Neurocomputing 471 (2022) 296–334

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Classify and generate: Using classification latent space representations for image generations

Saisubramaniam Gopalakrishnan [a,1], Pranshu Ranjan Singh [a,1], Yasin Yazici [a], Chuan-Sheng Foo [a,b], Vijay Chandrasekhar [a], ArulMurugan Ambikapathi [a,b,*]

[a] Institute for Infocomm Research, Agency for Science Technology and Research (A*STAR), Singapore
[b] Artificial Intelligence, Analytics And Informatics (AI3), A*STAR, Singapore

ARTICLE INFO

Article history:
Received 2 January 2021
Revised 23 September 2021
Accepted 29 October 2021
Available online 2 November 2021
Communicated by Zidong Wang

Keywords:
Classification latent space
Convex combination
Image generation

ABSTRACT

Utilization of classification latent space information for downstream reconstruction and generation is an intriguing and a relatively unexplored area. In general, discriminative representations are rich in class specific features but are too sparse for reconstruction, whereas, in autoencoders the representations are dense but has limited indistinguishable class specific features, making it less suitable for classification. In this work, we propose a discriminative modelling framework that employs manipulated supervised latent representations to reconstruct and generate new samples belonging to a given class. Unlike generative modelling approaches such as GANs and VAEs that aim to *model* the data manifold distribution, Representation based Generations (ReGene) directly *represents* the given data manifold in the classification space. Such supervised representations, under certain constraints, allow for reconstructions and controlled generations using an appropriate decoder without enforcing any prior distribution. Theoretically, given a class, we show that these representations when smartly manipulated using convex combinations retain the same class label. Furthermore, they also lead to novel generation of visually realistic images. Extensive experiments on datasets of varying resolutions demonstrate that ReGene has higher classification accuracy than existing conditional generative models while being competitive in terms of FID.

1. Introduction

Image classification is one of the major areas of computer vision that has been tremendously revolutionized by Deep Learning (DL). Since the work on AlexNet [1], there have been significant contributions like ResNet [2], DenseNet [3], InceptionNet [4], that have pushed the boundaries by increasing classification accuracy and offering robustness (to some extent), as the latent space representations of these classifiers extract a rich set of distinguishing features pertaining to each class. Ideally, such latent features contain potential information much more than required for the classification task, which can be utilized for downstream applications. While self-supervised representations are recently being investigated (SimCLR [5] and variants), other downstream applications of supervised classification latent space is still an intriguing open research.

Deep Learning based generative modelling has manifested itself as a fascinating and promising research area for image generation. Recent advancements in generative modeling include autoencoder (AE) based generative models [6–10], GAN based methods [11–15], and Flow based methods [16,17]. A plethora of algorithms developed based on these ideas (and their variants) have redefined the notion of image generations. As much as the interest for image generations have been shown towards generative models, to the best of our knowledge, revisiting discriminative approaches for the possibility of generations from classification latent space is comparatively less explored.Similar to AE and its variants being subjected to the downstream task of classification, it will be interesting to investigate the feasibility of training a classifier first and then exploiting its latent space for downstream image reconstructions and generations. Such analysis will help in better understanding of supervised latent representations, and can potentially aid in finding better robust representations.

In this regard, few natural questions arise: Can learning these supervised latent representations by a model trained exclusively for classification task be reused for another downstream task such as reconstructions, and more interestingly, for generations? Can

* Corresponding author.
E-mail address: Arulmurugan_Ambikapathi@i2r.a-star.edu.sg (A. Ambikapathi).
[1] Equal contribution.

https://doi.org/10.1016/j.neucom.2021.10.090
0925-2312/

the requirement for a latent space prior (as in VAEs/GANs) be substituted with a property extracted from classification latent space for image generations? In this work, we primarily endeavor to address the above two questions.We first begin by showing that supervised latent space representations can be reused for reconstruction using a suitable decoder with an appropriately designed loss function. From these representations for a set of image samples belonging to a given class, we generate new representations while guaranteeing that they belong to the same class. We then show how these new latent representations can be decoded to give new visually meaningful images. Hence, we propose a framework, namely Representation based Generations (ReGene), that investigates the above factors and demonstrates the feasibility of reusing classification latent space representations. Some examples of images generated (on different datasets of varying resolutions) using ReGene are summarized in Fig. 1.

The main contributions in this work are as follows: (i) We theoretically show that classifier latent space can be smartly interpolated using convex combinations, to yield new representations within the manifold. (ii) We discuss how to select good latent space representations that are sufficient for reconstruction through the design of an appropriate decoder using a combination of loss functions. (iii) Finally, we demonstrate how convex combinations of latent representations ($\mathbf{z}$) of images ($\mathbf{X}$) belonging to a class can lead to realistic and meaningful generations of new image samples belonging to the same class, using a decoder network exhibiting good generalization capability (i.e., $p(\mathbf{X}|\mathbf{z})$). The overall ReGene framework that is built based on discriminative modelling approach and capable of generating new images (from convexly combined latent space representations) belonging to a class, is depicted in Fig. 2, and the associated details are presented in the ensuing sections.

2. Background and related work

In this section, we discuss the literature pertaining to latent space representations and image generations inline with the ReGene framework.

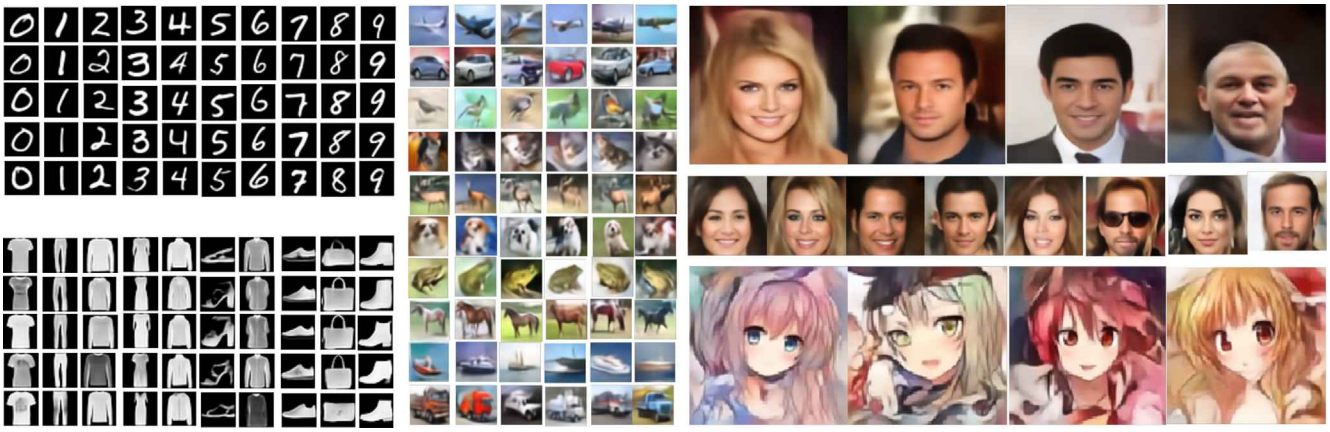


Fig. 1. ReGene image generations (proportionately scaled) for different datasets (of varying resolutions) – MNIST (28×28), Fashion MNIST (28×28), CIFAR-10 (32×32), CelebA (64×64 and 128×128), and Anime (128×128). Best viewed in color.

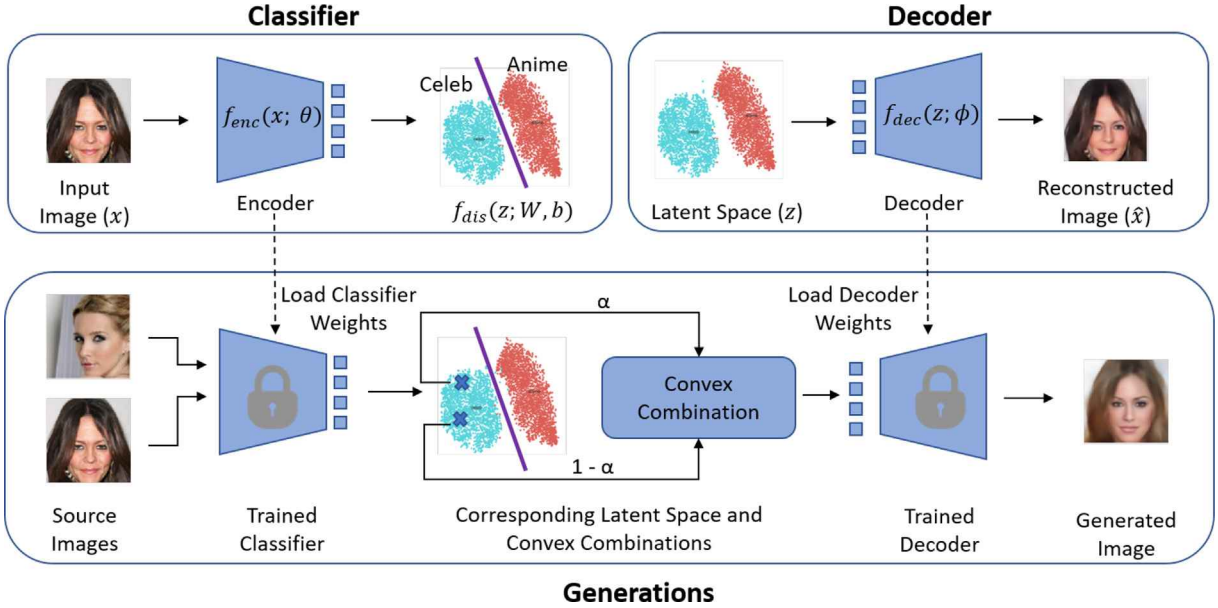


Fig. 2. ReGene Framework: Top blocks show classifier for supervised latent space representations (Details in Section 3.1) and decoder for image reconstruction (Details in Section 3.2). Bottom row depicts the image generation procedure using the trained classifier and decoder (Details in Section 3.3 to 3.4).

2.1. Latent space representations

There are few interesting works on the exploration of latent spaces for supervised (classification latent space) and unsupervised (autoencoder) tasks [18,19]. Autoencoders learn compressed latent space representations but can also produce realistic interpolated images by imposing additional constraints. For instance, [20] proposed an adversarial regularizer on the generated samples from autoencoder latent space interpolations and [21] trained adversarially on latent space interpolations. Autoencoder latent space representations can be utilized for downstream tasks by adding relevant regularizers [20,22,23].Autoencoder latent space for generations of image samples has been exploited to great extent, but to the best of our knowledge classification latent space based image generations is relatively unexplored in the computer vision community.

2.2. Generative modeling – gaining popularity for image generations

GANs and VAEs lead the list of modern generative learning approaches and have shown tremendous progress in recent years for generation of new images. Vanilla versions of both the approaches model the data distribution in an unsupervised fashion (unlabelled). VAEs enforce a prior distribution to control latent representations, whereas, GANs do not have an encoder to provide latent representations and require random noise from a prior distribution. GANs tend to suffer from issues such as mode collapse/-mode drop, training instability, etc., which have been addressed in [12,24].VAEs may produce blurry images and recent methods address these issues and have generated better quality images [10,25]. Conditional GANs ([26,27]) and Conditional VAEs ([9,28]) learn data distributions that are conditioned on class label or other images. Generating high-resolution and high-quality images is a challenge for both GANs and VAEs. BigGAN [29] and Progressive GAN [30] are recent works that address this problem. VQ-VAE, leveraging on discrete latent embedding also demonstrates comparable performance to state-of-the-art GAN models [25]. Though GANs and VAEs are the most studied generative modeling approaches, it should be emphasized that ReGene is a discriminative modeling framework and this work takes a different approach towards the possibility of representative modeling based generations.

3. Representation Based Generations (ReGene)

The prime focus of ReGene framework is to utilize classification latent space representations to generate new images belonging to a given class. To achieve this, the ReGene framework involves three parts (as shown in Fig. 2): (i) Classifier: Encoder for supervised latent space representation, (ii) Decoder: Image reconstruction and generation from the set of latent representations, and (iii) Convex analysis based manipulation of latent space representation. In this section, we derive the mathematical formulations that theoretically explains the framework's ability to capture the data distributions. Let $\boldsymbol{X} = \{\mathbf{X}^{(i)}\}_{i=1}^{m}, \boldsymbol{y} = \{y^{(i)}\}_{i=1}^{m}$ be the set of m i.i.d. data samples (images) and the corresponding class labels, respectively.

3.1. Classifier: Encoder – image space to latent space

The purpose of the encoder here is to find the appropriate latent space representations to classify the data samples according to their respective class labels (the well-known classification problem). For sake of clarity, let $f_{cls}(\boldsymbol{X}; \boldsymbol{\theta}, \mathbf{W}, \mathbf{b})$ be the classifier that maps the dataset $\boldsymbol{X}$ to its respective class labels $\boldsymbol{y}$. This classifier can be written as composition of two functions:

(i) Encoder $f_{enc}(\mathbf{X}; \boldsymbol{\theta})$, which maps the elements of dataset $\mathbf{X}$ to the corresponding latent space representation $\mathbf{z}$;

(ii) Discriminator $f_{dis}(\boldsymbol{z}; \mathbf{W}, \mathbf{b})$, which maps the set of latent space representation $\mathbf{z} = \{\mathbf{z}^{(i)}\}_{i=1}^{m}$ to the corresponding class labels in $\boldsymbol{y}$, using the respective hyperplanes characterized by $\mathbf{W}$ and $\mathbf{b}$. That is,

$$f_{cls}(\boldsymbol{X}; \boldsymbol{\theta}, \mathbf{W}, \mathbf{b}) = f_{dis}(f_{enc}(\boldsymbol{X}; \boldsymbol{\theta}); \mathbf{W}, \mathbf{b}). \tag{1}$$

Let $p(y^{(i)} \in \boldsymbol{y} | \mathbf{X}^{(i)} \in \boldsymbol{X})$ be the probability of classifying image $\mathbf{X}^{(i)}$ according to its class label $y^{(i)}$ for each data sample in $\boldsymbol{X}$. Then, the overall classifier likelihood function can be defined as

$$L(\boldsymbol{\theta}, \mathbf{W}, \mathbf{b}) = \mathop{\mathbb{E}}_{x,y \sim p_{x,y}} [\log p(y^{(i)} | \mathbf{X}^{(i)}; \boldsymbol{\theta}, \mathbf{W}, \mathbf{b})]. \tag{2}$$

The corresponding loss function is to minimize the cross-entropy error which is defined as,

$$L(\boldsymbol{\theta}, \mathbf{W}, \mathbf{b}) = -\frac{1}{m} \sum_{i=1}^{m} \sum_{c=c_1}^{c_N} \mathbb{1}\{\mathbb{c} = \mathbb{y}^{(i)}\} \log \left(\mathbb{f}_{\mathbb{cls}}(\mathbf{X}^{(i)}; \boldsymbol{\theta}, \mathbf{W}, \mathbf{b})\right)_{\mathbb{c}}, \tag{3}$$

where $c = c_1, c_2, \ldots, c_N$ denotes the N class labels.

On training the classifier using (3), the obtained optimal (in convergence sense) model parameter estimates $\boldsymbol{\theta}^*$ yield the encoder latent space representations $\mathbf{z}^{(i)*}$, which is defined as:

$$\mathbf{z}^{(i)*} = f_{enc}(\mathbf{X}^{(i)}; \boldsymbol{\theta} = \boldsymbol{\theta}^*). \tag{4}$$

These latent space representations will then be used to reconstruct/ generate images, as discussed next.

3.2. Decoder: Latent space to Image space Reconstruction

Let $f_{dec}(\boldsymbol{z} \in \mathbf{z}; \boldsymbol{\phi})$ be the decoder, which maps the optimal latent space representations $\mathbf{z}^{(i)*} \in \boldsymbol{z}$ to the respective data sample $\mathbf{X}^{(i)} \in \boldsymbol{X}$. Let $p(\mathbf{X}^{(i)} | \mathbf{z}^{(i)*})$ be the conditional probability of obtaining the sample $\mathbf{X}^{(i)}$ given the latent space representation $\mathbf{z}^{(i)*}$ via the process of generalized reconstruction. Then the overall decoder likelihood function can be defined as

$$L(\boldsymbol{\phi}) = \mathop{\mathbb{E}}_{z^*, \mathcal{X} \sim p_{z^*,x}} [\log p(\mathbf{X}^{(i)} | \mathbf{z}^{(i)*}; \boldsymbol{\phi})]. \tag{5}$$

A decoder designed by optimizing the above function will model $p(\boldsymbol{X}|\boldsymbol{z})$ and hence can be used for reconstruction and generation, provided the new $\boldsymbol{z}$ (will be discussed in subsequent sections) used for the generation still remains valid for the designed decoder. It should be emphasized that in ReGene framework a single decoder is simultaneously trained for all the classes. In other words, the decoder simply aims to invert the classification latent space representations, and hence, preserve class information which facilitates class specific image generations. Designing such a decoder is non-trivial as the obtained $\mathbf{z}$ are optimal in the classification sense but not necessarily suitable for reconstruction. Hence, the decoder architecture (detailed in Section 5.2) and appropriate loss function need to be specifically designed. In this work, we design the following loss function for decoder:

$$\begin{aligned} L(\boldsymbol{\phi}) = \frac{1}{m} \sum_{i=1}^{m} [& \lambda_1 MAE\left(f_{dec}(\mathbf{z}^{(i)*}; \boldsymbol{\phi}), \mathbf{X}^{(i)}\right) \\ & + \lambda_2 SSIM\left(f_{dec}(\mathbf{z}^{(i)*}; \boldsymbol{\phi}), \mathbf{X}^{(i)}\right) \\ & + \lambda_3 Perceptual\ Loss\left(f_{dec}(\mathbf{z}^{(i)*}; \boldsymbol{\phi}), \mathbf{X}^{(i)}\right)], \end{aligned} \tag{6}$$

where $f_{dec}(\mathbf{z}^{(i)*}; \boldsymbol{\phi})$ is the output from the decoder on passing latent space $\mathbf{z}^{(i)*}$. In (6), difference between $\mathbf{X}^{(i)}$ and $f_{dec}(\mathbf{z}^{(i)*}; \boldsymbol{\phi})$ are computed in the per pixel image space (MAE), structural similarity difference (SSIM) [31], and difference in activation of conv features at

various layers of a trained neural network (perceptual loss) [32]. λ_1, λ_2, and λ_3 are the respective weights for the different components of the loss function.

3.3. Convex combinations of latent space representations

This section focuses on analyzing the latent space representations of the samples/ images of a class. Particularly, we show via the following lemma that the classification latent space is exploitable, by demonstrating that the convex combination of two or more latent space representations of images belonging to a given class is still classifiable as the same class.

