - y_{[2]}^r, \ldots, y_{[N_{\text{uy}}^r]}^r - y_{[N_{\text{uy}}^r-1]}^r\right)$, where $y_{[l]}^r$ is the l-th smallest normalized distinct real label and N_{uy}^r is the number of normalized distinct labels in the training set. Then κ is set as a multiple of κ_{base} (i.e., $\kappa = m_\kappa\kappa_{\text{base}}$) where the multiplier m_κ stands for 50% of the minimum number of neighboring labels used for estimating $p_r(\boldsymbol{x}|y)$ given a label y. For example, $m_\kappa = 1$ implies using 2 neighboring labels (one on the left while the other one on the right). In our experiments, m_κ is generally set as 1 or 2. In some extreme case when many distinct labels have too few real samples, we may consider increasing m_κ. We also found $\nu = 1/\kappa^2$ works well in (10) in our experiments.

2.2 Solutions to (P2): Novel Label Input Mechanisms

In this section, we propose two solutions, consisting of a naive and an improved label input mechanism, to solve **(P2)**.
A naive label input (NLI) mechanism: We first propose a naive approach to incorporate the regression labels into the cGANs. For G, we add the label y element-wise to the output of its first linear layer. For D, the label y is first projected to

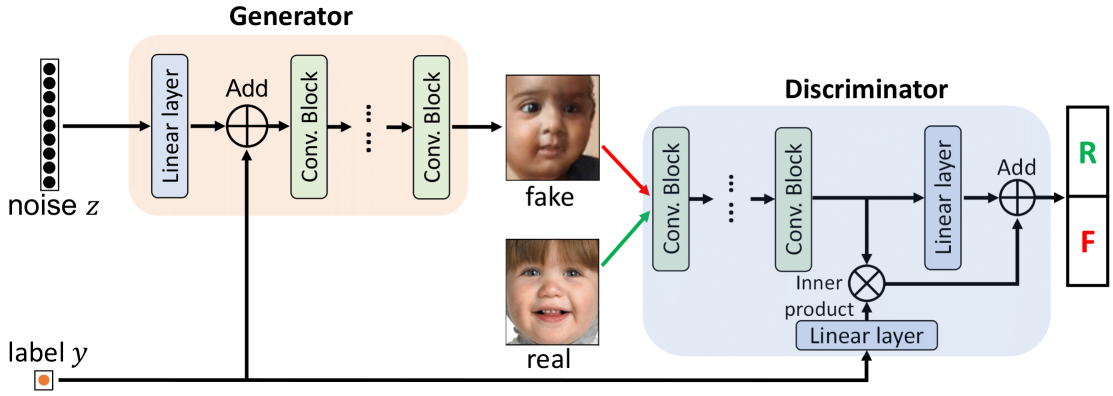

Fig. 3: **The workflow of the naive label input (NLI) mechanism.** NLI inputs a regression label y into G by adding y element-wise to the output of the first linear layer. NLI inputs y into D by label projection [3].

the latent space learned by an extra linear layer. Then, we incorporate the embedded label into the discriminator by label projection [3]. Fig. 3 visualizes the naive label input mechanism.

An improved label input (ILI) mechanism: Empirical studies in Section 5 show that CcGAN with the naive label input mechanism already substantially outperforms cGAN. Nevertheless, it still suffers from severe label inconsistency on some datasets (e.g., Cell-200 and Steering Angle). To improve the label consistency of CcGAN, we propose an improved label input (ILI) mechanism. The ILI approach consists of a pre-trained CNN and a label embedding network. The pre-trained CNN, as shown in Fig. 4, includes two subnetworks, T_1 and T_2, where T_1 maps an image $\boldsymbol{x}$ to a feature space and T_2 maps the extracted feature $\boldsymbol{h}$ to a regression label y. The dimension of the feature space is set to 128 in our experiments. The label embedding network T_3, as shown in Fig. 5, is a multilayer perceptron (MLP) [31] mapping a regression label y back to its hidden representation $\boldsymbol{h}$ in the feature space defined by T_1. Assume that there are m distinct regression labels in the training set, i.e., $y_1^u, y_2^u, \ldots, y_m^u$, then the label embedding network T_3 is trained by:

$$\min_{T_3} \frac{1}{m}\sum_{i=1}^{m} \mathbb{E}_{\gamma\sim\mathcal{N}(0,\sigma_\gamma^2)}\left[(T_2(T_3(y_i^u+\gamma))-(y_i^u+\gamma))^2\right], \tag{14}$$

where σ_γ is often a small value and is set at 0.2 in this paper. Then, given a regression label y, we can evaluate $T_3(y)$ to get a unique hidden representation of y which will be incorporated into CcGAN as the condition (visualized in Fig. 6). Specifically, for G, we input the embedded label by using conditional batch normalization [32]. For D, similar to the naive approach, we input the embedded label into D by label projection [3].

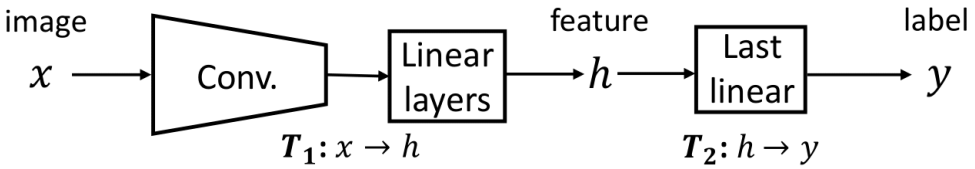

Fig. 4: **The pre-trained CNN $T_1 + T_2$ for label embedding.** The first subnetwork T_1 consists of some convolutional layers (Conv.) and some linear layers. The second subnetwork T_2 includes one linear layer.

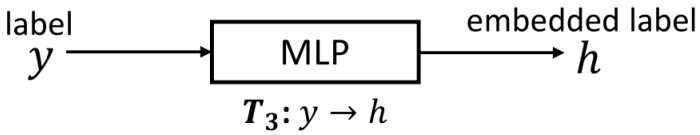

Fig. 5: **The label embedding network is a multilayer perceptron (MLP).**

3 Error Bounds of D Trained With HVDL and SVDL

In this section, we derive the error bounds of a discriminator D trained with $\widehat{\mathcal{L}}^{\text{HVDL}}(D)$ and $\widehat{\mathcal{L}}^{\text{SVDL}}(D)$ under the theoretical discriminator loss $\mathcal{L}(D)$. Denote by D^* the optimal discriminator [33] which minimizes $\mathcal{L}(D)$. Let $\widehat{D}^{\text{HVDL}} \triangleq \arg\min_{D\in\mathcal{D}} \widehat{\mathcal{L}}^{\text{HVDL}}(D)$; similarly, we define $\widehat{D}^{\text{SVDL}}$. We are interested in a reasonable bound (i.e, error bound) of the distance of $\widehat{D}^{\text{HVDL}}$ and $\widehat{D}^{\text{SVDL}}$ from D^* under $\mathcal{L}(D)$, i.e., $\mathcal{L}(\widehat{D}^{\text{HVDL}}) - \mathcal{L}(D^*)$ and $\mathcal{L}(\widehat{D}^{\text{SVDL}}) - \mathcal{L}(D^*)$. These error bounds theoretically illustrate how HVDL and SVDL influence the discriminator training, which can guide our implementation of HVDL and SVDL in practice such as the selection of κ and ν.

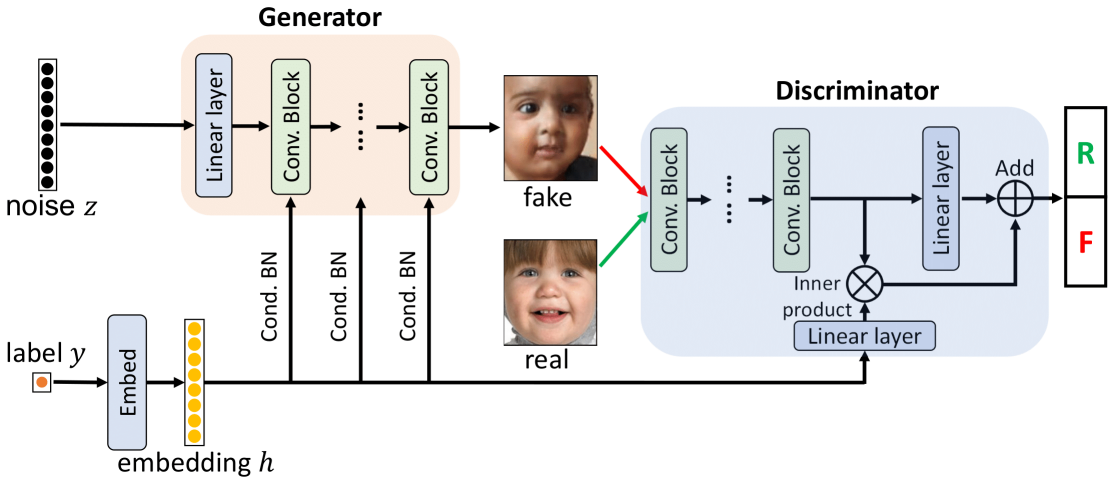

Fig. 6: **The workflow of the improved label input (ILI) mechanism.** ILI first uses an embedding network to convert y into its high-dimensional representation h. Then, h is input into G and D by conditional batch normalization [32] and label projection [3], respectively.

Before we move to the derivation, without loss of generality, we first assume $y \in [0,1]$. Then, we introduce some notations. Let $\mathcal{D}$ stand for the *Hypothesis Space* of D. Please note that $\mathcal{D}$ may not cover D^*. Let $\hat{p}_r^{\text{KDE}}(y)$ and $\hat{p}_g^{\text{KDE}}(y)$ stand for the KDEs of $p_r(y)$ and $p_g(y)$ respectively. Let $p_w^r(y'|y) \triangleq \frac{w^r(y',y)p^r(y')}{W^r(y)}$, $p_w^g(y'|y) \triangleq \frac{w^g(y',y)p^g(y')}{W^g(y)}$, $W^r(y) \triangleq \int w^r(y',y)p_r(y')dy'$ and $W^g(y) \triangleq \int w^g(y',y)p_g(y')dy'$.

Definition 1. (Hölder Class) Define the Hölder class of functions as:

$$\Sigma(L) \triangleq \left\{ p : \forall t_1, t_2 \in \mathcal{Y}, \exists L > 0, s.t. \frac{|p'(t_1) - p'(t_2)|}{|t_1 - t_2|} \leq L \right\}. \tag{15}$$

Please see Supp. S.10 for more details of these notations. Moreover, we will also work with the following assumptions:
(A1) All D's in $\mathcal{D}$ are measurable and uniformly bounded. Let $U \triangleq \max\{\sup_{D\in\mathcal{D}}[-\log D], \sup_{D\in\mathcal{D}}[-\log(1-D)]\}$ and $U < \infty$;
(A2) For $\forall \boldsymbol{x} \in \mathcal{X}$ and $y, y' \in \mathcal{Y}$, $\exists g^r(\boldsymbol{x}) > 0$ and $M^r > 0$, s.t. $|p_r(\boldsymbol{x}|y') - p_r(\boldsymbol{x}|y)| \leq g^r(\boldsymbol{x})|y' - y|$ with $\int g^r(\boldsymbol{x})d\boldsymbol{x} = M^r$;
(A3) For $\forall \boldsymbol{x} \in \mathcal{X}$ and $y, y' \in \mathcal{Y}$, $\exists g^g(\boldsymbol{x}) > 0$ and $M^g > 0$, s.t. $|p_g(\boldsymbol{x}|y') - p_g(\boldsymbol{x}|y)| \leq g^g(\boldsymbol{x})|y' - y|$ with $\int g^g(\boldsymbol{x})d\boldsymbol{x} = M^g$;
(A4) $p_r(y) \in \Sigma(L^r)$ and $p_g(y) \in \Sigma(L^g)$.

With these definitions and assumptions, we derive two lemmas based on which we derive the error bounds of a discriminator trained by using HVDL and SVDL in Theorems 1 and 2.

Lemma 1 (Lemma for HVDL). *Suppose that (A1)-(A2) and (A4) hold, then $\forall \delta \in (0,1)$, with probability at least $1-\delta$,*

$$\begin{aligned} &\sup_{D\in\mathcal{D}} \left| \frac{1}{N_{y,\kappa}} \sum_{i=1}^{N} \mathbb{1}_{\{|y-y_i|\leq\kappa\}} \left[-\log D(\boldsymbol{x}_i, y)\right] \right. \\ &\qquad \left. - \mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x}|y)} \left[-\log D(\boldsymbol{x}, y)\right] \right| \\ &\leq U\sqrt{\frac{1}{2N_{y,\kappa}} \log\left(\frac{2}{\delta}\right)} + \kappa U M, \end{aligned} \tag{16}$$

for a fixed y. If image-label pairs are real, then $N = N^r$, $N_{y,\kappa} = N_{y,\kappa}^r$, $p = p_r$, and $M = M^r$. Similarly, we have $N = N^g$, $N_{y,\kappa} = N_{y,\kappa}^g$, $p = p_g$, and $M = M^g$ for fake image-label pairs.

Lemma 2 (Lemma for SVDL). *Suppose that (A1), (A2) and (A4) hold, then $\forall \delta \in (0,1)$, with probability at least $1-\delta$,*

$$\begin{aligned} &\sup_{D\in\mathcal{D}} \left| \frac{\frac{1}{N}\sum_{i=1}^{N} w(y_i, y)\left[-\log D(\boldsymbol{x}_i, y)\right]}{\frac{1}{N}\sum_{i=1}^{N} w(y_i, y)} \right. \\ &\qquad \left. - \mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x}|y)} \left[-\log D(\boldsymbol{x}, y)\right] \right| \\ &\leq \frac{2U}{W(y)}\sqrt{\frac{1}{2N} \log\left(\frac{4}{\delta}\right)} + UM\mathbb{E}_{y'\sim p_w(y'|y)}\left[|y' - y|\right], \end{aligned} \tag{17}$$

for a fixed y. If image-label pairs are real, then $N = N^r$, $N_{y,\kappa} = N^r_{y,\kappa}$, $p = p_r$, $p_w = p^r_w$, $w = w^r$, $W = W^r$, and $M = M^r$. Similarly, we have $N = N^g$, $N_{y,\kappa} = N^g_{y,\kappa}$, $p = p_g$, $p_w = p^g_w$, $w = w^g$, $W = W^g$, and $M = M^g$ for fake image-label pairs.

Theorem 1 (Error Bound of D trained with HVDL)**.** *Assume that (A1)-(A4) hold, then $\forall \delta \in (0,1)$, with probability at least $1-\delta$,*

$$\begin{aligned}
\mathcal{L}(\widehat{D}^{HVDL}) - \mathcal{L}(D^*) \leq\ & 2U\left(\sqrt{\frac{C^{KDE}_{1,\delta}\log N^r}{N^r\sigma}} + L^r\sigma^2\right) \\
& + 2U\left(\sqrt{\frac{C^{KDE}_{2,\delta}\log N^g}{N^g\sigma}} + L^g\sigma^2\right) + 2\kappa U(M^r + M^g) \\
& + 2U\sqrt{\frac{1}{2}\log\left(\frac{8}{\delta}\right)}\left(\mathbb{E}_{y\sim\hat{p}^{KDE}_r(y)}\left[\sqrt{\frac{1}{N^r_{y,\kappa}}}\right]\right. \\
& \left. \quad + \mathbb{E}_{y\sim\hat{p}^{KDE}_g(y)}\left[\sqrt{\frac{1}{N^g_{y,\kappa}}}\right]\right) \\
& + \mathcal{L}(\widetilde{D}) - \mathcal{L}(D^*),
\end{aligned} \tag{18}$$

for some constants $C^{KDE}_{1,\delta}, C^{KDE}_{2,\delta}$ depending on δ.

Theorem 2 (Error Bound of D trained with SVDL)**.** *Assume that (A1)-(A4) hold, then $\forall \delta \in (0,1)$, with probability at least $1-\delta$,*

$$\begin{aligned}
\mathcal{L}(\widehat{D}^{SVDL}) - \mathcal{L}(D^*) \leq\ & 2U\left(\sqrt{\frac{C^{KDE}_{1,\delta}\log N^r}{N^r\sigma}} + L^r\sigma^2\right) \\
& + 2U\left(\sqrt{\frac{C^{KDE}_{2,\delta}\log N^g}{N^g\sigma}} + L^g\sigma^2\right) \\
& + 4U\sqrt{\frac{1}{2}\log\left(\frac{16}{\delta}\right)}\left(\frac{1}{\sqrt{N^r}}\mathbb{E}_{y\sim\hat{p}^{KDE}_r(y)}\left[\frac{1}{W^r(y)}\right]\right. \\
& \left. \quad + \frac{1}{\sqrt{N^g}}\mathbb{E}_{y\sim\hat{p}^{KDE}_g(y)}\left[\frac{1}{W^g(y)}\right]\right) \\
& + 2U\left(M^r\mathbb{E}_{y\sim\hat{p}^{KDE}_r(y)}\left[\mathbb{E}_{y'\sim\hat{p}^r_w(y'|y)}\left|y'-y\right|\right]\right. \\
& \left. \quad + M^g\mathbb{E}_{y\sim\hat{p}^{KDE}_g(y)}\left[\mathbb{E}_{y'\sim\hat{p}^g_w(y'|y)}\left|y'-y\right|\right]\right) \\
& + \mathcal{L}(\widetilde{D}) - \mathcal{L}(D^*),
\end{aligned} \tag{19}$$

for some constant $C^{KDE}_{1,\delta}$, $C^{KDE}_{2,\delta}$ depending on δ.

Remark 4. Please see Supp. S.10 for the proofs to these lemmas and theorems.

Remark 5 (Illustration of Theorems 1 and 2)**.** Both theorems imply HVDL and SVDL perform well if the output of D is not too close to 0 or 1 (i.e., favor small U). The first two terms in both upper bounds control the quality of KDE, which implies KDE works better if we have a large N^r and a large N^g but a small σ. The remaining terms of the two bounds are different. In the HVDL case, we favor small κ, M^r, and M^g. However, we should avoid setting κ too small, because we prefer large $N^r_{y,\kappa}$ and $N^g_{y,\kappa}$. In the SVDL case, we prefer small M^r and M^g but large $W^r(y)$ and $W^g(y)$. Large $W^r(y)$ and $W^g(y)$ imply that the weight function decays slowly (i.e., small ν). However, we should avoid setting ν too small because a small ν leads to large $\mathbb{E}_{y'\sim\hat{p}^r_w(y'|y)}|y'-y|$ and $\mathbb{E}_{y'\sim\hat{p}^g_w(y'|y)}|y'-y|$ (i.e., large weights for y''s which are far away from y). The rule of thumb formulae to select κ and ν in Remark 3 are consistent with our analysis here. Besides the rules of thumb, future work should propose a more refined hyper-parameter selection method.

4 Sliding Fréchet Inception Distance

A conditional GAN (no matter the type of the condition) needs to be evaluated from three perspectives [3], [5], [34]: (1) the visual quality, (2) the intra-label diversity (the diversity of fake images with the same label), and (3) the label consistency (whether assigned labels of fake images are consistent with their actual labels). Measuring the performance of cGANs from these three perspectives is often conducted by using a popular overall metric, termed the Intra-FID [3], [5], [34]. Intra-FID computes the *Fréchet inception distance* (FID) [35] separately at each of the distinct labels and reports the average FID score. Intra-FID is also used in our experiments on RC-49, UTKFace, and Cell-200 in Section 5; however, Intra-FID is not reliable or is even inapplicable when we have very few (even zero) real images for some distinct regression labels, e.g., the experiment on the Steering Angle dataset in Section 5.4. We therefore propose a novel metric, termed the *Sliding Fréchet Inception Distance*

(SFID), to replace Intra-FID in this scenario. SFID computes FID within an interval sliding on the range of the regression label y, and then reports the average of these FIDs. Specifically, we first prespecify a finite set of *SFID centers* c_{SFID} evenly over the range of y and a constant *SFID radius* r_{SFID}. Then, based on the c_{SFID} and r_{SFID}, we can define many joint *SFID intervals* of the form $[c_{\text{SFID}} - r_{\text{SFID}}, c_{\text{SFID}} + r_{\text{SFID}}]$. For each SFID interval, we compute FID between real and fake images with labels within this interval. Finally, SFID reports the average of these FIDs. Usually, we also report the standard deviation of these FIDs. We visualize the procedure for computing SFID in Fig. 7. A pseudo code for computing SFID is shown in Alg. 1. Similar to Intra-FID, a small SFID is preferred.

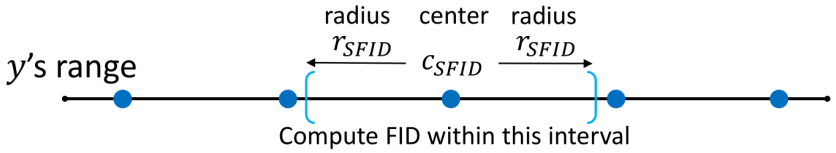

Fig. 7: **Sliding Fréchet Inception Distance (SFID).** We preset finite centers (blue dots) on y's range evenly and a radius r_{SFID}. Given an interval $[c_{\text{SFID}} - r_{\text{SFID}}, c_{\text{SFID}} + r_{\text{SFID}}]$, we compute FID between fake and real images with labels in this interval. SFID is the average of these FIDs.

Algorithm 1: An algorithm to compute the Sliding Fréchet Inception Distance (SFID).

Data: Real image-label pairs $D^r = \{(\boldsymbol{x}_1^r, y_1^r), \cdots, (\boldsymbol{x}_{N^r}^r, y_{N^r}^r)\}$; fake image-label pairs $D^g = \{(\boldsymbol{x}_1^g, y_1^g), \cdots, (\boldsymbol{x}_{N^g}^g, y_{N^g}^g)\}$; preset SFID centers $\{y_1^c, \cdots, y_{N^c}^c\}$; preset window radius r_{SFID}.
Result: $SFID$.

```
Initialize real and fake image sets in l-th sliding-window, i.e., D_l^r = φ and D_l^g = φ, l = 1, 2, ··· , N^c;
Initialize a FID array FID(l) = ∞, l = 1, 2, ··· , N^c;
for l = 1 to N^c do
    Update real image set at y_l^c: D_l^r = ⋃_{|y_i^r − y_l^c| ≤ r_SFID} {x_i^r}, i = 1, 2, ··· , N^r;
    Update fake image set at y_l^c: D_l^g = ⋃_{|y_i^g − y_l^c| ≤ r_SFID} {x_i^g}, i = 1, 2, ··· , N^g;
    Compute current FID: FID(l) = FID(D_l^r, D_l^g) ;
end
Compute SFID = (1/N^c) Σ_{l=1}^{N^c} FID(l).
```

5 Low-resolution Experiments

In this section, we study the effectiveness of CcGAN on four image datasets with resolution 64×64. Please note that our CGM task has never been studied in the literature, so there is no direct baseline. We modify conventional cGANs to create cGAN (K classes) and cGAN (concat) as baselines. For a fair comparison, from Sections 5.1 to 5.4, **all candidate methods use the same network architecture** (a customized DCGAN [36] architecture for Cell-200, and the SNGAN [24] architecture for the remaining datasets) except for the label input modules. The four CcGAN methods (i.e., HVDL+NLI, SVDL+NLI, HVDL+ILI, and SVDL+ILI) are tested in our experiments below. For stability, regression labels in all datasets are normalized to $[0, 1]$ during training.

Since GANs [33] do not explicitly estimate density functions, to measure the CGM quality, we evaluate the quality of fake images sampled from GANs. Following other cGAN methods [3], [5], [34], we use Intra-FID as the overall metric in our RC-49 (Section 5.1), UTKFace (Section 5.2), and Cell-200 (Section 5.3) experiments. The proposed SFID (see Section 4) is only used in the experiment conducted on the Steering Angle dataset in Section 5.4, where there are not enough real images to compute Intra-FID. The effectiveness of SFID is studied on the RC-49 dataset in Section 5.5, where we can control the sample size of real images. Besides Intra-FID and SFID, in each experiment (except Cell-200), we also compute three separate scores, i.e., Naturalness Image Quality Evaluator (NIQE) [37], Diversity, and Label Score, which evaluate fake images from three different perspectives. Furthermore, following [1], [4], [24], [26], we also report *Inception Score* (IS) [38] and *Fréchet Inception Distance* (FID) [35] of each cGAN for completeness; however, as illustrated in Section 5.6, IS and FID are not appropriate overall metrics for our experiment. The quantitative performances of the candidate cGANs on the four datasets are summarized in Table 1.

In the final experiment of this section, we demonstrate in Section 5.7 that CcGAN (SVDL+ILI) can significantly outperform state-of-the-art class-conditional GANs such as SNGAN [24], SAGAN [5], BigGAN [4], CR-BigGAN [39], BigGAN+DiffAug [19], and ReACGAN [40] in the CGM task.

5.1 RC-49

Since most benchmark datasets in the GAN literature do not have continuous, scalar regression labels, we propose a new benchmark dataset—RC-49, a synthetic dataset created by rendering 49 3-D chair models at different yaw angles. Each of 49 chair models is rendered at 899 yaw angles ranging from 0.1° to 89.9° with step size 0.1°. Therefore, RC-49 consists of 44,051 64×64 rendered RGB images and 899 distinct angles. Please see Supp. S.11 for more details of the data generation. Example images are shown in Fig. S.11.16 in Appendix.

TABLE 1: **Average quality of 64×64 fake images from cGANs and CcGANs with standard deviation after the "$\pm$" symbol.** We generate 179800, 60000, 200000, and 100000 fake images via each candidate method in the RC-49, UTKFace, Cell-200 and Steering Angle experiments, respectively. These fake images are evaluated through four metrics: Intra-FID, NIQE, Diversity, and Label Score. Note that IS and FID scores are also reported for completeness, but they are not suitable overall metrics for our task (see Section 5.7 for details). "↓" ("↑") indicates lower (higher) values are preferred. The best and second best results are marked in gray and green respectively.

Dataset	Model	Intra-FID ↓	NIQE ↓	Diversity ↑	Label Score ↓	IS ↑	FID ↓
RC-49	cGAN (150 classes)	1.720 ± 0.384	2.731 ± 0.162	0.779 ± 0.199	4.815 ± 5.152	2.382	1.066
	cGAN (concat)	1.141 ± 0.108	1.819 ± 0.111	2.459 ± 0.049	30.212 ± 21.391	11.440	0.295
	CcGAN (HVDL+NLI)	0.612 ± 0.145	1.869 ± 0.181	2.353 ± 0.121	5.617 ± 4.452	14.730	0.285
	CcGAN (SVDL+NLI)	0.515 ± 0.181	1.853 ± 0.159	2.610 ± 0.113	4.982 ± 4.439	19.425	0.207
	CcGAN (HVDL+ILI)	0.424 ± 0.081	1.805 ± 0.179	2.814 ± 0.052	1.816 ± 1.481	17.992	0.213
	CcGAN (SVDL+ILI)	0.389 ± 0.095	1.783 ± 0.173	2.949 ± 0.069	1.940 ± 1.489	20.173	0.197
UTKFace	cGAN (60 classes)	4.516 ± 0.965	2.315 ± 0.306	0.254 ± 0.353	11.087 ± 8.119	2.636	0.963
	cGAN (concat)	0.834 ± 0.199	2.051 ± 0.227	1.394 ± 0.026	17.291 ± 11.717	3.103	0.465
	CcGAN (HVDL+NLI)	0.572 ± 0.167	1.739 ± 0.145	1.338 ± 0.178	9.782 ± 7.166	3.328	0.114
	CcGAN (SVDL+NLI)	0.547 ± 0.181	1.753 ± 0.196	1.326 ± 0.198	10.739 ± 8.340	3.307	0.087
	CcGAN (HVDL+ILI)	0.480 ± 0.145	1.709 ± 0.169	1.280 ± 0.203	7.505 ± 5.857	3.256	0.056
	CcGAN (SVDL+ILI)	0.425 ± 0.157	1.725 ± 0.171	1.298 ± 0.176	7.452 ± 6.022	3.382	0.142
Cell-200	cGAN (100 classes)	90.255 ± 64.595	2.130 ± 2.440	—	66.748 ± 51.711	—	30.086
	cGAN (concat)	41.599 ± 21.430	3.250 ± 0.646	—	73.187 ± 51.133	—	37.689
	CcGAN (HVDL+NLI)	50.052 ± 20.584	1.488 ± 0.153	—	72.599 ± 37.425	—	40.279
	CcGAN (SVDL+NLI)	56.078 ± 19.334	1.829 ± 0.386	—	83.367 ± 49.577	—	51.318
	CcGAN (HVDL+ILI)	8.759 ± 6.652	1.283 ± 0.534	—	5.861 ± 4.900	—	3.263
	CcGAN (SVDL+ILI)	7.266 ± 2.305	1.220 ± 0.515	—	5.905 ± 5.020	—	1.684
Steering Angle	cGAN (210 classes)	3.285 ± 0.647	1.296 ± 0.095	0.603 ± 0.396	14.596 ± 15.402	2.572	0.976
	cGAN (concat)	2.446 ± 1.122	1.717 ± 0.003	1.255 ± 0.015	41.686 ± 25.864	3.251	0.255
	CcGAN (HVDL+NLI)	1.969 ± 0.676	1.093 ± 0.024	0.991 ± 0.361	22.322 ± 18.758	3.587	0.316
	CcGAN (SVDL+NLI)	1.866 ± 0.649	1.098 ± 0.038	1.007 ± 0.248	19.678 ± 18.281	3.968	0.212
	CcGAN (HVDL+ILI)	1.635 ± 0.699	1.152 ± 0.047	1.153 ± 0.153	10.868 ± 9.644	4.592	0.327
	CcGAN (SVDL+ILI)	1.546 ± 0.626	1.130 ± 0.078	1.156 ± 0.189	10.933 ± 8.978	4.439	0.331

Experimental setup: Not all images are used for the GAN training. A yaw angle is selected for training if its last digit is odd. Moreover, at each selected angle, only 25 images are randomly chosen for training. Thus, the training set includes 11250 images and 450 distinct angles. The remaining images are held out for evaluation.

