

# Supplementary Material : Lifelong Learning in StyleGAN through Latent Subspaces

Anonymous authors

Paper under double-blind review

## A Overview

Recall from our main manuscript that we present a lightweight expansion-based method called StyleCL for generative continual learning with StyleGAN. Unlike previous approaches that perform a transformation on either the feature maps or the weight parameters, we efficiently leverage the latent space of StyleGAN. The proposed method ensures comparable or better performance while using less computational and memory overhead than earlier approaches.

We include additional details in the supplementary material to keep the overall manuscript self-contained. The details comprise implementation details, additional experiments, additional ablation studies, and insights into our results, which could not be self-contained due to restrictions in the length of the main manuscript.

## B Preliminaries: StyleGAN

StyleGAN revolutionized generative adversarial networks by introducing a novel architecture that enables high-quality image synthesis with extensive control over stylistic features. At its core, StyleGAN employs a mapping network  $f$  that transforms an input latent code  $\mathbf{z} \in \mathbb{R}^{512}$ , sampled from a Gaussian prior  $P(\mathbf{Z})$ , into an intermediate latent code  $\mathbf{w} \in \mathbb{R}^{512} \in \mathcal{W}$ . This intermediate code  $\mathbf{w}$  is then replicated  $N_\ell$  times and injected into each layer of the generator  $G$ .

Unlike traditional generative models which pass the latent code  $\mathbf{z}$  directly to the input layer, StyleGAN utilizes a constant input  $\mathbf{c} \in \mathbb{R}^{4 \times 4 \times 512}$ . This constant input is progressively transformed through layers of increasing resolution to synthesize the final image. To enhance the overall textural quality, StyleGAN also introduces noise inputs at each layer, independently sampled from a Gaussian prior.

### B.1 Limitations of the StyleGAN

Despite its success, the original StyleGAN exhibited issues related to unnatural visual artifacts and inconsistencies, primarily due to its use of Adaptive Instance Normalization (AdaIN) for injecting style information at each layer. While AdaIN provided flexible control over image attributes, it also introduced undesirable artifacts and made smooth style transitions difficult to manage.

### B.2 Advancements in StyleGAN2

Addressing these limitations, StyleGAN2 presents an enhanced architecture that improves image generation quality and resolves the artifacts observed in the original model. A central innovation in StyleGAN2 is the replacement of AdaIN with a technique called *weight demodulation*. This method modulates the convolutional weights to prevent the introduction of artifacts, ensuring that output activations maintain the desired standard deviation.

Weight demodulation operates by first assuming that the input activations are independent and identically distributed (i.i.d.) random variables with unit standard deviation. After modulation and convolution, the standard deviation of the output activations is computed as:

$$\sigma_j = \sqrt{\sum_{i,k} w'_{ijk}{}^2}, \quad (1)$$

where  $w'_{ijk}$  are the modulated weights. To normalize the outputs back to unit standard deviation, each output feature map  $j$  is scaled, or “demodulated,” by  $1/\sigma_j$ . Alternatively, this normalization can be incorporated directly into the convolutional weights:

$$w''_{ijk} = \frac{w'_{ijk}}{\sqrt{\sum_{i,k} w'_{ijk}{}^2 + \epsilon}}, \quad (2)$$

with  $\epsilon$  being a small constant added to prevent numerical instability. This weight adjustment minimizes artifacts and improves the quality of the generated images.

Additionally, StyleGAN2 refines the training process by adopting a fixed-resolution architecture instead of the progressive growing technique used in the original StyleGAN. This change reduces training instability and eliminates phase artifacts, leading to more stable training and higher-quality images at a consistent resolution.