Lemma 1. *Consider a binary classification scenario. Let class A contains N_A image samples and class B contains N_B image samples. Let set $\boldsymbol{A} = \{\mathbf{a}_1, \ldots, \mathbf{a}_{n_A}\} \subset \mathbb{R}^{n_A \times n}$ contain n-dimensional feature space representation of n_A images in class A, where $2 \leqslant n_A \leqslant N_A$. And let set $\boldsymbol{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_{n_B}\} \subset \mathbb{R}^{n_B \times n}$ contain n-dimensional feature space representation of n_B images in class B, where $2 \leqslant n_B \leqslant N_B$. Given a separating hyperplane (Section 3.1) $\mathbf{W}^T\mathbf{z} + \mathbf{b} = 0$, that separates these two classes such that:*

$$\mathbf{W}^T\mathbf{z} + \mathbf{b} < 0, \ \forall \mathbf{z} \in \mathbf{A} \tag{7}$$

$$\mathbf{W}^T\mathbf{z} + \mathbf{b} > 0, \ \forall \mathbf{z} \in \mathbf{B}. \tag{8}$$

Then, it is true that

$$\mathbf{W}^T\mathbf{z} + \mathbf{b} < 0, \forall \mathbf{z} \in \text{conv}\{\mathbf{a}_1, \ldots, \mathbf{a}_{n_A}\} \tag{9}$$

$$\mathbf{W}^T\mathbf{z} + \mathbf{b} > 0, \forall \mathbf{z} \in \text{conv}\{\mathbf{b}_1, \ldots, \mathbf{b}_{n_B}\}, \tag{10}$$

where convex hull of $\{\mathbf{a}_1, \ldots, \mathbf{a}_m\} \subset \mathbb{R}^n$ is defined as

$$\text{conv}\{\mathbf{a}_1, \ldots, \mathbf{a}_m\} = \left\{ \mathbf{z} = \sum_{i=1}^{m} \theta_i \mathbf{a}_i | \mathbf{1}_m^T \boldsymbol{\theta} = 1, \boldsymbol{\theta}_i \geqslant \mathbf{0} \right\}. \tag{11}$$

Proof. Let us begin by proving (9). Let $\boldsymbol{\alpha} \in \text{conv}\{\mathbf{a}_1, \ldots, \mathbf{a}_{n_A}\}$. Then, $\mathbf{W}^T\boldsymbol{\alpha} + \mathbf{b}$ can be written as:

$$\begin{aligned}
&= \mathbf{W}^T(\theta_1\mathbf{a}_1 + \theta_2\mathbf{a}_2 + \cdots + \theta_{n_A}\mathbf{a}_{n_A}) + \mathbf{b}, \\
&\quad \text{where} \theta_i \geqslant 0, i = 1, \ldots, n_A, \mathbf{1}_{n_A}^T\boldsymbol{\theta} = 1 (\text{by}(11)) && (12)\\
&= \theta_1\mathbf{W}^T\mathbf{a}_1 + \theta_2\mathbf{W}^T\mathbf{a}_2 + \cdots + \theta_{n_A}\mathbf{W}^T\mathbf{a}_{n_A} \\
&\quad + \mathbf{b}\mathbf{1}_{n_A}^T\boldsymbol{\theta}, (\text{as}\mathbf{1}_{n_A}^T\boldsymbol{\theta} = 1) && (13)\\
&= \theta_1\mathbf{W}^T\mathbf{a}_1 + \theta_2\mathbf{W}^T\mathbf{a}_2 + \cdots + \theta_{n_A}\mathbf{W}^T\mathbf{a}_{n_A} \\
&\quad + (\theta_1\mathbf{b} + \theta_2\mathbf{b} + \cdots + \theta_{n_A}\mathbf{b}) && (14)\\
&= \theta_1(\mathbf{W}^T\mathbf{a}_1 + \mathbf{b}) + \theta_2(\mathbf{W}^T\mathbf{a}_2 + \mathbf{b}) + \cdots \\
&\quad + \theta_{n_A}(\mathbf{W}^T\mathbf{a}_{n_A} + \mathbf{b}) && (15)\\
&< 0 (\text{by } (7)) && (16)
\end{aligned}$$

Following similar steps, (10) can also be proved. □

Now, let us consider the generalization of Lemma 1 for an N-class classification scenario with class labels $c_1, \ldots, c_N$. As we are interested in generating new latent space representations (and subsequently new image generations) of a given class (say $c_i, i \in \{1, \ldots, N\}$ with latent representations $C_i = \{\mathbf{a}_1, \ldots, \mathbf{a}_{n_{C_i}}\}$ belonging to class c_i), the multi-class classification scenario can be considered as class c_i vs other classes. In other words, (7) now becomes:

$$\mathbf{W}^T\mathbf{z} + \mathbf{b} < 0, \forall \mathbf{z} \in C_i \tag{17}$$

$$\mathbf{W}^T\mathbf{z} + \mathbf{b} > 0, \forall \mathbf{z} \notin C_i. \tag{18}$$

Then, following the proof of Lemma 1, it is straight-forward to show that

$$\mathbf{W}^T\mathbf{z} + \mathbf{b} < 0, \forall \mathbf{z} \in \text{conv}\{\mathbf{a}_1, \ldots, \mathbf{a}_{n_{C_i}}\}. \tag{19}$$

3.4. Generations in image space

Consequently, as the decoder is trained to capture the distribution of $\boldsymbol{X}|\boldsymbol{z}$ (i.e., $p(\boldsymbol{X}|\boldsymbol{z})$) through a finite set of latent space representations $\mathbf{z}$, it can be shown that visually meaningful new images can be generated for each and every element belonging to the convex hull of the set $\boldsymbol{z}' \subseteq \boldsymbol{z}$, where $\mathbf{z}'$ is any set of latent space representations of a given class of interest. The generations in image space are obtained by performing the following two steps:

(i) Obtain generations in latent space for a given class via convex combinations of any $n \geqslant 2$, number of latent space representations (defined by set $\boldsymbol{z}'$), belonging to that class.

(ii) Trigger the trained decoder with the newly obtained latent space representations to get the corresponding image space generations.

The latent space representations obtained from step (i) lie within the given class (see Lemma 1). Unlike VAE, we do not enforce a prior on latent space to obtain random generations. Alternatively, we achieve our generations by varying the number of latent space representations in set $\boldsymbol{z}'$ (that belongs to a class of interest), that are used in the generation process with different convex combination ratios. Theoretically, we can generate *infinitely many* image samples using any value lying in $[0, 1]$ for convex combination ratio. But, to achieve maximum diversity in the generations, we use (in the experiments) the convex combination ratio to be $1/n$ (uniform weightage for each latent space representation) for the selected n samples from $\boldsymbol{z}'$. Let N be the number of samples in $\boldsymbol{z}'$, then we can generate a total of $2^N - N - 1$ samples, considering each possible n ($n = 2, 3, \ldots, N$) sample combinations.

The validity of image space generation obtained by the decoder in step (ii) is presented in Theorem 2. From [33] we have the following Theorem 1, which will be used in the proof of Theorem 2.

Theorem 1. *Given two random variables α_1 and α_2, with probability density functions $\rho_1(x)$ and $\rho_2(x)$ respectively, the probability density function $\rho(x)$ of the mixture obtained by choosing α_1 with probability w and α_2 with remaining probability $1 - w$ is a convex combination of probability density functions of original random variables, i.e.*

$$\rho(x) = f(\rho_1(x), \rho_2(x)) = w.\rho_1(x) + (1 - w).\rho_2(x). \tag{20}$$

Theorem 2. *For an ideal decoder, the convex combinations of latent space representations per class ($\boldsymbol{z}' \subseteq \boldsymbol{z}$) yields image representations belonging to the original image space distribution.*

Proof. Let the classification latent space obtained be $\mathbf{z}^{(i)*}$ (From 4),

$$\mathbf{z}^{(i)*} = f_{enc}(\mathbf{X}^{(i)}; \boldsymbol{\theta} = \boldsymbol{\theta}^*). \tag{21}$$

Let $f_{dec}(\mathbf{z}^{(i)*}; \phi)$ be the decoder, that learns a mapping from the latent space distribution to image space distribution and $\mathbf{z}^*|y_c$ be the latent space distribution for class y_c.

The latent representations per class,

$$\mathbf{z}^{(i)*}|y_c \sim p(\mathbf{z}|y_c) \tag{22}$$

For the mixture of two latent representations, $\mathbf{z}^{(i)*}|y_c$ and $\mathbf{z}^{(j)*}|y_c$ from a given class y_c chosen with probabilities α and $1 - \alpha$ respectively, the probability density is given by,

$$f(p(\mathbf{z}^{(i)*}|y_c), p(\mathbf{z}^{(j)*}|y_c) = \alpha.p(\mathbf{z}|y_c) + (1-\alpha).p(\mathbf{z}|y_c) \quad (\text{by Theorem 1}) \quad = p(\mathbf{z}|y_c) \tag{23}$$

As the mixture of two latent representations for a given class preserves the latent space distribution for that class (Lemma 1), the decoder, $f_{dec}(\mathbf{z}^{(i)*}; \phi)$ is able to map this latent space representation to a sample in the image space. Since the output of decoder belongs to the original image space distribution, the newly generated image sample also belongs to the original image space distribution. □

4. ReGene vs AutoEncoder: Case study with MNIST

This section provides a toy-study comparison between the Autoencoder (AE) and ReGene in terms of latent space representations' ability to classify, reconstruct and generate (using convex combinations, refer Sections 3.3 and 3.4). More rigorous analysis on complex datasets will be discussed in subsequent sections.

On training with MNIST data set, the ReGene classifier achieves 99% test accuracy whereas the AE achieves 95% (using a linear layer added on top AE latent space). As seen in Fig. 3(a), the classifier latent space is well-separated as compared to that of the autoencoder latent space. The test reconstruction Mean Absolute Error (MAE) for AE and ReGene decoder are 0.0134 and 0.0273, respectively. In other words, using the classification latent space, ReGene decoder is still able to achieve good reconstructions but with slightly higher MAE as compared to AE.The AE is trained (from scratch) end-end to reconstruct the given sample, using MAE as loss function. Since the latent space $\mathbf{z}$ does not have additional constraint (to be classifiable), the decoder learns an arbitrary mapping from $\boldsymbol{z}$ to $\boldsymbol{X}$ that minimizes MAE. Hence, it cannot and is not expected to learn class specific features. On the other hand, ReGene's decoder uses the $\mathbf{z}$ from a trained classifier model, (learnt using cross entropy loss function) and therefore has higher classification test accuracy, but also incurs slightly higher MAE (while reconstructing), as the latent space is designed for classification and not constrained for reconstruction. There is a trade-off between reconstruction and classification due to which there is a slightly higher MAE when reconstructing from classification latent space (ReGene) versus an AE. This is further explained in detail in Table 1 and in Section 5.2.1. Fig. 3(b) shows the generation in image space using Autoencoder and ReGene decoder. For Autoencoder it is observed in Fig. 3(a) that convex combination of samples in a given class do not always belong to the same class. For instance, a line (convex combination) between two points (lying on two different class '4' clusters) belonging to the class 4 (green) passes through the class 9. Therefore, the generated images based on such combinations cannot be guaranteed to be realistic and meaningful. One such illustration is shown in Fig. 3(b) (top) where the transition can be seen blurry as it moves from left (a given true sample) to right (another true sample). On the other hand, for the same two true samples, Fig. 3(b) (bottom) shows the classification latent space based transition, where one can observe a smooth transition within intermediate samples still preserving the class information (in this case "4"). This illustrates that unlike AE latent space, the classifier space is apt for generating new meaningful samples that still belong to the same class of interest.

5. Experimental setup

This section provides validation for the formulation and theory behind the framework. We discuss the experimental procedures including choice of datasets considered (of varying resolutions), and provide a high level overview of the classifier and decoder network architectures employed here for analyzing ReGene framework. We also discuss the evaluation metrics for performance comparison with state-of-the-art methods for image generation.

5.1. Datasets

Four standard datasets for image generation tasks – MNIST [34], Fashion MNIST [35], CIFAR-10 [36] and CelebA [37] are selected. MNIST and Fashion MNIST are relatively easier due to smaller image dimensions (28x28 px), grayscale images, perfect alignment, and absence of background. CelebA is chosen to demonstrate the generation of real-world, relatively higher-dimensional color images. To be consistent with other reported results in the literature, we followed the same pre-processing steps provided in [7,8] by taking 140×140 center crops and resizing the image to 64×64 px resolution.[2] To obtain latent space representation for CelebA, we introduce Anime [38] dataset as the complementary class. 15K random samples from both CelebA and Anime were chosen for training the classifier. For the decoder, we trained CelebA separately using the entirety of 200K samples. Additionally, we also trained CelebA and Anime for 128×128 px resolution without additional cropping other than provided in the original aligned dataset. CIFAR-10 (32×32 px) is an interesting dataset of choice since it presents a multiclass scenario with high intraclass variations and unaligned images with mixed backgrounds. For quantitative evaluation on the datasets, we directly state the respective numerical values of the standard evaluation metrics (as reported in the respective literature), to have fair comparisons with other existing generative approaches.

5.2. Network architectures

It should be noted that the prime focus of this work is on the theory and principle of the representation framework that suits the downstream task of generation. Here, we demonstrate our approach using simple feed-forward convolutional neural networks. Both classifier and decoder networks follow VGG-style architectures of varying depths and multiple blocks, with each major block having Conv2D for classifier/Conv2DTranspose for decoder, with BatchNorm, LeakyReLU, followed by a MaxPooling for classifier/UpSampling for decoder layer. Cross-entropy is adopted as the loss function for classification since it aims to maximize the capture of information sufficient enough to separate the classes (as reported in (3)). Deriving from information theory, minimizing entropy maximizes (class relevant) mutual information, which in turn captures a strong $\boldsymbol{z}$ and makes decoding to $\boldsymbol{X}$ possible. Other loss functions (e.g. hinge) for complex datasets (e.g. cifar10) are not as effective in terms of accuracy as they have overlapping latent space boundaries. Also, there is no theoretical guarantee to state that information in $\boldsymbol{z}$ using other loss functions has the necessary latent feature representations that can positively or negatively influence the reconstruction of $\boldsymbol{X}$. For the decoder, we employ a combination of three weighted loss functions (as reported in (6)): Mean Absolute Error (MAE), Structural Similarity Index (SSIM) [31] and Perceptual Loss [32]. The ratio of weights λ_1, λ_2 and λ_3 are dataset dependent; however the general heuristic is to allow initial training epochs to have high weights assigned to SSIM (λ_2) and Perceptual Loss (λ_3) to first reconstruct the global outline and texture of the image. In later epochs, those weights are gradually reduced to focus more on pixel wise error (λ_1). The MAE is weighted relatively high throughout the training Network weights are optimized using Adam optimizer. All experiments

[2] It should be noted that existing works in literature have used different versions of CelebA dataset with respect to pre-processing. Additional details pertaining to this are provided in Appendix B.

were run on a workstation with the following specifications: 64 GB RAM, Intel i9-9900 K CPU and 2x NVIDIA GeForce RTX 2080 Ti. Detailed network architectures for both classifier and decoder, training details such as learning rates, batch sizes, number of epochs, empirically chosen weights for λ_1, λ_2 and λ_3 during decoder training, time taken per epoch, etc. are provided in Appendix A.

5.2.1. Architecture selection for a good latent feature representation – Case study with CIFAR-10

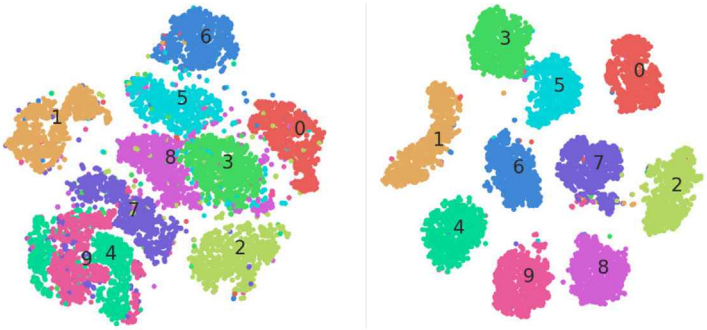
For a complex dataset with high intra-class background variance, learning proper class specific features becomes difficult. Taking CIFAR-10 as an example, we examine how to achieve trade-off between good classification and reconstruction. Discriminative classification models often yield highly sparse latent representations that sacrifice reconstruction and generation quality, implying even a decoder with sufficient capacity may under perform if it does not receive an input with sufficient information. This further implies that the best state-of-the-art classifier cannot be directly considered for ReGene framework as it would impose significant loss in reconstruction, and generation quality. With the intention of finding good latent space representations that achieve a balance between classification and reconstruction/generation, a hyper-parameter search was conducted for a suitable architecture in terms of depth (layers), latent space dimension size and relative non-sparsity in the latent feature space. Fig. 4 shows a comparison between different classifier architectures with their t-SNE plot for viewing the extent of class separation boundaries, and latent space representations per class (represented by the bar code). It can be observed that although architectures with more layers achieve higher classification accuracy and the t-SNE appears to be more compact, the latent representations (barcodes) gradually appears sparser (Fig. 4). With such sparser latent representations, the decoder training becomes more difficult, due to the limited information available for reconstruction.

We also experimented on DenseNet100 trained for the same task, which gave accuracy of 91%. Visual results in Fig. 5 show that the latent space of DenseNet is visibly sparse, except in certain regions, sufficient enough for the classifier to make decisions. However, image reconstructions only cover faint outline of the object of interest.

It is observed that achieving sparsity in the latent representation of autoencoder helps in learning useful features as a byproduct [19]. Indirectly, introducing methods towards sparsity (e.g. by enforcing regularization), makes the network learn class relevant properties. [20] enforced a regularizer and evaluated their improved latent space representation for downstream tasks such as classification. On the other hand, since we adopt a classifier trained via discriminative modelling in the first step, learning class relevant features and in-turn sparseness are already incorporated; therefore we focus on preventing over-sparsity while ensuring suf-

Table 1
Comparison of different network architectures for selection of a CIFAR-10 classifier latent space representations capable of reconstruction. Metrics used for selection: classifier accuracy and decoder MAE (Mean Absolute Error) computed on test set.

Network (Major Blocks)	Latent Space	Convolutions per Block	Classifier Accuracy	Decoder MAE
4	2048	1	0.772	**0.079**
		3	0.795	0.135
5	512	1	0.776	0.135
		3	0.808	0.182
6	128	1	0.779	0.169
		3	**0.821**	0.209



(a) t-SNE plot showing MNIST latent representations obtained from a trained Autoencoder (left) and ReGene Classifier (right). The numbers in the clusters indicate the class labels of MNIST.

(b) Image generation via linear interpolation of two latent space representations of MNIST digit 4 using a trained Autoencoder (top) and ReGene Classifier (bottom).

Fig. 3. MNIST Classification and Generation: Autoencoder vs ReGene.

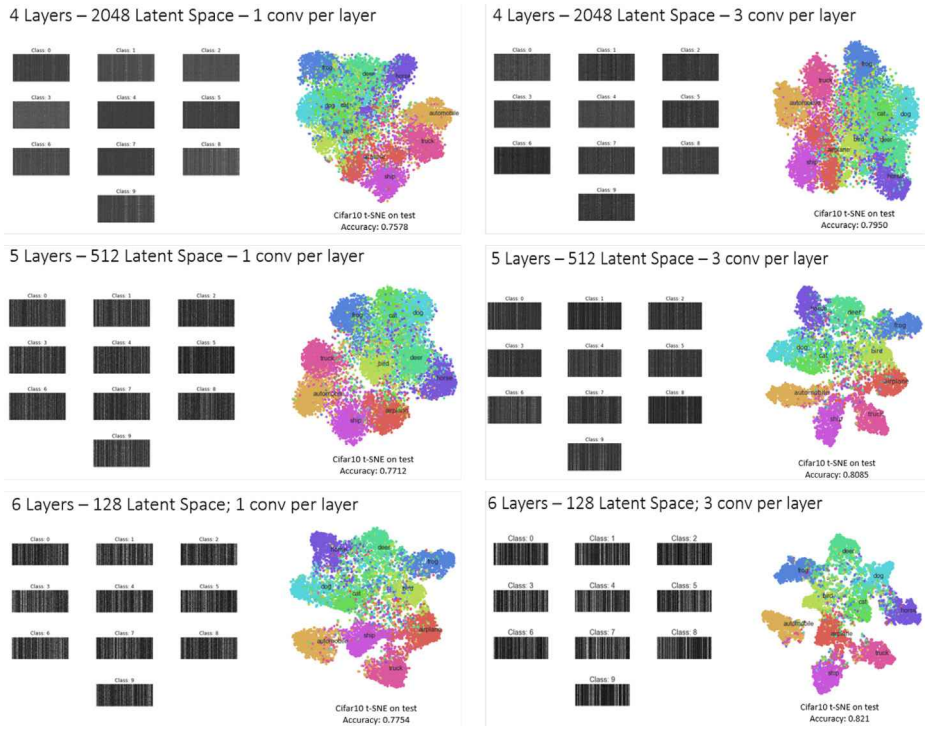


Fig. 4. Visualizing the latent space (using barcodes for each class) and t-SNE plots of CIFAR-10 for 6 classifier architectures (of varying depth of network, size of latent representations and number of convolutional layers per block). Classification accuracy on the test set is also shown below the respective t-SNE plots.

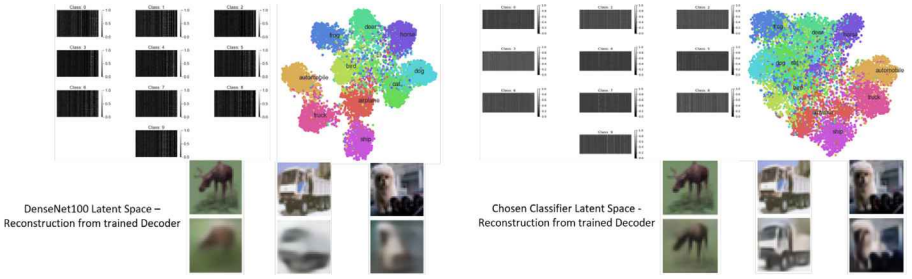


Fig. 5. Comparison of DenseNet100 (left) vs the chosen ReGene classifier (right) – latent space sparseness (top-left) and inability of decoder to reconstruct back the original image is observed in DenseNet (bottom-left). Whereas for the chosen classifier, the latent representations are comparatively less sparse (top-right) and ReGene decoder is able to reconstruct (bottom-right). Best viewed in color.

ficient class relevant features are captured. In general, a network with more layers (network blocks) favours better classification at the expense of increased sparsity. A decoder that receives such a sparse vector as input may be incapacitated for reconstruction of the original image, and hence, image generation, despite deep network architectures. This is further shown in Table 1 where decoder reconstruction MAE on the test set increase with increase in network blocks. The classification network architecture having latent space 2048 and single convolution per layer was empirically chosen as it achieves the best decoder MAE.[3] Similar approach was adopted to find the right choice of classifier and decoder combination for other datasets (presented in Appendix A).