When training cGAN (K classes), we divide $[0.1°, 89.9°]$ into 150 equal intervals where each interval is treated as a class. When training CcGAN, we use the rule of thumb formulae in Remark 3 to select the three hyper-parameters of HVDL and SVDL, i.e., $\sigma \approx 0.047$, $\kappa \approx 0.004$ and $\nu = 50625$. The two novel label input mechanisms for CcGAN (NLI and ILI) are implemented in this experiment. For ILI, we pre-train a modified ResNet-34 [41] with 3 linear layers after the average pooling layer and we only keep the last linear layer for label embedding (i.e., the $T_1 + T_2$ in Fig. 4). We use a five-layer MLP with 128 nodes in each layer to convert an angle into its hidden representation (i.e., the T_3 in Fig. 5). All candidates are trained for 30,000 iterations with batch size 256. Afterwards, we evaluate the trained GANs on all 899 angles by generating 200 fake images for each angle. Please see Supp. S.11 for the network architectures and more details about the training/testing setup.
Quantitative and visual results: To evaluate (1) the visual quality, (2) the intra-label diversity, and (3) the label consistency of fake images, we study an overall metric and three separate metrics here. (i) **Intra-FID** [3] is utilized as the overall metric. It computes FID [35] separately at each of the 899 evaluation angles and reports the average FID score along with the standard deviation of these 899 FIDs. (ii) **NIQE** [37] measures the visual quality only. (iii) **Diversity** is the average entropy of predicted chair types of fake images over evaluation angles. (iv) **Label Score** is the average absolute error between assigned angles and predicted angles. Please see Supp. S.11.5 for details of these metrics.

We report in Table 1 the performances of each GAN. The example fake images in Fig. S.11.16 in Appendix and line graphs in Fig. 8 support the quantitative results. cGAN (150 classes) often generates unrealistic, identical images for a target angle (i.e., low visual quality and low intra-label diversity). "Binning" $[0.1°, 89.9°]$ into other number of classes (e.g., 90 classes and 210 classes) is also tried but does not improve cGAN's performance. cGAN (concat) has good visual quality and high intra-label diversity but terrible label consistency. In contrast, the four CcGAN methods perform well from all three perspectives, i.e., good visual quality, high intra-label diversity, and high label consistency. Moreover, both ILI-based CcGANs outperform the two NLI-based CcGANs in terms of all four metrics, especially the label score.
Extra experimental results: To test cGAN and CcGAN under more challenging scenarios, we vary the sample size for each distinct angle in the training set from 45 to 5. We visualize the line graphs of Intra-FID versus the sample size for each distinct training angle in Fig. 9. From this figure, we can see the four CcGAN methods substantially outperform two cGANs and ILI performs better than NLI no matter what is the sample size for each distinct angle in the training set. The overall trend in this figure also shows that smaller sample size reduces the performance of both cGAN and CcGAN.

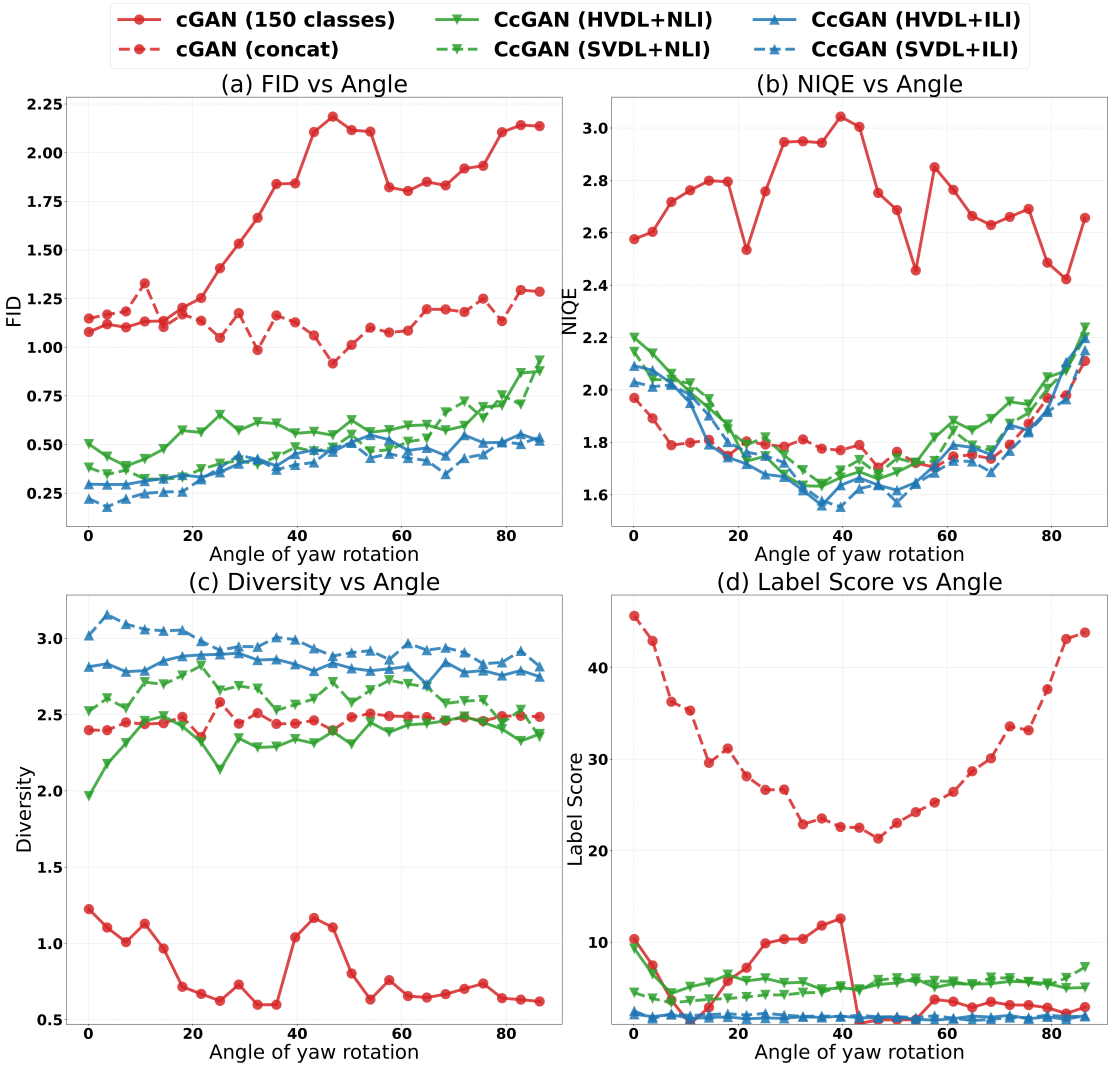

Fig. 8: **Line graphs of FID/NIQE/Diversity/Lable Score versus yaw angle for RC-49.** Figs. (a) to (c) show that four CcGAN methods consistently outperform cGAN (150 classes) across all angles. All graphs of CcGANs appear much smoother than those of cGAN (150 classes) because of HVDL and SVDL. Figs. (a) and (d) show that four CcGAN methods consistently outperform cGAN (concat) across all angles. Moreover, in most graphs, we can clearly see ILI-based CcGANs perform better than NLI-based CcGANs.

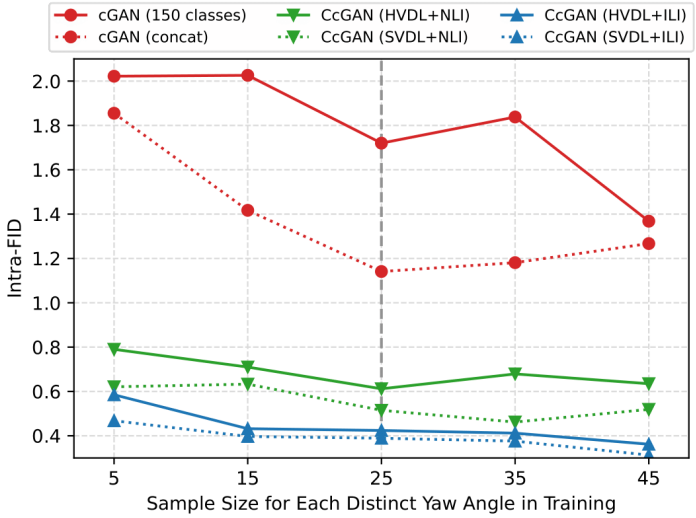

Fig. 9: **Line graphs of Intra-FID versus the sample size for each distinct training angle of RC-49.** The grey vertical dashed line stands for the sample size used in the main study of the RC-49 experiment. Four CcGAN methods substantially outperform two cGANs and ILI performs better than NLI no matter what the sample size for each distinct angle in the training set. The overall trend in this figure shows that a smaller sample size deteriorates the performance of both cGAN and CcGAN.

5.2 UTKFace

In this section, we compare CcGAN and cGAN on UTKFace [42], a dataset consisting of RGB images of human faces which are labeled by age.

Experimental setup: In this experiment, we only use images with age in $[1, 60]$. Some images with bad visual quality and watermarks are also discarded. After the preprocessing, 14,760 images are left. The number of images for each age ranges from 50 to 1051. We resize all selected images to 64×64. Some example UTKFace images are shown in the first image array in Fig. S.12.21.

When implementing cGAN (K classes), each age is treated as a class. For CcGAN we use the rule of thumb formulae in Remark 3 to select the three hyper-parameters of HVDL and SVDL, i.e., $\sigma \approx 0.041$, $\kappa \approx 0.017$ and $\nu = 3600$. Similar to the RC-49 experiment, we use NLI and ILI to incorporate ages into CcGAN. All GANs are trained for 40,000 iterations with batch size 512. In testing, we generate 1,000 fake images from each trained GAN for each age. Please see Supp. S.12 for more details of the data preprocessing, network architectures and training/testing setup.

Quantitative and visual results: Similar to the RC-49 experiment, we evaluate the quality of fake images by Intra-FID, NIQE, Diversity (entropy of predicted races), and Label Score. We report in Table 1 the average quality of 60,000 fake images. From this table, we can see the four CcGAN methods substantially outperform both cGANs and ILI performs better than NLI. Notably, although cGAN (concat) has the highest Diversity score, the huge label score reveals that cGAN (concat) cannot control the image generation with respective to ages. Thus, cGAN (concat) fails in this experiment. We also show in Fig. S.12.21 some example fake images from candidate models and line graphs of FID/NIQE/Diversity/Lable Score versus Age in Fig. 10. Analogous to the quantitative comparisons, we can see that CcGAN performs much better than cGAN.

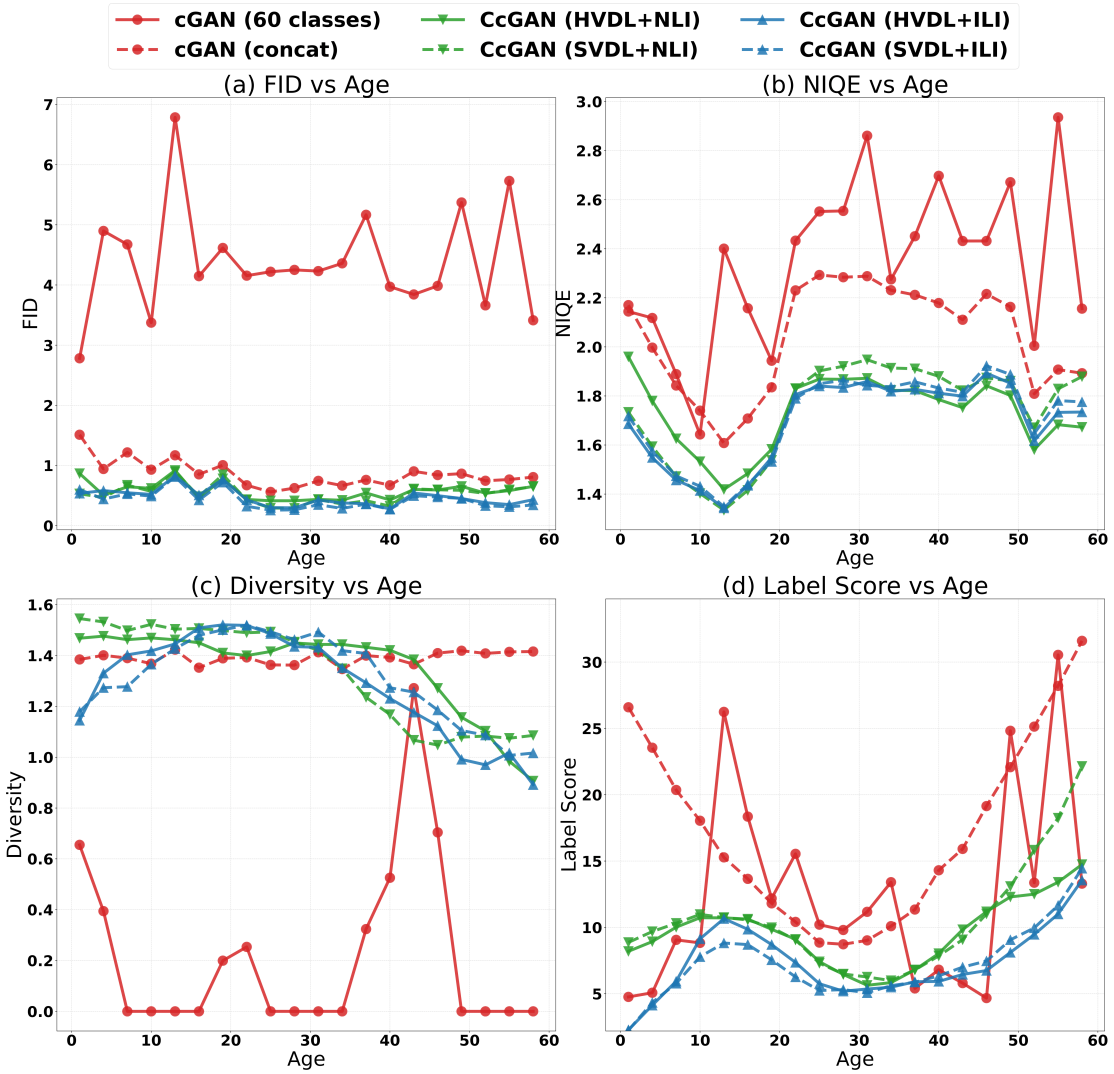

Fig. 10: **Line graphs of FID/NIQE/Diversity/Lable Score versus Age for UTKFace.** The four CcGAN methods significantly outperform cGAN (60 classes) in Figs. (a) to (c). All graphs of CcGANs appear much smoother than those of cGAN (60 classes) because of HVDL and SVDL. Figs. (a) and (d) show that four CcGAN methods consistently outperform cGAN (concat) across all ages. Fig. (d) also shows that the ILI-based CcGANs have higher label consistency than the NLI-based CcGANs.

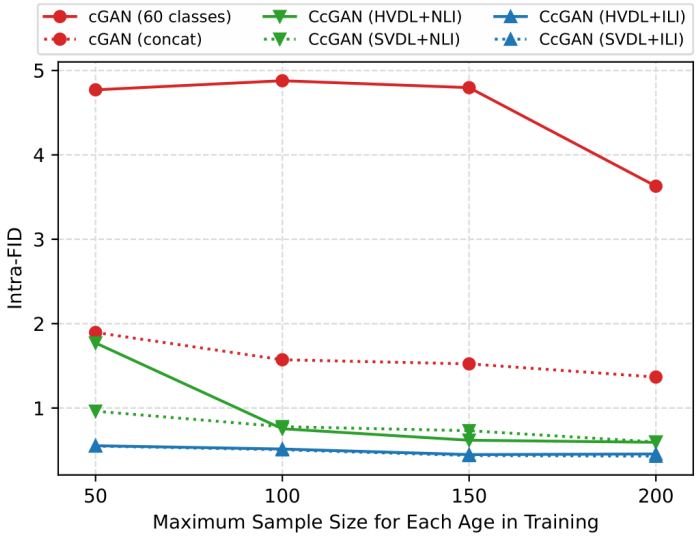

Fig. 11: **Line graphs of Intra-FID versus the maximum sample size for each distinct age in the training set of UTKFace.** Fig. 11 shows that four CcGAN methods perform much better than cGAN and ILI is better than NLI. Moreover, a smaller sample size deteriorates the performance of both cGAN and CcGAN.

Extra experimental results: The histogram in Fig. S.12.20 shows that the UTKFace dataset is highly imbalanced. To balance the training data and also test the performance of cGAN and CcGAN under smaller sample sizes, we vary the maximum sample size for each distinct age in the training from 200 to 50. Note that, we do not restrict the maximum sample size in the main study. Since we have a much smaller sample size, we reduce the number of iterations for the GAN training from 40,000 to 20,000 and slightly increase m_κ in Remark 3 from 1 to 2 (we therefore use a wider hard/soft vicinity). We visualize the line graphs of Intra-FID versus the maximum sample size for each age of cGAN and CcGAN in Fig. 11. From the figure, we can clearly see that a smaller sample size worsens the performance of both cGAN and CcGAN. Moreover, the Intra-FID scores of two cGANs often stay at a high level and are larger than those of the four CcGAN methods. The ILI-based CcGANs are also better than the NLI-based CcGANs.

5.3 Cell-200

In addition to RC-49, we propose another benchmark dataset–Cell-200, a dataset of synthetic fluorescence microscopy images with cell populations generated by SIMCEP [43]. Please see Supp. S.13.1 for more details about the data generation. Some example images are shown in Fig. S.13.25.

Experimental setup: The Cell-200 dataset consists of 200,000 64×64 grayscale images. The number of cells per image ranges from 1 to 200 and there are 1,000 images for each cell count. However, only a subset of Cell-200 with only odd cell counts and 10 images per count (1,000 training images in total) is used for the GAN training.

When training cGAN (K classes), we divide $[1, 200]$ into 100 equal intervals where each interval is treated as a class (i.e., $K = 100$). We use the rule of thumb formulae in Remark 3 to select the three hyperparameters of HVDL and SVDL, i.e., $\sigma \approx 0.077$, $\kappa \approx 0.020$ and $\nu = 2500$. Both cGAN and CcGAN are trained for 5,000 iterations. Afterwards, we evaluate the trained GANs on all 200 cell counts by generating 1,000 fake images for each count. Please see Supp. S.13 for the network architectures and more details about the training/testing setup.

Quantitative and visual results: We evaluate the quality of fake images by Intra-FID, NIQE, and Label Score. Please note that the Diversity score is not available in this experiment because there is no class label in Cell-200. We report in Table 1 the average quality of 200,000 fake images from cGAN and CcGAN. We also show in Fig. S.13.25 some example fake images from cGAN and CcGAN and line graphs of FID/NIQE/Label Score versus Cell Count in Fig. 12. Unlike the experimental results on RC-49 and UTKFace, although NLI-based CcGANs outperform cGAN (100 classes) in terms of Intra-FID and NIQE, their label scores are very high, which implies low label consistency. Fortunately, two ILI-based CcGANs still perform very well and substantially outperform two cGANs and two NLI-based CcGANs.

5.4 Steering Angle

In this section, we demonstrate the effectiveness of the proposed CcGAN on the Steering Angle dataset, a subset of an autonomous driving dataset [44]. The complete dataset consists of 109,231 RGB images. Each image is taken by using a dash camera mounted on a car and, at the same moment, the angle of the steering wheel rotation of the same car (i.e., steering angle) is recorded by a device attached to the steering wheel. Thus, each image in this autonomous driving dataset is paired with a steering angle ranging from $-338.82°$ to $501.78°$.

Experimental setup: To make the training and evaluation easier, we remove many images in this autonomous driving dataset where an image is removed due to at least one of the following reasons:

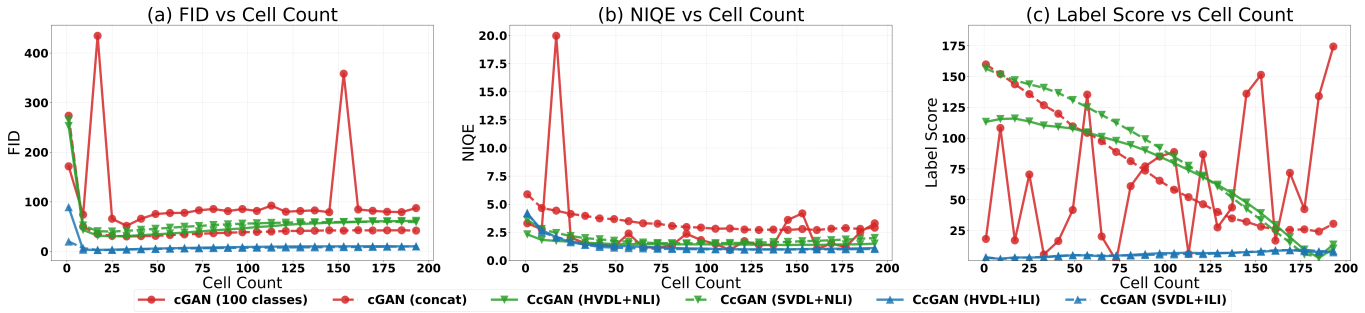

Fig. 12: **Line graphs of FID/NIQE/Lable Score versus Cell Count for Cell-200.** Figs. (a) to (c) show that, although the NLI-based CcGANs do not perform well, the ILI-based CcGANs outperform both cGANs across most cell counts. All graphs of CcGANs also appear much smoother than those of cGAN (100 classes) because of HVDL and SVDL. Moreover, in all figures, we can see ILI-based CcGANs perform better than NLI-based CcGANs especially in Fig. (c).

- The image is incorrectly labeled (e.g., some images show that the car was turning left/right but the corresponding steering angles are zero).
- The image has very bad visual quality due to overexposure or underexposure.
- There is no reference object (e.g., double yellow lines or side roads) in the image to let a human visually determine whether the car was turning left/right.
- The corresponding steering angle is outside $[-80^\circ, 80^\circ]$.

Eventually, there are 12,271 images left with 1,904 distinct steering angles in $[-80^\circ, 80^\circ]$. These images are then resized to 64×64 and they form a subset of the autonomous driving dataset [44], termed *Steering Angle* in this paper. Please note that the Steering Angle dataset is highly imbalanced and a histogram of steering angles is shown in Fig. S.14.28.

When training cGAN (K classes), we divide $[-80^\circ, 80^\circ]$ into 210 equal intervals where each interval is taken as a class (i.e., $K = 210$). When implementing CcGAN, the three hyper-parameters of HVDL and SVDL are selected by the rule of thumb formulae in Remark 3, i.e., $\sigma \approx 0.029$, $\kappa \approx 0.032$ and $\nu \approx 1000.438$. All GANs are trained for 20,000 iterations. To evaluate the candidate models, we choose 2,000 evenly spaced angles in $[-80^\circ, 80^\circ]$ and generate 50 images from each candidate GAN model for each of these angles. Please see Supp. S.14 for the network architectures and more details about the training/testing setup.

Quantitative and visual results: To evaluate the quality of fake images, we use the proposed *Sliding Fréchet Inception Distance* (SFID) as the overall metric instead of Intra-FID, since we have very few real images for many angles (e.g., angles close to the two end points of $[-80^\circ, 80^\circ]$). We preset 1,000 SFID centers in $[-80^\circ, 80^\circ]$ and let the SFID radius be 2°. NIQE, Diversity (entropy of predicted types of scenes), and Label Score are also reported. Please see Supp. S.14.5 for more details of these performance measures.

We report in Table 1 the average quality of 100,000 fake images from each candidate method. Some example fake images are also shown in Fig. S.14.30 in Appendix. We also compute FID, NIQE, Diversity, and Label Score in each SFID interval and plot the line graphs of FID/NIQE/Diversity/Label Score versus SFID Center in Fig. 13. Based on these quantitative and visual results, we can conclude:

- The two ILI-based CcGAN methods are better than cGAN (210 classes) in terms of all four metrics; however, the two NLI-based CcGAN methods have lower label consistency than cGAN (210 classes). Although cGAN (concat) has the highest Diversity score, the four CcGAN methods outperform cGAN (concat) in terms of the other three metrics.
- Although the NIQE score and Label Score of cGAN (210 classes) are not grossly uncompetitive, cGAN (210 classes) has a very low Diversity score and Fig. 13(c) shows that the Diversity scores are almost zero at some angles. Example fake images in Fig. S.14.30 also show that cGAN (210 classes) has the mode collapse problem [25], [45], [46] (i.e., it always generates the same image for some angles).
- Although cGAN (concat) has the highest Diversity score, its NIQE score and Label Score are terrible, implying bad visual quality and low label consistency. Example fake images in Fig. S.14.30 support these quantitative results.
- The line graphs in Fig. 13 show that the performance of cGAN (210 classes) is very unstable across all SFID intervals. In contrast, cGAN (concat) has very smooth graphs but its graph for Label Score is above all other graphs.
- The two ILI-based CcGANs perform better than the two NLI-based CcGANs in terms of all metrics except NIQE.