### B.3 Regularization Techniques in StyleGAN2

To further enhance the smoothness of the latent space and improve image quality, StyleGAN2 introduces additional regularization schemes. The Perceptual Path Length regularizer (PPL) ensures that the latent subspace induced by the generator is smooth by enforcing that a fixed step in the latent space results in a consistent magnitude change in the image space. Mathematically, the PPL regularizer is defined as:

$$\mathcal{L}_{\text{PPL}} = \mathbb{E}_{\mathbf{w}, \mathbf{y} \sim \mathcal{N}(0, \mathbf{I})} \left( \|\mathbf{J}_{\mathbf{w}}^\top \mathbf{y}\|_2 - a \right)^2, \quad (3)$$

where  $\mathbf{y}$  represents images with normally distributed pixel intensities,  $a \in \mathbb{R}$  is a global parameter indicating the desired gradient scale, and  $\mathbf{J}_{\mathbf{w}}^\top = \left( \frac{\partial \mathcal{G}(\mathbf{w})}{\partial \mathbf{w}} \right)^\top$  is the Jacobian transpose of the generator function  $\mathcal{G}$ .

Furthermore, StyleGAN2 applies  $\mathcal{R}_1$  regularization to improve convergence during training:

$$\mathcal{L}_{\mathcal{R}_1} = \mathbb{E}_{\mathbf{x} \sim p_{\text{data}}(\mathbf{x})} \left[ \|\nabla_{\mathbf{x}} D_{\psi}(\mathbf{x})\|^2 \right], \quad (4)$$

where  $D_{\psi}$  is the discriminator parameterized by  $\psi$ , and  $p_{\text{data}}(\mathbf{x})$  is the data distribution. This regularization penalizes the gradient norm of the discriminator with respect to real data samples, promoting smoother gradients and more stable training.

## C Results

### C.1 Implementation Details

We adapted the author’s Pytorch implementation of StyleGAN2-ADA for all our experimentations. All the datasets were scaled to 256 x 256 resolution. The number of dictionary vectors per style block was chosen as 16 for all our experiments. All other hyperparameters were the same as those of the original implementation of StyleGAN2-ADA. We reimplemented all the baselines like CAMGAN, GAN-Memory using StyleGAN2 architecture for fair comparison.

The calculation of FID (Fréchet Inception Distance) involves utilizing the image embeddings extracted by the InceptionV3 model. On the other hand, Density and Coverage metrics rely on the embeddings provided by the VGG-16 model.

Layer	Output shape
synthesis.b4.conv1	[4, 512, 4, 4]
synthesis.b4.torgb	[4, 3, 4, 4]
synthesis.b4:0	[4, 512, 4, 4]
synthesis.b4:1	[4, 512, 4, 4]
synthesis.b8.conv0	[4, 512, 8, 8]
synthesis.b8.conv1	[4, 512, 8, 8]
synthesis.b8.torgb	[4, 3, 8, 8]
synthesis.b8:0	[4, 512, 8, 8]
synthesis.b8:1	[4, 512, 8, 8]
synthesis.b16.conv0	[4, 512, 16, 16]
synthesis.b16.conv1	[4, 512, 16, 16]
synthesis.b16.torgb	[4, 3, 16, 16]
synthesis.b16:0	[4, 512, 16, 16]
synthesis.b16:1	[4, 512, 16, 16]
synthesis.b32.conv0	[4, 512, 32, 32]
synthesis.b32.conv1	[4, 512, 32, 32]
synthesis.b32.torgb	[4, 3, 32, 32]
synthesis.b32:0	[4, 512, 32, 32]
synthesis.b32:1	[4, 512, 32, 32]
synthesis.b64.conv0	[4, 512, 64, 64]
synthesis.b64.conv1	[4, 512, 64, 64]
synthesis.b64.torgb	[4, 3, 64, 64]
synthesis.b64:0	[4, 512, 64, 64]
synthesis.b64:1	[4, 512, 64, 64]
synthesis.b128.conv0	[4, 256, 128, 128]
synthesis.b128.conv1	[4, 256, 128, 128]
synthesis.b128.torgb	[4, 3, 128, 128]
synthesis.b128:0	[4, 256, 128, 128]
synthesis.b128:1	[4, 256, 128, 128]
synthesis.b256.conv0	[4, 128, 256, 256]
synthesis.b256.conv1	[4, 128, 256, 256]
synthesis.b256.torgb	[4, 3, 256, 256]
synthesis.b256:0	[4, 128, 256, 256]
synthesis.b256:1	[4, 128, 256, 256]

Table 1: Output Shape and Architecture of the Synthesis network of StyleGAN2 Generator

## C.2 Metrics

In the experiments, we evaluate the performance via the following metrics.