5.3. Evaluation metrics

To quantitatively evaluate the generations, we report their Classification Accuracy and Fréchet Inception Distance (FID) [39]. The classification accuracy validates the claim from Lemma 1 from the perspective of a neutral classifier (ResNet56v1 architecture), trained from scratch on the original dataset corresponding to the generations. FID is the standard evaluation measure predominantly used when comparing the performance of image generative methods. It should also be noted that FID is susceptible to the decoder noise, which is further discussed in Section 6.6. Also, for qualitative evaluations, we compare generations with nearest neighbors in the original training image space, their reconstructions, and with other generations created.

6. Results and discussion

In this section, extensive results for validation of Lemma 1 and generations from multiple combinations are presented (for various datasets). Additionally, comparison of generations with the closest match in the dataset using different distance measures, and the possibility of iterative generations are also discussed. Finally, we compare the FID scores between reconstructions and generations and show the presence of low level decoder noise can impact the FID score while still generating visually meaningful images.

[3] Finding an optimal classifier/decoder architecture is in itself a evolving research direction for any Deep Learning application.

Table 2
Trials taken by ReGene to process 5000 image generations (500 generations per class) using convex combination (cc) of 2, for MNIST, Fashion MNIST and Cifar10 datasets. Values denote number of combination trials needed to be processed for 500 valid generations per class, first pass being Latent Space (LS) validation, followed by Image Space (IS) validation.

Class Label	MNIST LS 500	MNIST IS 500	FashionLS 500	Fashion IS 500	Cifar10 LS 500	Cifar10 IS 500
0	500	501	504	564	502	830
1	502	507	500	500	501	730
2	500	521	501	538	508	929
3	504	520	501	608	512	681
4	501	518	526	808	506	901
5	500	513	506	590	517	895
6	500	502	552	847	502	1347
7	501	505	500	607	504	780
8	500	531	500	531	501	563
9	503	548	500	510	501	1390

Dataset	Original Test Accuracy	Convex Comb	Generation Accuracy
MNIST	0.9867	2	$0.9961 \pm 3.51e-5$
		3	$0.9984 \pm 1.23e-5$
		5	$0.9996 \pm 5.77e-5$
Fashion	0.9037	2	0.9766 ± 0.008
		3	0.9846 ± 0.009
		5	0.9853 ± 0.008
CIFAR-10	0.8953	2	0.9137 ± 0.006
		3	0.9151 ± 0.002
		5	0.9177 ± 0.007
CelebA	0.9998	2	0.9968 ± 0.003
		3	0.9972 ± 0.003
		5	0.9985 ± 0.0005

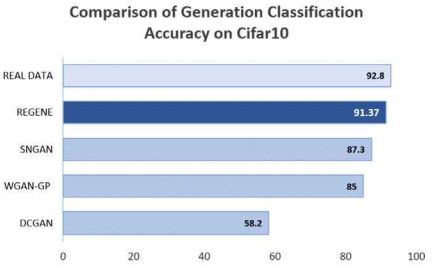


Fig. 6. (a) Generation Classification Accuracy on Neutral Classifier for different dataset generations with convex combinations – 2, 3 and 5. Results reported are averaged over 3 runs (left). (b) Comparison on Cifar10 dataset. ReGene outperforms other conditional generative approaches in terms of generation classification accuracy on Cifar10. Scores for baseline GANs are reported from [40] (right).

6.1. Practical validation of Lemma 1

The experimental validity of Lemma 1 can be inferred from the Table in left provided in Fig. 6(a), wherein the generations obtained using convex combination of 2, 3, and 5 are passed through a neutral classifier ResNet56v1 trained on the respective original datasets. Note that the classification accuracies are all high ($\geqslant$0.9). It is observed that convex combinations of higher order (larger n) tend to provide generations with higher classification accuracy. We take the example of Cifar-10 dataset in Fig. 6(b) and compare the classification accuracy score of ReGene with baseline conditional generative models, such as DCGAN, WGANGP, and SNGAN. In comparison to other conditional generative models, the generations of ReGene are classified better. Further, as compared to real test data (92.8%), ReGene generations are almost equally classifiable by a neutral classifier (91.37%). This demonstrates the gener-

Fig. 7. Left: Generations (highlighted in the blue square) produced from convex combination of different samples – n = 2,3,5 with combination ratio of 1/n. Right: Latent Space Interpolation transitions between two source images (top and bottom of each column) of digit 0, bag, anime and celeb, shown vertically for varying convex combination ratios (0.1 to 0.9) (All images are shown in same scale due to space constraints though they have different resolutions). Best viewed in color.

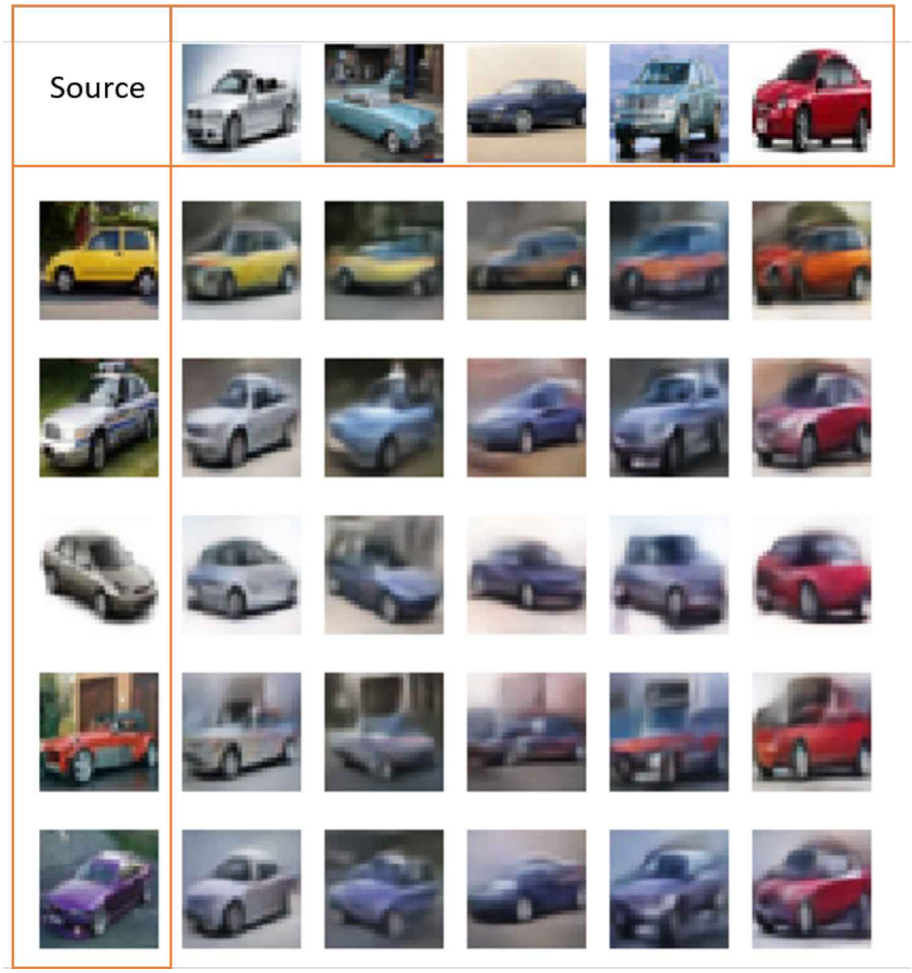


Fig. 8. Cifar10 Car generations displayed in Matrix format. Orange rectangular box specifies the source images. Best viewed in color.

ative efficacy of ReGene from the eyes of a neutral classifier. Although a perfect validation for Theorem 2 can be obtained only from an ideal (error free) decoder, the comparative scores of FID (reported and discussed in Section 6.6) help to substantiate that the decoder indeed closely learns the mapping from latent space to image space distribution and that the generations are not far from the original image space distribution.

6.2. Generations via convex combinations

Novel samples belonging to any class can be generated using the convex combination of multiple existing samples randomly chosen from that class, by combining ($n \geqslant 2$) samples in the latent space. Theoretically, we can have multiple samples participating in the generation process. For practical purposes, due to inevitable classification and decoder training errors, we apply a threshold that serves as a filtering criteria to select most meaningful generations. We apply a two-step judging criteria to decide whether a sample from the convex combination is fit to be considered as a new generation: (i) On obtaining the new latent space representation, we pass it to the trained ReGene classifier portion (softmax layer) and only allow samples that are correctly classified, (ii) Such sample representations once decoded to image (via decoder) are again passed to the ReGene classifier (as an image) to double check whether the new image has class confidence score above a certain threshold. General threshold adopted is the average class-confidence per class over the test/holdout samples. For generating 5,000 valid samples using the average class-confidence threshold, ReGene processed 5,166 combinations for MNIST, 6,103 combinations for Fashion MNIST, and 9046 combinations for Cifar10. Class-wise combination trials are provided in Table 2. It is observed that the Latent Space verification is close to 500 (per class), indicating that almost all samples are indeed within the convex hull of the same class it is being generated for (Lemma 1), whereas the Image Space verification takes higher number of samples, owing to reconstruction error in the decoder.

In Fig. 7, we show generations from combinations n = 2,3,5 and the convex combination ratio = 1/n, for each selected latent representation. Each row presents generations (highlighted in the blue square) from datasets selected in Section 5.1. Below each of the generated images (highlighted blue square) we have shown the spatial overlapping of the source images to visually observe the novelty and quality of the generated samples. Each column shows samples from different number of samples participating in the combinations. The generations - Digit 0, Bag, Anime, and Celeb transitions are produced by changing the convex combination ratios (0.1 to 0.9) in the latent space. The images reveal the generative ability of ReGene (for different value of n and in various datasets) and the smooth image transitions (for $n = 2$). Collage of each dataset is visualized in Appendix C.

6.3. Matrix visualizations

Generations from the convex combination of 2 samples are presented in a matrix structure for visual study of the subtle impact introduced by the nature of the different images. Examples shown

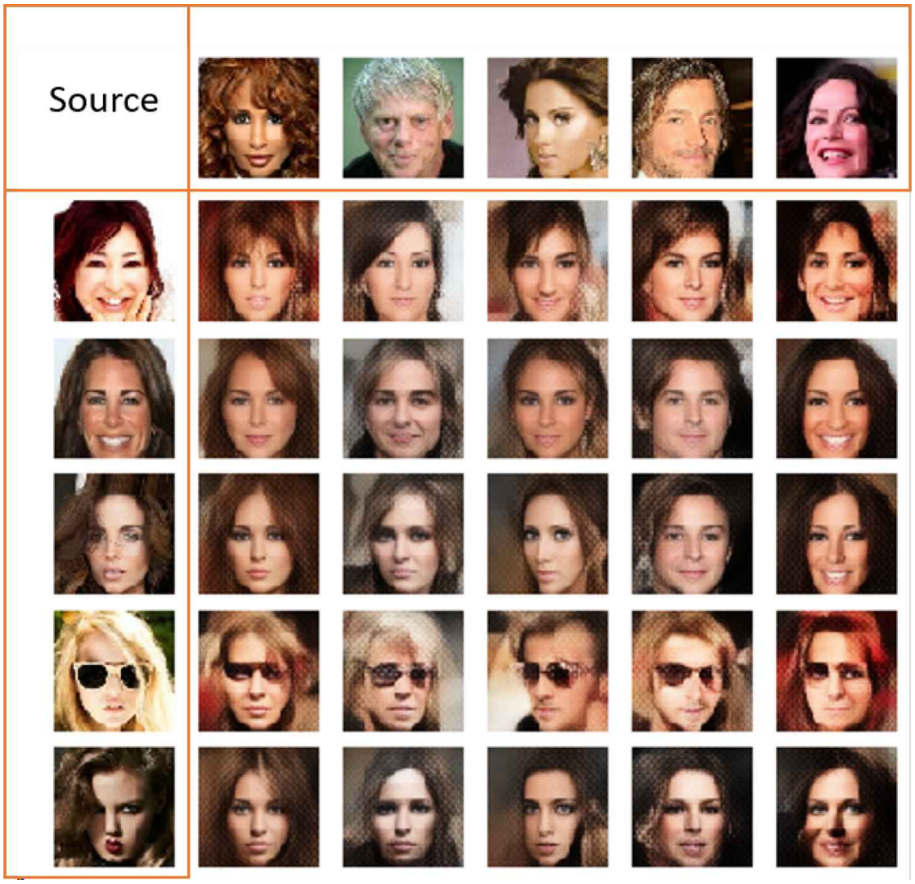


Fig. 9. CelebA Face (64×64 px) generations displayed in Matrix format. Orange rectangular box specifies the source images. Best viewed in color.

include Cifar10 (32 × 32 px) in Fig. 8, CelebA in Fig. 9 (64 × 64 px) and Fig. 10 (128 × 128 px), and Anime in Fig. 11. Considering one row (one image), one can observe the different changes the new generation undergoes by combining with another image of different property (color, expression, shape, background etc.)

6.4. Finding the closest match from the training data and among generations

To demonstrate that the decoder does not produce generations that simply mimic existing samples, and to show it is truly robust to mode collapse, we show qualitative samples comparing generations with samples from the training set in Fig. 12. We take two existing images, generate a new image and compare it against other existing images from the training dataset. Comparison to the closest match is done in three ways: (i) in the increasing order of squared error in image and latent space representations, (ii) in the decreasing order of SSIM in image space, and (iii) in the increasing order of squared error on latent features passed through Imagenet pre-trained VGG-16 network. We also compare the closest match with our own created generations set in Fig. 13, showing the generations from convex combination are truly unique, and the property of mode collapse can be avoided. Additional comparisons are available in Appendix D.

6.5. Generations arising across generations

It is also possible to create infinite generations, by iterative combinations of the generated images as inputs and feeding them back to the framework. Doing so allows for subsequent levels of generation, each level dependent on the generation from one level before. In Fig. 14a and b, 'Gen-1' refers to the first set of generations. Two newer ways for generation are: (i) applying convex combination of Gen-1 latent features in latent space (LS) directly (termed sibling generations' in 'Gen-2:fromLS'), or, (ii) by passing the Gen-1 images in image space (IS) back to classifier and obtaining new generations from convex combination of their respective latent features (termed 'child generations' in 'Gen-2:fromRecon'). The slight deterioration in the quality of newer generations is due to decoder not being 100% error-free, as the low level noise can be propagated with each subsequent level of generation.

6.6. Comparing FID scores of reconstruction vs generation

The decoder of ReGene serves for both reconstruction and generation. Unlike GANs/Conditional GANs, ReGene does not employ adversarial training. The FID scores obtained using ReGene generations for different datasets, and those obtained by different generative approaches (as reported in the literature), are summarized in Table 3. It is important to note that the FID scores computed between Org vs Recon are better than other AE based methods, but still relatively high due to low level decoder noise. This does not affect generation qualitatively (as can be witnessed from images in Fig. 7), but it impacts FID score significantly. As FID score is susceptible to such small decoder noise, it favor methods trained in adversarial fashion. The FID scores between Recon vs Gen provide a better comparison since they both (reconstruction and generation from same decoder) take into account the same decoder noise. Similar observation has also been reported in [25]. Fig. 15 (a) compares the generative performance exclusively for Cifar-10 with other conditional GAN methods, where the FID score for Recon vs Gen is close to SoTA methods trained exclusively for gen-

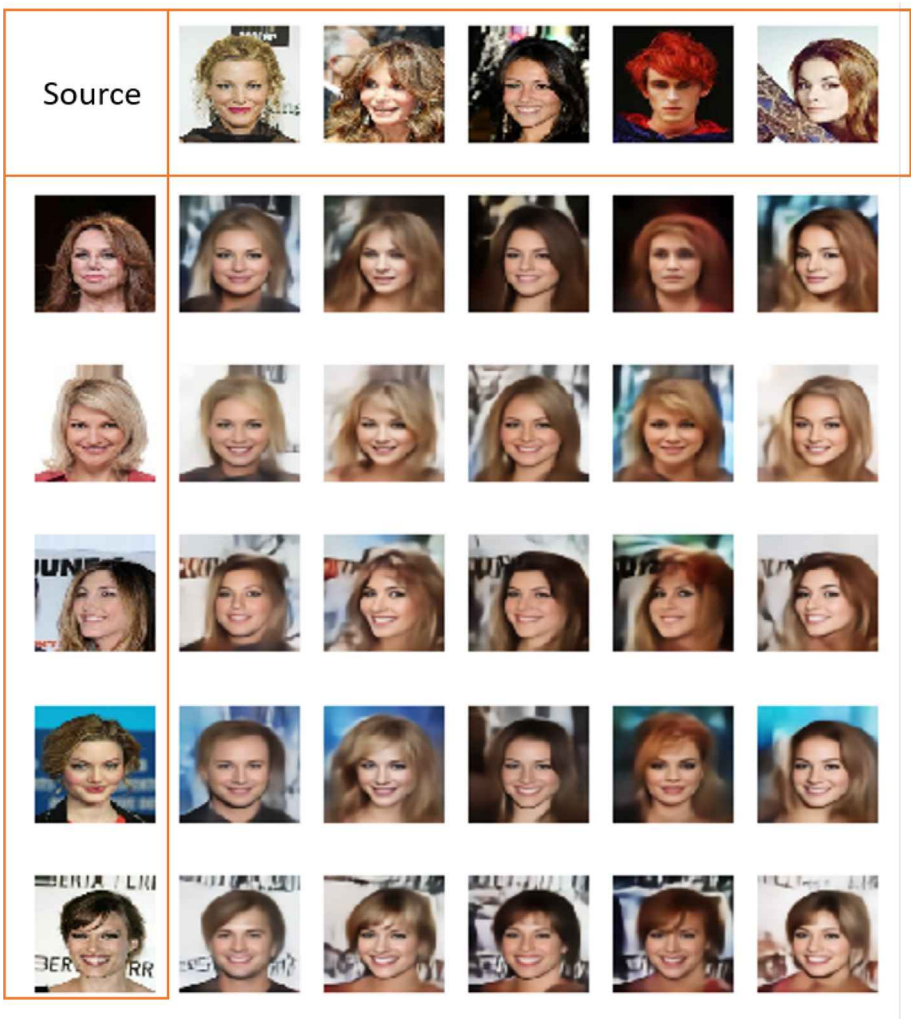


Fig. 10. CelebA Face (128×128 px) generations displayed in Matrix format. Orange rectangular box specifies the source images. Best viewed in color.

eration. In Fig. 15(b), we plot FID score vs accuracy for different convex combination ratios of original-reconstructed versions of the Cifar-10 samples. Initially, when all original training images are presented, the accuracy reaches 1 and FID is 0. As convex combination weight α of the reconstructed images increase, there is a gradual deterioration in the image quality, which decreases the accuracy and conversely increases FID. This further confirms that the error in decoder has significant influence on the two metrics (FID and accuracy). Though the purpose here is to demonstrate the generation ability of ReGene framework, it should be noted that with an improved decoder (implying lower FID for Org vs Recon), ReGene has the potential to generate images with even lower FID scores (for Org vs Gen). Though FID is predominantly used as a standard evaluation metric, earlier works also used Inception Score (IS). Since existing VAE [41] based literature reported the IS values only for MNIST and Fashion MNIST dataset, for a fair comparison we have compared those with the IS scores obtained by ReGene, as summarized in Table 4. Note that a higher IS score is an indication of better image generation.

6.7. Comparison of different loss functions for classifier

To compare the performance of classifier and its downstream effect on the generations, we study four different classification loss functions - cross-entropy (default), multi-class hinge, cosine similarity, and mean squared error. We analyze the effect of each loss function over different metrics, shown in Fig. 16. In terms of classifier accuracy, cross-entropy outperforms other loss functions. There is also a slight degradation in decoder MAE when using the latent space obtained from classifiers that are trained using other loss functions, compared to the cross-entropy based classifier. This in turn affects generation FID score, and the accuracy of those generations when passed through the trained classifier. With strict constraints/-judging criteria added on latent and image space (Section 6.2), it becomes harder to find quality generations for other loss functions, whereas cross-entropy clearly has better performance. With low FID score and 10K tries, it yields constraint satisfying generations with fewer tries when compared with hinge (4x), MSE (2.9×) and cosine (1.8×). Those generations also have low FID, and are classified correctly when passed through the trained classifier.

Additional details on the latent space representations of each loss function visualized by t-SNE and activation bar plots are available in Appendix B.