5.5 Effectiveness of SFID

In this section, we study the effectiveness of SFID on RC-49. Since RC-49 has a large enough sample size of real images, we can get a reliable Intra-FID. At the same time, we can also deliberately reduce the sample size of real images to mimic the scenario where a reliable Intra-FID is not applicable but SFID still works well. The experiment in this section can also be conducted on Cell-200 but is omitted in this paper. We study 11 combinations of the SFID radius (r_{SFID}) and the number of SFID centers (# c_{SFID}) in this experiment where we use SFID to evaluate cGAN and two CcGAN methods (i.e., HVDL+NLI and HVDL+ILI) pre-trained in Section 5.1. In Setting 1, we let $r_{\text{SFID}} = 0$ so SFID degenerates to Intra-FID. In the same setting,

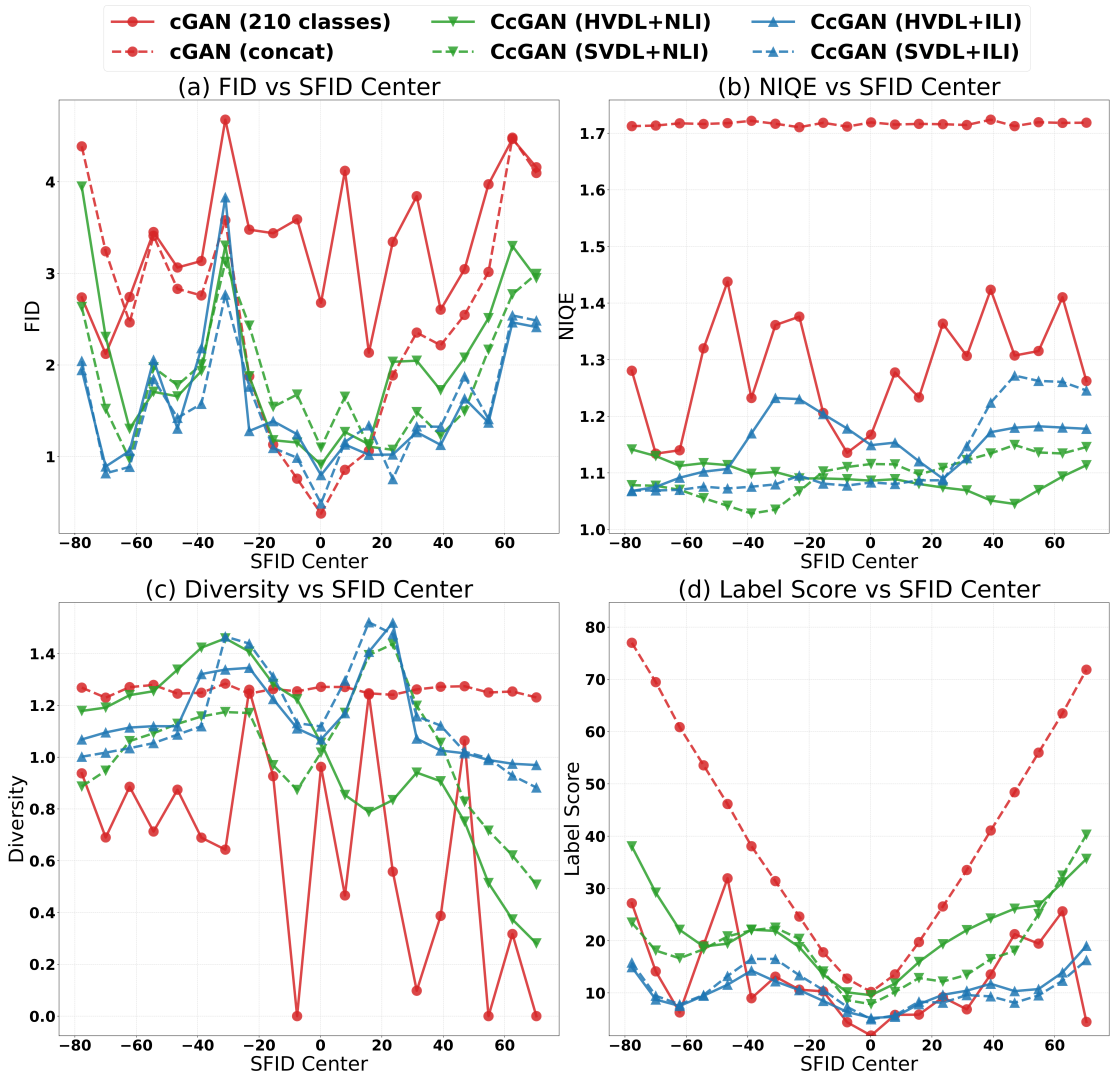

Fig. 13: **Line graphs of FID/NIQE/Diversity/Lable Score versus SFID Center for the Steering Angle dataset.** To plot these line graphs, we compute these metrics within each SFID interval defined by the corresponding SFID center. Figs. (a) to (d) show that, although the NLI-based CcGANs do not have good label consistency, the ILI-based CcGANs substantially outperform cGAN (210 classes) in most SFID intervals. All graphs of CcGANs also appear much smoother than those of cGAN (210 classes) because of HVDL and SVDL. Figs. (b) to (c) show that athough cGAN (concat) has the highest Diversity score, it also has the worst NIQE score and Label Score.

we evaluate the three GANs on all 899 distinct angles and all real images in RC-49 are used to compute Intra-FID, so Setting 1 is taken as the oracle in this experiment. In Setting 2, we also let $r_{\text{SFID}} = 0$ so SFID degenerates to Intra-FID again. In Setting 2, however, we only evaluate GANs on the 450 distinct angles which are seen in the training set of the experiment in Section 5.1. Moreover, to simulate the scenario where we have very few real images to compute Intra-FID, we deliberately reduce the number of real images at each distinct angle from 49 to 10. Therefore, in Setting 2, there are 10 real images for each angle seen in the training set and 0 real image for each angle unseen in the training set. Setting 2 is treated as the baseline in this experiment. Settings 3 to 11 are designed to show the effectiveness of SFID so we let $r_{\text{SFID}} > 0$. Similar to Setting 2, from Settings 3 to 11, real images are available only for those 450 distinct angles seen in the training set and only 10 real images are available for each angle. We consider three values for r_{SFID} (i.e., 0.5, 1, and 2) and three values for the number of c_{SFID}'s (i.e., 400, 600, and 800).

For all settings, we compute one FID in each SFID interval (in Settings 1 and 2, the SFID interval degenerates to an angle) and report in Table 2 the mean of these FIDs along with their standard deviation after the "$\pm$" symbol. Setting 1 is the oracle setting whose evaluation results can be seen as the ground truth, and we hope the evaluation results of SFID are close to Setting 1. In Setting 2, when we have very few real images (even zero) for each angle, Intra-FID overestimates the average FID of each GAN (e.g., from 1.7201 to 1.9664 for cGAN) and underestimates the quantitative difference between cGAN and CcGANs (e.g., from $(1.7201 - 0.6119)/1.7201 \approx 64.4\%$ to $(1.9664 - 1.2102)/1.9664 \approx 38.5\%$ for cGAN and HVDL+NLI). However, the performance of our proposed SFID in Settings 3 to 11 is very close to the oracle setting. If we compare within Settings 3 to 11, we can see r_{SFID} is inversely proportional to the SFID score while # c_{SFID} does not have obvious influence

on SFID. From Table 2, we may conclude that as long as r_{SFID} is set at a moderate level, SFID is a valid proxy to the oracle Intra-FID when there are insufficient real images to compute Intra-FID.

TABLE 2: **Evaluation results of SFID on RC-49 under different setups of r_{SFID} and number of c_{SFID}'s.** In the first two settings, SFID degenerates to Intra-FID since $r_{\text{SFID}} = 0$. In Setting 1, we evaluate GANs on all 899 distinct angles and all real images are used to compute Intra-FID, so Setting 1 is the oracle setting. In Setting 2, we evaluate GANs on the 450 angles seen in the training set and, for each angle, 10 real images are used to compute Intra-FID, so Setting 2 is treated as the baseline. The performance of our proposed SFID (Settings 3 to 11) is close to the oracle setting while Intra-FID (Setting 2) tends to overestimate the average FID and underestimate the quantitative difference between cGAN and CcGANs.

Setting	r_{SFID}	# c_{SFID}	cGAN (150 classes)	HVDL+NLI	HVDL+ILI
1 (Oracle)	0	899	1.720 ± 0.384	0.612 ± 0.145	0.424 ± 0.081
2 (Baseline)	0	450	1.966 ± 0.497	1.210 ± 0.298	1.032 ± 0.304
3	0.5	400	1.764 ± 0.401	0.675 ± 0.145	0.426 ± 0.087
4	1	400	1.726 ± 0.379	0.618 ± 0.123	0.412 ± 0.079
5	2	400	1.685 ± 0.356	0.581 ± 0.104	0.401 ± 0.073
6	0.5	600	1.764 ± 0.399	0.675 ± 0.144	0.425 ± 0.087
7	1	600	1.727 ± 0.378	0.618 ± 0.123	0.412 ± 0.078
8	2	600	1.685 ± 0.357	0.582 ± 0.104	0.401 ± 0.073
9	0.5	800	1.765 ± 0.399	0.675 ± 0.144	0.425 ± 0.087
10	1	800	1.726 ± 0.378	0.618 ± 0.123	0.412 ± 0.078
11	2	800	1.686 ± 0.357	0.582 ± 0.104	0.401 ± 0.073

5.6 Evaluation in Terms of IS and FID

For completeness, we also report in Table 1 the FID and IS scores of fake images generated from candidate methods. However, we emphasize that IS and FID cannot measure intra-label diversity and label consistency since computing IS and FID does not need the actual and assigned labels of fake images. Nonetheless, we see that CcGANs, especially ILI-based CcGANs, outperform the two conventional cGANs in terms of IS and FID too. Please see Supp. S.15 for detailed setups of this evaluation and more illustrations.

5.7 Comparison Against State of The Art cGANs

In Sections 5.1 to 5.4, to make the comparison fair and focus attention on the effectiveness of the proposed loss functions and label input mechanisms, both conventional cGANs and CcGANs adopt the same network architectures (e.g., SNGAN) and training techniques (e.g., with or without DiffAugment [19]). Unlike the above experiments, in this section, we aim to show that CcGAN (SVDL+ILI) can still outperform state of the art cGANs including SNGAN [24], SAGAN [5], BigGAN [4], CR-BigGAN [39], BigGAN+DiffAugment [19], and ReACGAN [40], which are equipped with advanced network architectures and training techniques.

Among these state of the art cGANs, SNGAN [24] and SAGAN [5] propose new network architectures. BigGAN [4] proposes not only the BigGAN architecture but also a bag of effective techniques for training cGANs. CR-BigGAN [39] proposes consistency regularization for BigGAN's training. BigGAN+DiffAugment [19] applies the DiffAugment technique to stabilize cGANs' training. DiffAugment is applicable to CcGAN too. ReACGAN [40] introduces novel training techniques for ACGAN [2], which makes ACGAN state of the art again.

We want to emphasize that, although the above state of the art cGANs work well in the class-conditional CGM tasks, they cannot model the distribution of images conditional on regression labels due to lack of a suitable label input mechanism. To make them fit into the regression scenario, we apply the binning strategy in Section 5.1 to convert rotation angles into class labels and then train them on RC-49 in a class-conditional manner. For the implementation of CcGAN (SVDL+ILI), unlike the setup in Section 5.1, we test with more advanced network architectures including SAGAN, and BigGAN. We also incorporate DiffAugment into CcGAN training, i.e., SAGAN+DiffAugment and BigGAN+DiffAugment. Similar to Section 5.1, we evaluate candidate models on 899 distinct angles by generating 200 fake images for each angle. Please see Supp. S.16 for detailed training and evaluation setups.

Comparison results are summarized in Table 3. We can see class-conditional SNGAN, SAGAN, and BigGAN suffer from mode collapse problems due to their very low Diversity. CR-BigGAN has higher Diversity but very low label consistency. The DiffAugment technique can substantially improve BigGAN's performance, but it is still much worse than CcGAN (SVDL+ILI). ReACGAN performs best among class-conditional GANs, but it is outperformed by CcGAN (SVDL+ILI) which is equipped with the SAGAN/BigGAN architecture and DiffAugment. These results demonstrate that, without HVDL/SVDL and NLI/ILI, existing architectures or training techniques cannot solve **(P1)** and **(P2)**.

6 High-resolution Experiments

Besides the low-resolution experiments in Section 5, we also compare CcGAN (SVDL+ILI) with cGAN (K classes) and cGAN (concat) on high-resolution images. We consider three datasets with different resolutions, i.e., RC-49 (128×128 and

TABLE 3: **Comparison against state of the art cGANs on RC-49.** In this table, we show the average quality of 179,800 fake images from class-conditional GANs and CcGAN (SVDL+ILI) with the standard deviation after the "$\pm$" symbol. IS and FID scores are also reported for completeness. "$\downarrow$" ("$\uparrow$") indicates lower (higher) values are preferred. The best and second best results are marked in gray and green respectively.

Model	Network Architecture and Training Technique	Intra-FID $\downarrow$	NIQE $\downarrow$	Diversity $\uparrow$	Label Score $\downarrow$	IS $\uparrow$	FID $\downarrow$
cGAN (150 classes)	SNGAN [24] (2017)	1.720 ± 0.384	2.731 ± 0.162	0.779 ± 0.199	4.815 ± 5.152	2.382	1.066
	SAGAN [5] (2018)	1.288 ± 0.223	2.517 ± 0.257	0.898 ± 0.372	36.388 ± 22.619	1.950	1.061
	BigGAN [4] (2019)	2.193 ± 0.372	2.399 ± 0.201	0.543 ± 0.315	4.467 ± 3.008	1.615	1.375
	CR-BigGAN [39] (2020)	0.981 ± 0.239	2.570 ± 0.142	2.000 ± 0.119	21.530 ± 17.861	3.565	0.479
	BigGAN+DiffAugment [19] (2020)	0.545 ± 0.206	2.138 ± 0.120	3.250 ± 0.051	9.595 ± 3.896	25.400	0.119
	ReACGAN [40] (2021)	0.178 ± 0.036	1.938 ± 0.141	3.145 ± 0.028	1.422 ± 1.337	21.945	0.056
CcGAN (SVDL+ILI)	SNGAN	0.389 ± 0.095	1.783 ± 0.173	2.949 ± 0.069	1.940 ± 1.489	20.173	0.197
	SAGAN+DiffAugment	0.106 ± 0.029	1.784 ± 0.173	3.590 ± 0.041	2.431 ± 0.449	36.546	0.026
	BigGAN+DiffAugment	0.086 ± 0.028	1.950 ± 0.166	3.620 ± 0.090	2.468 ± 1.920	30.686	0.013

256×256), UTKFace (128×128 and 192×192), and Steering Angle (128×128). Since high-resolution experiments are more challenging than low-resolution ones, we make some changes to the setups in Section 5 to improve GANs' performance. First, all candidates use the more advanced SAGAN [5] architecture. Second, cGAN (K classes) and cGAN (concat) are trained with the hinge loss [47]. CcGAN (SVDL+ILI) is trained with the reformulated hinge loss (see Eq. (S.45) in Supp. S.17.0.1). Third, DiffAugment [19], a recently proposed technique for training GANs with few samples, is also applied to improve the performance of the candidate methods.

Both visual and quantitative results demonstrate that the high-resolution fake images generated from CcGAN are visually realistic, diverse, and label consistent. These results also show that CcGAN is compatible with state-of-the-art GAN architectures and training techniques. Furthermore, the failure patterns of cGAN (K classes) and cGAN (concat) in this experiment are consistent with those in Section 5. cGAN (K classes) tends to have high label consistency but bad visual quality and low diversity. Oppositely, cGAN (concat) often has high diversity but bad/fair visual quality and terrible label consistency.

TABLE 4: **Average quality of high-resolution fake images from cGAN and CcGAN with the standard deviation after the "$\pm$" symbol.** We generate 179800, 60000, and 100000 fake images via each candidate method for the RC-49, UTKFace, and Steering Angle experiments, respectively. These fake images are evaluated under four metrics: Intra-FID, NIQE, Diversity, and Label Score. "$\downarrow$" ("$\uparrow$") indicates lower (higher) values are preferred. The best result are marked in gray.

Dataset	Model	Intra-FID $\downarrow$	NIQE $\downarrow$	Diversity $\uparrow$	Label Score $\downarrow$
RC-49 (128×128)	cGAN (150 classes)	1.250 ± 0.492	2.293 ± 0.133	2.341 ± 0.224	2.032 ± 1.653
	cGAN (concat)	1.128 ± 0.166	2.104 ± 0.104	3.431 ± 0.039	29.414 ± 7.052
	CcGAN (SVDL+ILI)	0.111 ± 0.033	1.775 ± 0.051	3.552 ± 0.047	2.643 ± 2.077
RC-49 (256×256)	cGAN (150 classes)	1.224 ± 0.336	2.147 ± 0.085	2.462 ± 0.095	2.790 ± 2.852
	cGAN (concat)	1.325 ± 0.227	3.153 ± 0.122	3.120 ± 0.043	28.776 ± 20.273
	CcGAN (SVDL+ILI)	0.495 ± 0.139	1.655 ± 0.070	2.844 ± 0.101	3.260 ± 2.641
UTKFace (128×128)	cGAN (60 classes)	1.195 ± 0.356	1.381 ± 0.208	0.788 ± 0.425	6.150 ± 5.268
	cGAN (concat)	0.408 ± 0.144	1.377 ± 0.079	1.332 ± 0.026	18.064 ± 12.550
	CcGAN (SVDL+ILI)	0.367 ± 0.123	1.113 ± 0.033	1.199 ± 0.232	7.747 ± 6.580
UTKFace (192×192)	cGAN (60 classes)	0.908 ± 0.327	1.755 ± 0.215	1.047 ± 0.381	6.639 ± 5.686
	cGAN (concat)	0.591 ± 0.214	2.352 ± 0.126	1.358 ± 0.019	17.116 ± 11.652
	CcGAN (SVDL+ILI)	0.499 ± 0.186	1.661 ± 0.047	1.207 ± 0.260	7.885 ± 6.272
Steering Angle (128×128)	cGAN (210 classes)	4.963 ± 0.916	2.520 ± 0.282	0.564 ± 0.401	31.756 ± 23.005
	cGAN (concat)	2.140 ± 0.821	2.542 ± 0.006	1.292 ± 0.014	42.757 ± 27.341
	CcGAN (SVDL+ILI)	1.689 ± 0.443	2.411 ± 0.100	1.088 ± 0.243	18.438 ± 16.072

6.1 High-resolution RC-49

In this experiment, we test three candidate methods on RC-49 with two resolutions, i.e., 128×128 and 256×256. Most training setups are consistent with Section 5.1 and please see Supp. S.17.0.2 for details. The quantitative and visual results are shown in Table 4 and Fig. 14. We can see CcGAN can generate high-quality images and the example fake images in Fig. 14 are indistinguishable from real images. However, cGAN (150 classes) and cGAN (concat) fail again. Fake images generated from cGAN (150 classes) are visually unrealistic and less diverse. Conversely, cGAN (concat) can generate images with fair visual quality and high diversity, but it cannot control the image generation via conditioning angles.

6.2 High-resolution UTKFace

In this experiment, we test three candidate methods on UTKFace with two resolutions, i.e., 128×128 and 192×192. Most training setups are consistent with Section 5.2 except that we let $\nu = 900$ when implementing CcGAN (SVDL+ILI). Please

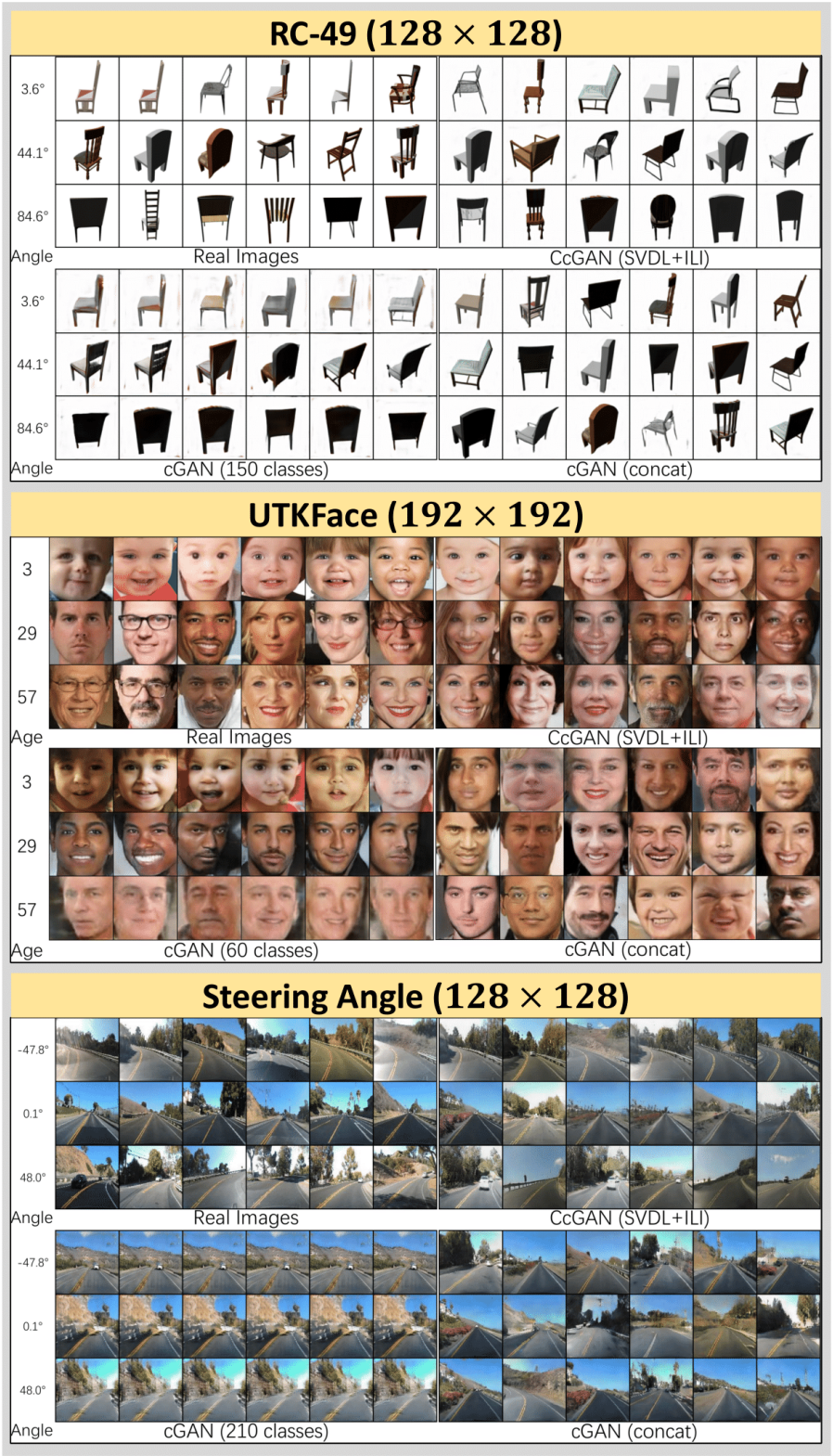

Fig. 14: **Some example high-resolution images for the RC-49, UTKFace, and Steering Angle experiments, respectively.** In the RC-49 experiment, the fake images from CcGAN are almost indistinguishable from real images. Conversely, cGAN (150 classes) has bad visual quality and low diversity. cGAN (concat) has fair visual quality and poor label consistency. In the UTKFace experiment, CcGAN can generate visually realistic, diverse, and label consistent images. Fake images from cGAN (60 classes) are visually poor and lack diversity (e.g., the last row only has white male). cGAN (concat) fails to condition the image generation on age (e.g., the first row has many adults). In the Steering Angle experiment, CcGAN substantially outperforms both cGANs. Notably, cGAN (210 classes) has the mode collapse problem [25], [45], [46] on this dataset.

see Supp. S.17.0.3 for details. The quantitative and visual results are shown in Table 4 and Fig. 14. We can see CcGAN substantially outperforms two cGANs. Fake images generated from cGAN (150 classes) are visually unrealistic and less diverse. cGAN (concat) cannot condition image generation on age.

6.3 High-resolution Steering Angle

Although the low-resolution Steering Angle experiment is already challenging enough due to high imbalance, we further increase the image resolution to 128×128, making the generative modeling more difficult. Most training setups are consistent with Section 5.4 and please see Supp. S.17.0.4 for details. The quantitative and visual results are shown in Table 4 and Fig. 14. Both visual and quantitative results of cGAN (210 classes) imply severe mode collapse problem [25], [45], [46]. Similar to previous experiments, cGAN (concat) has a very high Diversity score, but its label consistency is terrible. On the contrary, the proposed CcGAN performs well in all three evaluation perspectives.

7 Conclusion

We propose CcGAN in this paper for generative image modeling conditional on regression labels. In CcGAN, two novel empirical discriminator losses (HVDL and SVDL), a novel empirical generator loss and two novel label input mechanisms (NLI and ILI) are proposed to overcome the two problems of existing cGANs. The error bounds of a discriminator trained under HVDL and SVDL are studied in this work. Two new benchmark datasets, RC-49 and Cell-200, are created for the continuous scenario. A new evaluation metric, termed SFID, is also proposed to replace Intra-FID when there are insufficient real images. Finally we demonstrate the superiority of the proposed CcGAN to representative conventional cGANs on RC-49, UTKFace, Cell-200, and Steering Angle datasets with both low and high image resolutions.

Acknowledgments

This work was supported by UBC ARC Sockeye, Compute Canada, and the Natural Sciences and Engineering Research Council of Canada (NSERC) under Grants CRDPJ 476594-14, RGPIN-2019-05019, and RGPAS2017-507965.

References

[1] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *arXiv preprint arXiv:1411.1784*, 2014.
[2] A. Odena, C. Olah, and J. Shlens, "Conditional image synthesis with auxiliary classifier GANs," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, 2017, pp. 2642–2651.
[3] T. Miyato and M. Koyama, "cGANs with projection discriminator," in *International Conference on Learning Representations*, 2018.
[4] A. Brock, J. Donahue, and K. Simonyan, "Large scale GAN training for high fidelity natural image synthesis," in *International Conference on Learning Representations*, 2019.
[5] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena, "Self-attention generative adversarial networks," in *Proceedings of the 36th International Conference on Machine Learning*, vol. 97, 2019, pp. 7354–7363.
[6] V. Vapnik, *The nature of statistical learning theory*. Springer, 2000.
[7] M. Mohri, A. Rostamizadeh, and A. Talwalkar, *Foundations of machine learning*. MIT Press, 2018.
[8] S. Shalev-Shwartz and S. Ben-David, *Understanding machine learning: from theory to algorithms*. Cambridge University Press, 2014.
[9] G. Olmschenk, "Semi-supervised regression with generative adversarial networks using minimal labeled data," Ph.D. dissertation, 2019.
[10] O. Chapelle, J. Weston, L. Bottou, and V. Vapnik, "Vicinal risk minimization," in *Advances in neural information processing systems*, 2001, pp. 416–422.
[11] M. Rezagholizadeh, M. A. Haidar, and D. Wu, "Semi-supervised regression with generative adversarial networks," Nov. 22 2018, US Patent App. 15/789,518.
[12] M. Rezagholiradeh and M. A. Haidar, "Reg-GAN: Semi-supervised learning based on generative adversarial networks for regression," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 2806–2810.
[13] G. Olmschenk, J. Chen, H. Tang, and Z. Zhu, "Dense crowd counting convolutional neural networks with minimal data using semi-supervised dual-goal generative adversarial networks," in *IEEE Conference on Computer Vision and Pattern Recognition: Learning with Imperfect Data Workshop*, 2019.
[14] A. Dosovitskiy, J. T. Springenberg, M. Tatarchenko, and T. Brox, "Learning to generate chairs, tables and cars with convolutional networks," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 4, pp. 692–705, 2016.
[15] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, "InfoGAN: Interpretable representation learning by information maximizing generative adversarial nets," in *Advances in Neural Information Processing Systems*, vol. 29, 2016.
[16] X. Yan, J. Yang, K. Sohn, and H. Lee, "Attribute2image: Conditional image generation from visual attributes," in *European Conference on Computer Vision*, 2016, pp. 776–791.
[17] B. Li, X. Qi, T. Lukasiewicz, and P. Torr, "Controllable text-to-image generation," in *Advances in Neural Information Processing Systems*, vol. 32, 2019.
[18] M. Tao, H. Tang, S. Wu, N. Sebe, X.-Y. Jing, F. Wu, and B. Bao, "DF-GAN: Deep fusion generative adversarial networks for text-to-image synthesis," *arXiv preprint arXiv:2008.05865*, 2020.
[19] S. Zhao, Z. Liu, J. Lin, J.-Y. Zhu, and S. Han, "Differentiable augmentation for data-efficient GAN training," *Advances in Neural Information Processing Systems*, vol. 33, 2020.
[20] T. Karras, M. Aittala, J. Hellsten, S. Laine, J. Lehtinen, and T. Aila, "Training generative adversarial networks with limited data," *Advances in Neural Information Processing Systems*, vol. 33, 2020.
[21] N.-T. Tran, V.-H. Tran, N.-B. Nguyen, T.-K. Nguyen, and N.-M. Cheung, "On data augmentation for GAN training," 2020.
[22] Z. Zhao, Z. Zhang, T. Chen, S. Singh, and H. Zhang, "Image augmentations for GAN training," 2020.
[23] X. Ding, Y. Wang, Z. Xu, W. J. Welch, and Z. J. Wang, "CcGAN: Continuous conditional generative adversarial networks for image generation," in *International Conference on Learning Representations*, 2021.
[24] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral normalization for generative adversarial networks," *arXiv preprint arXiv:1802.05957*, 2018.