- **Frechet Inception Distance** Quantifies the discrepancy between the distributions of real and fake images, represented by deep embeddings. Both distributions are approximated by Gaussians, and the Wasserstein distance between them is computed. Lower FID scores indicate that the generated images are more similar to the real images, reflecting better quality of the generated images.

$$\text{FID} = \|\mu_r - \mu_g\|_2^2 + \text{Tr} \left( \Sigma_r + \Sigma_g - 2(\Sigma_r \Sigma_g)^{1/2} \right) \quad (5)$$

- **Density:** This metric assesses how well the GAN can replicate the density of the real data distribution. High-density scores indicate that the generative model is capable of producing samples that are very close to those from the real dataset, essentially meaning that the generated samples are indistinguishable from real ones in terms of how they are distributed in the data space.

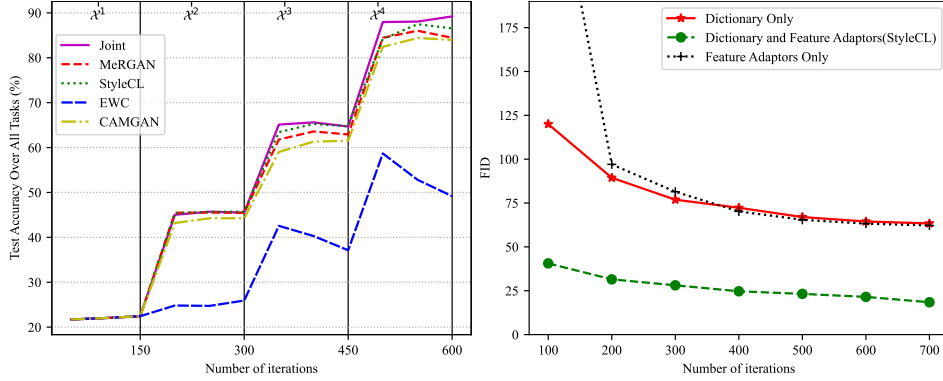


Figure 1: (Left) Classification accuracy on the lifelong classification problem. (Right) Quantitative evaluation of the contribution of different components of StyleCL for image generation.

$$\text{density} := \frac{1}{kM} \sum_{j=1}^M \sum_{i=1}^N 1_{Y_j \in B(X_i, \text{NND}_k(X_i))} \quad (6)$$

- **Coverage:** It measures the extent to which the generated samples cover the variation present in the real data. It measures the fraction of real samples whose neighbourhoods contain at least one fake sample. Coverage is bounded between 0 and 1. It assesses the diversity of the generated samples. A high coverage score means that the GAN generates a wide variety of samples that capture the different modes of real data distribution. This is important because a GAN that only generates high-quality but not diverse samples might be overfitting to a small subset of the data distribution. Coverage is defined as

$$\text{coverage} := \frac{1}{N} \sum_{i=1}^N 1_{\exists j \text{ s.t. } Y_j \in B(X_i, \text{NND}_k(X_i))}. \quad (7)$$

### C.3 Additional Results on StyleCL

We could only display a limited number of samples generated using StyleCL in the main paper due to space constraints. Hence, we present additional samples to strengthen the qualitative evaluation of the proposed method in Fig. 6, Fig. 7, and Fig. 8. We observe that the qualitative analysis supports the quantitative metrics in terms of FID scores, Density, and Coverage metrics. We further provide the absolute value of parameters (in Millions(M)), FLOPs (measured in gigaFLOPs(G)), and additional information Tab. 2.