6.8. Classification performance on varying the % of training samples

This section studies the impact of varying different % of generations on the classifier's performance. Considering Cifar10 dataset for the experiment, the performance on varying different sample sizes for training classifier is plotted in Fig. 17. From the extreme left (indicated by red dot) with 100% generated images (no original samples), we analyse the classification accuracy by gradually

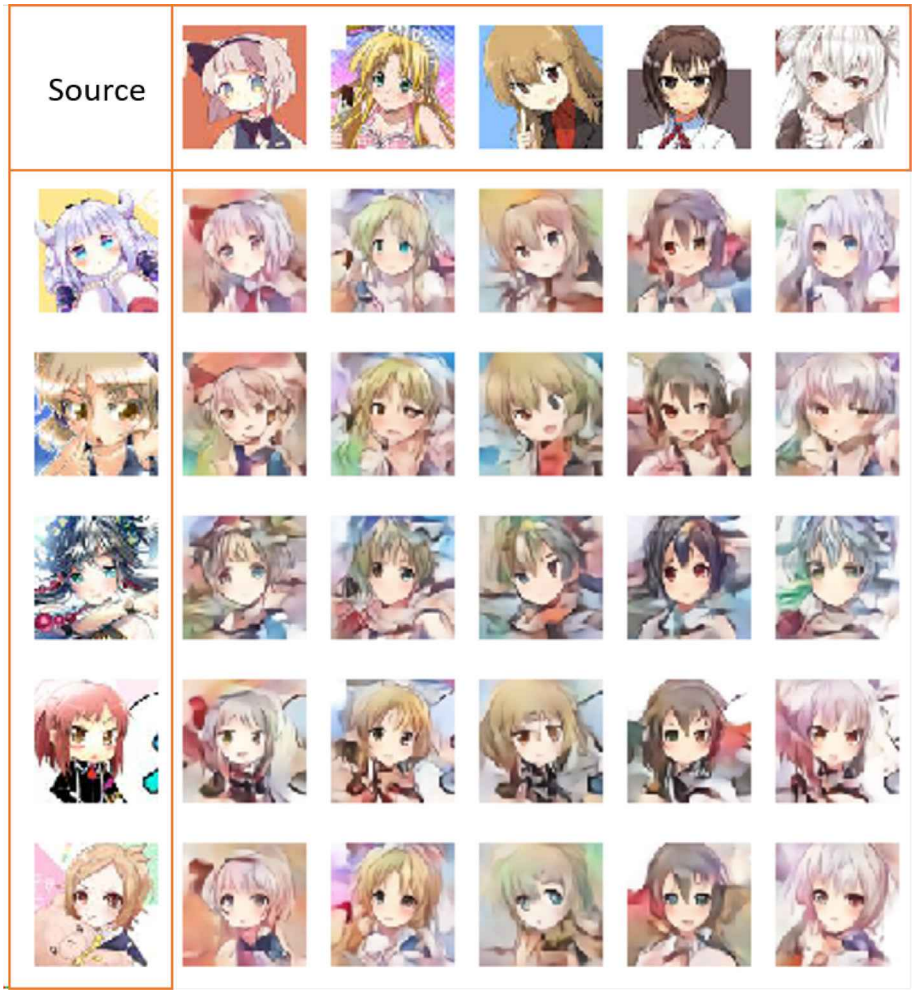


Fig. 11. Anime (128×128 px) generations displayed in Matrix format. Orange rectangular box specifies the source images. Best viewed in color.

Fig. 12. Comparison of generation (left – center image) generated via Image 1 and Image 2 among the training set samples – Showing Top 3 closest matches among Original (Row 1) and Reconstructed images (Row 2) using MSE, SSIM in Image Space and VGG-16 Latent Feature Space. As SSIM and VGG-16 Feature space operate on images and MSE can operate on both images and latent space, Row 3 depicts the closest matches in Latent Space using MSE.

increasing the number of original samples used to train the classifier and keeping the overall training samples as constant (the percentage of generated samples is correspondingly reduced). At the center (indicated by blue dot) with 100% original samples (no generations) corresponds to the default classifier with accuracy of 77% for the particular chosen architecture (Table A.7). It can be observed that the performance of classifier steadily increases from left to the center with increasing percentage of original samples. On the other hand, the plot progresses from the center to the right with varying % of generations augmented to the existing complete set of original samples. At extreme right (indicated by green dot), with 200% (100% original +100% generations) training samples,

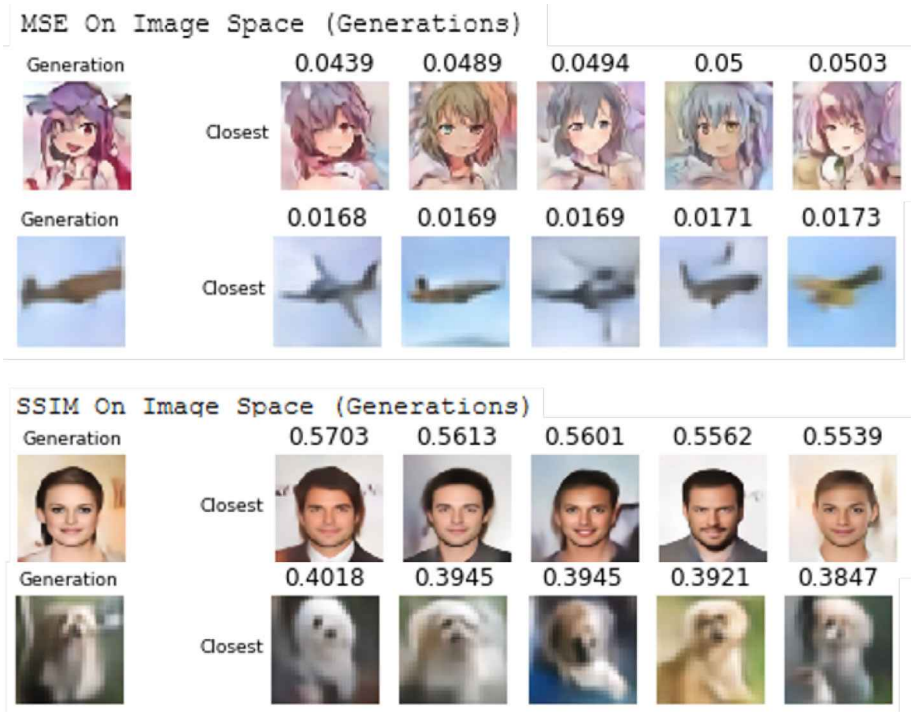


Fig. 13. Comparison of generation among other generations – First two rows show top 5 closest matches in increasing order of squared error in image space; bottom two rows show top 5 closest match in terms of decreasing order of SSIM in image space. Best viewed in color.

Fig. 14. Generations across Generations shown for CelebA and CIFAR-10 car (in (a)), and CIFAR-10 bird and horse (in (b)). Shown in the figure are generations (Gen-1) whose latent features are used to combine and produce new generations Gen-2:fromLS (orange border) and Gen-2:fromRecon. (green border). Best viewed in color.

Table 3
Comparison of FID scores between ReGene and other methods. For benchmarking purposes, results indicated by † are taken from [8], ◇ from [10]. The blanks indicate that the values are not reported in the respective papers. ReGeNe generations are obtained by performing convex combinations of 2 samples, experiments repeated over 3 separate runs. The last row represents FID scores computed between dataset reconstruction and generations.

Method	MNIST	Fashion	CIFAR-10	CelebA
NS GAN†	6.8 ± 0.5	26.5 ± 1.6	58.5 ± 1.9	55.0 ± 3.3
LSGAN†	7.8 ± 0.6	30.7 ± 2.2	87.1 ± 47.5	53.9 ± 2.8
WGAN GP†	20.3 ± 5.0	24.5 ± 2.1	55.8 ± 0.9	30.3 ± 1.0
BEGAN†	13.1 ± 1.0	22.9 ± 0.9	71.4 ± 1.6	38.9 ± 0.9
VAE◇	19.21	–	106.37	48.12
CV – VAE◇	33.79	–	94.75	48.87
WAE◇	20.42	–	117.44	53.67
RAE – SN◇	19.67	–	84.25	44.74
LVPGA◇	6.32 ± 0.16	–	52.94 ± 0.89	$\mathbf{13.8 \pm 0.20}$
2-Stage VAE†	12.6 ± 1.5	29.3 ± 1.0	72.9 ± 0.9	44.4 ± 0.7
ReGene (*Org vs Gen*)	$\mathbf{6.13 \pm 0.12}$	$\mathbf{15.16 \pm 0.08}$	$\mathbf{40.06 \pm 0.23}$	46.65 ± 0.23
ReGene (*Org vs Recon*)	6.03	12.43	29.93	27.41
ReGene (*Recon vs Gen*)	$\mathbf{3.98 \pm 0.03}$	$\mathbf{7.34 \pm 0.03}$	$\mathbf{17.45 \pm 0.02}$	16.35 ± 0.03

the accuracy increased to around 80% with an improvement of around 3% from the base accuracy of 77% (center), indicating that generations can also be potentially used as an augmentation technique for improving classification accuracy.

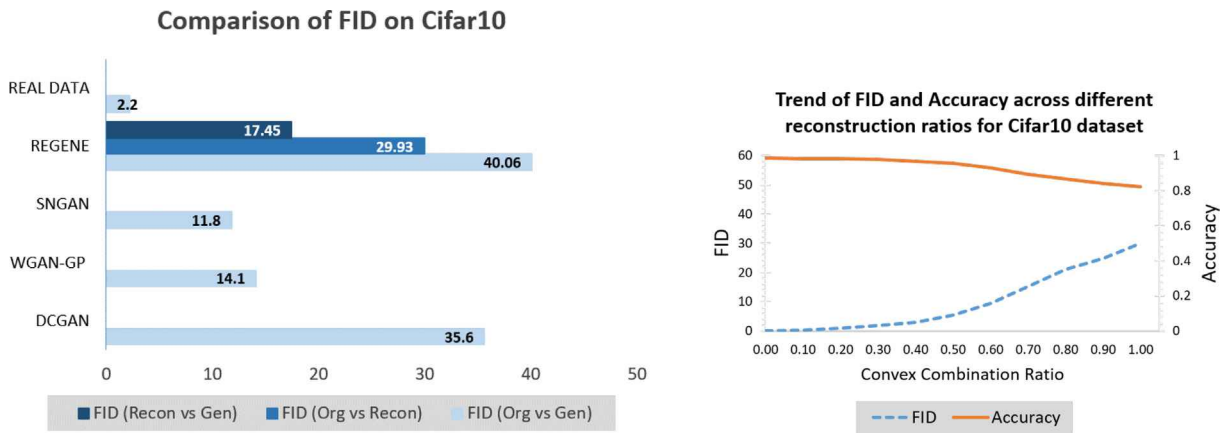


Fig. 15. Left (a): Comparison of FID on the Cifar10 dataset - The FID score for Recon vs Gen (17.45) is comparable to SoTA conditional GANs. ReGene (Org vs Gen) FID score (40.06) is higher, however, (i) ReGene was never optimized directly for generations, (ii) the (Org vs Recon) FID score (29.93) is also high, indicating the reconstruction error should be accounted for during comparison. Right (b): Trend of FID score (blue) vs Accuracy (orange) for different convex combination ratios of original:reconstructed samples. Higher FID score is observed as the ratio tends towards reconstructed samples, caused primarily by the decoder error.

Table 4
Comparison of Inception Score (higher values are better) with other Conditional VAE variants. Results shown for MNIST and Fashion MNIST.

Method	IS for MNIST	IS for Fashion MNIST
Real Image	9.8793 ± 0.0614	9.0617 ± 0.0430
CVAE	2.0594 ± 0.0426	3.5721 ± 0.0483
CCVAE	$\mathbf{2.6463 \pm 0.1007}$	3.4170 ± 0.1455
CCapsCVAE	2.2970 ± 0.0512	4.1865 ± 0.0627
ReGene	2.3840 ± 0.0173	$\mathbf{4.4061 \pm 0.0367}$

7. ReGene: Strengths and limitations

To summarize, in ReGene framework two probabilistic models – encoder/ classifier $p(\boldsymbol{y}|\boldsymbol{X})$ and decoder $p(\boldsymbol{X}|\boldsymbol{z})$ are defined and trained separately. Unlike in VAEs and C-VAEs, in ReGne there is no need for probabilistic modelling of $p(\boldsymbol{z}|\boldsymbol{x})$ and $p(\boldsymbol{z}|\boldsymbol{X},\boldsymbol{y})$ respectively, which facilitates comparatively easier training. Also, instead of randomly sampling the latent space as in VAEs/C-VAEs, ReGene

Method	Classifier Accuracy	Decoder MAE	FID score with no constraint on generation	Accuracy when these generations (**no constraint**) are passed to classifier	FID score with latent + image space constraint on generation	Number of tries to obtain constraint accepted generations	Accuracy when these generations (**with constraint**) are passed to classifier
Categorical cross-entropy	**0.7522**	**0.0667**	**51.13**	**0.6856**	40.67	**10243**	**1.0**
Multiclass hinge	0.7303	0.0695	53.64	0.6276	**39.36**	40847	0.9675
Mean Squared Error	0.6499	0.0746	57.47	0.5339	49.59	29217	0.9764
Cosine similarity	0.7275	0.0721	54.32	0.6069	43.40	18784	0.9987

Fig. 16. Different loss functions employed for training classifier on Cifar10 are compared with each other on different metrics such as classification accuracy, generation quality and number of tries taken to obtain quality generations. Among the four loss functions, crossentropy outperforms other approaches in almost all of the listed metrics.

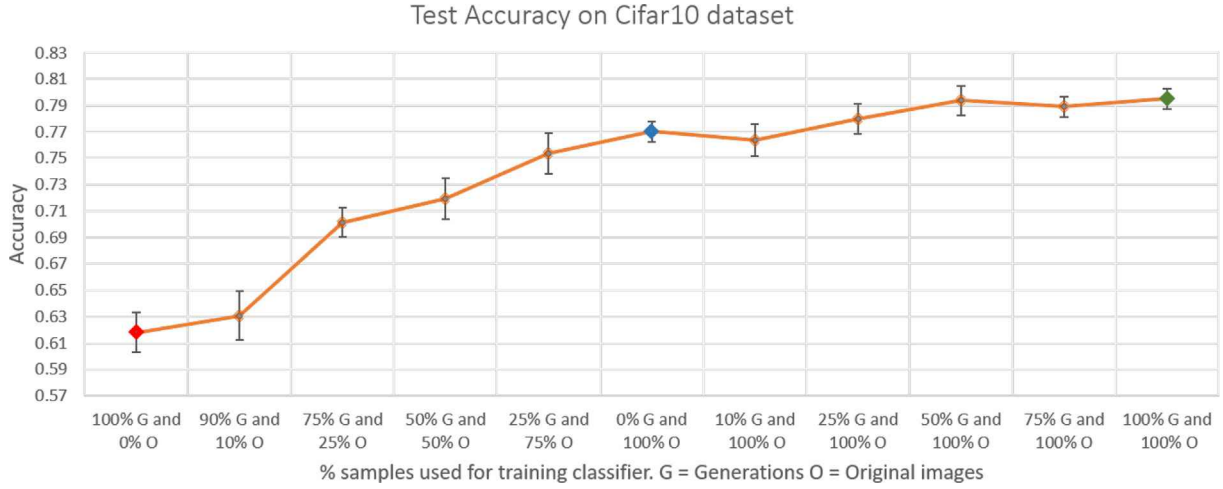


Fig. 17. Classification performance on varying the % of training samples containing generation and original samples, experimented on Cifar10. Reading from center to left shows the performance of varying original versus generations while retaining same number of samples. Reading center to right is the performance of generation augmentation on top of original images. Colored dots (red, blue, green) indicate the complete absence of original data, default 100% original data, and complete augmentation with equal ratio of original and generations.

framework uses convex combinations to generate new samples (Lemma 1). Other main advantages of ReGene includes (i) Guarantee for same class generations in both latent space and image space (from Lemma 1 and Theorem 2); (ii) No prior distribution required to model latent space distributions; (iii) No mode drop issue (as selection samples for a given class are deterministic in nature), and (iv) Stable and straight-forward training procedure (no adversarial training required). As ReGene is one of the first frameworks to investigate classification latent space based image generation, the following limitations requires further attention: (i) ReGene cannot directly evaluate $\boldsymbol{p}(\boldsymbol{x})$ for modeling the data distribution; (ii) Trade-off between reconstruction and classification accuracy as highlighted in Section 5.2.1, and (iii) Finally, the quality of image generations is dependent on the reconstruction ability of decoder.

8. Conclusion

The answer to the question: 'Can classification latent space representations be reused for the downstream task of reconstruction and generation?' is *Yes* – through the proposed ReGene framework which can reconstruct using classification latent space representations and generate visually meaningful images via convex combinations of such representations. We quantitatively and qualitatively demonstrated the efficacy of ReGene framework on standard datasets and showed comparable performance with other existing state-of-the-art generative methods. While our experiments utilized simple network structures to prove the possibility of this alternative generative approach, design and development of more sophisticated architectures for better reconstruction and therefore, generation will further strengthen the generation quality of ReGene. It should be noted that Lemma 1 is directly defined on classification latent space, and hence its application to domains other than computer vision will be an exciting research direction. We firmly believe this work will foster further intriguing researches in exploration and exploitation of supervised latent space representations to other downstream tasks.

Funding

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

CRediT authorship contribution statement

Saisubramaniam Gopalakrishnan: Conceptualization, Methodology, Investigation, Writing - original draft, Writing - review & editing. **Pranshu Ranjan Singh:** Conceptualization, Methodology, Investigation, Writing - original draft, Writing - review & editing. **Yasin Yazici:** Writing - original draft, Validation. **Chuan-Sheng Foo:** Validation. **Vijay Chandrasekhar:** Validation (during initial stages of the work). **ArulMurugan Ambikapathi:** Conceptualization, Methodology, Writing - original draft, Writing - review & editing, Supervision.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix A

In this Section, we provide extra information that may be useful to understand better the sections in the main portion, and aid the readers in understanding the implementation details.[4] Appedix A provides details of the different neural network architecture used for each dataset, along with hyperparameters. Appedix B is a short discusission on the discrepancy in CelebA across different methods, and its effect on FID score. Appedix C presents collage of visualizations for all the datasets used under the Experiment Section Appedix D compares the closest match of a generation across all training samples, and across other generations, per class and dataset.

Table A.5
Network Architecture of Classifier/Encoder and Decoder used for MNIST/Fashion dataset.

Classifier/Encoder Architecture Details
Input(shape=(28, 28, 1))
[Conv2D(filters=128, kenel size=3x3, padding=*same*), BatchNorm, LeakyReLU(α=0.3)] x 2, MaxPool2D(pool size=2x2)
Conv2D(filters=128, kenel size=3x3, padding=*same*), BatchNorm, LeakyReLU(α=0.3), MaxPool2D(pool size=2x2)
Conv2D(filters=128, kenel size=3x3, padding=*same*), BatchNorm, LeakyReLU(α=0.3), MaxPool2D(pool size=2x2)
[Conv2D(filters=128, kenel size=3x3, padding=*same*), BatchNorm, LeakyReLU(α=0.3)] x 2, Flatten(dimension=1152)
Dense(nodes=10), Softmax Activation
Decoder Architecture Details
Input(shape=(1152)), Reshape(shape=(3, 3, 128))
[Conv2DTranspose(filters=512, kenel size=3x3, padding=*valid*), LeakyReLU(α=0.3), BatchNorm] x 2
Conv2DTranspose(filters=512, kenel size=3x3, padding=*same*), LeakyReLU(α=0.3), BatchNorm, UpSampling2D(size=2x2)
[Conv2DTranspose(filters=256, kenel size=3x3, padding=*same*), LeakyReLU(α=0.3), BatchNorm] x 3, UpSampling2D(size=2x2)
[Conv2DTranspose(filters=256, kenel size=3x3, padding=*same*), LeakyReLU(α=0.3), BatchNorm] x 3
Conv2D(filters=3, kenel size=3x3, padding=*same*), Sigmoid Activation

[4] The code for Regene framework will be released on acceptance of the paper, to foster further research in this direction.

Table A.7
Network Architecture of Classifier/Encoder and Decoder used for CIFAR-10 dataset.