[25] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," ser. Proceedings of Machine Learning Research, vol. 70. International Convention Centre, Sydney, Australia: PMLR, 2017, pp. 214–223.
[26] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of Wasserstein GANs," in *Advances in neural information processing systems*, 2017, pp. 5767–5777.
[27] L. Salasnich, *Quantum Physics of Light and Matter: A Modern Introduction to Photons, Atoms and Many-Body Systems*. Springer, 2014.
[28] R. A. Davis, K.-S. Lii, and D. N. Politis, "Remarks on some nonparametric estimates of a density function," in *Selected Works of Murray Rosenblatt*. Springer, 2011, pp. 95–100.
[29] E. Parzen, "On estimation of a probability density function and mode," *The annals of mathematical statistics*, vol. 33, no. 3, pp. 1065–1076, 1962.
[30] B. W. Silverman, *Density estimation for statistics and data analysis*. CRC press, 1986, vol. 26.
[31] T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media, 2009.
[32] H. De Vries, F. Strub, J. Mary, H. Larochelle, O. Pietquin, and A. C. Courville, "Modulating early visual processing by language," in *Advances in Neural Information Processing Systems*, 2017, pp. 6594–6604.
[33] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.
[34] T. DeVries, A. Romero, L. Pineda, G. W. Taylor, and M. Drozdzal, "On the evaluation of conditional GANs," *arXiv preprint arXiv:1907.08175*, 2019.
[35] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "GANs trained by a two time-scale update rule converge to a local Nash equilibrium," in *Advances in Neural Information Processing Systems*, 2017, pp. 6626–6637.
[36] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.
[37] A. Mittal, R. Soundararajan, and A. C. Bovik, "Making a "completely blind" image quality analyzer," *IEEE Signal processing letters*, vol. 20, no. 3, pp. 209–212, 2012.
[38] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training GANs," in *Proceedings of the 30th International Conference on Neural Information Processing Systems*, ser. NIPS'16, 2016, p. 2234–2242.
[39] H. Zhang, Z. Zhang, A. Odena, and H. Lee, "Consistency regularization for generative adversarial networks," in *International Conference on Learning Representations*, 2020.
[40] M. Kang, W. Shim, M. Cho, and J. Park, "Rebooting ACGAN: Auxiliary classifier GANs with stable training," in *Advances in Neural Information Processing Systems*, M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Eds., vol. 34, 2021, pp. 23 505–23 518.
[41] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
[42] Z. Zhang, Y. Song, and H. Qi, "Age progression/regression by conditional adversarial autoencoder," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 5810–5818.
[43] A. Lehmussola, P. Ruusuvuori, J. Selinummi, H. Huttunen, and O. Yli-Harja, "Computational framework for simulating fluorescence microscope images with cell populations," *IEEE transactions on medical imaging*, vol. 26, no. 7, pp. 1010–1016, 2007.
[44] S. Chen, "The Steering Angle dataset @ONLINE," https://github.com/SullyChen/driving-datasets, 2018.
[45] A. Srivastava, L. Valkov, C. Russell, M. U. Gutmann, and C. Sutton, "VEEGAN: Reducing mode collapse in GANs using implicit variational learning," in *Advances in Neural Information Processing Systems*, 2017, pp. 3308–3318.
[46] C.-C. Chang, C. Hubert Lin, C.-R. Lee, D.-C. Juan, W. Wei, and H.-T. Chen, "Escaping from collapsing modes in a constrained space," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 204–219.
[47] J. H. Lim and J. C. Ye, "Geometric GAN," *arXiv preprint arXiv:1705.02894*, 2017.
[48] L. Wasserman, "Density estimation @ONLINE," http://www.stat.cmu.edu/~larry/=sml/densityestimation.pdf.
[49] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su *et al.*, "Shapenet: An information-rich 3D model repository," *arXiv preprint arXiv:1512.03012*, 2015.
[50] Z. Akata, F. Perronnin, Z. Harchaoui, and C. Schmid, "Label-embedding for image classification," *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 7, pp. 1425–1438, 2015.
[51] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2015.
[52] A. Lehmussola, P. Ruusuvuori, J. Selinummi, H. Huttunen, and O. Yli-Harja, "Computational framework for simulating fluorescence microscope images with cell populations," *IEEE transactions on medical imaging*, vol. 26, no. 7, pp. 1010–1016, 2007.
[53] V. Lempitsky and A. Zisserman, "Learning to count objects in images," in *Advances in neural information processing systems*, 2010, pp. 1324–1332.
[54] S. Chen, "How a high school junior made a self-driving car? @ONLINE," https://towardsdatascience.com/how-a-high-school-junior-made-a-self-driving-car-705fa9b6e860, 2018.
[55] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*, 2009, pp. 248–255.

SUPPLEMENTARY MATERIAL

S.8 GITHUB REPOSITORY

Please find the codes for this paper at

https://github.com/UBCDingXin/improved_CcGAN

S.9 ALGORITHMS FOR CCGAN TRAINING

Algorithm 2: An algorithm for CcGAN training with the proposed HVDL.

Data: N^r real image-label pairs $\Omega^r = \{(\boldsymbol{x}_1^r, y_1^r), \ldots, (\boldsymbol{x}_{N^r}^r, y_{N^r}^r)\}$, N_{uy}^r ordered distinct labels $\Upsilon = \{y_{[1]}^r, \ldots, y_{[N_{\text{uy}}^r]}^r\}$ in the dataset, preset σ and κ, number of iterations K, the discriminator batch size m^d, and the generator batch size m^g.
Result: Trained generator G.

for $k = 1$ **to** K **do**
- **Train D**;
- Draw m^d labels Y^d with replacement from Υ;
- Create a set of target labels $Y^{d,\epsilon} = \{y_i + \epsilon | y_i \in Y^d, \epsilon \in \mathcal{N}(0, \sigma^2), i = 1, \ldots, m^d\}$ (D is conditional on these labels) ;
- Initialize $\Omega_d^r = \phi, \Omega_d^f = \phi$;
- **for** $i = 1$ **to** m^d **do**
  - Randomly choose an image-label pair $(\boldsymbol{x}, y) \in \Omega^r$ satisfying $|y - y_i - \epsilon| \leq \kappa$ where $y_i + \epsilon \in Y^{d,\epsilon}$ and let $\Omega_d^r = \Omega_d^r \cup (\boldsymbol{x}, y_i + \epsilon)$. ;
  - Randomly draw a label y' from $U(y_i + \epsilon - \kappa, y_i + \epsilon + \kappa)$ and generate a fake image $\boldsymbol{x}'$ by evaluating $G(\boldsymbol{z}, y')$, where $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$. Let $\Omega_d^f = \Omega_d^f \cup (\boldsymbol{x}', y_i + \epsilon)$. ;
- **end**
- Update D with samples in set Ω_d^r and Ω_d^f via gradient-based optimizers based on Eq. (11);
- **Train G**;
- Draw m^g labels Y^g with replacement from Υ;
- Create another set of target labels $Y^{g,\epsilon} = \{y_i + \epsilon | y_i \in Y^g, \epsilon \in \mathcal{N}(0, \sigma^2), i = 1, \ldots, m^g\}$ (G is conditional on these labels) ;
- Generate m^g fake images conditional on $Y^{g,\epsilon}$ and put these image-label pairs in Ω_g^f ;
- Update G with samples in Ω_g^f via gradient-based optimizers based on Eq.(13) ;

end

Algorithm 3: An algorithm for CcGAN training with the proposed SVDL.

Data: N^r real image-label pairs $\Omega^r = \{(\boldsymbol{x}_1^r, y_1^r), \ldots, (\boldsymbol{x}_{N^r}^r, y_{N^r}^r)\}$, N_{uy}^r ordered distinct labels $\Upsilon = \{y_{[1]}^r, \ldots, y_{[N_{\text{uy}}^r]}^r\}$ in the dataset, preset σ and ν, number of iterations K, the discriminator batch size m^d, and the generator batch size m^g.
Result: Trained generator G.

for $k = 1$ **to** K **do**
- **Train D**;
- Draw m^d labels Y^d with replacement from Υ;
- Create a set of target labels $Y^{d,\epsilon} = \{y_i + \epsilon | y_i \in Y^d, \epsilon \in \mathcal{N}(0, \sigma^2), i = 1, \ldots, m^d\}$ (D is conditional on these labels) ;
- Initialize $\Omega_d^r = \phi, \Omega_d^f = \phi$;
- **for** $i = 1$ **to** m^d **do**
  - Randomly choose an image-label pair $(\boldsymbol{x}, y) \in \Omega^r$ satisfying $e^{-\nu(y - y_i - \epsilon)^2} > 10^{-3}$ where $y_i + \epsilon \in Y^{d,\epsilon}$ and let $\Omega_d^r = \Omega_d^r \cup (\boldsymbol{x}, y_i + \epsilon)$. This step is used to exclude real images with too small weights. ;
  - Compute $w_i^r(y, y_i + \epsilon) = e^{-\nu(y_i + \epsilon - y)^2}$;
  - Randomly draw a label y' from $U(y_i + \epsilon - \sqrt{-\frac{\log 10^{-3}}{\nu}}, y_i + \epsilon + \sqrt{-\frac{\log 10^{-3}}{\nu}})$ and generate a fake image $\boldsymbol{x}'$ by evaluating $G(\boldsymbol{z}, y')$, where $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$. Let $\Omega_d^f = \Omega_d^f \cup (\boldsymbol{x}', y_i + \epsilon)$. ;
  - Compute $w_i^g(y', y_i + \epsilon) = e^{-\nu(y_i + \epsilon - y')^2}$;
- **end**
- Update D with samples in set Ω_d^r and Ω_d^f via gradient-based optimizers based on Eq. (12);
- **Train G**;
- Draw m^g labels Y^g with replacement from Υ;
- Create another set of target labels $Y^{g,\epsilon} = \{y_i + \epsilon | y_i \in Y^g, \epsilon \in \mathcal{N}(0, \sigma^2), i = 1, \ldots, m^g\}$ (G is conditional on these labels) ;
- Generate m^g fake images conditional on $Y^{g,\epsilon}$ and put these image-label pairs in Ω_g^f ;
- Update G with samples in Ω_g^f via gradient-based optimizers based on Eq.(13) ;

end

Remark S.6. If should be noted that, for computational efficiency, the normalizing constants $N_{y_j^r + \epsilon^r, \kappa}^r$, $N_{y_j^g + \epsilon^g, \kappa}^g$, $\sum_{i=1}^{N^r} w^r(y_i^r, y_j^r + \epsilon^r)$, and $\sum_{i=1}^{N^g} w^g(y_i^g, y_j^g + \epsilon^g)$ in Eq. (11) and (12) are excluded from the training and only used for theoretical analysis.

S.10 THEORETICAL ANALYSIS FOR HVDL AND SVDL

In this section, we provide a self-contained theoretical analysis of HVDL and SVDL. To make the derivation clearer, we use some notations and definitions slightly different from those in the main content of the paper. Some necessary assumptions, lemmas, and theorems are also introduced or derived in Sections S.10.2 and S.10.3. The main theorems on the error bounds of D are derived in Section S.10.4.

S.10.1 Some Necessary Definitions and Notations

This section summarizes some necessary definitions and notations used in the derivation. Please note that **all these definitions and notations are valid in Supp. S.10 only**.

- Unlike other contents of this paper, we use different symbols to denote random variables/vectors and the fixed values that random variables/vectors may take. Specifically, a random image and a random label are represented respectively by a bold capital $\boldsymbol{X}$ and a capital Y. A sequence of N random image-label pairs are represented by $(\boldsymbol{X}_1, Y_1), \ldots, (\boldsymbol{X}_N, Y_N)$. Please note that some subscripts or superscripts may apply to $\boldsymbol{X}$ and Y to provide some extra information. An observed (fixed) image and an observed (fixed) label are denoted respectively by a bold lowercase $\boldsymbol{x}$ and a lowercase y. **Moreover, without loss of generality, we assume** $Y, y \in [0,1]$**, i.e.,** $\mathcal{Y} = [0,1]$.
- Let $p(\boldsymbol{x}|Y=y)$ denote the conditional probability density function (PDF) of $\boldsymbol{X}$ given the occurrence of the value y of Y. p may have superscripts or subscripts to provide some extra information.
- Let $p(y'|Y=y)$ denote the conditional PDF of Y' given the occurrence of the value y of Y. p may have superscripts or subscripts to provide some extra information.
- Let $\mathcal{D}$ stand for the *Hypothesis Space* of D. $\mathcal{D}$ is a set of functions that can be represented by D (a neural network with determined architecture but undetermined weights).
- Let $f(\boldsymbol{x}, y) = -\log D(\boldsymbol{x}, y)$ and $\mathcal{F} = -\log \mathcal{D}$.
- Let $\hat{p}_r^{\text{KDE}}(y)$ and $\hat{p}_g^{\text{KDE}}(y)$ stand for the KDEs of $p_r(y)$ and $p_g(y)$ respectively.
- For HVDL, denote respectively by

$$p_r^{y,\kappa}(\boldsymbol{x}) \triangleq \int p_r(\boldsymbol{x}|Y=y') \frac{\mathbb{1}_{\{|y'-y|\leq\kappa\}} p_r(y')}{\int \mathbb{1}_{\{|y'-y|\leq\kappa\}} p_r(y') dy'} dy'$$

and

$$p_g^{y,\kappa}(\boldsymbol{x}) \triangleq \int p_g(\boldsymbol{x}|Y=y') \frac{\mathbb{1}_{\{|y'-y|\leq\kappa\}} p_g(y')}{\int \mathbb{1}_{\{|y'-y|\leq\kappa\}} p_g(y') dy'} dy'$$

the PDFs of the marginal distributions for real and fake images with labels in $[y-\kappa, y+\kappa]$.
- For SVDL, given y and the weight functions, if the number of real and fake images are infinite, the real and fake empirical densities converges to

$$p_r^{y,w^r}(\boldsymbol{x}) \triangleq \int p_r(\boldsymbol{x}|Y=y') \frac{w^r(y',y) p_r(y')}{W^r(y)} dy'$$

and

$$p_g^{y,w^g}(\boldsymbol{x}) \triangleq \int p_g(\boldsymbol{x}|Y=y') \frac{w^g(y',y) p_g(y')}{W^g(y)} dy'$$

respectively, where

$$W^r(y) \triangleq \int w^r(y',y) p_r(y') dy',$$

$$W^g(y) \triangleq \int w^g(y',y) p_g(y') dy',$$

and w^r and w^g are the weight functions defined as follows

$$w^r(y',y) = e^{-\nu(y'-y)^2} \quad \text{and} \quad w^g(y',y) = e^{-\nu(y'-y)^2}.$$

We also let

$$p_w^r(y'|Y=y) \triangleq \frac{w^r(y',y) p^r(y')}{W^r(y)}$$

and

$$p_w^g(y'|Y=y) \triangleq \frac{w^g(y',y) p^g(y')}{W^g(y)}.$$

- The Hölder Class defined below is a set of functions with bouned second derivatives, which controls the variation of the function when parameter changes.

  Definition S.2. (Hölder Class) Define the Hölder class of functions as:

$$\Sigma(L) \triangleq \left\{ p : \forall t_1, t_2 \in \mathcal{Y}, \exists L > 0, s.t. \frac{|p'(t_1) - p'(t_2)|}{|t_1 - t_2|} \leq L \right\}. \tag{S.20}$$

- With some new notations above, we restate the theoretical discriminator losses $\mathcal{L}(D)$ as follows:

$$\begin{aligned} \mathcal{L}(D) = &- \mathbb{E}_{Y\sim p_r(y)} \left[\mathbb{E}_{\boldsymbol{X}\sim p_r(\boldsymbol{x}|Y)} \left[\log \left(D(\boldsymbol{X}, Y) \right) \right] \right] \\ &- \mathbb{E}_{Y\sim p_g(y)} \left[\mathbb{E}_{\boldsymbol{X}\sim p_g(\boldsymbol{x}|Y)} \left[\log \left(1 - D(\boldsymbol{X}, Y) \right) \right] \right], \end{aligned} \tag{S.21}$$

- Recall that, given a G, the optimal discriminator which minimizes $\mathcal{L}(D)$ is in the form of

$$D^*(\boldsymbol{x}, y) = \frac{p_r(\boldsymbol{x}, y)}{p_r(\boldsymbol{x}, y) + p_g(\boldsymbol{x}, y)}.$$

However, D^* may not be covered by the hypothesis space $\mathcal{D}$. Define $\widetilde{D}$, $\widehat{D}^{\text{HVDL}}$, and $\widehat{D}^{\text{SVDL}}$ as follows

$$\begin{aligned}
&\widetilde{D} \triangleq \arg\min_{D\in\mathcal{D}} \mathcal{L}(D),\\
&\widehat{D}^{\text{HVDL}} \triangleq \arg\min_{D\in\mathcal{D}} \widehat{\mathcal{L}}^{\text{HVDL}}(D),\\
&\widehat{D}^{\text{SVDL}} \triangleq \arg\min_{D\in\mathcal{D}} \widehat{\mathcal{L}}^{\text{SVDL}}(D).
\end{aligned}$$

Note that $\mathcal{L}(\widetilde{D}) - \mathcal{L}(D^*)$ should be a non-negative constant. In CcGAN, we minimize $\widehat{\mathcal{L}}^{\text{HVDL}}(D)$ or $\widehat{\mathcal{L}}^{\text{SVDL}}(D)$ with respect to $D \in \mathcal{D}$, so we are interested in the distance of $\widehat{D}^{\text{HVDL}}$ and $\widehat{D}^{\text{SVDL}}$ from D^*, i.e., $\mathcal{L}(\widehat{D}^{\text{HVDL}}) - \mathcal{L}(D^*)$ and $\mathcal{L}(\widehat{D}^{\text{SVDL}}) - \mathcal{L}(D^*)$.

S.10.2 Some Necessary Assumptions

In this theoretical analysis, we work with the following assumptions:
(A1) All D's in $\mathcal{D}$ are measurable and uniformly bounded. Let

$$U \triangleq \max\{\sup_{D\in\mathcal{D}} [-\log D], \sup_{D\in\mathcal{D}} [-\log(1-D)]\}$$

and $U < \infty$;
(A2) For $\forall \boldsymbol{x} \in \mathcal{X}$ and $y, y' \in \mathcal{Y}$, $\exists g^r(\boldsymbol{x}) > 0$ and $M^r > 0$, s.t. $|p_r(\boldsymbol{x}|Y = y') - p_r(\boldsymbol{x}|Y = y)| \leq g^r(\boldsymbol{x})|y' - y|$ with $\int g^r(\boldsymbol{x})d\boldsymbol{x} = M^r$;
(A3) For $\forall \boldsymbol{x} \in \mathcal{X}$ and $y, y' \in \mathcal{Y}$, $\exists g^g(\boldsymbol{x}) > 0$ and $M^g > 0$, s.t. $|p_g(\boldsymbol{x}|Y = y') - p_g(\boldsymbol{x}|Y = y)| \leq g^g(\boldsymbol{x})|y' - y|$ with $\int g^g(\boldsymbol{x})d\boldsymbol{x} = M^g$;
(A4) $p_r(y) \in \Sigma(L^r)$ and $p_g(y) \in \Sigma(L^g)$.

S.10.3 Some Necessary Lemmas and Theorems

In this section, we first introduce the Hoeffding's inequality that are widely used later to derive some lemmas.

Theorem S.3 (Hoeffding's inequality [8]). *Let $Z_1, \ldots, Z_m$ be a sequence of i.i.d. random variables and let $\bar{Z} = \frac{1}{m}\sum_{i=1}^m Z_i$. Assume that $\mathbb{E}[\bar{Z}] = \mu$ and $Pr(a \leq Z_i \leq b) = 1$ for every i. Then, for any $\epsilon > 0$*

$$Pr\left[\left|\frac{1}{m}\sum_{i=1}^m Z_i - \mu\right| > \epsilon\right] \leq 2\exp\left(-\frac{2m\epsilon^2}{(b-a)^2}\right).$$

Proof. Please see [8, Lemma B.6] for the proof. □

Remark S.7. Let $\delta = 2\exp\left(-\frac{2m\epsilon^2}{(b-a)^2}\right)$, then $\epsilon = \sqrt{\frac{1}{2m}\log\left(\frac{2}{\delta}\right)}$. Thus, we can get another form of the Hoeffding's inequality. For $\forall \delta \in (0,1)$, with probability at least $1-\delta$, we have

$$\left|\frac{1}{m}\sum_{i=1}^m Z_i - \mu\right| \leq \sqrt{\frac{1}{2m}\log\left(\frac{2}{\delta}\right)}.$$

Lemma S.3 (Lemma for HVDL). *Suppose that (A1)-(A2) and (A4) hold and let $(\boldsymbol{X}_1, Y_1), \ldots, (\boldsymbol{X}_N, Y_N)$ be a sequence of i.i.d. random image-label pairs, then $\forall \delta \in (0,1)$, with probability at least $1-\delta$,*

$$\begin{aligned}
&\sup_{D\in\mathcal{D}} \left|\frac{1}{N_{y,\kappa}}\sum_{i=1}^N \mathbb{1}_{\{|y-Y_i|\leq\kappa\}} [-\log D(\boldsymbol{X}_i, y)] - \mathbb{E}_{\boldsymbol{X}\sim p(\boldsymbol{x}|Y=y)} [-\log D(\boldsymbol{X}, y)]\right|\\
&\leq U\sqrt{\frac{1}{2N_{y,\kappa}}\log\left(\frac{2}{\delta}\right)} + \kappa U M,
\end{aligned} \tag{S.22}$$

for a fixed y. If image-label pairs are real, then $N = N^r$, $N_{y,\kappa} = N^r_{y,\kappa}$, $p = p_r$, and $M = M^r$. Similarly, we have $N = N^g$, $N_{y,\kappa} = N^g_{y,\kappa}$, $p = p_g$, and $M = M^g$ for fake image-label pairs.

Proof. Triangle inequality yields

$$\begin{aligned}
&\sup_{D\in\mathcal{D}} \left|\frac{1}{N_{y,\kappa}}\sum_{i=1}^N \mathbb{1}_{\{|y-Y_i|\leq\kappa\}} [-\log D(\boldsymbol{X}_i, y)] - \mathbb{E}_{\boldsymbol{X}\sim p_r(\boldsymbol{x}|Y=y)} [-\log D(\boldsymbol{X}, y)]\right|\\
&\leq \sup_{D\in\mathcal{D}} \left|\frac{1}{N_{y,\kappa}}\sum_{i=1}^N \mathbb{1}_{\{|y-Y_i|\leq\kappa\}} [-\log D(\boldsymbol{X}_i, y)] - \mathbb{E}_{\boldsymbol{X}\sim p^{y,\kappa}(\boldsymbol{x})} [-\log D(\boldsymbol{X}, y)]\right|
\end{aligned}$$

$$+\sup_{D\in\mathcal{D}}\left|\mathbb{E}_{\boldsymbol{X}\sim p^{y,\kappa}(\boldsymbol{x})}\left[-\log D(\boldsymbol{X},y)\right]-\mathbb{E}_{\boldsymbol{X}\sim p(\boldsymbol{x}|Y=y)}\left[-\log D(\boldsymbol{X},y)\right]\right|$$

We then bound the two terms of the RHS separately as follows:

1) Real images with labels in $[y-\kappa,y+\kappa]$ can be seen as independent samples from $p^{y,\kappa}(\boldsymbol{x})$. Then the first term can be bounded by applying Hoeffding's inequality as follows: $\forall\delta\in(0,1)$, with at least probability $1-\delta$,

$$\begin{aligned}&\sup_{D\in\mathcal{D}}\left|\frac{1}{N_{y,\kappa}}\sum_{i=1}^{N}\mathbb{1}_{\{|y-Y_i|\leq\kappa\}}\left[U\frac{-\log D(\boldsymbol{X}_i,y)}{U}\right]-\mathbb{E}_{\boldsymbol{X}\sim p^{y,\kappa}(\boldsymbol{x})}\left[U\frac{-\log D(\boldsymbol{X},y)}{U}\right]\right|\\&\leq U\sqrt{\frac{1}{2N_{y,\kappa}}\log\left(\frac{2}{\delta}\right)}.\end{aligned}\tag{S.23}$$

2) For the second term, we have

$$\begin{aligned}&\sup_{D\in\mathcal{D}}\left|\mathbb{E}_{\boldsymbol{X}\sim p^{y,\kappa}(\boldsymbol{x})}\left[-\log D(\boldsymbol{X},y)\right]-\mathbb{E}_{\boldsymbol{X}\sim p(\boldsymbol{x}|Y=y)}\left[-\log D(\boldsymbol{X},y)\right]\right|\\=&\sup_{D\in\mathcal{D}}\left|\int\left[-\log D(\boldsymbol{x},y)\right]\cdot\left[p^{y,\kappa}(\boldsymbol{x})-p(\boldsymbol{x}|Y=y)\right]d\boldsymbol{x}\right|\\\leq&\sup_{D\in\mathcal{D}}\int\left|-\log D(\boldsymbol{x},y)\right|\cdot\left|p^{y,\kappa}(\boldsymbol{x})-p(\boldsymbol{x}|Y=y)\right|d\boldsymbol{x}\\\leq&U\int\left|p^{y,\kappa}(\boldsymbol{x})-p(\boldsymbol{x}|Y=y)\right|d\boldsymbol{x}.\end{aligned}\tag{S.24}$$

Then, we focus on $|p^{y,\kappa}(\boldsymbol{x})-p(\boldsymbol{x}|Y=y)|$. By the definition of $p^{y,\kappa}(\boldsymbol{x})$ and defining $p_\kappa(y')=\frac{\mathbb{1}_{\{|y'-y|\leq\kappa\}}p(y')}{\int\mathbb{1}_{\{|y'-y|\leq\kappa\}}p(y')dy'}$, we have

$$\begin{aligned}&\left|p^{y,\kappa}(\boldsymbol{x})-p(\boldsymbol{x}|Y=y)\right|\\=&\left|\int p(\boldsymbol{x}|Y=y')p_\kappa(y')dy'-p(\boldsymbol{x}|Y=y)\right|\\\leq&\int\left|p(\boldsymbol{x}|Y=y')-p(\boldsymbol{x}|Y=y)\right|p_\kappa(y')dy'\\&\text{(by (A2), and let } g=g^r \text{ for real images and } g=g^g \text{ for fake images)}\\\leq&\int g(\boldsymbol{x})|y'-y|p_\kappa(y')dy'\\\leq&\kappa g(\boldsymbol{x}).\end{aligned}$$

Thus, Eq. (S.24) is upper bounded as follows,

$$\begin{aligned}&\sup_{D\in\mathcal{D}}\left|\mathbb{E}_{\boldsymbol{X}\sim p^{y,\kappa}(\boldsymbol{x})}\left[-\log D(\boldsymbol{X},y)\right]-\mathbb{E}_{\boldsymbol{X}\sim p(\boldsymbol{x}|Y=y)}\left[-\log D(\boldsymbol{X},y)\right]\right|\\\leq&U\int\kappa g(\boldsymbol{x})d\boldsymbol{x}\\&\text{(by (A2))}\\=&\kappa UM.\end{aligned}\tag{S.25}$$

By combining Eq. (S.23) and (S.25), we can get Eq. (S.22), which finishes the proof. □

Lemma S.4 (Lemma for SVDL)**.** *Suppose that (A1), (A2) and (A4) hold and let $(\boldsymbol{X}_1,Y_1),\ldots,(\boldsymbol{X}_N,Y_N)$ be a sequence of i.i.d. random image-label pairs, then $\forall\delta\in(0,1)$, with probability at least $1-\delta$,*

$$\begin{aligned}&\sup_{D\in\mathcal{D}}\left|\frac{\frac{1}{N}\sum_{i=1}^{N}w(Y_i,y)\left[-\log D(\boldsymbol{X}_i,y)\right]}{\frac{1}{N}\sum_{i=1}^{N}w(Y_i,y)}-\mathbb{E}_{\boldsymbol{X}\sim p(\boldsymbol{x}|Y=y)}\left[-\log D(\boldsymbol{X},y)\right]\right|\\&\leq\frac{2U}{W(y)}\sqrt{\frac{1}{2N}\log\left(\frac{4}{\delta}\right)}+UM\mathbb{E}_{Y'\sim p_w(Y'|Y=y)}\left[|Y'-y|\right],\end{aligned}\tag{S.26}$$

for a fixed y. If image-label pairs are real, then $N=N^r$, $N_{y,\kappa}=N^r_{y,\kappa}$, $p=p_r$, $p_w=p^r_w$, $w=w^r$, $W=W^r$, and $M=M^r$. Similarly, we have $N=N^g$, $N_{y,\kappa}=N^g_{y,\kappa}$, $p=p_g$, $p_w=p^g_w$, $w=w^g$, $W=W^g$, and $M=M^g$ for fake image-label pairs.