Algorithm	Parameters (M)	FLOPs (G)	Parameters Increase per Task (M) ↓	FLOPs Increase (%) ↓
MerGAN	24.94	14.00	-	-
GAN Memory	29.15	16.19	4.21	15.7
CAM-GAN	26.46	17.29	1.52	23.32
<b>StyleCL</b>	26.02	14.57	<b>1.08</b>	<b>4.1</b>

Table 2: Comparison of our approach against the contemporaries GAN Memory, CAM-GAN, and MerGAN with respect to Parameters, FLOPs, Parameter reduction factor, and percentage increase in FLOPs.



#### C.4 Additional Results on Forward Transfer

In the main text, the forward transfer capabilities of StyleCL were evaluated. This section extends that evaluation to perceptually distinct tasks, affirming the model’s adaptability. Our investigation includes datasets with unique visual properties, such as Flowers, LSUN Church  $\mathcal{X}^2$ , LSUN Cats  $\mathcal{X}^3$ , Brain MRI  $\mathcal{X}^4$ , and Chest X-Ray  $\mathcal{X}^5$ . As Table 3 shows, StyleCL undergoes negative forward transfer when  $\text{sim}(t, k) \leq 0$ , a scenario indicative of low task similarity resulting in a performance drop in the absence of transfer learning. This underscores the crucial role of transfer learning in achieving consistent performance across varied datasets. Interestingly, the effective forward transfer between perceptually distinct tasks such as  $\mathcal{X}^2$  and  $\mathcal{X}^1$ , suggests that utilizing the latent space to ascertain task similarity can be more effective than conventional metrics like FID for identifying the most suitable task for transfer.

Table 3: Quantitative comparison of StyleCL performance with and without forward transfer on perceptually diverse datasets.

	$\mathcal{X}^2$	$\mathcal{X}^3$	$\mathcal{X}^4$	$\mathcal{X}^5$
StyleCL w/o Transfer	17.36	<b>34.43</b>	<b>29.42</b>	<b>25.83</b>
StyleCL with Transfer	<b>22.48</b>	36.18	35.38	29.12
$\text{sim}(t, k)$	0.58	0.09	-0.02	0.01

#### C.5 Lifelong Classification

The conditional variant of StyleCL can serve as a generative replay buffer within discriminative continual learning. To gauge its effectiveness in aiding the classifier model during lifelong learning, we conduct tests within the generative replay paradigm. Specifically, we explore conditional data generation in generative replay experiments, applying it to the class-incremental setup designed for classification tasks. Our approach involves selecting images of fish, birds, snakes, and dogs from the ImageNet dataset and creating 4 distinct streams of tasks as in CAMGAN. Each task is framed as a 6-classification problem, such as identifying 6 different types of birds in the bird task. After completing each task  $t$ , the learned classifier is required to accurately classify all the categories observed so far, up to and including task  $t$ . The quantitative comparison of StyleCL with regularization-based baselines EWC and Replay-based methods such as MerGAN, and CAM-GAN for lifelong classification tasks is demonstrated in Fig. 1 Left. It can be observed that the StyleCL performs better compared to the baselines when used for lifelong classification tasks, because of its superior generative capability. The method that closely competes with StyleCL for lifelong classification tasks is MerGAN. However, the performance of MerGAN degrades as the task sequence increase since MerGAN learns new task at the expense of forgetting previous tasks. Generated samples using StyleCL are given in Fig. 2. The upper bound for classification performance denoted as Joint is obtained by using all the data till the current task to train the classifier. The performance of StyleCL is seen to be close to this upper bound.

#### C.6 StyleCL on longer task Sequence

We illustrate the performance of StyleCL on long task sequence by considering each class of CIFAR10 as a task and continually arriving. The results are summarised in the table below. As observed StyleCL performs well even in this extended task sequence showcasing its robustness for long task sequences.

Task ID	1	2	3	4	5	6	7	8	9	10
FID	11.5	13.5	13.2	14.0	12.6	10.4	15.3	11.2	12.2	14.3

Table 4: Performance of StyleCL on long task-sequence



Figure 2: **Conditional Generation Qualitative results:** Randomly generated samples by StyleCL and CAM-GAN for lifelong classification. In each image, each row represents a class, six classes per task.