Classifier/Encoder Architecture Details
Input(shape=(32, 32, 3))
[Conv2D(filters=128, kenel size=3x3, padding=*same*), BatchNorm, LeakyReLU(α=0.3)] x 2, MaxPool2D(pool size=2x2)
Conv2D(filters=128, kenel size=3x3, padding=*same*), BatchNorm, LeakyReLU(α=0.3), MaxPool2D(pool size=2x2)
Conv2D(filters=128, kenel size=3x3, padding=*same*), BatchNorm, LeakyReLU(α=0.3), MaxPool2D(pool size=2x2)
Conv2D(filters=128, kenel size=9x9, padding=*same*), BatchNorm, LeakyReLU(α=0.3)
Conv2D(filters=128, kenel size=11x11, padding=*same*), BatchNorm, LeakyReLU(α=0.3), Flatten(dimension=2048)
Dense(nodes=10), Softmax Activation
Decoder Architecture Details
Input(shape=(2048)), Reshape(shape=(4, 4, 128))
[Conv2DTranspose(filters=512, kenel size=3x3, padding=*same*), LeakyReLU(α=0.3), BatchNorm] x 3, UpSampling2D(size=2x2)
[Conv2DTranspose(filters=256, kenel size=3x3, padding=*same*), LeakyReLU(α=0.3), BatchNorm] x 3, UpSampling2D(size=2x2)
[Conv2DTranspose(filters=256, kenel size=3x3, padding=*same*), LeakyReLU(α=0.3), BatchNorm] x 3, UpSampling2D(size=2x2)
[Conv2DTranspose(filters=256, kenel size=3x3, padding=*same*), LeakyReLU(α=0.3), BatchNorm] x 3
Conv2D(filters=3, kenel size=3x3, padding=*same*), Sigmoid Activation

Table A.6
Network Architecture of Classifier/Encoder and Decoder used for CelebA/Anime dataset.

Classifier/Encoder Architecture Details
Input(shape=(64, 64, 3))
[Conv2D(filters=128, kenel size=3x3, padding=*same*), BatchNorm, LeakyReLU(α=0.3)] x 2, MaxPool2D(pool size=2x2)
Conv2D(filters=128, kenel size=3x3, padding=*same*), BatchNorm, LeakyReLU(α=0.3), MaxPool2D(pool size=2x2)
Conv2D(filters=128, kenel size=3x3, padding=*same*), BatchNorm, LeakyReLU(α=0.3), MaxPool2D(pool size=2x2)
Conv2D(filters=128, kenel size=3x3, padding=*same*), BatchNorm, LeakyReLU(α=0.3), MaxPool2D(pool size=2x2)
[Conv2D(filters=128, kenel size=3x3, padding=*same*), BatchNorm, LeakyReLU(α=0.3)] x 2, Flatten(dimension=2048)
Dense(nodes=2), Softmax Activation
Decoder Architecture Details
Input(shape=(2048)), Reshape(shape=(4, 4, 128))
[Conv2D(filters=512, kenel size=3x3, padding=*same*), LeakyReLU(α=0.3), BatchNorm] x 3, UpSampling2D(size=2x2)
[Conv2D(filters=256, kenel size=3x3, padding=*same*), LeakyReLU(α=0.3), BatchNorm] x 3, UpSampling2D(size=2x2)
[Conv2D(filters=256, kenel size=3x3, padding=*same*), LeakyReLU(α=0.3), BatchNorm] x 3, UpSampling2D(size=2x2)
[Conv2D(filters=256, kenel size=3x3, padding=*same*), LeakyReLU(α=0.3), BatchNorm] x 3, UpSampling2D(size=2x2)
[Conv2D(filters=256, kenel size=3x3, padding=*same*), LeakyReLU(α=0.3), BatchNorm] x 3
Conv2D(filters=3, kenel size=3x3, padding=*same*), Sigmoid Activation

A.1. Network architectures

The network architectures used in experiments (Section 4) for datasets MNIST (and Fashion), CelebA/Anime and CIFAR-10 are depicted here in Tables A.5–A.7, respectively. Additional training details for each dataset are provided below. Most of the hyperparameter settings are common for similar datasets (MNIST, Fashion MNIST). Depending on resolution and RGB channels, slight changes are made to accommodate size and memory constraints.

A.1.1. MNIST

- Classifier
  - Loss function: Categorical Crossentropy
  - Optimizer: Adam (learning rate = 0.005 with drop of 0.99 every 20 epochs)
  - Batch Size: 512
  - Epochs Trained: 50
  - Time taken for training on RTX 2080Ti: 3 s per epoch.
- Decoder
  - Loss functions: MAE, SSIM
  - Loss weights:
    Epoch 1–200: λ_1 (MAE)=1.0, λ_2 (SSIM)=1.0; Adam optimizer (learning rate = 0.001)
    Epoch 200–300: λ_1 (MAE)=1.0, λ_2 (SSIM)=0.0; Adam optimizer (learning rate = 0.0001)
  - Batch Size: 512
  - Epochs Trained: 300

– Time taken for training on RTX 2080Ti: 20 s per epoch.

A.1.2. Fashion MNIST

- Classifier
  - Loss function: Categorical Crossentropy
  - Optimizer: Adam (learning rate = 0.005 with drop of 0.99 every 20 epochs)
  - Batch Size: 512
  - Epochs Trained: 50
  - Time taken for training on RTX 2080Ti: 3 s per epoch.
- Decoder
  - Loss functions: MAE, SSIM
  - Loss weights:
    Epoch 1–200: λ_1 (MAE)=1.0, λ_2 (SSIM)=1.0; Adam optimizer (learning rate = 0.001)
    Epoch 200–300: λ_1 (MAE)=1.0, λ_2 (SSIM)=0.0; Adam optimizer (learning rate = 0.0001)
  - Batch Size: 512
  - Epochs Trained: 300
  - Time taken for training on RTX 2080Ti: 20 s per epoch.

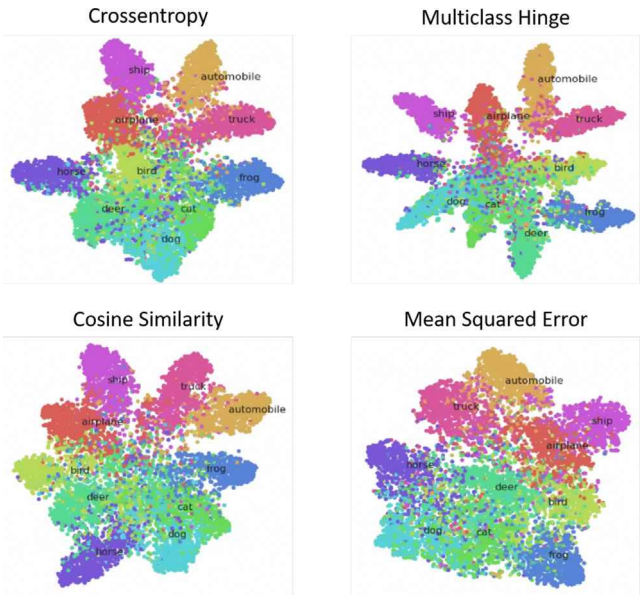


Fig. B.18. t-SNE of latent spaces obtained through different loss functions when training classifier.

A.1.3. Cifar10

- Classifier
  - Loss function: Categorical Crossentropy
  - Optimizer: Adam (learning rate = 0.005 with drop of 0.99 every 20 epochs)
  - Batch Size: 512
  - Epochs Trained: 100
  - Time taken for training on RTX 2080Ti: 13 s per epoch.
- Decoder
  - Loss functions: MAE, SSIM, Perceptual Loss (InceptionV3 - block1_conv1)
  - Loss weights:
    Epoch 1–100: λ_1 (MAE)=1.0, λ_2 (SSIM)=1.0, λ_3 (Perceptual Loss)=1.0; Adam optimizer (learning rate = 0.001)
    Epoch 100–200: λ_1 (MAE)=1.0, λ_2 (SSIM)=0.0, λ_3 (Perceptual Loss)=1.0; Adam optimizer (learning rate = 0.001)
    Epoch 200–300: λ_1 (MAE)=1.0, λ_2 (SSIM)=0.0, λ_3 (Perceptual Loss)=0.0; Adam optimizer (learning rate = 0.0001)
  - Batch Size: 512
  - Epochs Trained: 300
  - Time taken for training on RTX 2080Ti: 35 s per epoch.

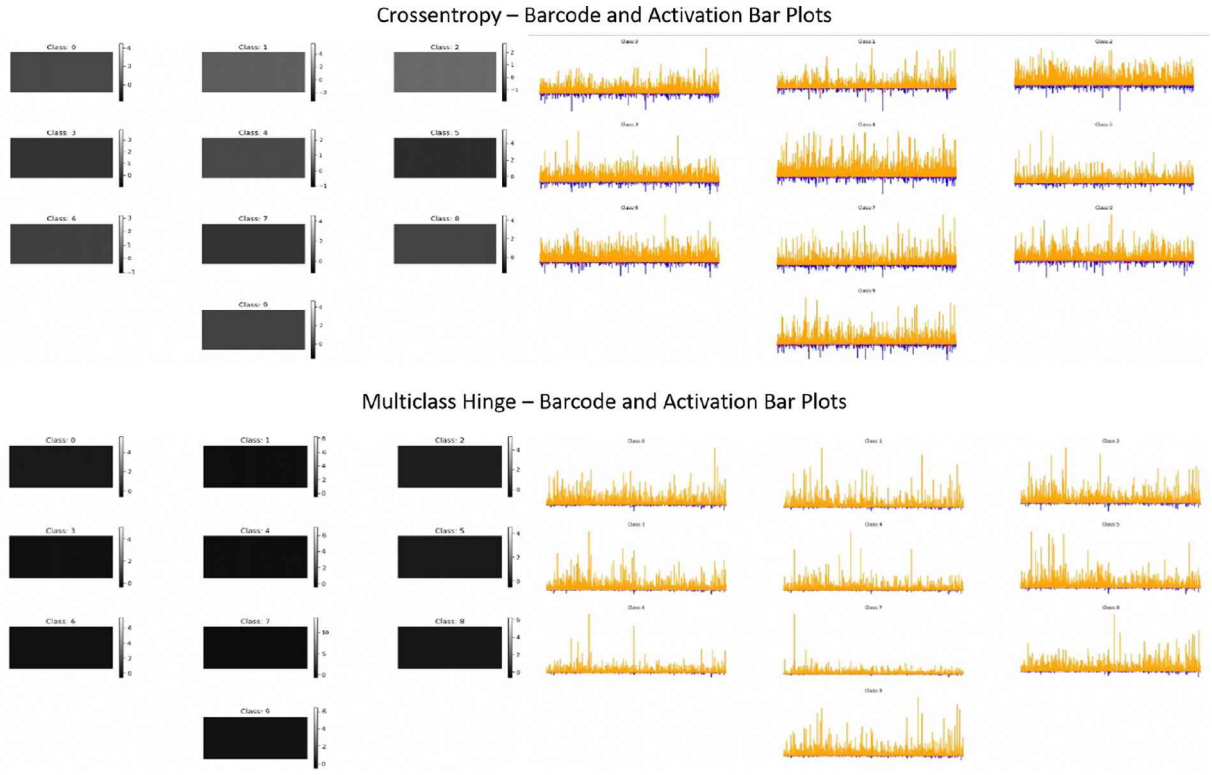


Fig. B.19. Barcode and bar plots obtained by passing samples belonging to the particular class – Crossentropy and multi-class hinge.

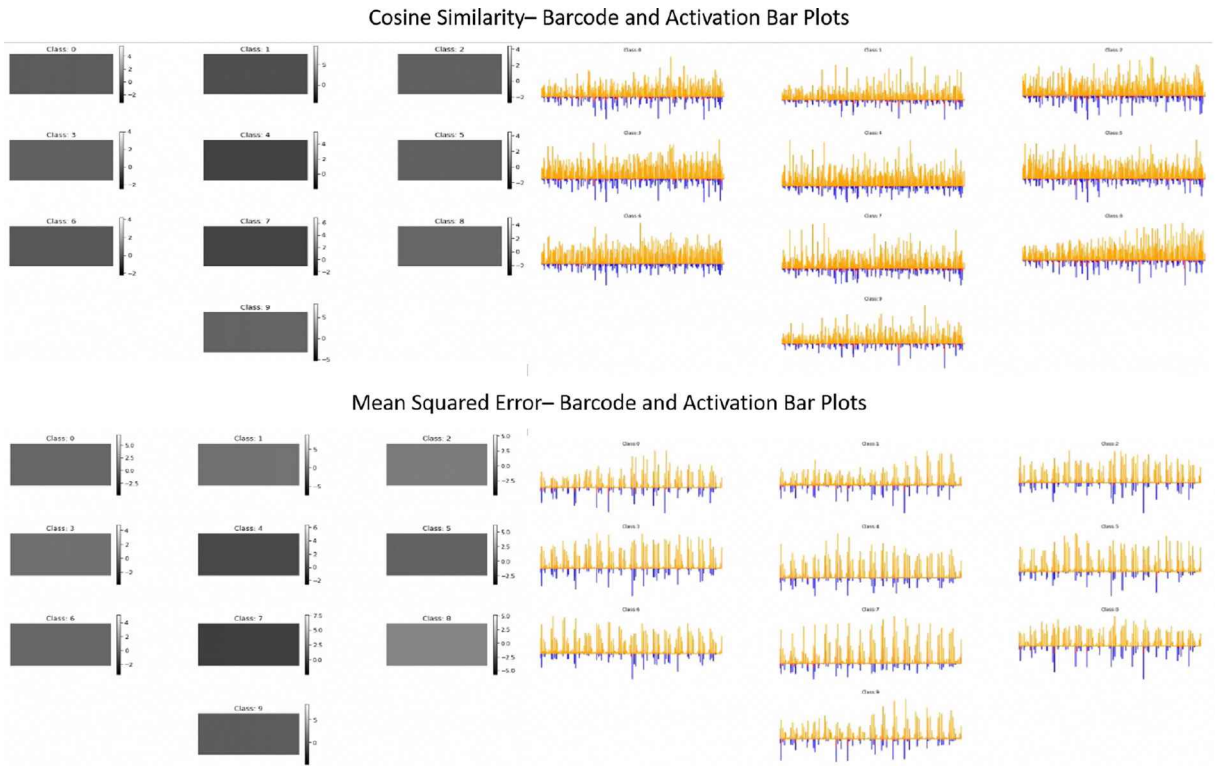


Fig. B.20. Barcode and bar plots obtained by passing samples belonging to the particular class – Cosine similarity and mean squared error.

A.1.4. CelebA

- Classifier
  - Loss function: Binary Crossentropy
  - Optimizer: Adam (learning rate = 0.005 with drop of 0.99 every 5 epochs)
  - Batch Size: 256
  - Epochs Trained: 50
  - Time taken for training on RTX 2080Ti: 40 s per epoch.
- Decoder
  - Loss functions: MAE, SSIM, Perceptual Loss (InceptionV3 - block1_conv1)
  - Loss weights:
    Epoch 1–300: λ_1 (MAE)=1.0, λ_2 (SSIM)=1.0, λ_3 (Perceptual Loss)=1.0
    Epoch 300–600: λ_1 (MAE)=1.0, λ_2 (SSIM)=0.0, λ_3 (Perceptual Loss)=1.0
    600–700: λ_1 (MAE)=1.0, λ_2 (SSIM)=0.0, λ_3 (Perceptual Loss)=0.0
  - Optimizer: Adam (learning rate = 0.0005)
  - Batch Size: 128
  - Epochs Trained: 700
  - Time taken for training on RTX 2080Ti: 300 s per epoch.

Appendix B. Comparison of different loss functions for classifier

The extended study of different loss functions used for optimizing the classifier are provided here. t-SNE comparison of the different methods discussed in the main paper is shown in Fig. B.18, followed by the barcode and activation plots in Figs. B.19 and B.20 respectively. Notable from the classifier latent space is multi-class hinge, where the decision boundary separates the classes neatly,but the representation shape does not favour convex combination. This is reflected from Table 16, it takes 4x as much tries compared to crossentropy to generate a valid and high accuracy yielding generation. Other methods such as cosine similarity and mean squared error have worse performance than crossentropy in terms of classifier accuracy and FID scores.

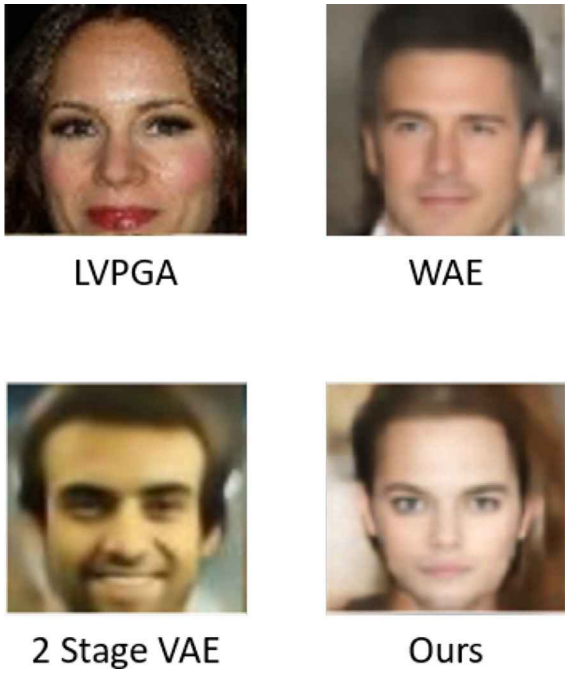


Fig. C.21. Comparison of different CelebA face generations from (i) LVPGA, (ii) WAE, (iii) 2-Stage VAE, and (iv) Ours (ReGene). Notice the face scaling is more closer for LVPGA.

Appendix C. Discussion on CelebA dataset: Discrepancy in CelebA dataset used by LVPGA

We would also like to bring to notice on the different CelebA versions reported by different existing approaches in the literature, with respect to the pre-processing (center cropping) performed to extract the faces. (As mentioned in Section 4.1 in the main paper) When we computed FID for original CelebA 'Align&Cropped Images' dataset (128×128 resize without crop), we have obtained FID score of ~ 78. After following the cropping procedure used by WAE [7] and 2-Stage VAE [8] (140×140 center crop on the original CelebA 'Align&Cropped Images' dataset) and resized to 64×64 to compare, our FID score was 48.3. For LVPGA, we noticed that the cropping resolution for CelebA faces used is not mentioned, and visual inspection readily showed that the cropping is even smaller, and contains only limited face features (cutting off portions of hair and the chin) (Refer Fig. C.21). Hence, different versions of the same CelebA dataset giving different FID scores is not a surprise, and LVPGA achieving a low FID score of 13.8 can be partially attributed to its custom cropping procedure.

Appendix D. Additional visualizations

D.1. Collage

Collage of generations for MNIST (Fig. D.22), Fashion MNIST (Fig. D.23), Cifar10 (Fig. D.24), CelebA faces (Figs. D.25), D.26 and Anime faces (Fig. D.27) are provided.

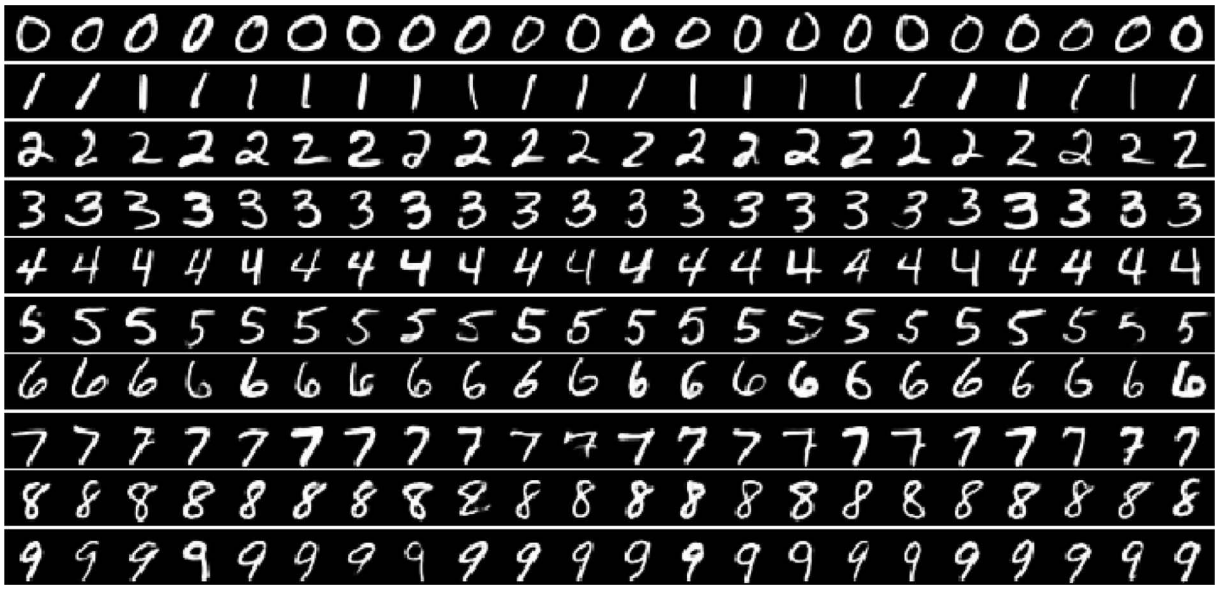


Fig. D.22. Generation Collage from MNIST (28×28 resolution). The generations per row are digits 0 to 9.