Proof. Triangle inequality yields

$$\sup_{D\in\mathcal{D}}\left|\frac{\frac{1}{N}\sum_{i=1}^{N}w(Y_i,y)\left[-\log D(\boldsymbol{X}_i,y)\right]}{\frac{1}{N}\sum_{i=1}^{N}w(Y_i,y)}-\mathbb{E}_{\boldsymbol{X}\sim p_r(\boldsymbol{x}|Y=y)}\left[-\log D(\boldsymbol{X},y)\right]\right|$$

(Recall $f(\boldsymbol{x},y)=-\log D(\boldsymbol{x},y)$ and $\mathcal{F}=-\log\mathcal{D}$.)

$$
\begin{aligned}
&= \sup_{f\in\mathcal{F}} \left| \frac{\frac{1}{N}\sum_{i=1}^{N} w(Y_i,y) f(\boldsymbol{X}_i,y)}{\frac{1}{N}\sum_{i=1}^{N} w(Y_i,y)} - \mathbb{E}_{\boldsymbol{X}\sim p_r(\boldsymbol{x}|Y=y)}\left[f(\boldsymbol{X},y)\right] \right| \\
&\leq \sup_{f\in\mathcal{F}} \left| \frac{\frac{1}{N}\sum_{i=1}^{N} w(Y_i,y) f(\boldsymbol{X}_i,y)}{\frac{1}{N}\sum_{i=1}^{N} w(Y_i,y)} - \mathbb{E}_{\boldsymbol{X}\sim p^{y,w}(\boldsymbol{x})}\left[f(\boldsymbol{X},y)\right] \right| \\
&\quad + \sup_{f\in\mathcal{F}} \left| \mathbb{E}_{\boldsymbol{X}\sim p^{y,w}(\boldsymbol{x})}\left[f(\boldsymbol{X},y)\right] - \mathbb{E}_{\boldsymbol{X}\sim p_r(\boldsymbol{x}|Y=y)}\left[f(\boldsymbol{X},y)\right] \right|. \qquad \text{(S.27)} \\
&\quad (p^{y,w} = p_r^{y,w^r} \text{ for real images and } p^{y,w} = p_g^{y,w^g} \text{ for fake images})
\end{aligned}
$$

We then derive bounds for both terms on the RHS of the inequality.

1) For the first term, we can further split it into two parts,

$$
\begin{aligned}
&\left| \frac{\frac{1}{N}\sum_{i=1}^{N} w(Y_i,y) f(\boldsymbol{X}_i,y)}{\frac{1}{N}\sum_{i=1}^{N} w(Y_i,y)} - \mathbb{E}_{\boldsymbol{X}\sim p^{y,w}(\boldsymbol{x})}\left[f(\boldsymbol{X},y)\right] \right| \\
\leq &\left| \frac{\frac{1}{N}\sum_{i=1}^{N} w(Y_i,y) f(\boldsymbol{X}_i,y)}{\frac{1}{N}\sum_{i=1}^{N} w(Y_i,y)} - \frac{\frac{1}{N}\sum_{i=1}^{N} w(Y_i,y) f(\boldsymbol{X}_i,y)}{W(y)} \right| \\
&+ \left| \frac{\frac{1}{N}\sum_{i=1}^{N} w(Y_i,y) f(\boldsymbol{X}_i,y)}{W(y)} - \mathbb{E}_{\boldsymbol{X}\sim p^{y,w}(\boldsymbol{x})}\left[f(\boldsymbol{X},y)\right] \right| \qquad \text{(S.28)}
\end{aligned}
$$

Focusing on the first part of RHS of Eq.(S.28). By (A1),

$$
\begin{aligned}
&\left| \frac{\frac{1}{N}\sum_{i=1}^{N} w(Y_i,y) f(\boldsymbol{X}_i,y)}{\frac{1}{N}\sum_{i=1}^{N} w(Y_i,y)} - \frac{\frac{1}{N}\sum_{i=1}^{N} w(Y_i,y) f(\boldsymbol{X}_i,y)}{W(y)} \right| \\
\leq &U \frac{\left| \frac{1}{N}\sum_{i=1}^{N} w(Y_i,y) - W(y) \right|}{W(y)}
\end{aligned}
$$

Note that $\forall y, y', w(y',y) = e^{-\nu|y-y'|^2} \leq 1$ (since $\nu > 0$) and hence given y, $w(Y',y)$ is a random variable bounded by 1. Moreover, given y, $W(y)$ is the expectation of $w(Y',y)$. Then, apply Hoeffding's inequality to the numerator of above, yielding that with probability at least $1-\delta'$,

$$
\left| \frac{1}{N}\sum_{i=1}^{N} w(Y_i,y) - W(y) \right| \leq \sqrt{\frac{1}{2N}\log\left(\frac{2}{\delta'}\right)}.
$$

Thus, by the boundedness of f, with probability at least $1-\delta'$,

$$
\begin{aligned}
&\left| \frac{\frac{1}{N}\sum_{i=1}^{N} w(Y_i,y) f(\boldsymbol{X}_i,y)}{\frac{1}{N}\sum_{i=1}^{N} w(Y_i,y)} - \frac{\frac{1}{N}\sum_{i=1}^{N} w(Y_i,y) f(\boldsymbol{X}_i,y)}{W(y)} \right| \\
\leq &\frac{U}{W(y)} \sqrt{\frac{1}{2N}\log\left(\frac{2}{\delta'}\right)}. \qquad \text{(S.29)}
\end{aligned}
$$

Then, consider the second part of RHS of Eq.(S.28). Recall that $p^{y,w}(\boldsymbol{x}) \triangleq \int p(\boldsymbol{x}|Y=y') \frac{w(y',y)p(y')}{W(y)} dy'$. Thus,

$$
\begin{aligned}
&\left| \frac{\frac{1}{N}\sum_{i=1}^{N} w(Y_i,y) f(\boldsymbol{X}_i,y)}{W(y)} - \mathbb{E}_{\boldsymbol{X}\sim p^{y,w}(\boldsymbol{x})}\left[f(\boldsymbol{X},y)\right] \right| \\
= &\frac{1}{W(y)} \left| \frac{1}{N}\sum_{i=1}^{N} w(Y_i,y) f(\boldsymbol{X}_i,y) - \mathbb{E}_{(\boldsymbol{X},Y')\sim p(\boldsymbol{x},y')}\left[w^r(Y',y) f(\boldsymbol{X}_i,y)\right] \right|,
\end{aligned}
$$

where $p(\boldsymbol{x},y') = p(\boldsymbol{x}|Y=y')p(y')$ denotes PDF of the joint distribution of real image and its label. Again, since $w(Y',y)f(\boldsymbol{X}_i,y)$ is uniformly bounded by U under (A1), we can apply Hoeffding's inequality. This implies that with probability at least $1-\delta'$, the above can be upper bounded by

$$
\frac{U}{W(y)} \sqrt{\frac{1}{2N}\log\left(\frac{2}{\delta'}\right)}. \qquad \text{(S.30)}
$$

Combining Eq. (S.29) and (S.30) and by setting $\delta' = \frac{\delta}{2}$, we have with probability at least $1-\delta$,

$$
\left| \frac{\frac{1}{N}\sum_{i=1}^{N} w(Y_i,y) f(\boldsymbol{X}_i,y)}{\frac{1}{N}\sum_{i=1}^{N} w(Y_i,y)} - \mathbb{E}_{\boldsymbol{X}\sim p^{y,w}(\boldsymbol{x})}\left[f(\boldsymbol{X},y)\right] \right| \leq \frac{2U}{W(y)} \sqrt{\frac{1}{2N}\log\left(\frac{4}{\delta}\right)}.
$$

Since this holds for $\forall f \in \mathcal{F}$, taking supremum over f, we have

$$\sup_{f\in\mathcal{F}}\left|\frac{\frac{1}{N}\sum_{i=1}^{N} w(Y_i,y)f(\boldsymbol{X}_i,y)}{\frac{1}{N}\sum_{i=1}^{N} w(Y_i,y)} - \mathbb{E}_{\boldsymbol{X}\sim p^{y,w}(\boldsymbol{x})}\left[f(\boldsymbol{X},y)\right]\right| \leq \frac{2U}{W(y)}\sqrt{\frac{1}{2N}\log\left(\frac{4}{\delta}\right)}. \tag{S.31}$$

2) For the second term on the RHS of Eq. (S.27). By (A1) that $|f| \leq U$,

$$\begin{aligned}&\sup_{f\in\mathcal{F}}\left|\mathbb{E}_{\boldsymbol{X}\sim p^{y,w}(\boldsymbol{x})}\left[f(\boldsymbol{X},y)\right] - \mathbb{E}_{\boldsymbol{X}\sim p(\boldsymbol{x}|Y=y)}\left[f(\boldsymbol{X},y)\right]\right|\\ &\text{(See Eq. (S.24))}\\ \leq& U\int |p^{y,w}(\boldsymbol{x}) - p(\boldsymbol{x}|Y=y)|d\boldsymbol{x}.\end{aligned}$$

Note that by the definition of

$$p^{y,w}(\boldsymbol{x}) \triangleq \int p(\boldsymbol{x}|Y=y')\frac{w(y',y)p(y')}{W(y)}dy'$$

and

$$p_w\left(y'|Y=y\right) \triangleq \frac{w\left(y',y\right)p^r\left(y'\right)}{W^r(y)},$$

we have

$$\begin{aligned}|p^{y,w}(\boldsymbol{x}) - p(\boldsymbol{x}|Y=y)| &= \left|\int p(\boldsymbol{x}|Y=y')p_w\left(y'|Y=y\right)dy' - p(\boldsymbol{x}|Y=y)\right|\\ &\leq \int |p(\boldsymbol{x}|Y=y') - p(\boldsymbol{x}|Y=y)|\, p_w\left(y'|Y=y\right)dy'.\end{aligned}$$

By (A.2) and $y \in [0,1]$, the above is upper bounded by

$$g(\boldsymbol{x})\mathbb{E}_{Y'\sim p_w(y'|Y=y)}\left[|y-Y'|\right].$$

Thus,

$$\begin{aligned}&\sup_{f\in\mathcal{F}}\left|\mathbb{E}_{\boldsymbol{X}\sim p^{y,w}(\boldsymbol{x})}\left[f(\boldsymbol{X},y)\right] - \mathbb{E}_{\boldsymbol{X}\sim p(\boldsymbol{x}|Y=y)}\left[f(\boldsymbol{X},y)\right]\right|\\ \leq& U\int g(\boldsymbol{x})\mathbb{E}_{Y'\sim p_w(y'|Y=y)}\left[|Y'-y|\right]d\boldsymbol{x}\\ =& UM\mathbb{E}_{Y'\sim p_w(y'|Y=y)}\left[|Y'-y|\right].\end{aligned} \tag{S.32}$$

Therefore, combining both Eq.(S.31) and (S.32), with probability at least $1-\delta$,

$$\begin{aligned}&\sup_{D\in\mathcal{D}}\left|\frac{\frac{1}{N}\sum_{i=1}^{N} w(Y_i,y)\left[-\log D(\boldsymbol{X}_i,y)\right]}{\frac{1}{N}\sum_{i=1}^{N} w(Y_i,y)} - \mathbb{E}_{\boldsymbol{X}\sim p(\boldsymbol{x}|Y=y)}\left[-\log D(\boldsymbol{X},y)\right]\right|\\ \leq&\frac{2U}{W(y)}\sqrt{\frac{1}{2N}\log\left(\frac{4}{\delta}\right)} + UM\mathbb{E}_{Y'\sim p_w(y'|Y=y)}\left[|Y'-y|\right].\end{aligned}$$

This finishes the proof. □

As introduced in Section 2, we use KDE for the density of the marginal label distribution with Gaussian kernel. The next theorem characterizes the difference between a $p_r(y), p_g(y)$ and their KDE using N i.i.d. samples.

Theorem S.4. *Let $\hat{p}_r^{KDE}(y)$ and $\hat{p}_g^{KDE}(y)$ stand for the KDE of $p_r(y)$ and $p_g(y)$ respectively. Under condition (A4), if the KDEs are based on N i.i.d. samples from p_r/p_g and a bandwidth σ, for all $\delta \in (0,1)$, with probability at least $1-\delta$,*

$$\sup_t\left|\hat{p}_r^{KDE}(y) - p_r(y)\right| \leq \sqrt{\frac{C_{1,\delta}^{KDE}\log N}{N\sigma}} + L^r\sigma, \tag{S.33}$$

$$\sup_t\left|\hat{p}_g^{KDE}(y) - p_g(y)\right| \leq \sqrt{\frac{C_{2,\delta}^{KDE}\log N}{N\sigma}} + L^g\sigma, \tag{S.34}$$

for some constants $C_{1,\delta}^{KDE}, C_{2,\delta}^{KDE}$ depending on δ.

Proof. By ([48]; P.12), for any $p(t) \in \Sigma(L)$ (the Hölder Class, see Definition S.2), with probability at least $1-\delta$,

$$\sup_t\left|\hat{p}^{\text{KDE}}(t) - p(t)\right| \leq \sqrt{\frac{C_{\delta}^{\text{KDE}}\log N}{N\sigma}} + c\sigma,$$

for some constants $C_{\delta}^{\mathrm{KDE}}$ and c, where C depends on δ and $c = L\int K(s)|s|^2 ds$. Since in this work, K is chosen as Gaussian kernel, $c = L\int K(s)|s|^2 ds = L$. □

Based on above lemmas and theorems, we derive the following two theorems, which will be used in the derivation of the error bounds of D trained with HVDL and SVDL in Section S.10.4.

Theorem S.5. *Assume that (A1)-(A4) hold, then $\forall \delta \in (0,1)$, with probability at least $1-\delta$,*

$$\begin{aligned}
&\sup_{D\in\mathcal{D}}\left|\widehat{\mathcal{L}}^{HVDL}(D) - \mathcal{L}(D)\right| \\
&\leq U\left(\sqrt{\frac{C_{1,\delta}^{KDE}\log N^r}{N^r\sigma}} + L^r\sigma^2\right) + U\left(\sqrt{\frac{C_{2,\delta}^{KDE}\log N^g}{N^g\sigma}} + L^g\sigma^2\right) + \kappa U(M^r + M^g) \\
&\quad + U\sqrt{\frac{1}{2}\log\left(\frac{8}{\delta}\right)}\left(\mathbb{E}_{Y\sim\hat{p}_r^{KDE}(y)}\left[\sqrt{\frac{1}{N_{Y,\kappa}^r}}\right] + \mathbb{E}_{Y\sim\hat{p}_g^{KDE}(y)}\left[\sqrt{\frac{1}{N_{Y,\kappa}^g}}\right]\right),
\end{aligned} \tag{S.35}$$

for some constants $C_{1,\delta}^{KDE}, C_{2,\delta}^{KDE}$ depending on δ.

Proof. Let $(\boldsymbol{X}_1^r, Y_1^r), \ldots, (\boldsymbol{X}_{N^r}^r, Y_{N^r}^r)$ and $(\boldsymbol{X}_1^g, Y_1^g), \ldots, (\boldsymbol{X}_{N^g}^g, Y_{N^g}^g)$ denote respectively real and fake i.i.d. random image-label pairs.

We first decompose $\sup_{D\in\mathcal{D}}\left|\widehat{\mathcal{L}}^{\mathrm{HVDL}}(D) - \mathcal{L}(D)\right|$ as follows

$$\begin{aligned}
&\sup_{D\in\mathcal{D}}\left|\widehat{\mathcal{L}}^{\mathrm{HVDL}}(D) - \mathcal{L}(D)\right| \\
&\leq \sup_{D\in\mathcal{D}}\left|\int\left[\int\left[-\log D(\boldsymbol{x},y)\right]p_r(\boldsymbol{x}|Y=y)d\boldsymbol{x}\right](p_r(y) - \hat{p}_r^{\mathrm{KDE}}(y))dy\right| \\
&+ \sup_{D\in\mathcal{D}}\left|\int\left[\int\left[-\log(1-D(\boldsymbol{x},y))\right]p_g(\boldsymbol{x}|Y=y)d\boldsymbol{x}\right](p_g(y) - \hat{p}_g^{\mathrm{KDE}}(y))dy\right| \\
&+ \sup_{D\in\mathcal{D}}\left|\int\left[\frac{1}{N_{y,\kappa}^r}\sum_{i=1}^{N^r}\mathbb{1}_{\{|y-Y_i^r|\leq\kappa\}}\left[-\log D(\boldsymbol{X}_i^r,y)\right] - \mathbb{E}_{\boldsymbol{X}\sim p_r(\boldsymbol{x}|Y=y)}\left[-\log D(\boldsymbol{X},y)\right]\right]\hat{p}_r^{\mathrm{KDE}}(y)dy\right| \\
&+ \sup_{D\in\mathcal{D}}\left|\int\left[\frac{1}{N_{y,\kappa}^g}\sum_{i=1}^{N^r}\mathbb{1}_{\{|y-Y_i^g|\leq\kappa\}}\left[-\log(1-D(\boldsymbol{X}_i^g,y))\right] - \mathbb{E}_{\boldsymbol{X}\sim p_g(\boldsymbol{x}|Y=y)}\left[-\log(1-D(\boldsymbol{X},y))\right]\right]\hat{p}_g^{\mathrm{KDE}}(y)dy\right|.
\end{aligned}$$

These four terms in the RHS can be bounded separately as follows

1) The first term can be bounded by using Theorem S.4 and the boundness of D. For the first term, $\forall\delta_1\in(0,1)$, with at least probability $1-\delta_1$,

$$\begin{aligned}
&\sup_{D\in\mathcal{D}}\left|\int\left[\int\left[-\log D(\boldsymbol{x},y)\right]p_r(\boldsymbol{x}|Y=y)d\boldsymbol{x}\right](p_r(y) - \hat{p}_r^{\mathrm{KDE}}(y))dy\right| \\
\leq& U\left(\sqrt{\frac{C_{1,\delta_1}^{\mathrm{KDE}}\log N^r}{N^r\sigma}} + L^r\sigma^2\right),
\end{aligned} \tag{S.36}$$

   for some constants $C_{1,\delta_1}^{\mathrm{KDE}}$ depending on δ_1.

2) Similarly, for the second term, $\forall\delta_2\in(0,1)$, with at least probability $1-\delta_2$,

$$\begin{aligned}
&\sup_{D\in\mathcal{D}}\left|\int\left[\int\left[-\log(1-D(\boldsymbol{x},y))\right]p_g(\boldsymbol{x}|Y=y)d\boldsymbol{x}\right](p_g(y) - \hat{p}_g^{\mathrm{KDE}}(y))dy\right| \\
\leq& U\left(\sqrt{\frac{C_{2,\delta_2}^{\mathrm{KDE}}\log N^g}{N^r\sigma}} + L^g\sigma^2\right),
\end{aligned} \tag{S.37}$$

   for some constants $C_{2,\delta_2}^{\mathrm{KDE}}$ depending on δ_2.

3) The third term can be bounded by using Lemma S.3. For the third term, $\forall\delta_3\in(0,1)$, with at least probability $1-\delta_3$,

$$\begin{aligned}
&\sup_{D\in\mathcal{D}}\left|\int\left[\frac{1}{N_{y,\kappa}^r}\sum_{i=1}^{N^r}\mathbb{1}_{\{|y-Y_i^r|\leq\kappa\}}\left[-\log D(\boldsymbol{X}_i^r,y)\right] - \mathbb{E}_{\boldsymbol{X}\sim p_r(\boldsymbol{x}|Y=y)}\left[-\log D(\boldsymbol{X},y)\right]\right]\hat{p}_r^{\mathrm{KDE}}(y)dy\right| \\
\leq& \int\sup_{D\in\mathcal{D}}\left|\frac{1}{N_{y,\kappa}^r}\sum_{i=1}^{N^r}\mathbb{1}_{\{|y-Y_i^r|\leq\kappa\}}\left[-\log D(\boldsymbol{X}_i^r,y)\right] - \mathbb{E}_{\boldsymbol{X}\sim p_r(\boldsymbol{x}|Y=y)}\left[-\log D(\boldsymbol{X},y)\right]\right|\hat{p}_r^{\mathrm{KDE}}(y)dy \\
\leq& \int\left[U\sqrt{\frac{1}{2N_{y,\kappa}^r}\log\left(\frac{2}{\delta_3}\right)} + \kappa U M^r\right]\hat{p}_r^{\mathrm{KDE}}(y)dy
\end{aligned}$$

Note that $N^r_{y,\kappa} = \sum_{i=1}^{N^r} \mathbb{1}_{\{|y-Y_i^r|\}}$, which is a random variable of Y_i's. The above can be expressed as

$$\begin{aligned}&\sup_{D\in\mathcal{D}}\left|\int\left[\frac{1}{N^r_{y,\kappa}}\sum_{i=1}^{N^r}\mathbb{1}_{\{|y-Y_i^r|\le\kappa\}}\left[-\log D(\boldsymbol{X}_i^r,y)\right]-\mathbb{E}_{\boldsymbol{X}\sim p_r(\boldsymbol{x}|Y=y)}\left[-\log D(\boldsymbol{X},y)\right]\right]\hat{p}_r^{\text{KDE}}(y)dy\right|\\&\le\kappa UM^r+U\sqrt{\frac{1}{2}\log\left(\frac{2}{\delta_3}\right)}\mathbb{E}_{Y\sim\hat{p}_r^{\text{KDE}}(y)}\left[\sqrt{\frac{1}{N^r_{Y,\kappa}}}\right].\end{aligned}\tag{S.38}$$

4) Similarly, for the fourth term, $\forall\delta_4\in(0,1)$, with at least probability $1-\delta_4$,

$$\begin{aligned}&\sup_{D\in\mathcal{D}}\left|\int\left\{\int\left[\frac{1}{N^g_{y,\kappa}}\sum_{i=1}^{N^r}\mathbb{1}_{\{|y-Y_i^g|\le\kappa\}}\left[-\log(1-D(\boldsymbol{X}_i^g,y))\right]\right.\right.\\&\qquad\left.\left.-\mathbb{E}_{\boldsymbol{X}\sim p_g(\boldsymbol{x}|Y=y)}\left[-\log(1-D(\boldsymbol{X},y))\right]\right]d\boldsymbol{x}\right\}\hat{p}_g^{\text{KDE}}(y)dy\Bigg|\\&\le\kappa UM^g+U\sqrt{\frac{1}{2}\log\left(\frac{2}{\delta_4}\right)}\mathbb{E}_{Y\sim\hat{p}_g^{\text{KDE}}(y)}\left[\sqrt{\frac{1}{N^g_{Y,\kappa}}}\right].\end{aligned}\tag{S.39}$$

With $\delta_1=\delta_2=\delta_3=\delta_4=\frac{\delta}{4}$, combining Eq. (S.36) - (S.39) leads to the upper bound in Theorem S.5. □

Theorem S.6. *Assume that (A1)-(A4) hold, then $\forall\delta\in(0,1)$, with probability at least $1-\delta$,*

$$\begin{aligned}&\sup_{D\in\mathcal{D}}\left|\widehat{\mathcal{L}}^{SVDL}(D)-\mathcal{L}(D)\right|\\&\le U\left(\sqrt{\frac{C^{KDE}_{1,\delta}\log N^r}{N^r\sigma}}+L^r\sigma^2\right)+U\left(\sqrt{\frac{C^{KDE}_{2,\delta}\log N^g}{N^g\sigma}}+L^g\sigma^2\right)\\&\quad+4U\sqrt{\frac{1}{2}\log\left(\frac{16}{\delta}\right)}\left(\frac{1}{\sqrt{N^r}}\mathbb{E}_{Y\sim\hat{p}_r^{KDE}(y)}\left[\frac{1}{W^r(Y)}\right]+\frac{1}{\sqrt{N^g}}\mathbb{E}_{Y\sim\hat{p}_g^{KDE}(y)}\left[\frac{1}{W^g(Y)}\right]\right)\\&\quad+2U\left(M^r\mathbb{E}_{Y\sim\hat{p}_r^{KDE}(y)}\left[\mathbb{E}_{Y'\sim p_w^r(y'|Y)}\left|Y'-Y\right|\right]+M^g\mathbb{E}_{Y\sim\hat{p}_g^{KDE}(y)}\left[\mathbb{E}_{Y'\sim p_w^g(y'|Y)}\left|Y'-Y\right|\right]\right)\end{aligned}\tag{S.40}$$

for some constant $C^{KDE}_{1,\delta}, C^{KDE}_{2,\delta}$ depending on δ.

Proof. Similar to the decomposition for Theorem S.5, we can decompose $\sup_{D\in\mathcal{D}}|\widehat{\mathcal{L}}^{\text{SVDL}}(D)-\mathcal{L}(D)|$ into four terms which can be bounded by using Theorem S.4, the boundness of D, and Lemma S.4. The detail is omitted because it is almost identical to the one of Theorem S.5. □

S.10.4 Error Bounds of D Trained with HVDL and SVDL

Based on above theorems and lemmas, we derive the error bounds of D that is trained with HVDL and SVDL respectively. The error bound is characterized by the distance of $\widehat{D}^{\text{HVDL}}$ and $\widehat{D}^{\text{SVDL}}$ from the optimal D^* under the theoretical discriminator loss $\mathcal{L}(D)$, i.e., $\mathcal{L}(\widehat{D}^{\text{HVDL}})-\mathcal{L}(D^*)$ and $\mathcal{L}(\widehat{D}^{\text{SVDL}})-\mathcal{L}(D^*)$ respectively. Please see Theorem S.7 and S.8 for details. An illustrative diagram to visualize the theoretical analysis is shown in Fig. S.10.15.

Theorem S.7 (Error bound of D trained with HVDL)**.** *Assume that (A1)-(A4) hold, then $\forall\delta\in(0,1)$, with probability at least $1-\delta$,*

$$\mathcal{L}(\widehat{D}^{HVDL})-\mathcal{L}(D^*)\tag{S.41}$$

$$\begin{aligned}&\le2U\left(\sqrt{\frac{C^{KDE}_{1,\delta}\log N^r}{N^r\sigma}}+L^r\sigma^2\right)+2U\left(\sqrt{\frac{C^{KDE}_{2,\delta}\log N^g}{N^g\sigma}}+L^g\sigma^2\right)\\&\quad+2\kappa U(M^r+M^g)\\&\quad+2U\sqrt{\frac{1}{2}\log\left(\frac{8}{\delta}\right)}\left(\mathbb{E}_{Y\sim\hat{p}_r^{KDE}(y)}\left[\sqrt{\frac{1}{N^r_{Y,\kappa}}}\right]+\mathbb{E}_{Y\sim\hat{p}_g^{KDE}(y)}\left[\sqrt{\frac{1}{N^g_{Y,\kappa}}}\right]\right)\\&\quad+\mathcal{L}(\widetilde{D})-\mathcal{L}(D^*),\end{aligned}\tag{S.42}$$

for some constants $C^{KDE}_{1,\delta}, C^{KDE}_{2,\delta}$ depending on δ.