## C.7 Interpolation

### C.7.1 Interpolation within a task

The smoothness of the learned latent subspace is one of the properties enforced in StyleCL. We conduct a qualitative evaluation to support this claim which is demonstrated in Fig. 9 and Fig. 10 for Anime and Flowers datasets respectively. We sample two vectors randomly from the latent subspace learned for each dataset and perform linear interpolation to generate images corresponding to these vectors along the interpolation path. It can be observed that the linear interpolation in latent space produces a smooth transition in pixel space as well, demonstrating the smoothness of the latent subspace that has been learned.

### C.7.2 Interpolation across tasks

We perform linear interpolation among various task generation processes on randomly sampled vectors from the latent subspaces learned for two distinct datasets. We perform the analysis for dataset pairs, namely (Brain MRI, Chest X-Ray) and (Flowers, Anime) which is demonstrated in Fig. 11 and Fig. 12 respectively. The linear interpolation is performed on the latent vectors as well as the task-specific parameters, while the other parameters remain fixed. As observed in Fig. 11 and Fig. 12, facial artifacts arise in the generated images when moving away from learned latent subspaces, and the image quality is drastically degraded. This could be attributed to the fact that the generation process is influenced by the parameters of the base task, CelebA-HQ.

## C.8 Overcoming task ID constraints with StyleCL

The previously defined problem setting assumes a sequential arrival of datasets (or tasks) with unique task IDs. However, in real-life scenarios, this assumption may not always hold. For instance, data collection from multiple sources can occur simultaneously, resulting in a set of data without task distinction. Specifically, we address scenarios where task  $t$  comprises contributions from  $Q$  datasets, and task ID is not available. We

operate under the assumption that  $Q$  is known apriori. We demonstrate that StyleCL can be seamlessly extended to accommodate these scenarios. We initialize  $Q$  sets of dictionary vectors and feature adaptors. The intuition is that this modelling choice forces each of the  $q$  latent adaptors to concentrate on latent vectors originating from separate regions in the latent space. As a result, the model would effectively allocate distinct data sources to distinct dictionaries. During training, we exclusively employ the  $q^{th}$  feature adaptor set for latent vectors generated from  $q^{th}$  dictionary where  $q \in \{1, 2, \dots, Q\}$ . This ensures that  $q^{th}$  feature adaptors capture task-specific knowledge exclusively for  $q^{th}$  dataset.

Task	Datasets	FID
Flowers & Brain-MRI	Flowers	24.38
	Brain-MRI	35.22
LSUN Church & Anime	LSUN Church	23.48
	Anime	14.63

Table 5: Performance of StyleCL on the task ID free setting on two data mixtures.

## D Additional Analysis and Ablation Studies

In the main text, we have discussed a few analyses and ablations, in this section, we provide additional results and ablation studies for interested readers.

### D.1 Effect of dimensionality of learned Latent subspace

To investigate the impact of the dimensionality of the learned latent space (number of dictionary vectors per style block -  $K$ ) on the quality of generated images, we conducted experiments with StyleCL using only the latent subspace, varying the number of dictionary vectors. The results of this analysis are depicted in Fig. 3. Notably, the generation quality exhibits a remarkable improvement as the dimensionality of the latent subspace increases, till an optimal point. However, beyond this optimum, the generation quality starts to degrade, as illustrated in Fig. 3. We hypothesize that the additional dimensions introduce noise that interferes with the original latent vector, affecting the generation quality. We further observed that the optimal number of dictionary vectors per style block was consistent across a wide range of datasets. Thus, we use  $K = 16$  for all the datasets.