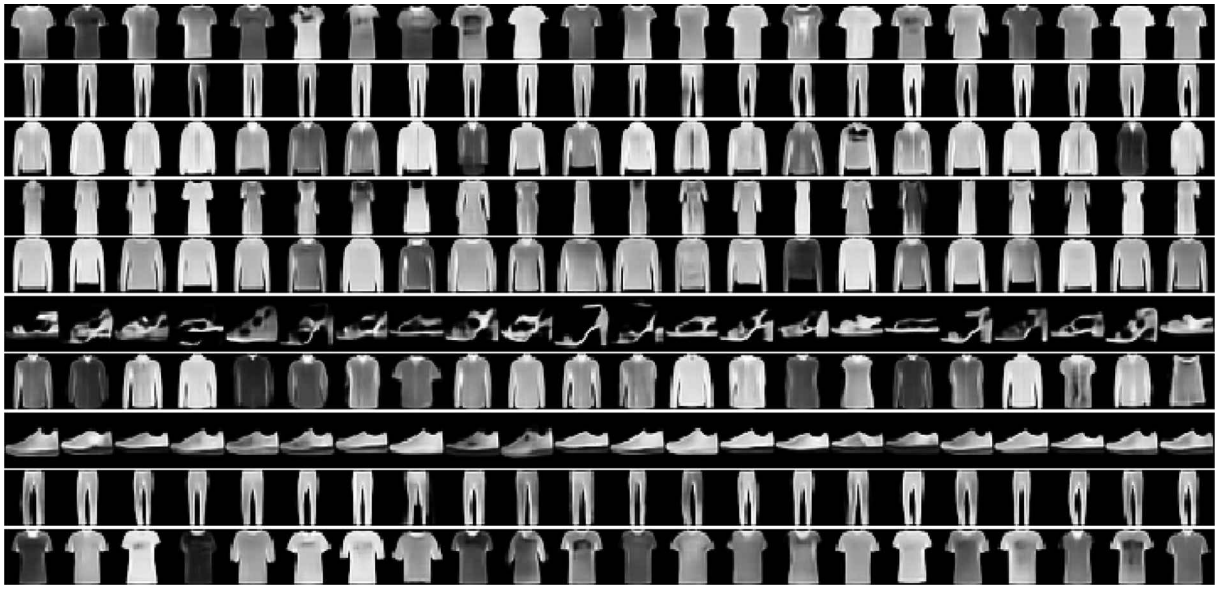


Fig. D.23. Generation Collage from Fashion MNIST (28×28 resolution). The generations per row are: 'Tshirt','Trouser','Pullover','Dress','Coat','Sandal','Shirt','Sneaker','Bag', and 'Ankle boot'.

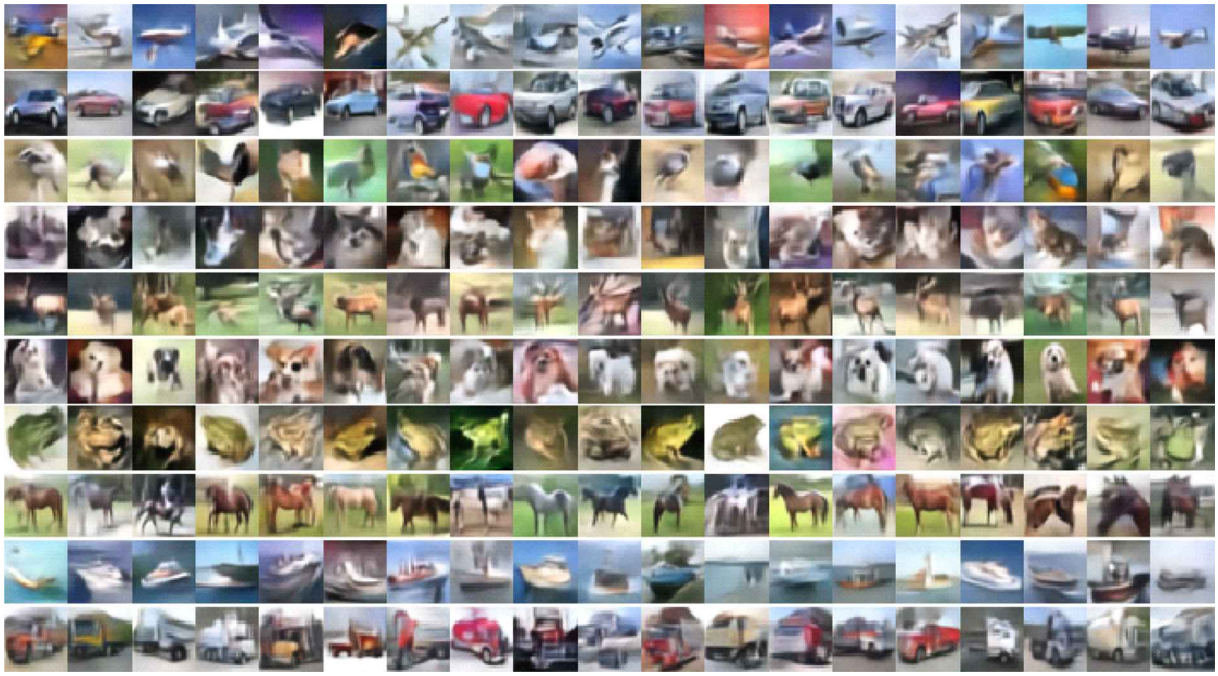


Fig. D.24. Generation Collage from Cifar10 (32×32 resolution). The generations per row are: 'Airplane', 'Automobile', 'Bird', 'Cat', 'Deer', 'Dog', 'Frog', 'Horse', 'Ship', and 'Truck'. Best viewed in color.

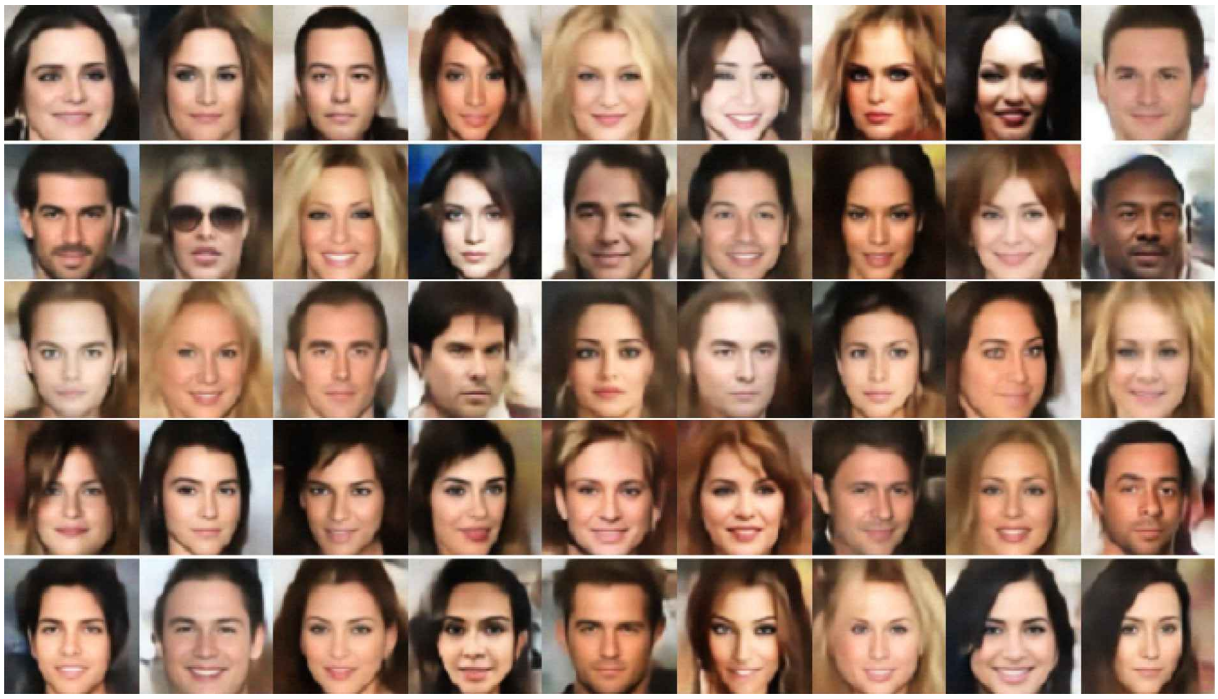


Fig. D.25. Generation Collage from CelebA (64×64 resolution). Best viewed in color.

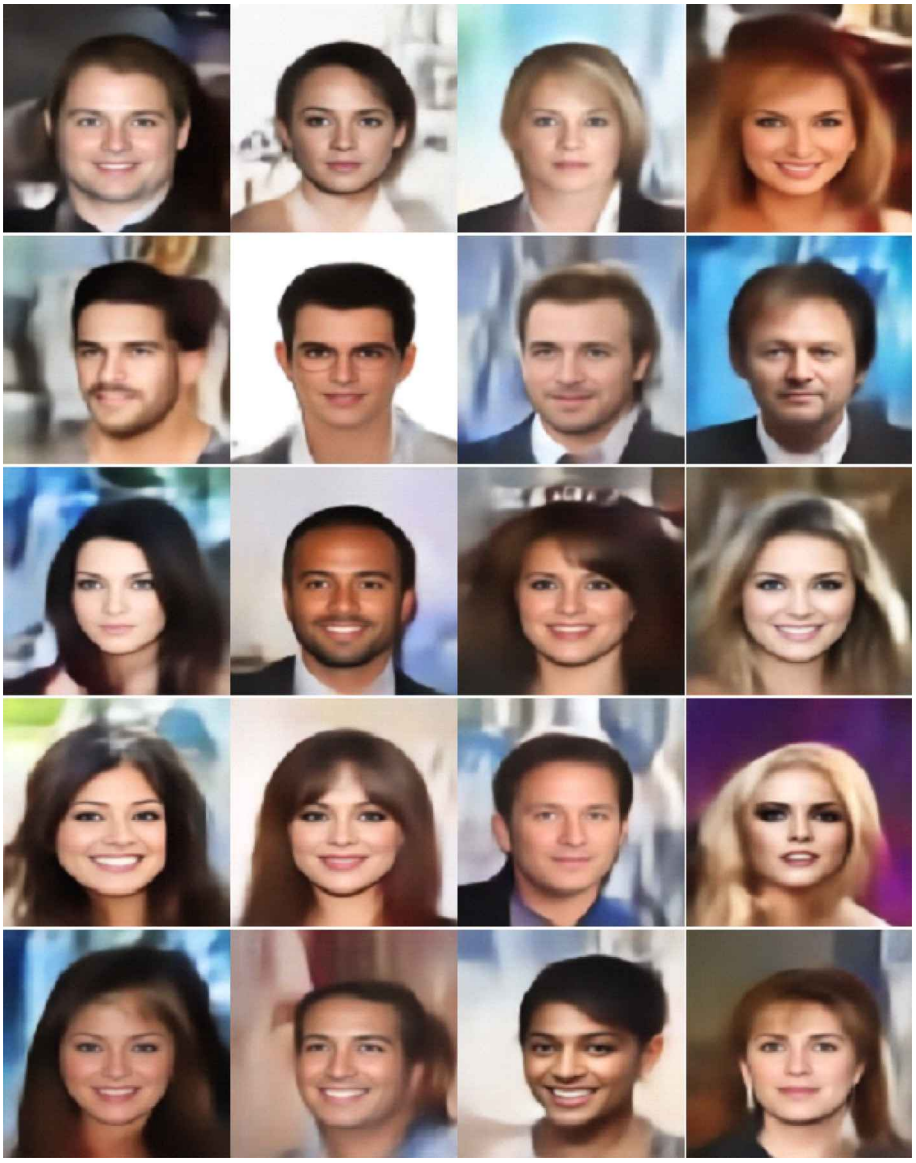


Fig. D.26. Generation Collage from CelebA (128×128 resolution). Best viewed in color.

Fig. D.27. Generation Collage from Anime (128×128 resolution). Best viewed in color.

Appendix E. Closest match

E.1. Closest match across training samples

We present generations and compare with the existing training dataset to get closest match. The closest matches among training samples (both original and reconstructed versions) in terms of (i) increasing image space MSE, (ii) decreasing image space SSIM, and (iii) VGG-16 feature space are shown in Figs. E.28–E.39.

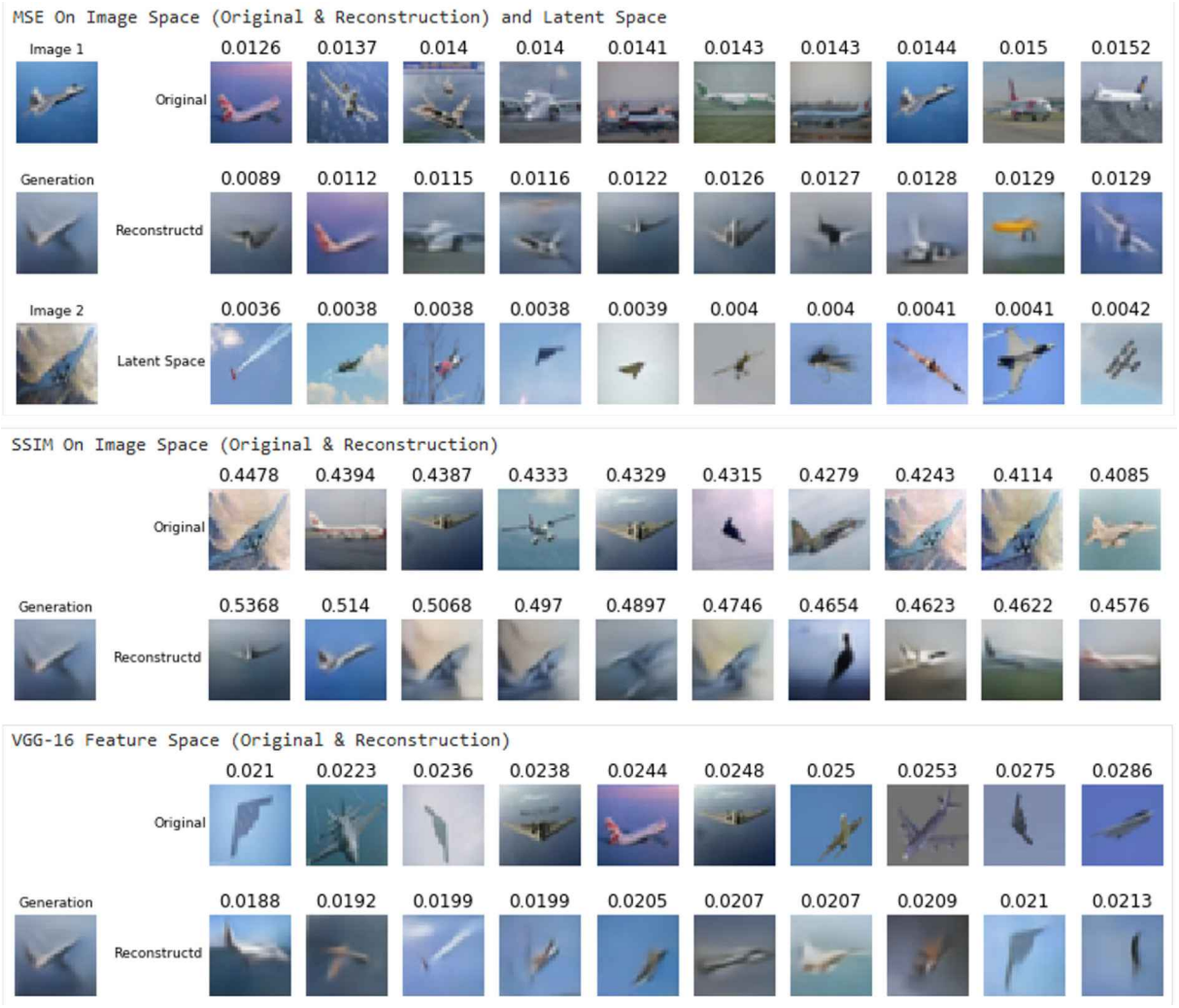


Fig. E.28. Cifar10 Closest Match Among Training Samples – Airplane.

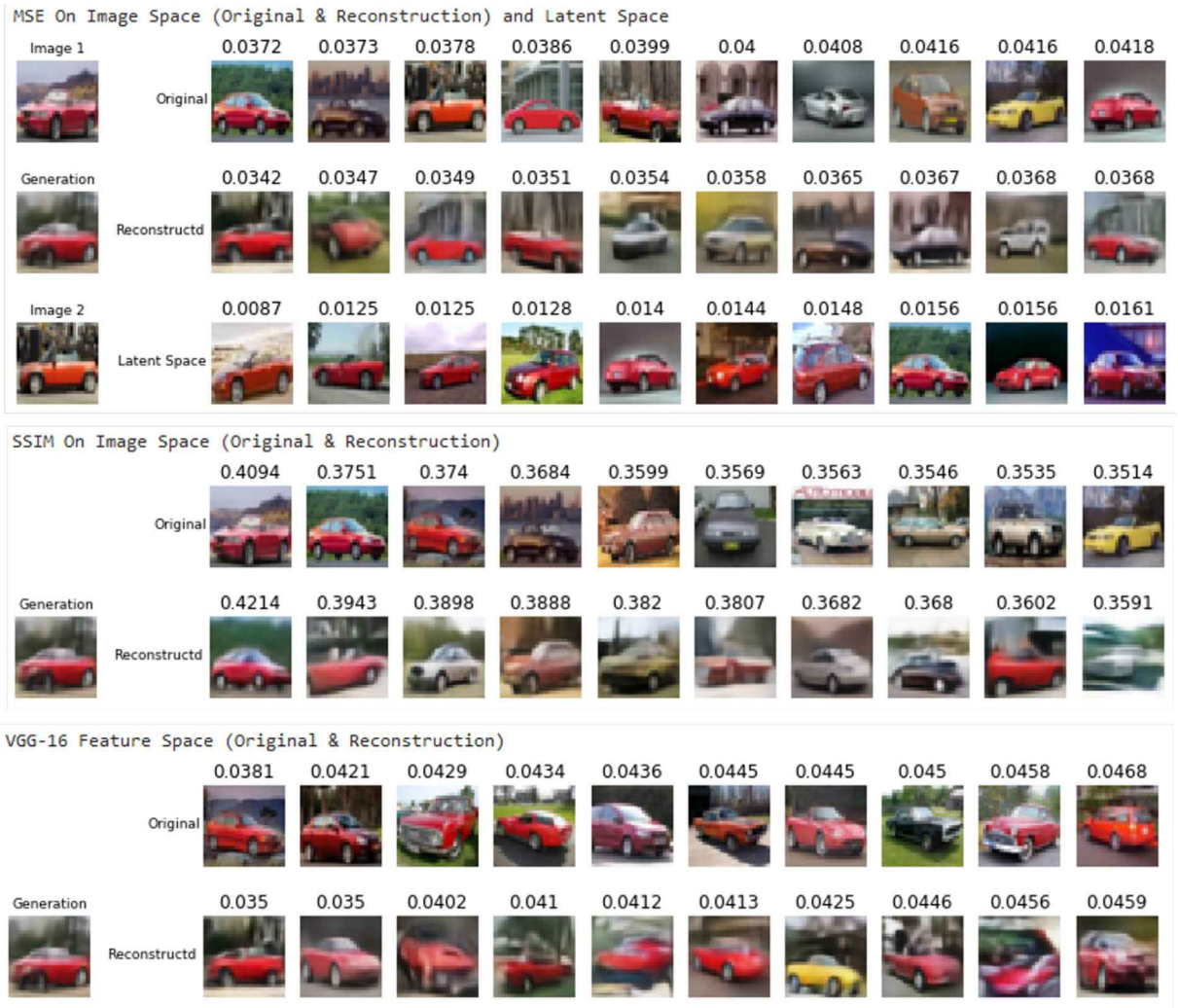


Fig. E.29. Cifar10 Closest Match Among Training Samples – Car.

Fig. E.30. Cifar10 Closest Match Among Training Samples – Bird.

Fig. E.31. Cifar10 Closest Match Among Training Samples – Cat.

Fig. E.32. Cifar10 Closest Match Among Training Samples – Deer.

Fig. E.33. Cifar10 Closest Match Among Training Samples – Dog.