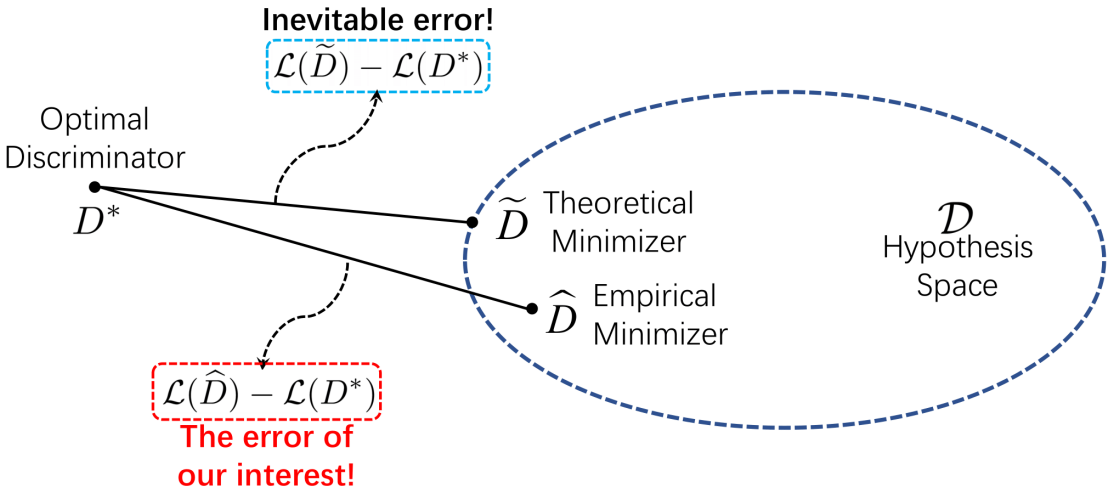

Fig. S.10.15: An illustrative diagram for error bounds of HVDL and SVDL.

Proof. We first decompose $\mathcal{L}(\widehat{D}^{\text{HVDL}}) - \mathcal{L}(D^*)$ as follows

$$
\begin{aligned}
&\mathcal{L}(\widehat{D}^{\text{HVDL}}) - \mathcal{L}(D^*) \\
=&\mathcal{L}(\widehat{D}^{\text{HVDL}}) - \widehat{\mathcal{L}}(\widehat{D}^{\text{HVDL}}) + \widehat{\mathcal{L}}(\widehat{D}^{\text{HVDL}}) - \widehat{\mathcal{L}}(\widetilde{D}) + \widehat{\mathcal{L}}(\widetilde{D}) - \mathcal{L}(\widetilde{D}) \\
&+ \mathcal{L}(\widetilde{D}) - \mathcal{L}(D^*) \\
&(\text{by } \widehat{\mathcal{L}}(\widehat{D}^{\text{HVDL}}) - \widehat{\mathcal{L}}(\widetilde{D}) \leq 0) \\
\leq& 2 \sup_{D \in \mathcal{D}} \left| \widehat{\mathcal{L}}^{\text{HVDL}}(D) - \mathcal{L}(D) \right| + \mathcal{L}(\widetilde{D}) - \mathcal{L}(D^*) \\
&(\text{by Theorem S.5}) \\
\leq& 2U \left(\sqrt{\frac{C_{1,\delta}^{\text{KDE}} \log N^r}{N^r \sigma}} + L^r \sigma^2 \right) + 2U \left(\sqrt{\frac{C_{2,\delta}^{\text{KDE}} \log N^g}{N^g \sigma}} + L^g \sigma^2 \right) \\
&+ 2\kappa U (M^r + M^g) \\
&+ 2U \sqrt{\frac{1}{2} \log \left(\frac{8}{\delta} \right)} \left(\mathbb{E}_{Y \sim \hat{p}_r^{\text{KDE}}(y)} \left[\sqrt{\frac{1}{N_{Y,\kappa}^r}} \right] + \mathbb{E}_{Y \sim \hat{p}_g^{\text{KDE}}(y)} \left[\sqrt{\frac{1}{N_{Y,\kappa}^g}} \right] \right) \\
&+ \mathcal{L}(\widetilde{D}) - \mathcal{L}(D^*).
\end{aligned}
\tag{S.43}
$$

□

Theorem S.8 (Error bound of D trained with SVDL)**.** *Assume that (A1)-(A4) hold, then* $\forall \delta \in (0,1)$*, with probability at least* $1-\delta$*,*

$$
\begin{aligned}
&\mathcal{L}(\widehat{D}^{SVDL}) - \mathcal{L}(D^*) \\
\leq& 2U \left(\sqrt{\frac{C_{1,\delta}^{KDE} \log N^r}{N^r \sigma}} + L^r \sigma^2 \right) + 2U \left(\sqrt{\frac{C_{2,\delta}^{KDE} \log N^g}{N^g \sigma}} + L^g \sigma^2 \right) \\
&+ 4U \sqrt{\frac{1}{2} \log \left(\frac{16}{\delta} \right)} \left(\frac{1}{\sqrt{N^r}} \mathbb{E}_{Y \sim \hat{p}_r^{KDE}(y)} \left[\frac{1}{W^r(Y)} \right] + \frac{1}{\sqrt{N^g}} \mathbb{E}_{Y \sim \hat{p}_g^{KDE}(y)} \left[\frac{1}{W^g(Y)} \right] \right) \\
&+ 2U \left(M^r \mathbb{E}_{Y \sim \hat{p}_r^{KDE}(y)} \left[\mathbb{E}_{Y' \sim \hat{p}_w^r(y'|Y)} |Y' - Y| \right] + M^g \mathbb{E}_{Y \sim \hat{p}_g^{KDE}(y)} \left[\mathbb{E}_{Y' \sim \hat{p}_w^g(y'|Y)} |Y' - Y| \right] \right) \\
&+ \mathcal{L}(\widetilde{D}) - \mathcal{L}(D^*),
\end{aligned}
\tag{S.44}
$$

for some constant $C_{1,\delta}^{KDE}$, $C_{2,\delta}^{KDE}$ *depending on* δ*.*

Proof. Smilarly, based on Theorem S.6, we can derive Theorem S.8. The detailed proof is omitted. □

S.11 More Details of the Experiment on Low-resolution RC-49 in Section 5.1

S.11.1 Description of RC-49

To generate RC-49, firstly we randomly select 49 3-D chair object models from the "Chair" category provided by ShapeNet [49]. Then we use Blender v2.79 [1] to render these 3-D models. Specifically, during the rendering, we rotate each chair model

1. https://www.blender.org/download/releases/2-79/

along with the yaw axis for a degree between 0.1° and 89.9° (angle resolution as 0.1°) where we use the scene image mode to compose our dataset. The rendered images are converted from the RGBA to RGB color model. In total, the RC-49 dataset consists of 44051 images of image size 64×64 in the PNG format.

S.11.2 Network architectures

The RC-49 dataset is a more sophisticated dataset compared with the simulation, thus it requires networks with deeper layers. We employ the SNGAN architecture [24] in both cGAN and CcGAN consisting of residual blocks for the generator and the discriminator. Moreover, for the generator in cGAN, the regression labels are input into the network by the label embedding [50] and the conditional batch normalization [32]. For the discriminator in cGAN, the regression labels are fed into the network by the label embedding and the label projection [3]. For CcGAN, the regression labels are fed into networks by the two proposed label input methods (NLI and ILI) in Section 2.2. The pre-trained CNN $T_1 + T_2$ for ILI is a modified ResNet-34 with two extra linear layers before the final linear layer. The label embedding network T_3 is a 5-layer MLP with 128 nodes in each layer. The dimension of the noise z is 128 for NLI-based CcGANs and 256 for ILI-based CcGANs. Please refer to our codes for more details about the network specifications of cGAN and CcGAN.

S.11.3 Training setups

The cGAN and CcGAN are trained for 30,000 iterations on the training set with the Adam [51] optimizer (with $\beta_1 = 0.5$ and $\beta_2 = 0.999$), a constant learning rate 10^{-4} and batch size 256.

The rule of thumb formulae in Rmk 3 are used to select the hyper-parameters for HVDL and SVDL, where we let $m_\kappa = 2$. Thus, the three hyper-parameters in this experiments are set as follows: $\sigma = 0.0473$, $\kappa = 0.004$, $\nu = 50625$.

The modified ResNet-34 (i.e., the $T_1 + T_2$ in Fig. 4) for ILI is trained for 200 epochs with the SGD optimizer, initial learning rate 0.1 (decay at epoch 60, 120, and 160 with factor 0.2), weight decay 10^{-4}, and batch size 256. The 5-layer MLP for the label embedding in ILI is trained for 500 epochs with the SGD optimizer, initial learning rate 0.1 (decay at epoch 100, 200, and 400 with factor 0.2), weight decay 10^{-4}, and batch size 256.

Please see our codes for more details of the training setups.

S.11.4 Testing setups

The RC-49 dataset consists of 899 distinct yaw angles and at each angle there are 49 images (corresponding to 49 types of chairs). At the test stage, we ask the trained cGAN or CcGAN to generate 200 fake images at each of these 899 yaw angles. Please note that, among these 899 yaw angles, only 450 of them are seen at the training stage so real images at the rest 449 angles are not used in the training.

We evaluate the quality of the fake images from three perspectives, i.e., visual quality, intra-label diversity, and label consistency. One overall metric (Intra-FID) and three separate metrics (NIQE, Diversity, and Label Score) are used. Their details are shown in Supp. S.11.5.

S.11.5 Performance measures

Before we conduct the evaluation in terms of the four metrics, we first train an autoencoder (AE) , a regression-oriented ResNet-34 [41] and a classification-oriented ResNet-34 [41] on all real images of RC-49. The bottleneck dimension of the AE is 512 and the AE is trained to reconstruct the real images in RC-49 with the MSE loss. The regression-oriented ResNet-34 is trained to predict the yaw angle of a given image. The classification-oriented ResNet-34 is trained to predict the chair type of a given image. The autoencoder and both two ResNets are trained for 200 epochs with a batch size of 256.

- **Intra-FID** [3]: *We take Intra-FID as the overall score to evaluate the quality of fake images and we prefer the small Intra-FID score.* At each evaluation angle, we compute the FID [35] between 49 real images and 200 fake images in terms of the bottleneck feature of the pre-trained AE. The Intra-FID score is the average FID over all 899 evaluation angles. Please note that we also try to use the classification-oriented ResNet-34 to compute the Intra-FID but the Intra-FID scores vary in a very wide range and sometimes obviously contradict with the three separate metrics.
- **NIQE** [37]: *NIQE is used to evaluate the visual quality of fake images with the real images as the reference and we prefer the small NIQE score.* We train one NIQE model with the 49 real images at each of the 899 angles so we have 899 NIQE models. During evaluation, an average NIQE score is computed for each evaluation angle based on the NIQE model at that angle. Finally, we report the average and standard deviations of the 899 average NIQE scores over the 899 yaw angels (i.e., "the mean/standard deviation of 899 means"). Note that the NIQE is implemented by the NIQE library in `MATLAB`. The block size and the sharpness threshold are set to 8 and 0.1 respectively in this and rest experiments.
- **Diversity**: *Diversity is used to evaluate the intra-label diversity and the larger the better.* In RC-49, there are 49 chair types. At each evaluation angle, we ask a pre-trained classification -oriented ResNet-34 to predict the chair types of the 200 fake images and an entropy is computed based on these predicted chair types. The diversity reported in Table 1 is the average of the 899 entropies over all evaluation angles.
- **Label Score**: *Label Score is used to evaluate the label consistency and the smaller the better.* We ask the pre-trained regression-oriented ResNet-34 to predict the yaw angles of all fake images and the predicted angles are then compared with the assigned angles. The Label Score is defined as the average absolute distance between the predicted angles and assigned angles over all fake images, which is equivalent to the Mean Absolute Error (MAE). Note that, to plot the line graphs, we compute Label Score at each of the 899 evaluation angles.

S.11.6 Example 64×64 RC-49 images

Example RC-49 images are shown in Fig. S.11.16.

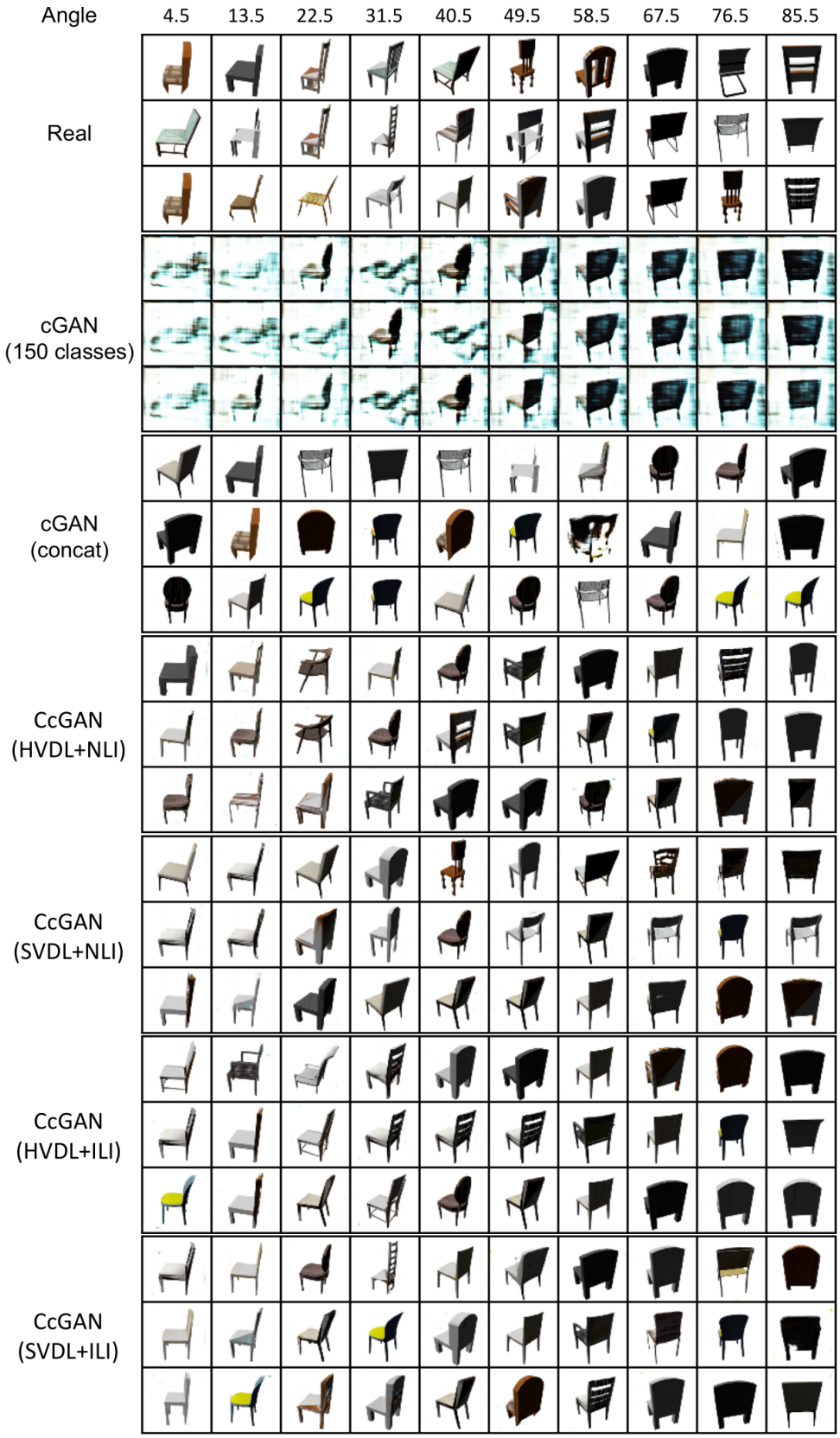

Fig. S.11.16: Three RC-49 example images in 64×64 resolution for each of 10 angles: real images and example fake images from cGAN and four proposed CcGANs, respectively. CcGANs produce chair images with **higher visual quality and more diversity**.

S.11.7 Extra experiments

S.11.7.1 Interpolation

In Fig. S.11.17, we present some interpolation results of the four CcGAN methods (i.e., HVDL+NLI, SVDL+NLI, HVDL+ILI, and SVDL+ILI). For an input pair (z, y), we fix the noise z but perform label-wise interpolations, i.e., varying label y from 4.5 to 85.5. Clearly, all generated images are visually realistic and we can see the chair distribution smoothly changes over continuous angles. Please note that, Fig. S.11.17 is meant to show the smooth change of the chair distribution instead of one single chair so the chair type may change over angles. This confirms CcGAN is capable of capturing the underlying conditional image distribution rather than simply memorizing training data.

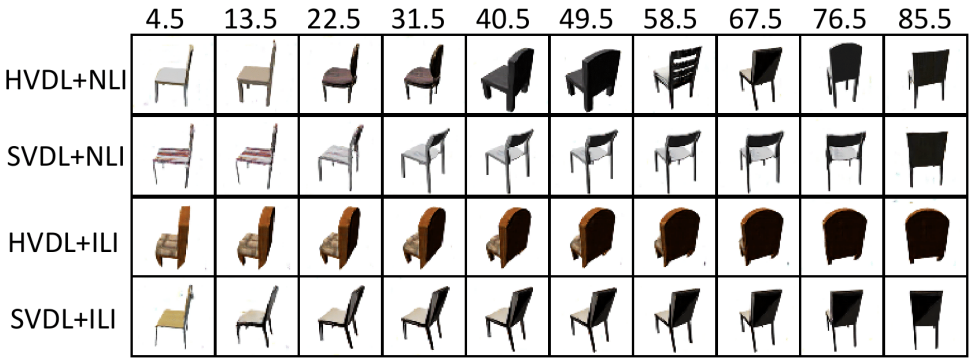

Fig. S.11.17: Some example RC-49 fake images from the four CcGAN methods. We fix the noise $\boldsymbol{z}$ but vary the label y.

S.11.7.2 Degenerated CcGAN

In this experiment, we consider the extreme case of the proposed CcGAN (degenerated CcGAN), i.e., $\sigma \to 0$ and $\kappa \to 0$ or $\nu \to +\infty$. Some examples from a degenerated NLI-based CcGAN are shown in Fig. S.11.18. Since, at each angle, the degenerated CcGAN only uses the images at this angle, it leads to the mode collapse problem (e.g, the row in the yellow rectangle) and bad visual quality (e.g., images in the red rectangle) at some angles.

Note that the degenerated CcGAN is still different from cGAN, since we still treat y as a continuous scalar instead of a class label here and we use the proposed label input method (e.g., NLI) to incorporate y into the generator and the discriminator.

S.11.7.3 cGAN: different number of classes

In this experiment, we show that cGAN still fails even though we bin $[0.1, 89.9]$ into other number of classes. We experimented with three different bin setting – grouping labels into 90, 150, and 210 classes, respectively. Experimental results are shown in Fig. S.11.19 and we observe all three cGANs completely fail.

S.12 More details of the experiment on the Low-resolution UTKFace dataset in Section 5.2

S.12.1 Description of the UTKFace dataset

The UTKFace dataset is an age regression dataset [42], with human face images collected in the wild. We use a preprocessed version (cropped and aligned), with ages spanning from 1 to 60. After the data cleaning (i.e., removing images of very low quality or with clearly wrong labels), the number of images left is 14760. These images are then resized to 64×64. The histogram of the UTKFace dataset after data cleaning is shown in S.12.20.

From Fig. S.12.20, we can see UTKFace dataset is very imbalanced so the samples from the minority age groups are unlikely to be chosen at each iteration during the GAN training. Consequently, cGAN and CcGAN may not be well-trained at these minority age groups. To increase the chance of drawing these minority samples during training, we randomly replicate samples in the minority age groups to ensure that the sample size of each age is more than 200.

S.12.2 Network architectures

The network architectures used in this experiment is similar to those in the RC-49 experiment. Please refer to our codes for more details about the network specifications.

S.12.3 Training setups

The cGAN and CcGAN are trained for 40,000 iterations on the training set with the Adam [51] optimizer (with $\beta_1 = 0.5$ and $\beta_2 = 0.999$), a constant learning rate 10^{-4} and batch size 512. The rule of thumb formulae in Section 3 are used to select the hyper-parameters for HVDL and SVDL, where we let $m_\kappa = 1$.

Please see our codes for more details of the training setups.

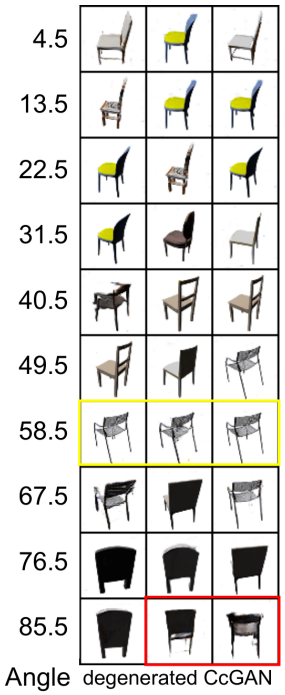

Fig. S.11.18: Some example RC-49 fake images from a degenerated NLI-based CcGAN.

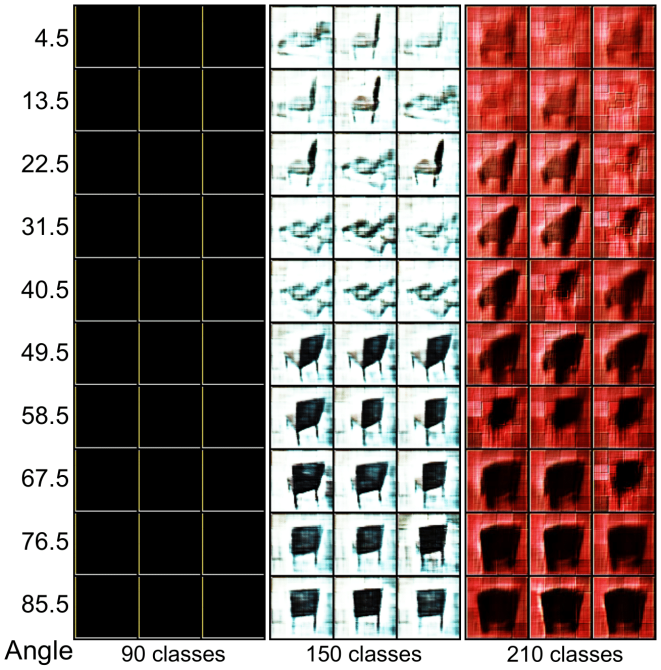

Fig. S.11.19: Example RC-49 fake images from cGAN when we bin the yaw angle range into different number of classes.

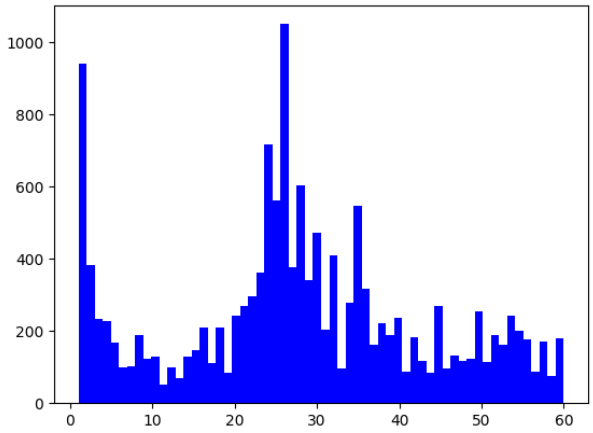

Fig. S.12.20: The histogram of the UTKFace dataset with ages varying from 1 to 60.

S.12.4 Performance measures

Similar to the RC-49 experiment, we evaluate the quality of fake images by Intra-FID, NIQE, Diversity, and Label Score. We also train an AE (bottleneck dimension is 512), a classification-oriented ResNet-34, and a regression-oriented ResNet-34 on the UTKFace dataset. Please note that, the UTKFace dataset consists of face images from 5 races based on which we train the classification-oriented ResNet-34. The AE and both two ResNets are trained for 200 epochs with a batch size 256.

S.12.5 Example UTKFace images

Example UTKFace images are shown in Fig. S.12.21.

S.12.6 Extra experiments

S.12.6.1 Interpolation

To perform label interpolation experiments, we keep the noise vector z fixed and vary label from age 3 to age 57 for the four CcGANs. The interpolation results are illustrated in S.12.22. As age y increases, we observe the synthetic face gradually

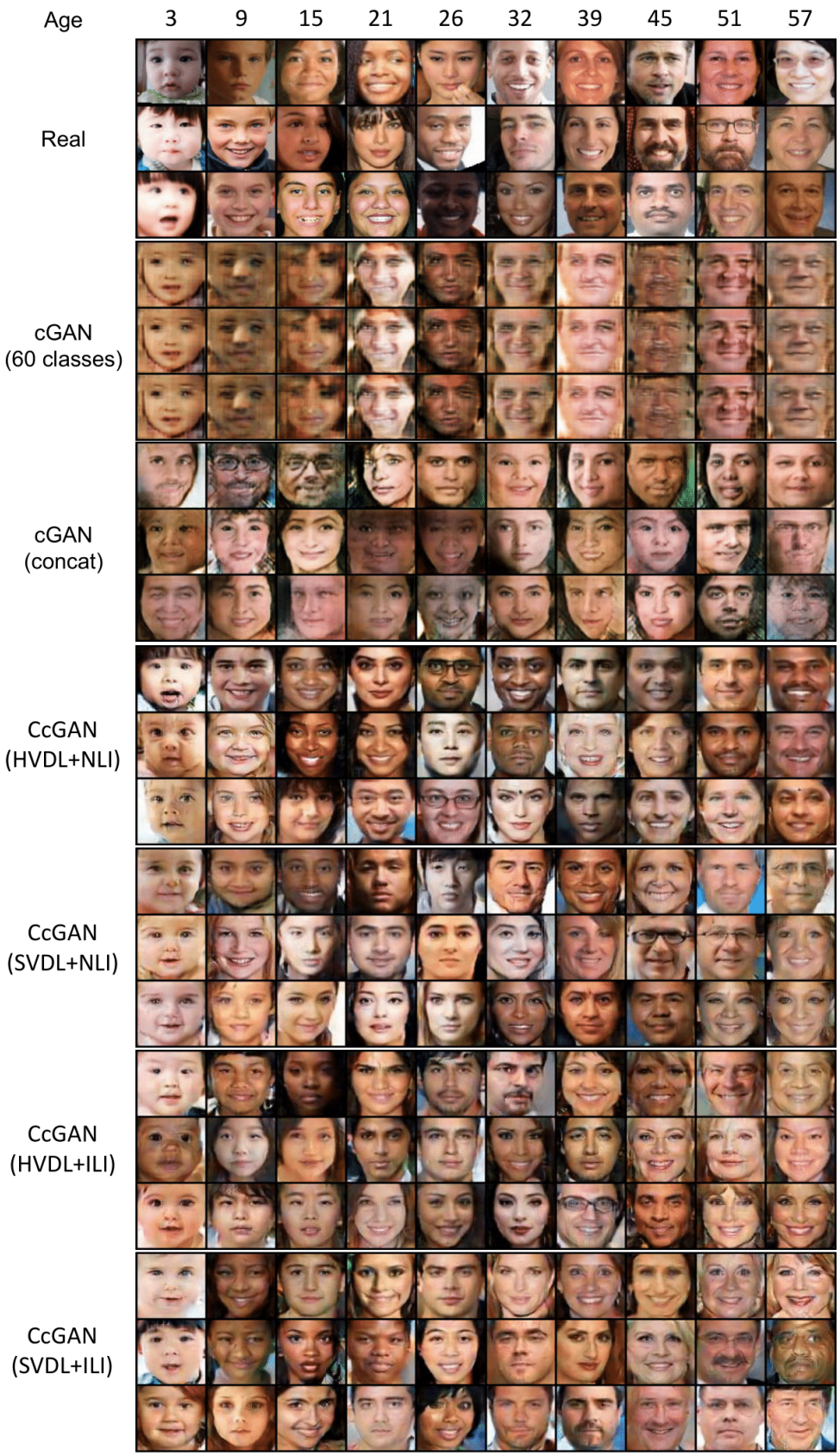

Fig. S.12.21: Three UTKFace example images in 64×64 resolution for each of 10 ages: real images and example fake images from cGAN and four proposed CcGANs, respectively. CcGANs produce face images with **higher visual quality and more diversity**.

becomes older in appearance. This observation convincingly shows that all four CcGANs do not simply memorize or overfit to the training set. Indeed, our CcGANs demonstrate continuous control over synthetic images with respect to ages.