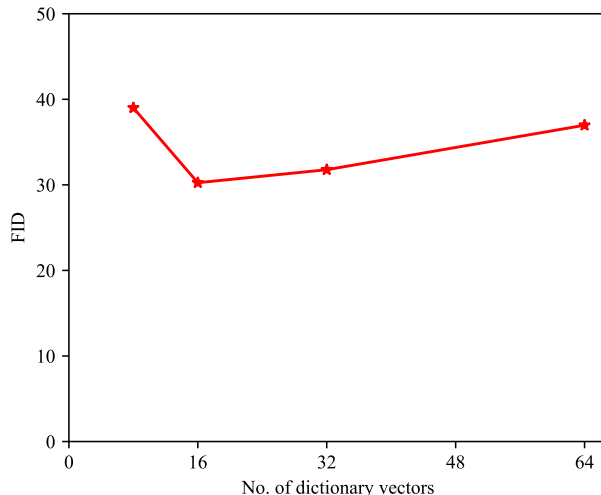


Figure 3: Quantitative ablation of the effect of dictionary size( $K$ ) on the generation quality for the Anime dataset.

## D.2 Ablations over the latent dictionary and feature adaptors

In Section 5.2 of the main text, we presented an ablation study on the latent dictionary and feature adaptors using the Flowers Dataset. To provide a more comprehensive analysis, we have extended this ablation study to multiple datasets. In this section, we present the results of this ablation study across these datasets.

The ablation study involves systematically removing or modifying components of the latent dictionary and feature adaptors and evaluating the impact on the model’s performance. By conducting this study on multiple datasets, we aim to assess the generalizability of our findings and the effectiveness of the proposed method across different domains. By conducting this ablation study on multiple datasets, we ensure that our findings

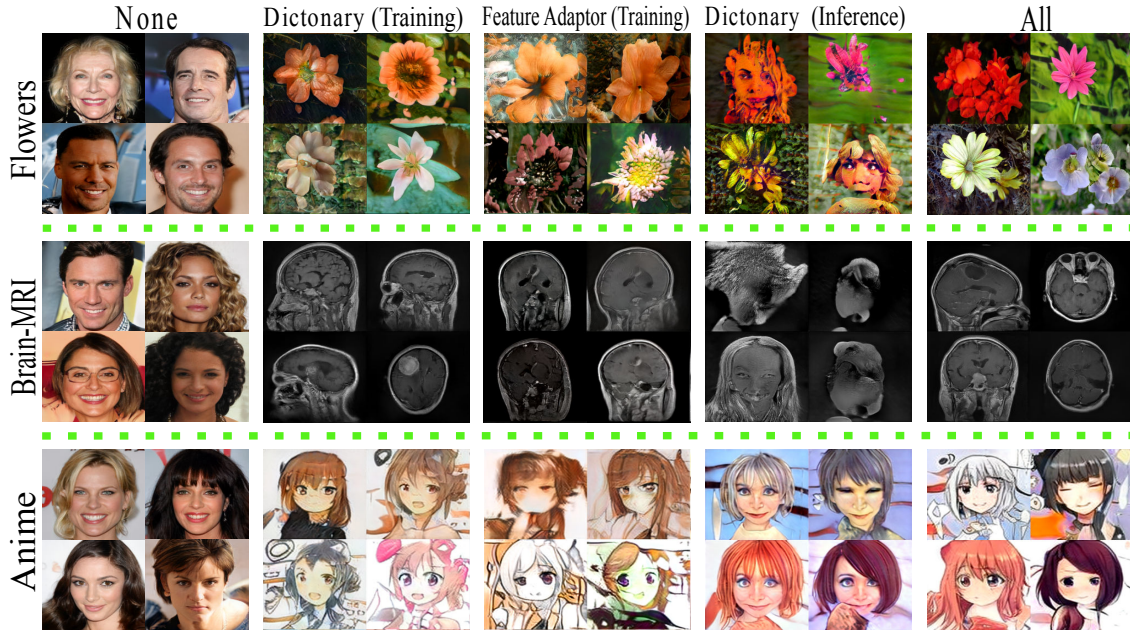


Figure 4: Ablation over feature adaptors and Dictionary across Anime, Brain-MRI, and Flowers dataset.

are not limited to a specific dataset or domain. The results provide a broader perspective on the performance and robustness of our proposed method, allowing for a more comprehensive evaluation and understanding of its capabilities.