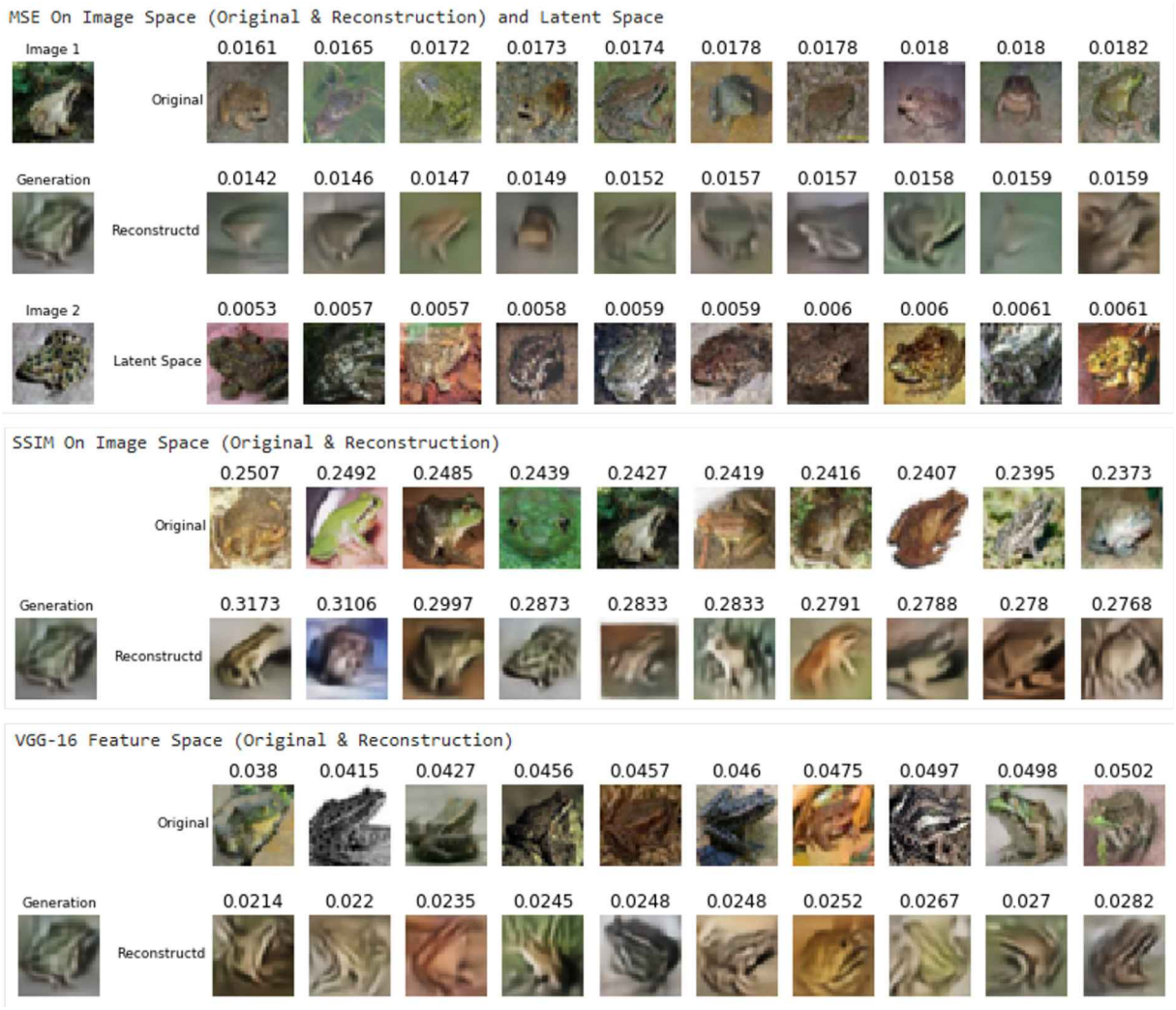


Fig. E.34. Cifar10 Closest Match Among Training Samples – Frog.

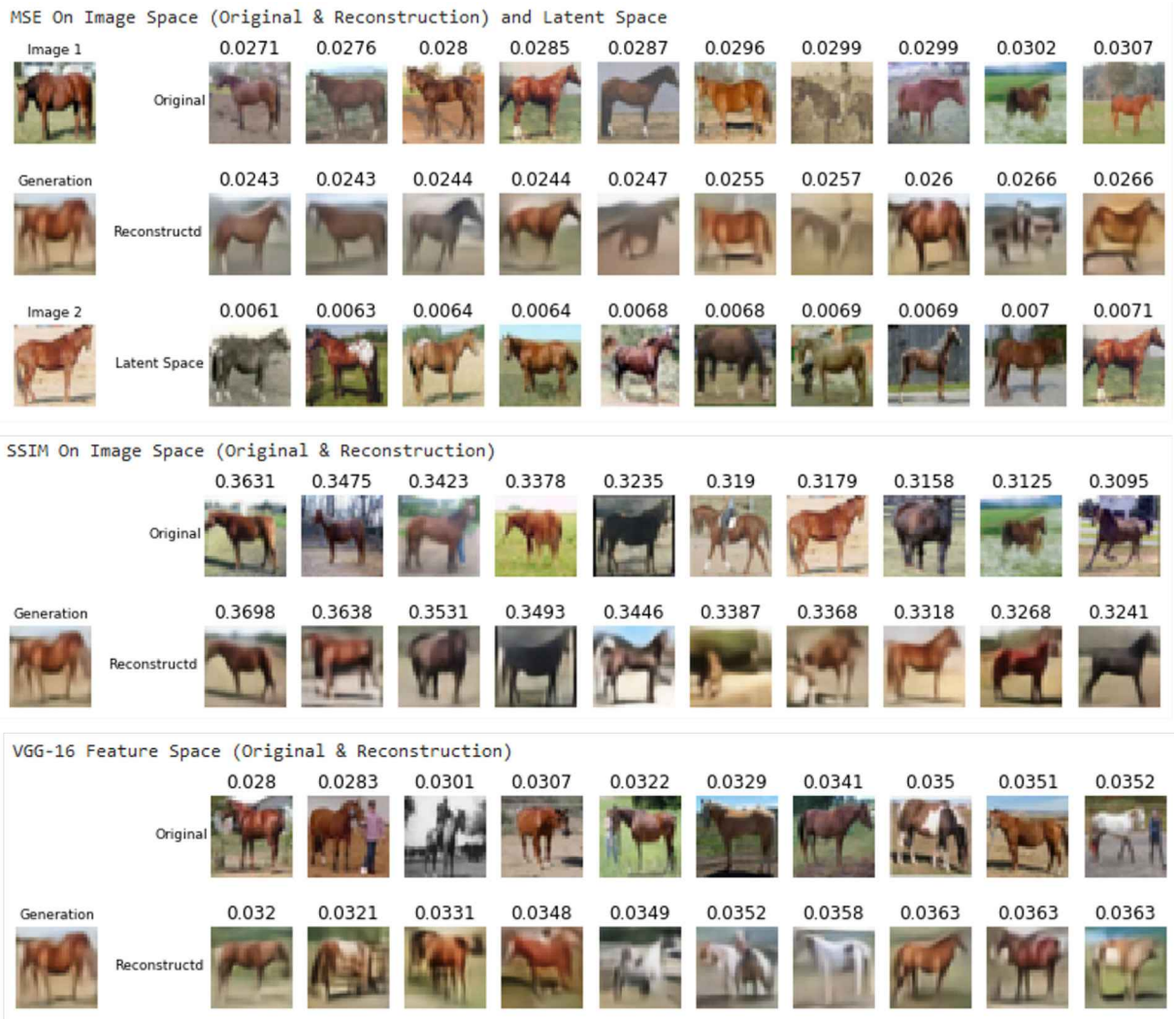


Fig. E.35. Cifar10 Closest Match Among Training Samples – Horse.

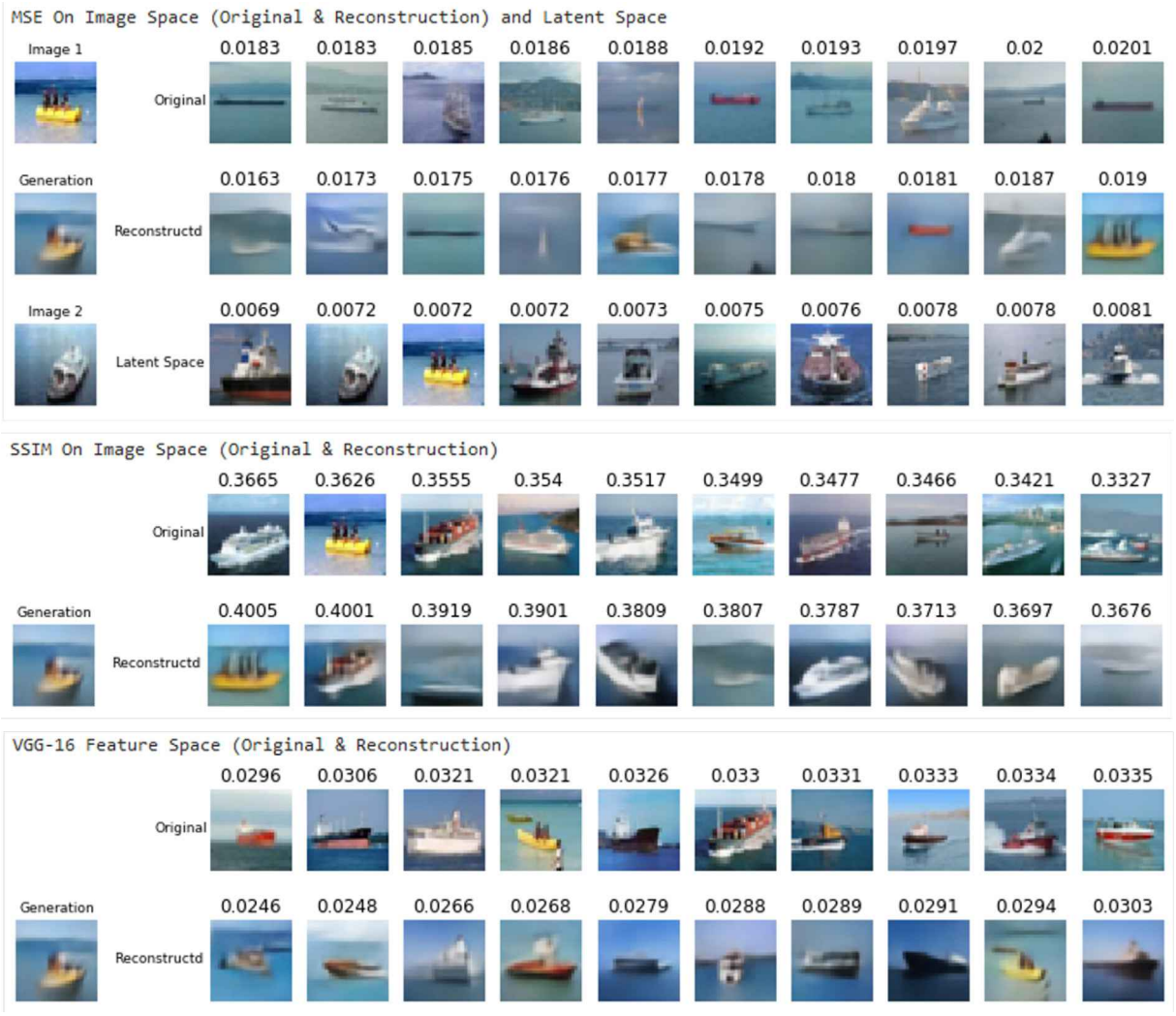


Fig. E.36. Cifar10 Closest Match Among Training Samples – Ship.

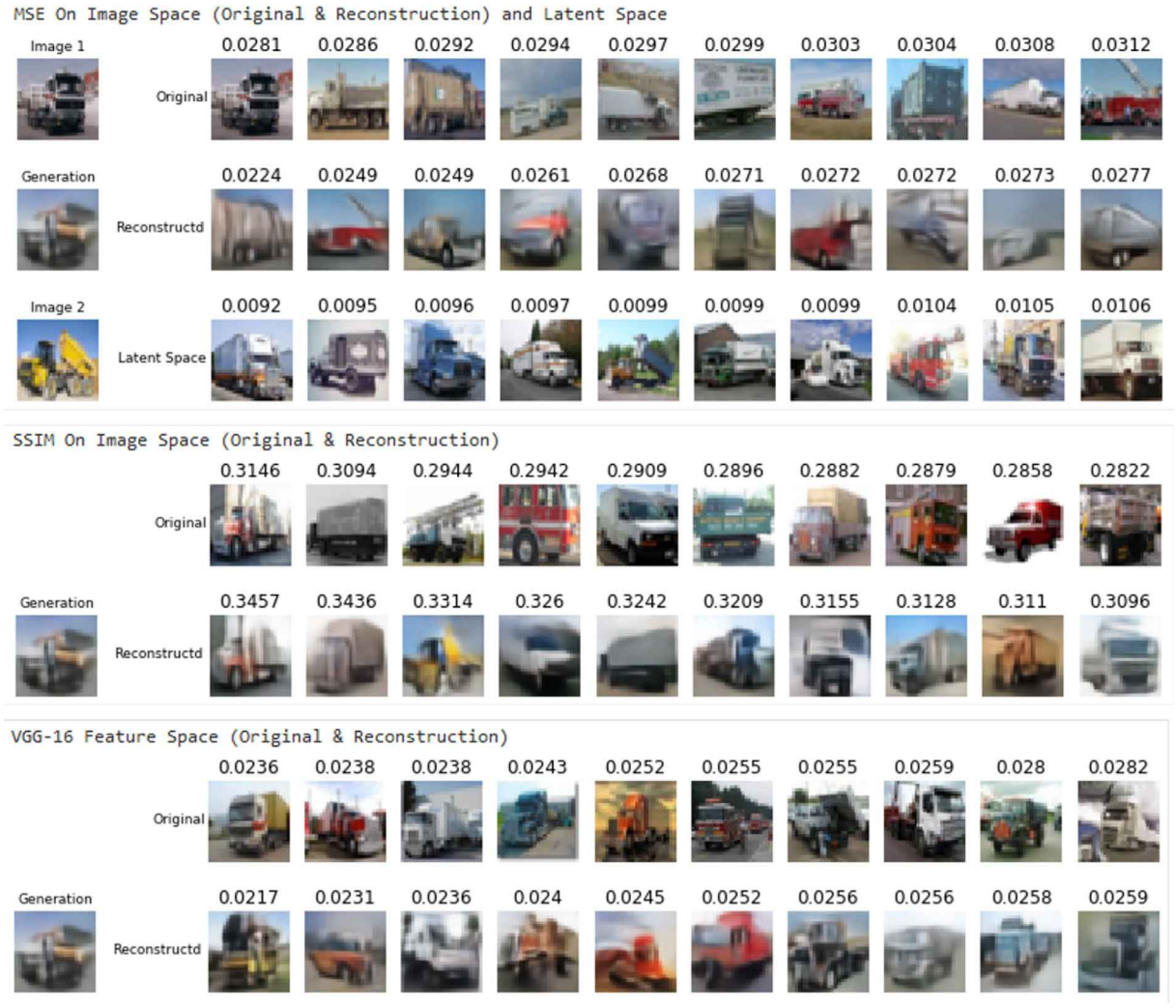


Fig. E.37. Cifar10 Closest Match Among Training Samples – Truck.

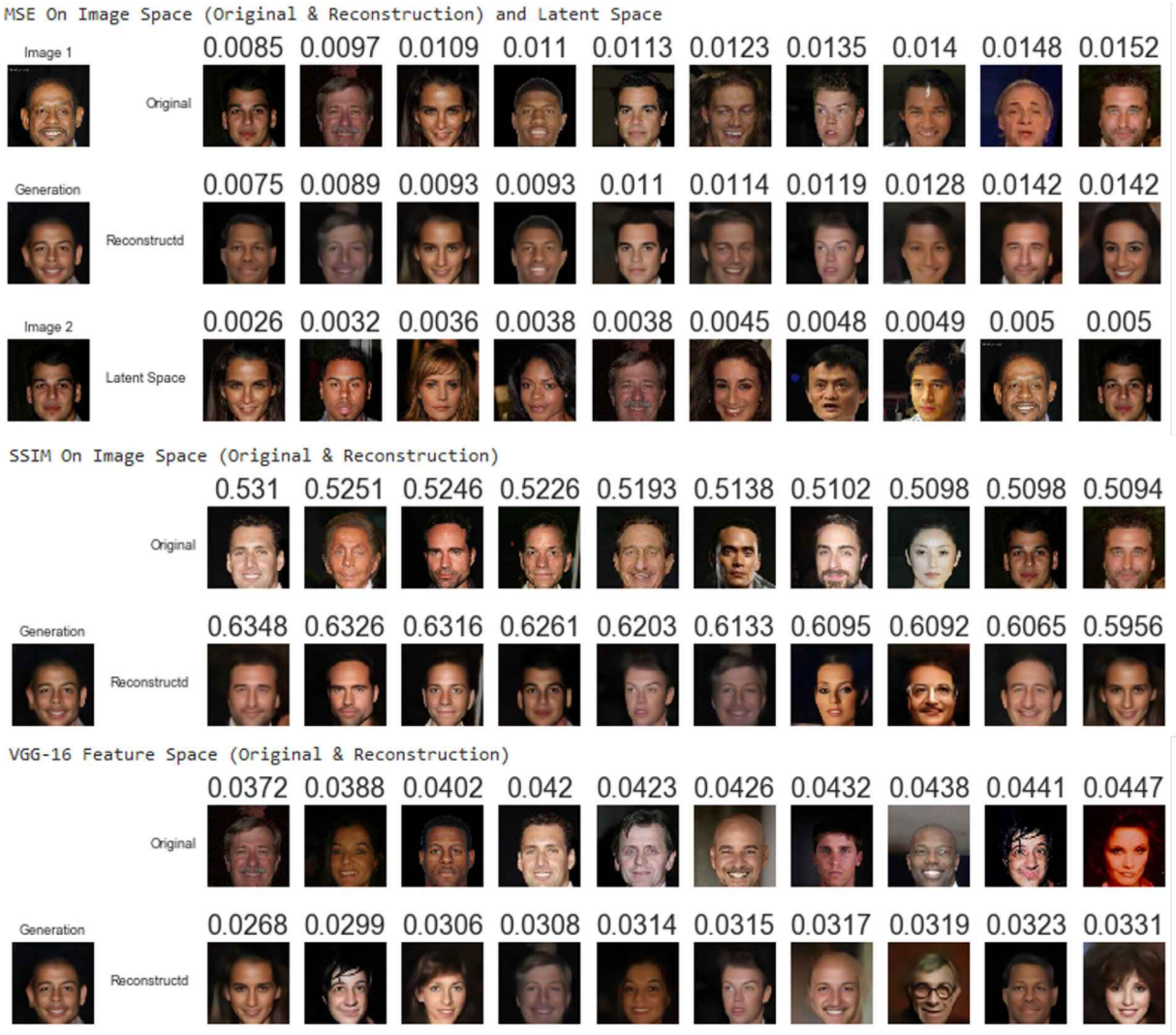


Fig. E.38. CelebA Faces Closest Match Among Training Samples.

Fig. E.39. Anime Faces Closest Match Among Training Samples.

E.2. Closest match across generations

Generations are compared to find closest match among each other. Figs. E.40, E.42 and E.43 illustrate generations are unique when calculating MSE scores in image space, and Figs. E.41, E.44 and E.45 illustrate generations are unique when calculating SSIM scores in image space.

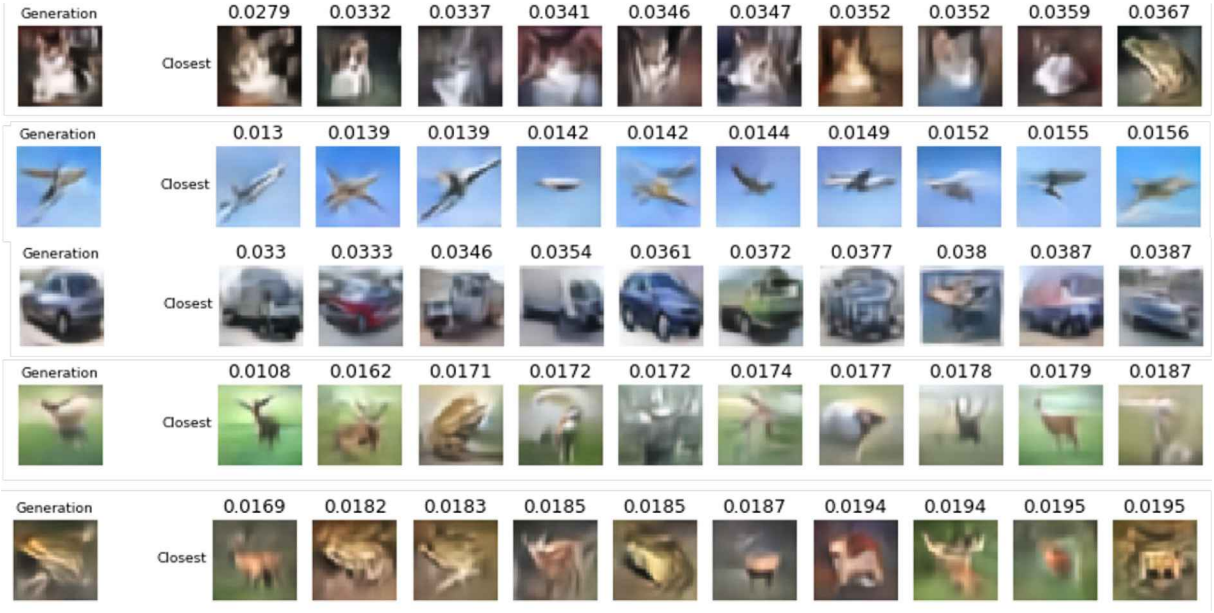


Fig. E.40. Cifar10 Closest Match Among Generations – MSE.