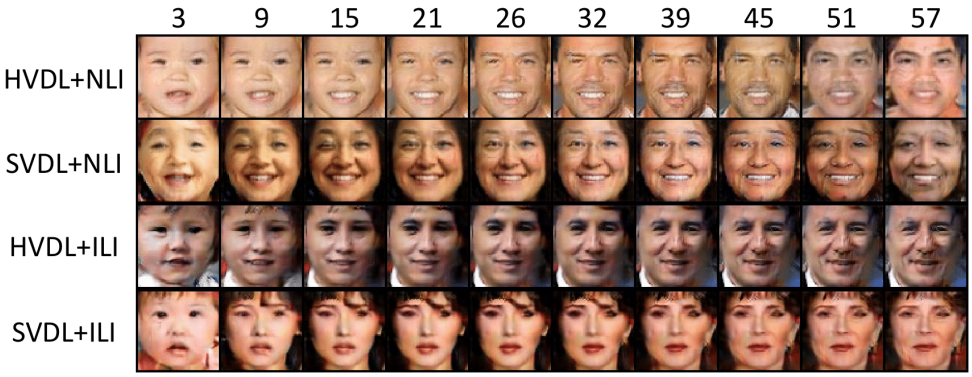

Fig. S.12.22: Some examples of generated UTKFace images from the four CcGAN methods. We fix the noise $\boldsymbol{z}$ but vary the label y from 3 to 57.

S.12.6.2 Degenerated CcGAN

We consider the extreme cases of the proposed CcGANs on the UTKFace dataset. As shown in Fig. S.12.23, the degenerated NLI-based CcGANs fails to generate facial images at some ages (e.g., 51 and 57) because of too small sample sizes.

S.12.6.3 cGAN: different number of classes

In the last experiment, we bin samples into different number of classes based on ground-truth labels, in order to increase the number of training samples at each class. Then we train cGAN using samples from the binned classes. We experimented with two different bin setting, i.e., binning image samples into 60 classes and 40 classes, respectively. The results are reported in Fig.S.12.24. The results demonstrate cGANs consistently fail to generate diverse synthetic images with labels aligned with their conditional information. Moreover, the image quality is much worse than those from the proposed CcGANs. In conclusion, compared with existing cGANs, our proposed CcGANs have substantially better performance in terms of the image quality and diversity.

S.13 More details of the experiment on the Low-resolution Cell-200 dataset in Section 5.3

S.13.1 Description of Cell-200

Cell-200 is a synthetic image dataset, emulating the colonies of bacterial cells in the view of fluorescence microscope. This dataset contains cell populations with overall number varying between 1 and 200, generated with [52]. For each cell population (e.g., 1 to 200), we generate 1000 different synthetic fluorescence microscopic images, with diverse cell variations (e.g., shapes, locations, overlaps and blurring effects). As in [53], we set nucleus radius as 5, and image size as 256×256. To alleviate computational burden, images in the Cell-200 dataset are then resized to 64×64.

S.13.2 Network architectures

The network architectures for cGAN and CcGAN in this experiment are adapted from the famous DCGAN [36] architecture. The dimension of the noise z is 128 for NLI-based CcGANs and 256 for ILI-based CcGANs. Please refer to our codes for more details about the network specifications.

S.13.3 Training setups

The cGAN and CcGAN are trained for 5000 iterations on the training set with the Adam [51] optimizer (with $\beta_1 = 0.5$ and $\beta_2 = 0.999$) and a constant learning rate 10^{-4}. The rule of thumb formulae in Remark 3 are used to select the hyperparameters for HVDL and SVDL, where we let $m_\kappa = 2$.

For cGAN training, the cell count range $[1, 200]$ is split into 100 disjoint intervals, i.e., $[1, 3), [3, 5), \ldots, [197, 199), [199, 200]$. In this case, cGAN estimates image distribution conditional on these intervals. In Supp. S.13.7.2, we also compare the performance of cGAN under different splitting schemes.

Please note that we use different batch sizes for cGAN and CcGAN in this experiment. The batch size for cGAN is 512. Differently, for all CcGAN methods in this experiment, the batch size is 512 for the generator but 32 for the discriminator. The reason that we use different batch sizes for the generator and discriminator in CcGAN is based on some observations we got during training. In this experiment, if the generator and the discriminator in CcGAN have the same batch size, the discriminator loss often decreases to almost zero very quickly while the generator loss still maintains at a high level. Consequently, at each iteration, the discriminator can easily distinguish the real and fake images while the generator cannot

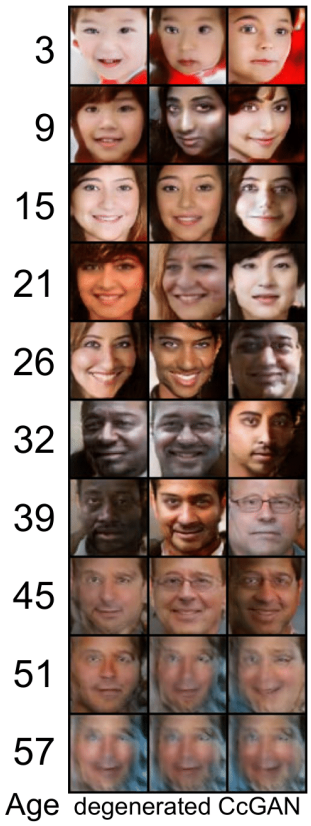

Fig. S.12.23: Some example UTKFace fake images from a degenerated NLI-based CcGAN.

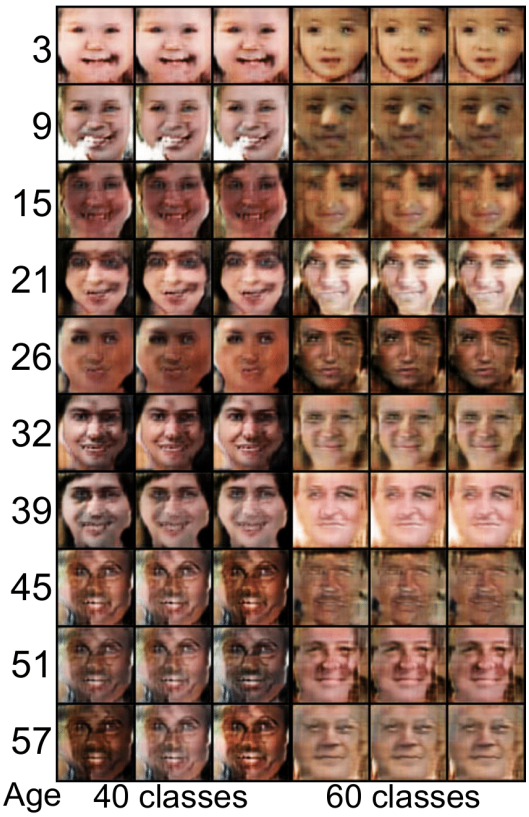

Fig. S.12.24: Example UTKFace fake images from cGAN when we bin the age range into different number of classes.

fool the discriminator and won't improve in the next iteration, which implies a high imbalance between the generator update and the discriminator update. To balance the training of the generator and the discriminator, we deliberately decrease the number of images seen by the discriminator at each iteration to slow down the update of the discriminator so that the generator can catch up.

Please see our codes for more details of the training setups.

S.13.4 Testing setups

We evaluate the trained cGAN and four CcGAN methods on all 200 cell counts (half of them are blinded during training). Each method generates 1,000 images for each cell count, so there are 200,000 fake images from each method.

When evaluating the trained cGAN, if a test label y' is unseen in the training set, we just need to find which interval (recall we split $[1, 200]$ is split into 100 disjoint intervals) covers this label. Then, we generate samples from the trained cGAN conditional on this interval instead of y'.

S.13.5 Performance measures

Similar to the previous two experiments, we evaluate the quality of fake images by Intra-FID, NIQE, and Label Score but Diversity. The Diversity score is not available here because we don't have any class label in Cell-200. An AE with a bottleneck dimension of 512 and a regression-oriented ResNet-34 are pre-trained on the complete Cell-200 dataset (i.e., 200,000 images) to compute the Intra-FID and Label Score respectively. The possibility of lacking class labels in regression-oriented datasets is another reason that we propose to use an AE to compute Intra-FID instead of a classification-oriented CNN. The AE is trained for 50 epochs with a batch size of 256. The regression-oriented ResNet-34 is trained for 200 epochs with a batch size of 256.

S.13.6 Example UTKFace images

Example Cell-200 images are shown in Fig. S.13.25.

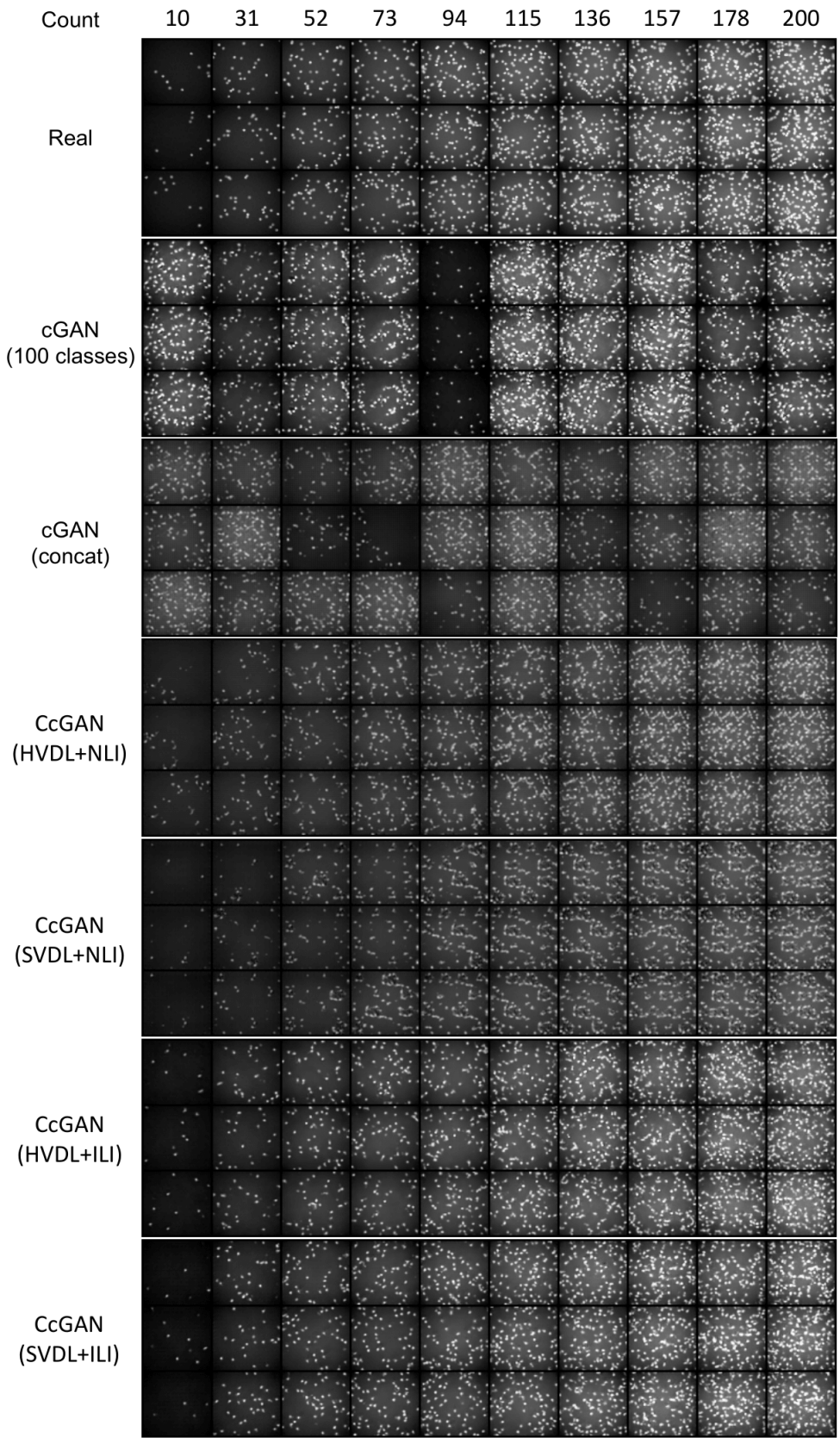

Fig. S.13.25: Three Cell-200 images in 64×64 resolution for each of 10 cell counts absent in the training data: real images and example fake images from cGAN and four proposed CcGANs, respectively. cGAN has severe mode collapse problem in this experiment. Two NLI-based CcGANs do not perform well enough but two ILI-based CcGANs produce images with **higher visual quality, more diversity, and higher label consistency**.

S.13.7 Extra experiments

S.13.7.1 Interpolation

To perform the label interpolation, we keep the noise vector z fixed and vary label from 10 to 200 for the four CcGANs. The interpolation results are illustrated in S.13.26. As cell count y increases, we observe the cells in images become more crowded. This observation convincingly shows that all four CcGANs do not simply memorize or overfit to the training set. Indeed, our CcGANs demonstrate continuous control over synthetic images with respect to cell counts.

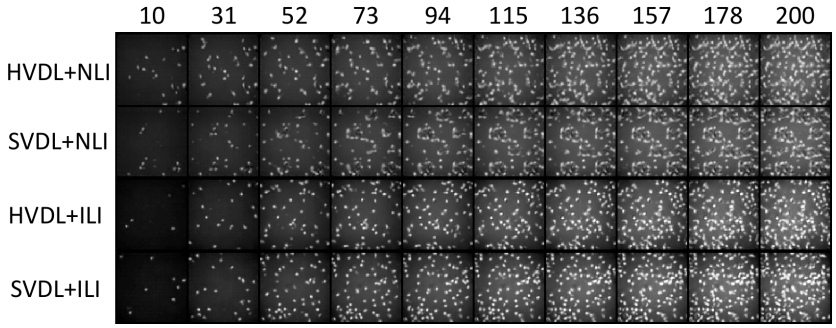

Fig. S.13.26: Some examples of generated Cell-200 images from the four CcGAN methods. We fix the noise $\boldsymbol{z}$ but vary the label y from 10 to 200.

S.13.7.2 cGAN: different number of classes

In this experiment, we experimented with two different bin setting – grouping labels into 100 classes and 50 classes, respectively. Experimental results are shown in Fig. S.13.27. We observe both cGANs fail in this experiment. First of all, cGANs still suffer from the mode collapse problems. Besides, cell counts of generated images do not match those of their given labels.

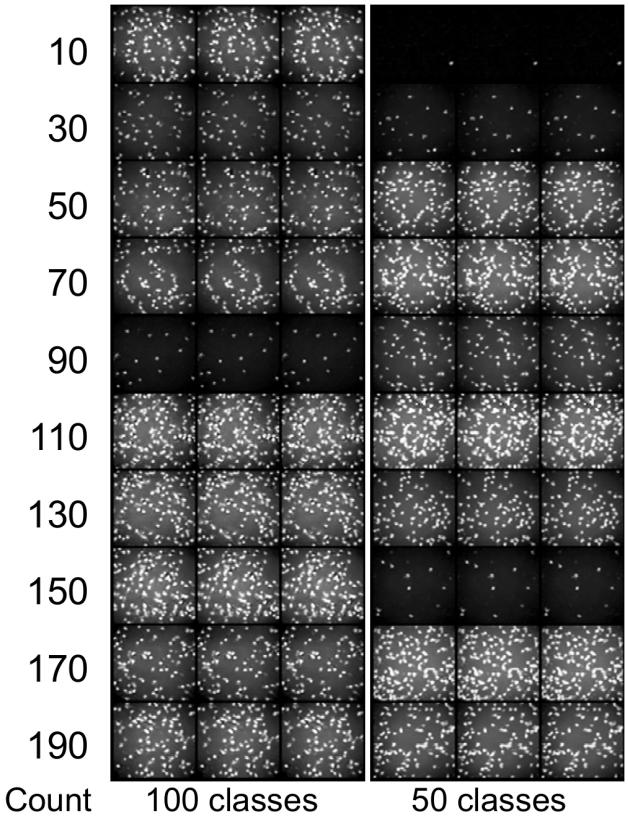

Fig. S.13.27: Example Cell-200 fake images from cGAN when we bin the range of cell count into different number of classes.

S.14 More details of the experiment on the Low-resolution Steering Angle dataset in Section 5.4

S.14.1 Description of Steering Angle

The *Steering Angle* dataset is a subset of an autonomous driving dataset [44], [54]. Steering Angle consists of 12,271 RGB images with 1,904 distinct steering angles ranging from -80° to 80°. We resize all images to 64×64. The histogram of the steering angles in this dataset is show in Fig. S.14.28.

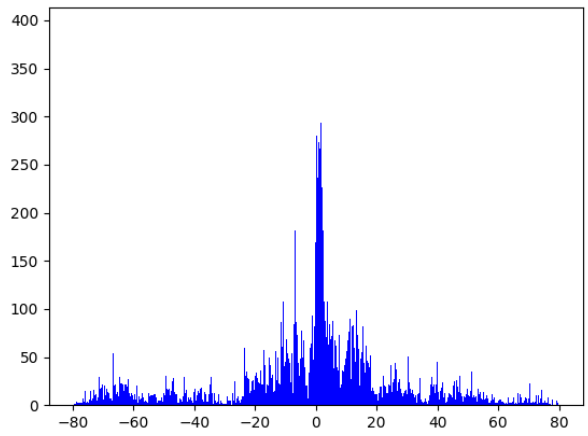

Fig. S.14.28: The histogram of the Steering Angle dataset with steering angles varying from -80° to 80°. At many angles, we only have 1 or 2 images.

S.14.2 Network architectures

The network architectures used in this experiment is similar to those in the RC-49 and UTKFace experiments. Please refer to our codes for more details about the network specifications.

S.14.3 Training setups

The cGAN and CcGAN are trained for 20,000 iterations on the training set with the Adam [51] optimizer (with $\beta_1 = 0.5$ and $\beta_2 = 0.999$) and a constant learning rate 10^{-4}. The rule of thumb formalue in Remark 3 are used to select the hyperparameters for HVDL and SVDL, where we let $m_\kappa = 5$.

Please note that, similar to the Cell-200 experiment, we use different batch sizes for the generator and discriminator in four CcGAN methods. The batch size is set to 64 and 512 respectively for the discriminator and generator in CcGAN. Please refer to Supp. S.13.3 for the reason.

Please see our codes for more details of the training setups.

S.14.4 Testing setups

At the testing stage, we first set 2,000 evenly spaced evaluation labels in $[-80^\circ, 80^\circ]$ and we ask each GAN model to generate 50 images conditional on each evaluation label.

S.14.5 Performance measures

The quality of generated images from each GAN is evaluated by SFID, NIQE, Diversity, and Label Score.

- **SFID**: To computing SFID, we preset 1,000 SFID centers $[-80^\circ, 80^\circ]$ and let $r_{\text{SFID}} = 2^\circ$. These SFID centers and $r_{\text{SFID}} = 2$ define 1,000 joint SFID intervals. We compute one FID between real and fake images with labels in this interval. We report the mean (i.e., SFID) and standard deviation of these FIDs for each GAN.
- **NIQE** [37]: Different from previous three experiments, we train only one NIQE model by using all real images in the Steering Angle dataset since this dataset is highly imbalanced. In the evaluation, we compute one NIQE score for each SFID interval. The reported NIQE score in Table 1 is the mean of these NIQE scores.
- **Diversity**: The original autonomous driving dataset [44], [54] does not have class labels. To compute Diversity, we manually group the images in Steering Angle into five categories according to their background objects or the types of the road in the images. The five groups are labeled respectively by tree, tree+barrier, bush, bush+barrier, and winding mountain road. Some example images for these five groups are show in Fig. S.14.29. Images in the tree group all have

trees in the background. Images in the tree+barrier group all have trees and barriers in the background. Images in the bush group all have bushes in the background. Images in the bush+barrier group all have bushes and barriers in the background. Images in the winding mountain road group all correspond to the scenes of winding mountain roads. A classification-oriented ResNet-34 is trained to classify images from these five groups, and then the Diversity score can be computed based on the entropies of predicted scenes in each SFID interval.

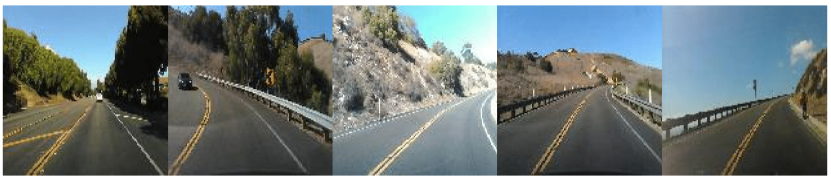

Fig. S.14.29: Example Steering Angle images from 5 scenes, i.e., tree, tree+barrier, bush, bush+barrier, and winding mountain road (from left to right).

- **Label Score**: Similar to previous experiments, we pre-train a regression-oriented ResNet-34 to predict the angle for each fake image, and then computes Label Score.
  Please note that, when plotting the line graph of Label Score versus SFID Center in Fig 13, one Label Score is computed for each SFID interval.

S.14.6 Example Steering Angle images

Example Steering Angle images are shown in Fig. S.14.30.

S.14.7 Extra experiments

S.14.7.1 Interpolation

In this section, for each CcGAN method, we fix the noise vector $\boldsymbol{z}$ but vary the regression label y from -71.8° to 72°. We can see the road in the image gradually changes from a left turn to a right turn.

S.14.7.2 cGAN: different number of classes

In this experiment, we experimented with three different bin setting – grouping labels into 90 classes, 150 classes, and 210 classes, respectively. Experimental results are shown in Fig. S.14.32. We observe that different bin settings cannot improve cGAN's performance.

S.15 Evaluation Results of Low-resolution Experiments under FID and IS

Inception Score (IS) [38] and Fréchet Inception Distance [35], originally proposed for unconditional image generation, are not appropriate overall metrics for our experiment. Consistent with the evaluation of cGANs in [3], [5], [34], a conditional generative model in our task needs to be evaluated from three perspectives: (1) visual quality, (2) intra-label diversity (the diversity of images with the same label), and (3) label consistency (whether the labels used as conditions are consistent with the true labels of fake images). (The labels used as conditions are also called assigned labels in our paper.) A conditional generative model's performance in these three perspective partially reflects its conditional density estimation quality. Since IS and FID are developed initially to evaluate images generated from unconditional GANs, they aim to assess the visual quality and **marginal** diversity of fake images, partially reflecting the marginal density estimation quality. Because computing IS and FID does not need the true and assigned labels of fake images, IS and FID cannot measure intra-label diversity and label consistency. For example, assume we have some fake images with high intra-label diversity and label consistency. We may dramatically degenerate these fake images' intra-label diversity and label consistency by shuffling their assigned labels. However, the IS and FID scores of these fake images won't change because assigned labels are not used in computing IS and FID scores. Furthermore, in our task, a model with high IS and low FID scores may still fail in our task since it may have low intra-label diversity or low label consistency. For example, although cGAN (concat) have better diversity and higher visual quality than cGAN (K classes) does, their label consistency scores are terrible, implying their failure in our task. Therefore, IS and FID are not appropriate overall metrics for our task. To comprehensively evaluate cGAN-generated images, [3] proposes Intra-FID, which computes FID separately at each of the distinct labels and reports the average FID score. We further extend Intra-FID by SFID to the scenario with insufficient real images. Besides Intra-FID and SFID, we also use three separate metrics (NIQE, Diversity, and Label Score) to evaluate the visual quality, intra-label diversity, and label consistency, respectively. The evaluation in terms of these three individual metrics is often consistent with that based on the overall metric (i.e., Intra-FID or SFID) in our experiments. Therefore, Intra-FID and SFID are taken as the overall metric in our task.

For completeness, we also report them in this appendix. Since our datasets are quite different from ImageNet [55], we train a classification ResNet-34 and an autoencoder (the one used for computing Intra-FID) from scratch on our datasets to compute IS and FID respectively. Table S.15.5 summarizes the evaluation results of all candidate methods on four low-resolution datasets in terms of IS and FID. We can see CcGAN still outperforms two cGANs, even though IS and FID are not appropriate overall metrics for our task.

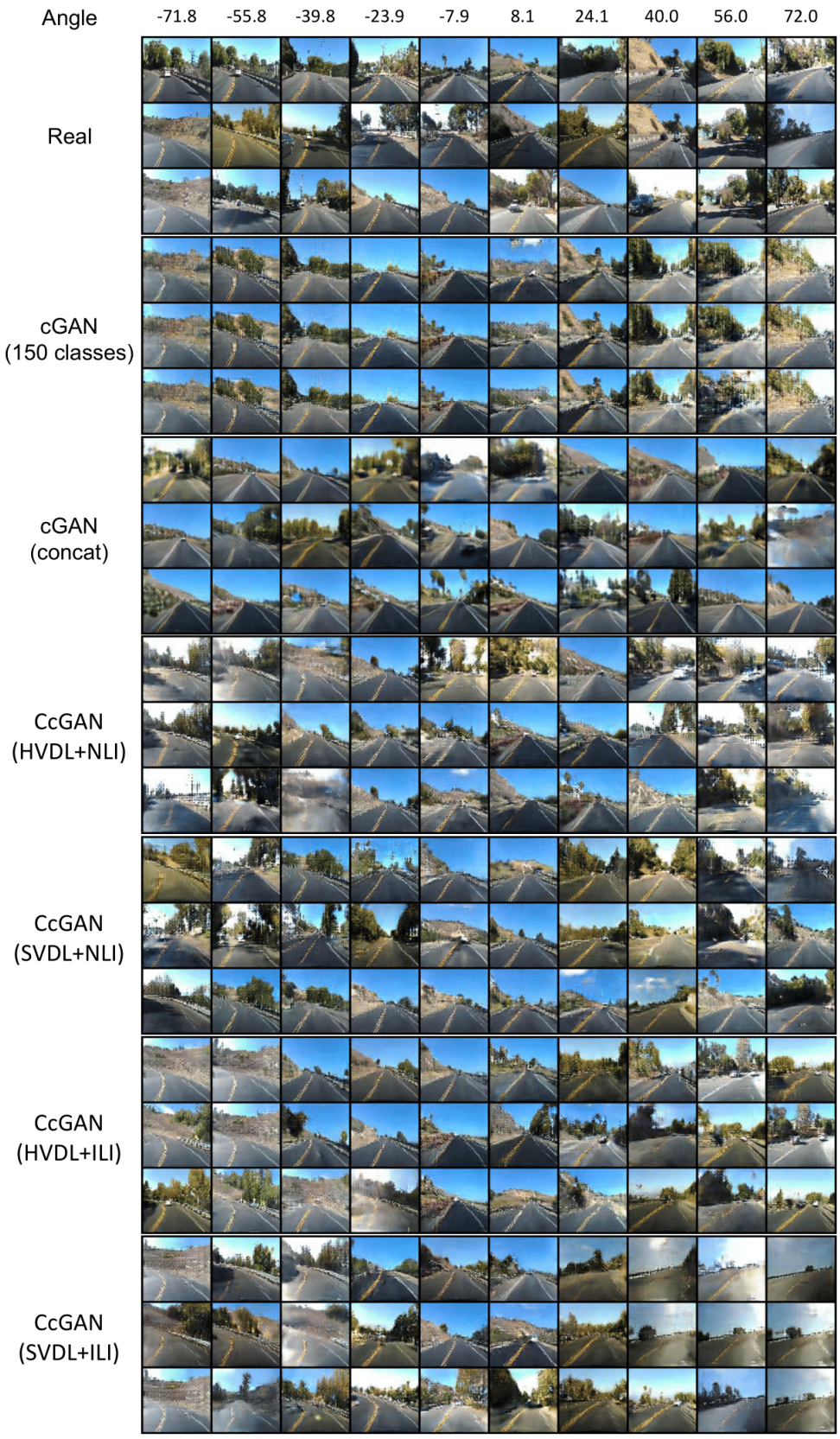

Fig. S.14.30: Three Steering Angle images in 64×64 resolution for each of 10 angles: real images and example fake images from cGAN and four proposed CcGANs, respectively. cGAN has severe mode collapse problem in this experiment. Two NLI-based CcGANs do not work well but two ILI-baesd CcGANs produce images with **higher visual quality and more diversity**.