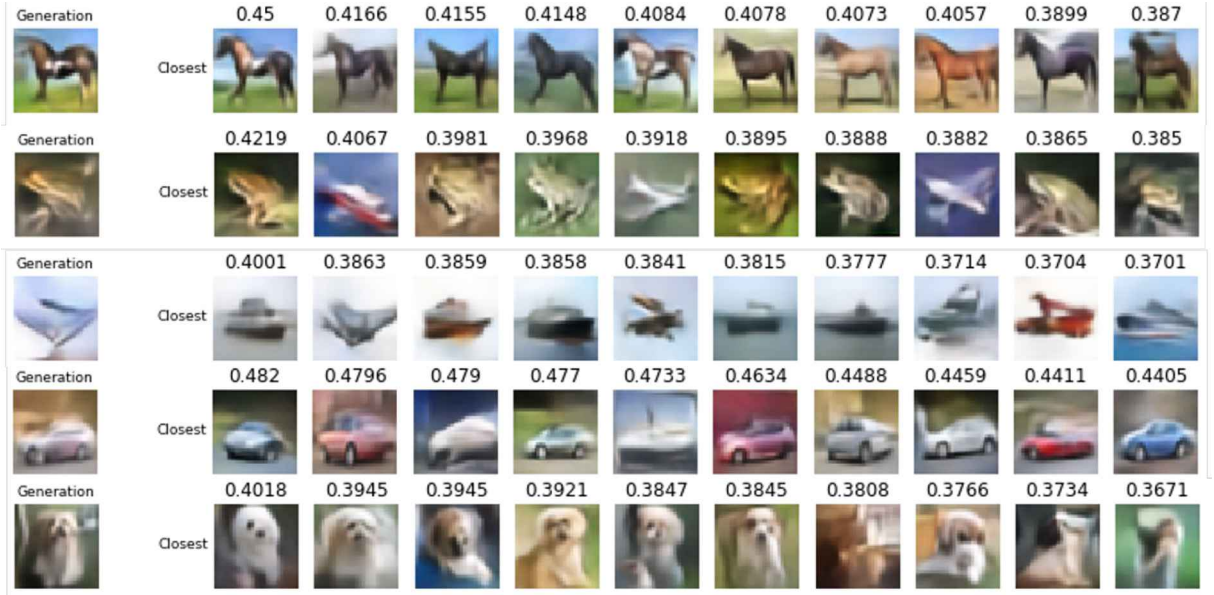


Fig. E.41. Cifar10 Faces Closest Match Among Generations – SSIM.

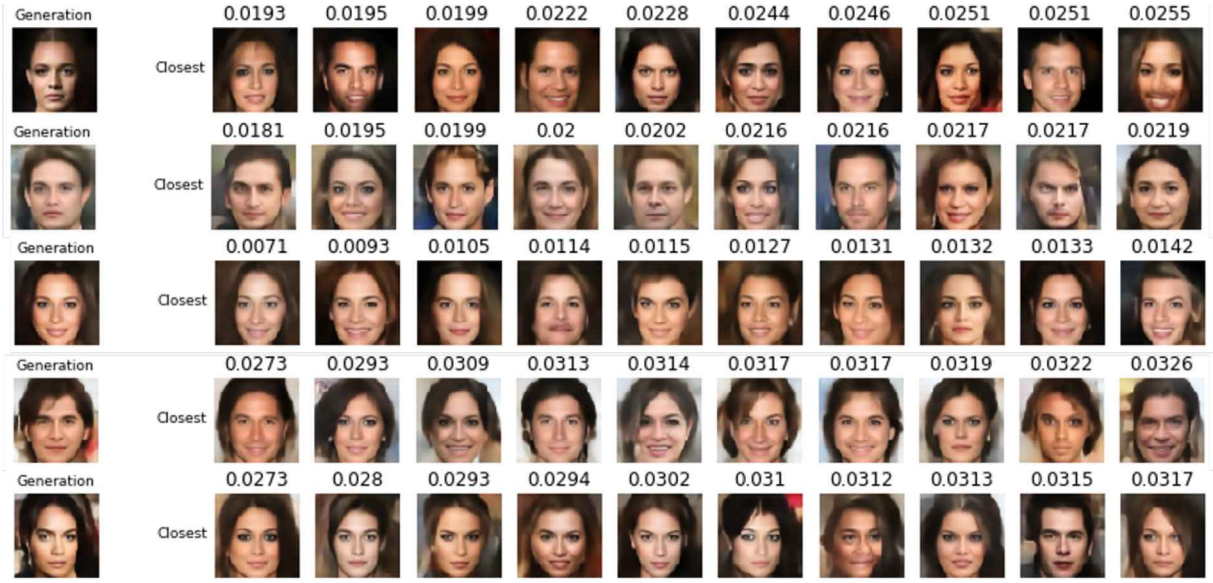


Fig. E.42. CelebA Faces Closest Match Among Generations – MSE.

Fig. E.43. Anime Faces Closest Match Among Generations – MSE.

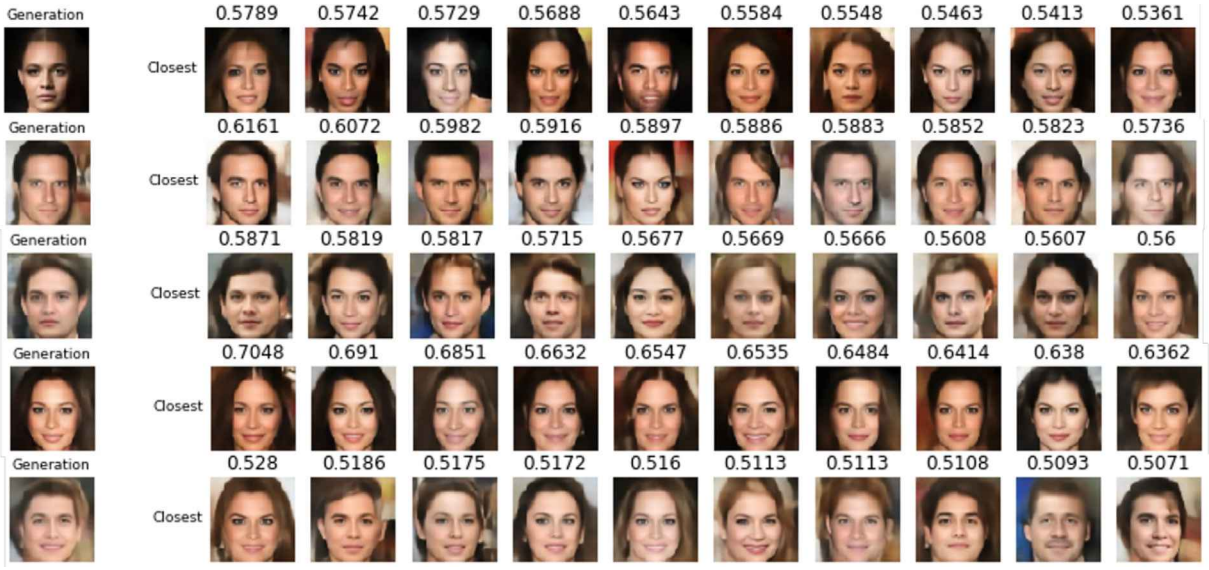


Fig. E.44. CelebA Faces Closest Match Among Generations – SSIM.

Fig. E.45. Anime Faces Closest Match Among Generations – SSIM.

References

[1] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, Commun. ACM 60 (6) (2017) 84–90, https://doi.org/10.1145/3065386, URL: http://doi.acm.org/10.1145/3065386.

[2] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.

[3] G. Huang, Z. Liu, L. Van Der Maaten, K.Q. Weinberger, Densely connected convolutional networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 4700–4708.

[4] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna, Rethinking the inception architecture for computer vision, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 2818–2826.

[5] T. Chen, S. Kornblith, M. Norouzi, G. Hinton, A simple framework for contrastive learning of visual representations, arXiv preprint arXiv:2002.05709. .

[6] D.P. Kingma, M. Welling, Auto-encoding variational bayes, in: 2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14–16, 2014, Conference Track Proceedings, 2014. url:http://arxiv.org/abs/1312.6114. .

[7] I. Tolstikhin, O. Bousquet, S. Gelly, B. Schoelkopf, Wasserstein auto-encoders, in: International Conference on Learning Representations, 2018. URL: https://openreview.net/forum?id=HkL7n1-0b. .

[8] B. Dai, D. Wipf, https://openreview.net/forum?id=B1e0X3C9tQDiagnosing and enhancing VAE models, in: International Conference on Learning Representations, 2019. URL: https://openreview.net/forum?id=B1e0X3C9tQ. .

[9] K. Sohn, H. Lee, X. Yan, Learning structured output representation using deep conditional generative models, in: Advances in Neural Information Processing Systems, 2015, pp. 3483–3491. .

[10] Z. Zhang, R. Zhang, Z. Li, Y. Bengio, L. Paull, Perceptual generative autoencoders, in: Deep Generative Models for Highly Structured Data, ICLR 2019 Workshop, New Orleans, Louisiana, United States, May 6, 2019, 2019. URL: https://openreview.net/forum?id=rkxkr8UKuN. .

[11] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial nets, in: Z. Ghahramani, M. Welling, C. Cortes, N.D. Lawrence, K.Q. Weinberger (Eds.), Advances in Neural Information Processing Systems 27, Curran Associates Inc, 2014, pp. 2672–2680. URL: http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf. .

[12] A. Radford, L. Metz, S. Chintala, Unsupervised representation learning with deep convolutional generative adversarial networks, in: ICLR, 2016.

[13] M. Arjovsky, S. Chintala, L. Bottou, Wasserstein generative adversarial networks, in: D. Precup, Y.W. Teh (Eds.), Proceedings of the 34th International Conference on Machine Learning, Vol. 70 of Proceedings of Machine Learning Research, PMLR, International Convention Centre, Sydney, Australia, 2017, pp. 214–223. URL: http://proceedings.mlr.press/v70/arjovsky17a.html. .

[14] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, A.C. Courville, Improved training of wasserstein gans, in: Advances in Neural Information Processing Systems, 2017, pp. 5767–5777. .
[15] D. Berthelot, T. Schumm, L. Metz, BEGAN: boundary equilibrium generative adversarial networks, CoRR abs/1703.10717. arXiv:1703.10717. url:http://arxiv.org/abs/1703.10717. .
[16] L. Dinh, J. Sohl-Dickstein, S. Bengio, Density estimation using real NVP, in: 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24–26, 2017, Conference Track Proceedings, 2017. URL: https://openreview.net/forum?id=HkpbnH9lx. .
[17] D.P. Kingma, P. Dhariwal, Glow: Generative flow with invertible 1×1 convolutions, in: Advances in Neural Information Processing Systems, 2018, pp. 10215–10224..
[18] A. Ng et al., Sparse autoencoder, CS294A Lecture Notes 72 (2011) (2011) 1–19.
[19] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, MIT Press, 2016, URL: http://www.deeplearningbook.org..
[20] D. Berthelot, C. Raffel, A. Roy, I.J. Goodfellow, Understanding and improving interpolation in autoencoders via an adversarial regularizer, CoRR abs/1807.07543. arXiv:1807.07543. URL: http://arxiv.org/abs/1807.07543. .
[21] T. Sainburg, M. Thielk, B. Theilman, B. Migliori, T. Gentner, Generative adversarial interpolative autoencoding: adversarial training on latent space interpolations encourage convex latent distributions, CoRR abs/1807.06650. arXiv:1807.06650. URL: http://arxiv.org/abs/1807.06650. .
[22] A. Makhzani, B.J. Frey, k-sparse autoencoders, in: 2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14–16, 2014, Conference Track Proceedings, 2014. URL: http://arxiv.org/abs/1312.5663. .
[23] Y. Bengio, G. Mesnil, Y. Dauphin, S. Rifai, Better mixing via deep representations, in: Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16–21 June 2013, 2013, pp. 552–560. URL: http://jmlr.org/proceedings/papers/v28/bengio13.html. .
[24] T. Salimans, I.J. Goodfellow, W. Zaremba, V. Cheung, A. Radford, X. Chen, Improved techniques for training gans, in: Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5–10, 2016, Barcelona, Spain, 2016, pp. 2226–2234. .
[25] A. Razavi, A. van den Oord, O. Vinyals, Generating diverse high-fidelity images with VQ-VAE-2, CoRR abs/1906.00446. arXiv:1906.00446. URL: http://arxiv.org/abs/1906.00446. .
[26] M. Mirza, S. Osindero, Conditional generative adversarial nets, arXiv preprint arXiv:1411.1784. .
[27] P. Isola, J.-Y. Zhu, T. Zhou, A.A. Efros, Image-to-image translation with conditional adversarial networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 1125–1134.
[28] O. Ivanov, M. Figurnov, D.P. Vetrov, Variational autoencoder with arbitrary conditioning, in: 7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6–9, 2019, 2019. URL: https://openreview.net/forum?id=SyxtJh0qYm. .
[29] A. Brock, J. Donahue, K. Simonyan, Large scale GAN training for high fidelity natural image synthesis, in: 7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6–9, 2019, 2019. URL: https://openreview.net/forum?id=B1xsqj09Fm. .
[30] T. Karras, T. Aila, S. Laine, J. Lehtinen, Progressive growing of gans for improved quality, stability, and variation, CoRR abs/1710.10196. arXiv:1710.10196. URL: http://arxiv.org/abs/1710.10196. .
[31] Z. Wang, A.C. Bovik, H.R. Sheikh, E.P. Simoncelli, Image quality assessment: from error visibility to structural similarity, IEEE Transactions on Image Processing 13 (4) (2004) 600–612.
[32] J. Johnson, A. Alahi, L. Fei-Fei, Perceptual losses for real-time style transfer and super-resolution, in: European Conference on Computer Vision, 2016.
[33] A. Pownuk, V. Kreinovich, Combining Interval, Probabilistic, and Other Types of Uncertainty in Engineering Applications, Vol. 773 of Studies in Computational Intelligence, Springer, 2018. doi:10.1007/978-3-319-91026-0. URL: https://doi.org/10.1007/978-3-319-91026-0. .
[34] Y. LeCun, C. Cortes, C. Burges, Mnist handwritten digit database, ATT Labs [Online]. Available: http://yann. lecun. com/exdb/mnist 2. .
[35] H. Xiao, K. Rasul, R. Vollgraf, Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, CoRR abs/1708.07747. arXiv:1708.07747. URL: http://arxiv.org/abs/1708.07747. .
[36] A. Krizhevsky, Learning multiple layers of features from tiny images, Tech. rep. (2009).
[37] Z. Liu, P. Luo, X. Wang, X. Tang, Deep learning face attributes in the wild, in: 2015 IEEE International Conference on Computer Vision (ICCV), 2015, pp. 3730–3738.
[38] Anonymous, D. community, G. Branwen, A. Gokaslan, Danbooru 2018: A large-scale crowdsourced and tagged anime illustration dataset, URL: https://www.gwern.net/Danbooru2018 (January 2019). .
[39] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, 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.
[40] K. Shmelkov, C. Schmid, K. Alahari, How good is my gan?, in: Computer Vision - ECCV 2018–15th European Conference, Munich, Germany, September 8–14, 2018, Proceedings, Part II, 2018, pp. 218–234. doi:10.1007/978-3-030-01216-8_14. URL: https://doi.org/10.1007/978-3-030-01216-8_14. .
[41] K. Zheng, Y. Cheng, X. Kang, H. Yao, T. Tian, Conditional introspective variational autoencoder for image synthesis, IEEE Access 8 (2020) 153905–153913, https://doi.org/10.1109/ACCESS.2020.3018228.

Saisubramaniam Gopalakrishnan received the Master of Technology degree with specialization in Knowledge Engineering from Institute of Systems Science, National University of Singapore in 2019. He is currently a Research Engineer at Machine Intellection, Institute for Infocomm Research, Agency for Science, Technology, and Research, Singapore. He was a Software Analyst in Aspire Systems India Pvt. Ltd. India, working on big data technologies from 2015 to 2017. His research interests are in computer vision, deep learning, continual learning, inverse design of experiments, and big data engineering.

Pranshu Ranjan Singh received the M.Tech. degree in Knowledge Engineering from National University of Singapore, Singapore in 2019. Prior to that he received the B.Tech. degree in Computer Science and Engineering from PDPM Indian Institute of Information Technology, Design and Manufacturing, Jabalpur, India in 2017. He is currently a Research Engineer at Institute for Infocomm Research, a research wing of Agency for Science, Technology, and Research, Singapore. His research interests are in computer vision, deep learning, and design of experiments, image generations, and continual learning.

Yasin Yazici received the Ph.D. degree in Computer Vision from Nanyang Technological University, Singapore, in 2020. Currently, he is a research Scientist at Machine Intellection, Institute for Infocomm Research, a research wing of Agency for Science, Technology, and Research, Singapore. His research interest are in Generative Models, Unsupervised Learning, and Deep Learning.

Chuan Sheng Foo received his BS, MS and PhD degrees in Computer Science from Stanford University, in 2008, 2012 and 2017, respectively. He leads a research group at the Institute for Infocomm Research, A*STAR, Singapore that focuses on developing data-efficient deep learning algorithms that can learn from less labeled data

Vijay Chandrasekhar completed his B.S and M.S. from Carnegie Mellon University (2002–2005), and Ph.D. in Electrical Engineering from Stanford University (2006–2013). His research contributions span deep learning and machine learning algorithms, computer vision, large-scale image and audio search, augmented reality and deep learning hardware. He has published more than 100 papers/MPEG contributions in a wide range of top-tier journals/conferences like IJCV, ICCV, CVPR, IEEE SPM, ACM MM, IEEE TIP, DCC, ISMIR, MPEG-CDVS, ICLR, etc, and has filed 7 US patents (4 granted, 3 pending). His Ph.D. work on feature compression led to the widely adopted MPEG-CDVS (Compact Descriptors for Visual Search) standard, which he actively contributed from 2010–2013. He was awarded the A*STAR National Science Scholarship (NSS) in 2002. He is an IEEE Senior Member, and was nominated for the Young Scientist Award at the national level from the Singapore National Academy of Sciences in 2017.

ArulMurugan Ambikapathi (S'02-M'11-SM'20) received the Ph.D. degree from the Institute of Communications Engineering (ICE), National Tsing Hua University (NTHU), Hsinchu, Taiwan, in 2011. Prior to that he received the B.E. degree in electronics and communication engineering from Bharathidasan University, India, in 2003, the M.E degree in communication systems from Anna University, India, in 2005. He is currently a Scientist and Group Lead at Machine Intellection, Institute for Infocomm Research, a research wing of Agency for Science, Technology, and Research, Singapore. He was a Team lead and Senior Algorithm Engineer at Utechzone Co. Ltd., Taipei, Taiwan, from Sep. 2014 to June 2018, where he found and lead an AI team to develop the first AI-AOI Solution in Taiwan. Earlier, he was a Postdoctoral Research Fellow with ICE, NTHU, from Sep. 2011 to Aug. 2014. His research interests are in computer vision, deep learning theories and applications, image generations, and hyperspectral and biomedical image analysis. Dr. Ambikapathi was the recipient of Gold and Silver medals for academic excellence in his B.E and M.E programs, respectively. He was also the recipient of the NTHU Outstanding Student Scholarship award for two consecutive years (2009 and 2010). He was a co-recipient of the IEEE Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing 2011, Best Paper Award. He was awarded 'The Best Ph.D Thesis Award? from IEEE GeoScience and Remote Sensing Society, Taipei Chapter. Based on his industrial research in AI, he have 7 US Patents to his name. He is a senior member of IEEE and is currently leading a research team that provides AI based advanced industrial solutions.