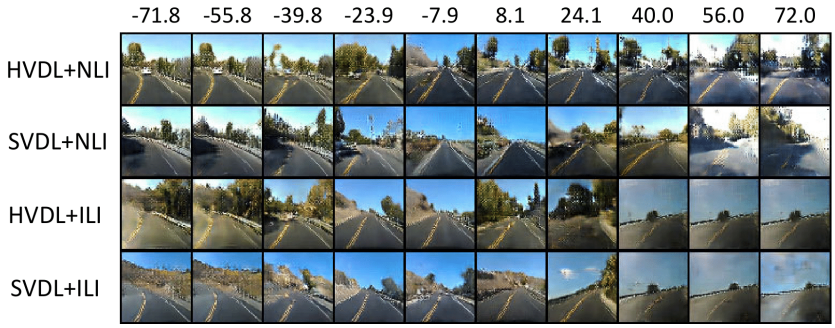

Fig. S.14.31: Some examples of generated Steering Angle images from the four CcGAN methods. We fix the noise $\boldsymbol{z}$ but vary the label y from -71.8° to 72°.

Fig. S.14.32: Example Steering Angle fake images from cGAN when we bin the range of steering angles into different number of classes.

TABLE S.15.5: IS and FID scores of all candidates methods in 64×64 experiments.

	RC-49		UTKFace		Cell-200		Steering Angle	
Method	IS ↑	FID ↓	IS ↑	FID ↓	IS ↑	FID ↓	IS ↑	FID ↓
cGAN (K classes)	2.382	1.066	2.636	0.963	-	30.086	2.572	0.976
cGAN (concat)	11.440	0.295	3.103	0.465	-	37.689	3.251	0.255
CcGAN (HVDL+NLI)	14.730	0.285	3.328	0.114	-	40.279	3.587	0.316
CcGAN (SVDL+NLI)	19.425	0.207	3.307	0.087	-	51.318	3.968	**0.212**
CcGAN (HVDL+ILI)	17.992	0.213	3.256	**0.056**	-	3.263	**4.592**	0.327
CcGAN (SVDL+ILI)	**20.173**	**0.197**	**3.382**	0.142	-	**1.684**	4.439	0.331

S.16 COMPARISON AGAINST STATE OF THE ART CGANS

All class-conditional GANs and CcGAN are trained for 30,000 iterations with batch size 256. Except CR-BigGAN and ReACGAN, when training all cGANs, we update the discriminator network twice in each iteration.

To implement the class-conditional SNGAN, we use the vanilla cGAN loss [1] instead of the hinge loss [47] because hinge loss results in mode collapse in this setting.

To implement CR-BigGAN [39] and ReACGAN [40], we borrow codes of StudioGAN from https://github.com/

POSTECH-CVLab/PyTorch-StudioGAN. The training setups for CR-BigGAN and ReACGAN are mainly based on the configuration file of StudioGAN designed for 64×64 TinyImageNet. Please note that ReACGAN [40] is published more than one year after our submission of CcGAN to ICLR 2021.

DiffAugment [19] with the strongest transformation combination (Color + Translation + Cutout) is also used in some GAN training including BigGAN+DiffAugment (class-conditional), SAGAN+DiffAugment (CcGNA), and BigGAN+DiffAugment (CcGAN).

Other setups are similar to Supp. S.11. Please see our code in ".\RC-49\RC-49_64x64_extra" for more details.

S.17 MORE DETAILS OF THE HIGH-RESOLUTION EXPERIMENTS IN SECTION 6

S.17.0.1 Reformulated hinge loss

Our CcGAN (SVDL+ILI) in the high-resolution experiments is trained with a reformulated hinge loss shown as follows.

$$\begin{aligned}\widehat{\mathcal{L}}^{\text{SVDL}}(D) = &- \frac{C_7}{N^r}\sum_{j=1}^{N^r}\sum_{i=1}^{N^r}\mathbb{E}_{\epsilon^r\sim\mathcal{N}(0,\sigma^2)}\left[W_3\cdot\min(0,-1+D(\boldsymbol{x}_i^r,y_j^r+\epsilon^r))\right] \\ &- \frac{C_8}{N^g}\sum_{j=1}^{N^g}\sum_{i=1}^{N^g}\mathbb{E}_{\epsilon^g\sim\mathcal{N}(0,\sigma^2)}\left[W_4\cdot\min(0,-1-D(\boldsymbol{x}_i^g,y_j^g+\epsilon^g))\right],\end{aligned} \tag{S.45}$$

where $\epsilon^r \triangleq y - y_j^r$, $\epsilon^g \triangleq y - y_j^g$,

$$W_3 = \frac{w^r(y_i^r, y_j^r+\epsilon^r)}{\sum_{i=1}^{N^r} w^r(y_i^r, y_j^r+\epsilon^r)}, \quad W_4 = \frac{w^g(y_i^g, y_j^g+\epsilon^g)}{\sum_{i=1}^{N^g} w^g(y_i^g, y_j^g+\epsilon^g)},$$

and C_7 and C_8 are some constants.

S.17.0.2 High-resolution RC-49

In the high-resolution experiment, we test CcGAN (SVDL+ILI), cGAN (150 classes), and cGAN (concat) on RC-49 with two resolutions, i.e., 128×128 and 256×256. We use SAGAN [5] as the backbone for all candidates. We also use hinge loss [47] to train cGAN (150 classes) and cGAN (concat), and Eq. (S.45) to train CcGAN (SVDL+ILI). DiffAugment [19] with the strongest transformation combination (Color + Translation + Cutout) is also used in all GAN training. DiffAugment substantially alleviates the mode collapse problem of cGAN (K classes) on RC-49. When training each candidate GAN, at each iteration, we update the discriminator twice while update the generator once. For 128×128 experiment, the batch size is 256. For 256×256 experiment, the batch size is set 128. The rest experimental setups are consistent with the low-resolution experiment. Some example images in the 128×128 resolution for this experiment are shown in Figs. S.17.33 and S.17.34.

S.17.0.3 High-resolution UTKFace

In the high-resolution experiment, we test CcGAN (SVDL+ILI), cGAN (60 classes), and cGAN (concat) on UTKFace with two resolutions, i.e., 128×128 and 192×192. We use SAGAN [5] as the backbone for all candidates. We also use hinge loss [47] to train cGAN (150 classes) and cGAN (concat), and Eq. (S.45) to train CcGAN (SVDL+ILI). DiffAugment [19] with the strongest transformation combination (Color + Translation + Cutout) is also used in all GAN training. DiffAugment substantially alleviates the mode collapse problem of cGAN (K classes) on UTKFace. When training each candidate GAN, at each iteration, we update the discriminator four times while update the generator once. For 128×128 experiment, the batch size is 256. For 192×192 experiment, the batch size is set 96. We also use $\nu = 900$ for the CcGAN training. The rest experimental setups are consistent with the low-resolution experiment. Some example images in the 192×192 resolution for this experiment are shown in Figs. S.17.35 and S.17.36.

S.17.0.4 High-resolution Steering Angle

In the high-resolution experiment, we test CcGAN (SVDL+ILI), cGAN (210 classes), and cGAN (concat) on Steering Angle in 128×128 resolution. We use SAGAN [5] as the backbone for all candidates. We also use hinge loss [47] to train cGAN (150 classes) and cGAN (concat), and Eq. (S.45) to train CcGAN (SVDL+ILI). DiffAugment [19] with the strongest transformation combination (Color + Translation + Cutout) is also used in all GAN training. In this experiment, even with DiffAugment, cGAN (K classes) still has the mode collapse problem. When training each candidate GAN, at each iteration, we update the discriminator twice while update the generator once. The batch size is set 256. The rest experimental setups are consistent with the low-resolution experiment. Some example images in the 128×128 resolution for this experiment are shown in Figs. S.17.37 and S.17.38.

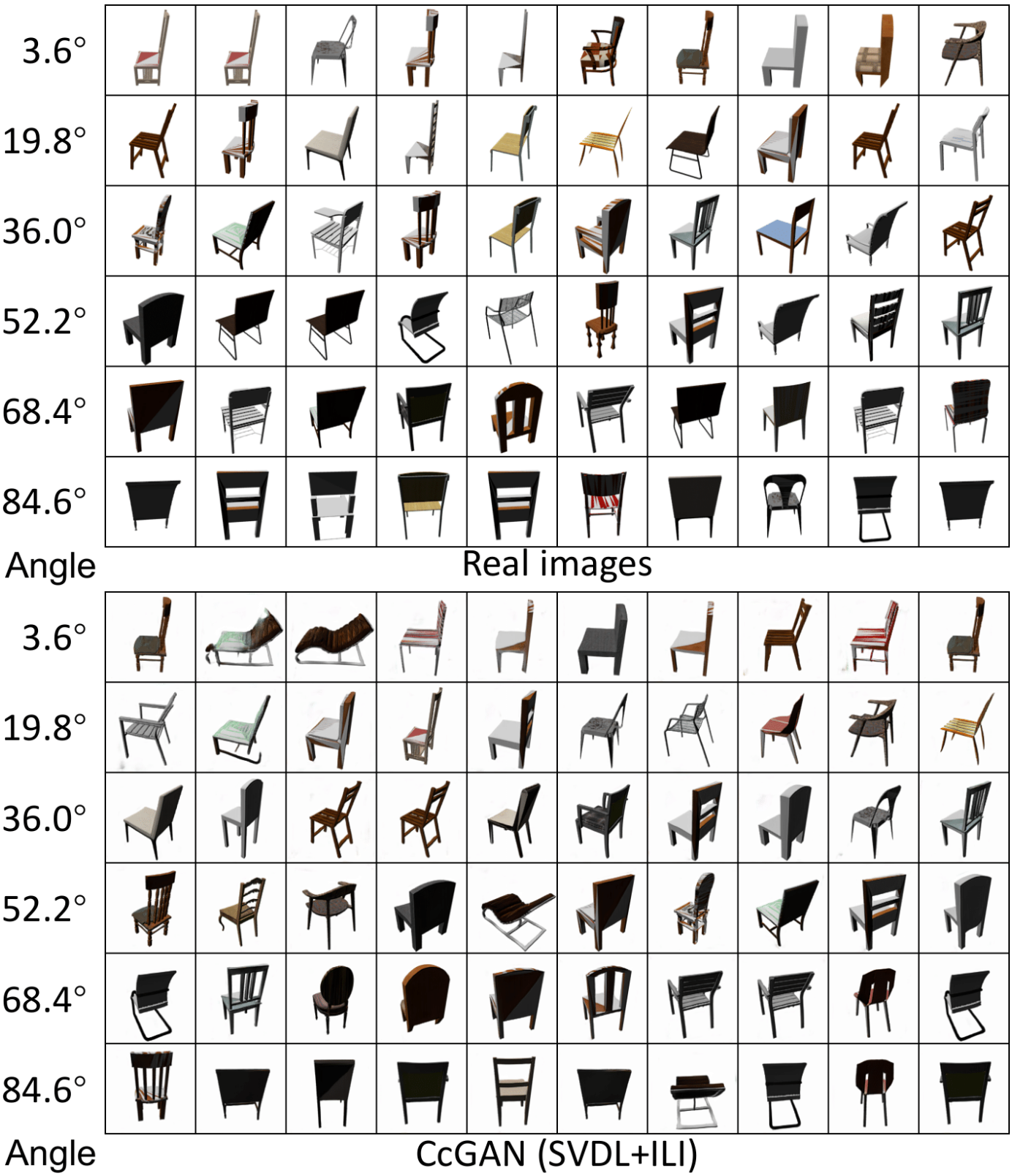

Fig. S.17.33: Some example real RC-49 images and fake RC-49 images from CcGAN (SVDL+ILI) in the 128×128 resolution. We can see CcGAN can generate visually realistic, diverse and label consistent images.

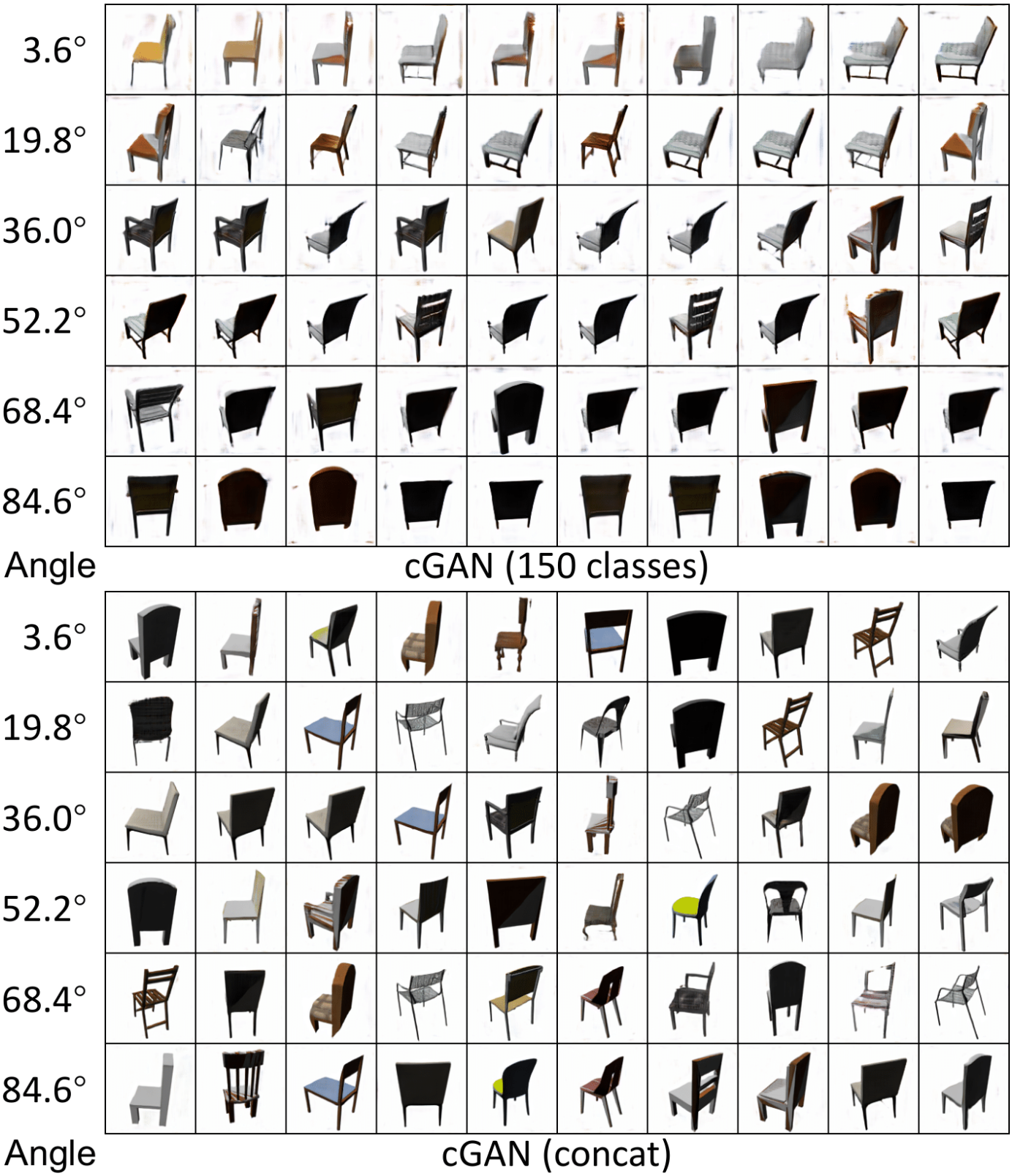

Fig. S.17.34: Some example fake RC-49 images from cGAN (150 classes) and cGAN (concat) in the 128×128 resolution. They show two types of failures of conventional cGANs. cGAN (150 classes) has high label consistency but low visual quality and low intra-label diversity. cGAN (concat) has high intra-label diversity and fair visual quality but low label consistency.

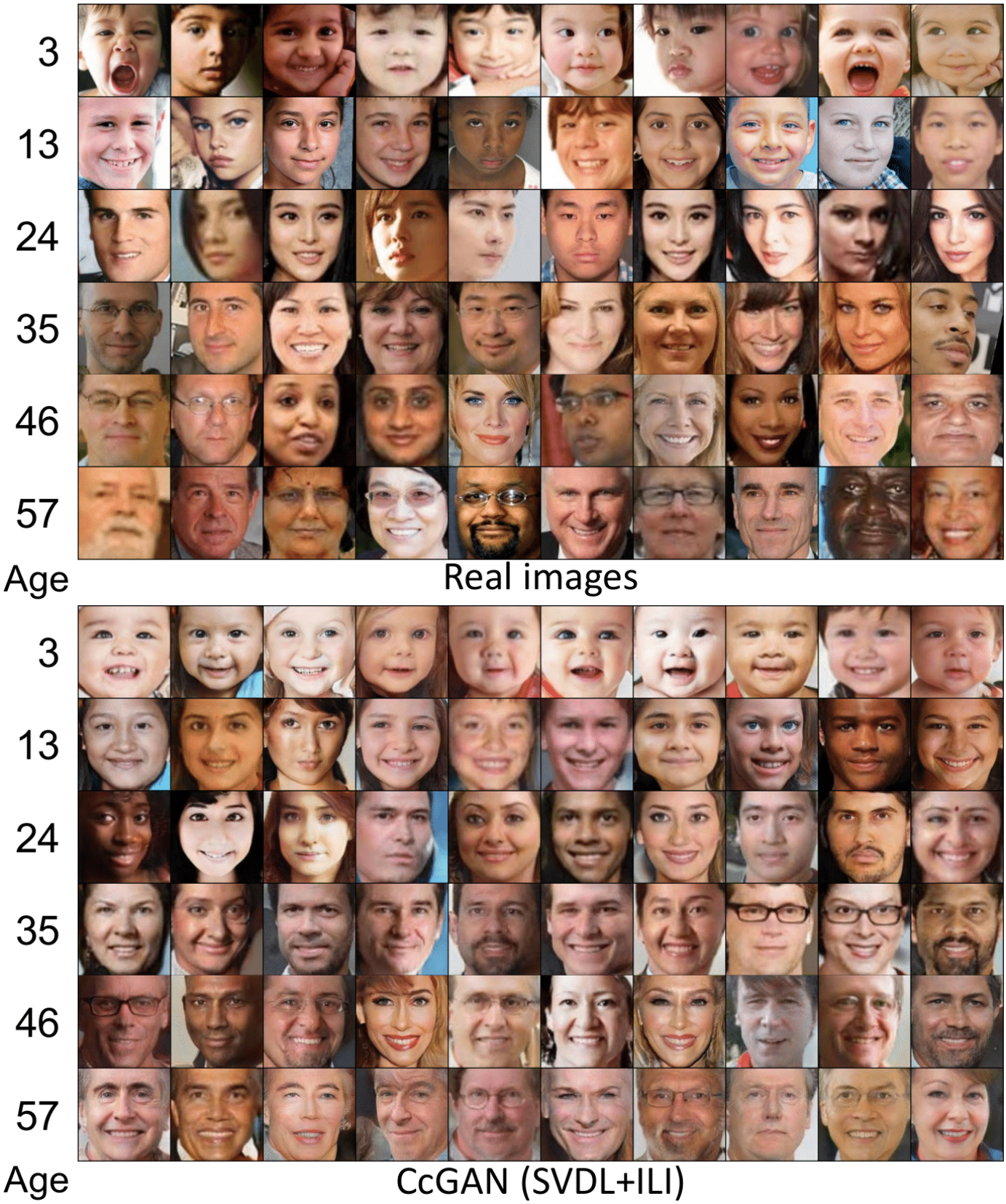

Fig. S.17.35: Some example real UTKFace images and fake UTKFace images from CcGAN (SVDL+ILI) in the 192×192 resolution. We can see CcGAN can generate visually realistic, diverse and label consistent images.

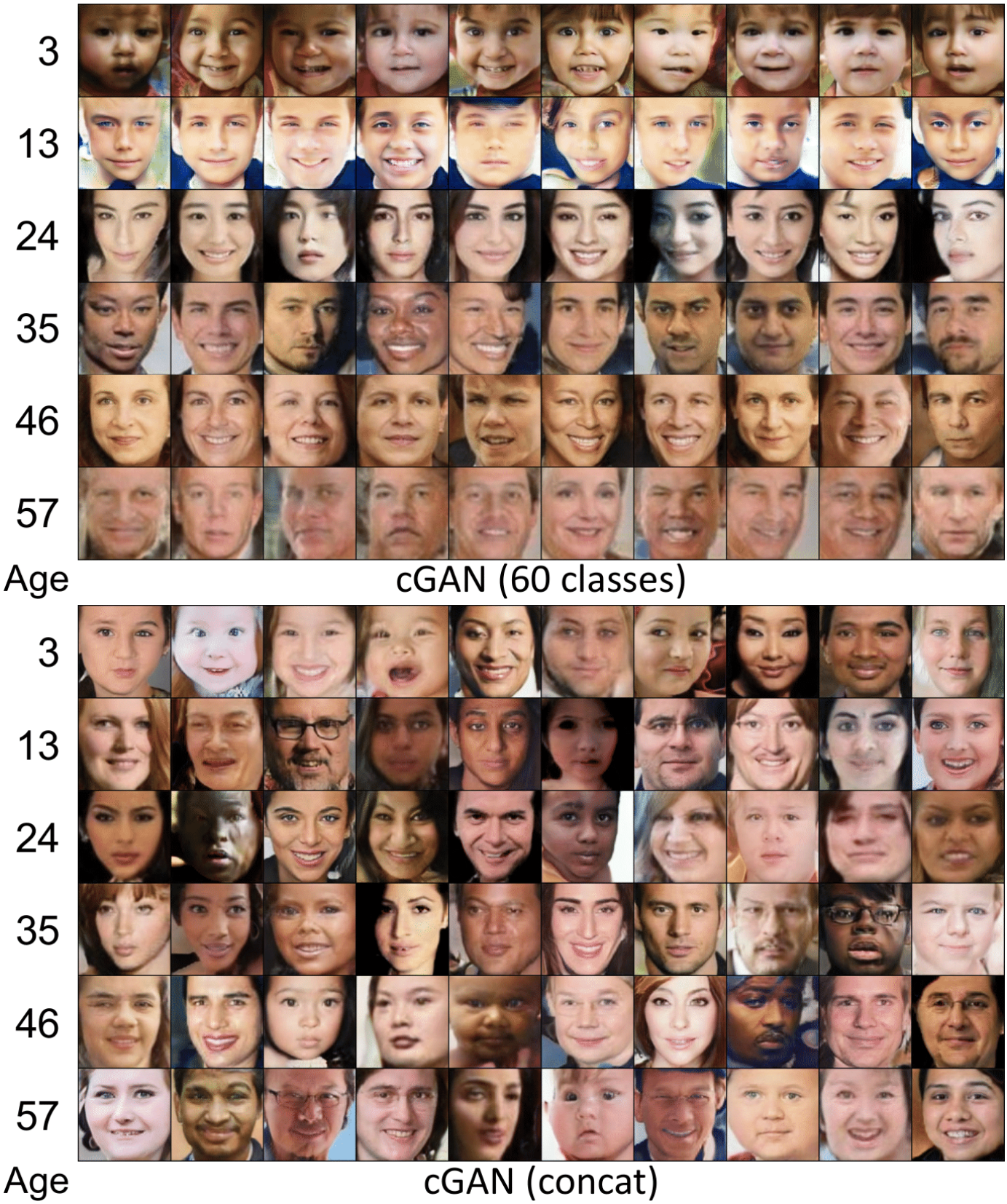

Fig. S.17.36: Some example fake UTKFace images from cGAN (60 classes) and cGAN (concat) in the 192×192 resolution. They show two types of failures of conventional cGANs. cGAN (60 classes) has high label consistency but low visual quality (e.g., last row) and low intra-label diversity (e.g., second row only has boys). cGAN (concat) has high intra-label diversity and moderate visual quality but low label consistency (e.g., first row has many adults).

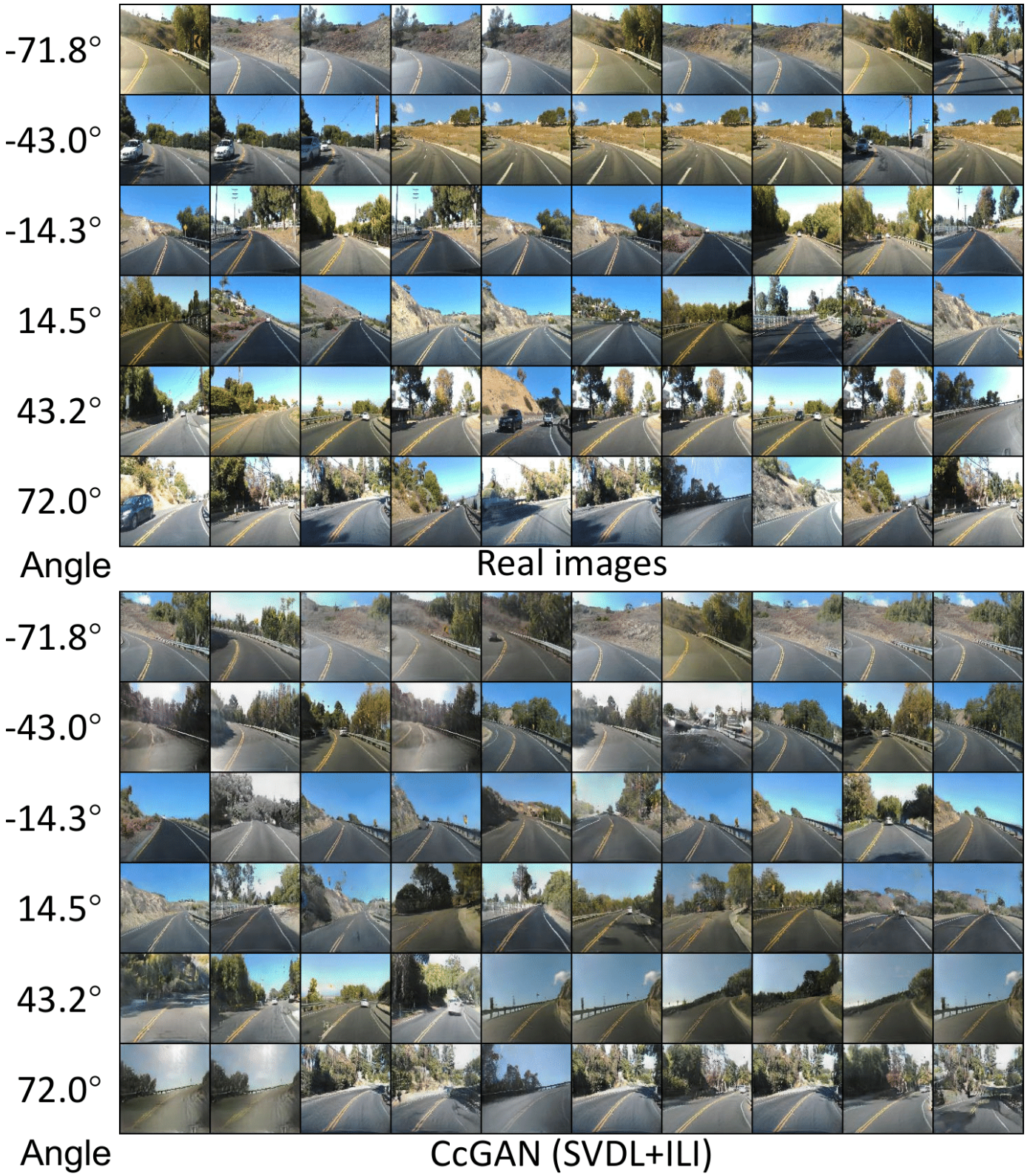

Fig. S.17.37: Some example real Steering Angle images and fake Steering Angle images from CcGAN (SVDL+ILI) in the 128×128 resolution. We can see CcGAN can generate visually realistic, diverse and label consistent images.

Fig. S.17.38: Some example fake Steering Angle images from cGAN (210 classes) and cGAN (concat) in the 128×128 resolution. They show two types of failures of conventional cGANs. cGAN (150 classes) has high label consistency but fair visual quality and low intra-label diversity. cGAN (concat) has high intra-label diversity and moderate visual quality but low label consistency.