## E Limitations

The proposed method, although efficient for generative continual learning (CL), is dependent on the nature of the base task and the similarity between the base task and the continual task. This dependency could lead to diminished performance. Furthermore, the performance gains achieved by ensuring positive forward transfer may be insignificant if the current task has little similarity with previous tasks.

## F Reproducibility

To facilitate reproducibility, we are attaching the code along with supplementary material as a zip file. The necessary requirements file is also attached with it. We intend to release a more user-friendly version of the code publicly along with the pre-trained models post-acceptance. All the datasets used are publicly available.



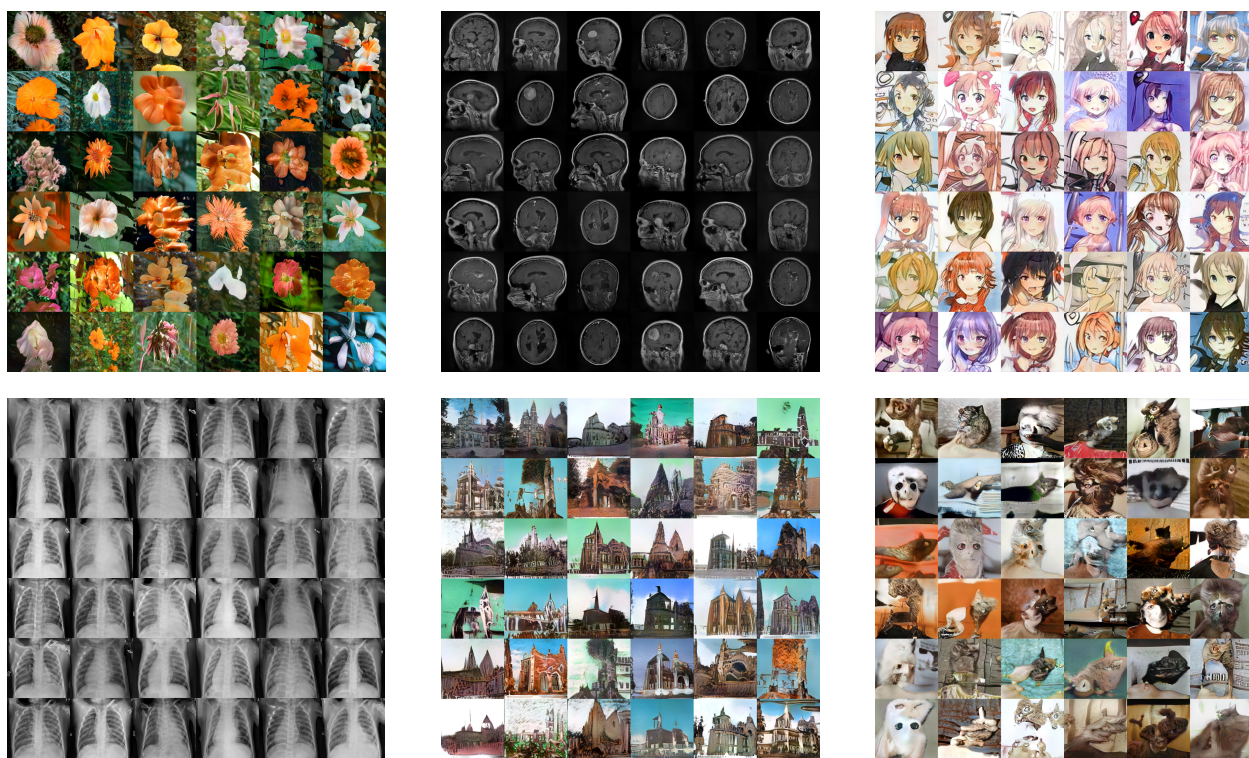


Figure 5: Generated samples by dictionary learning alone in StyleCL.

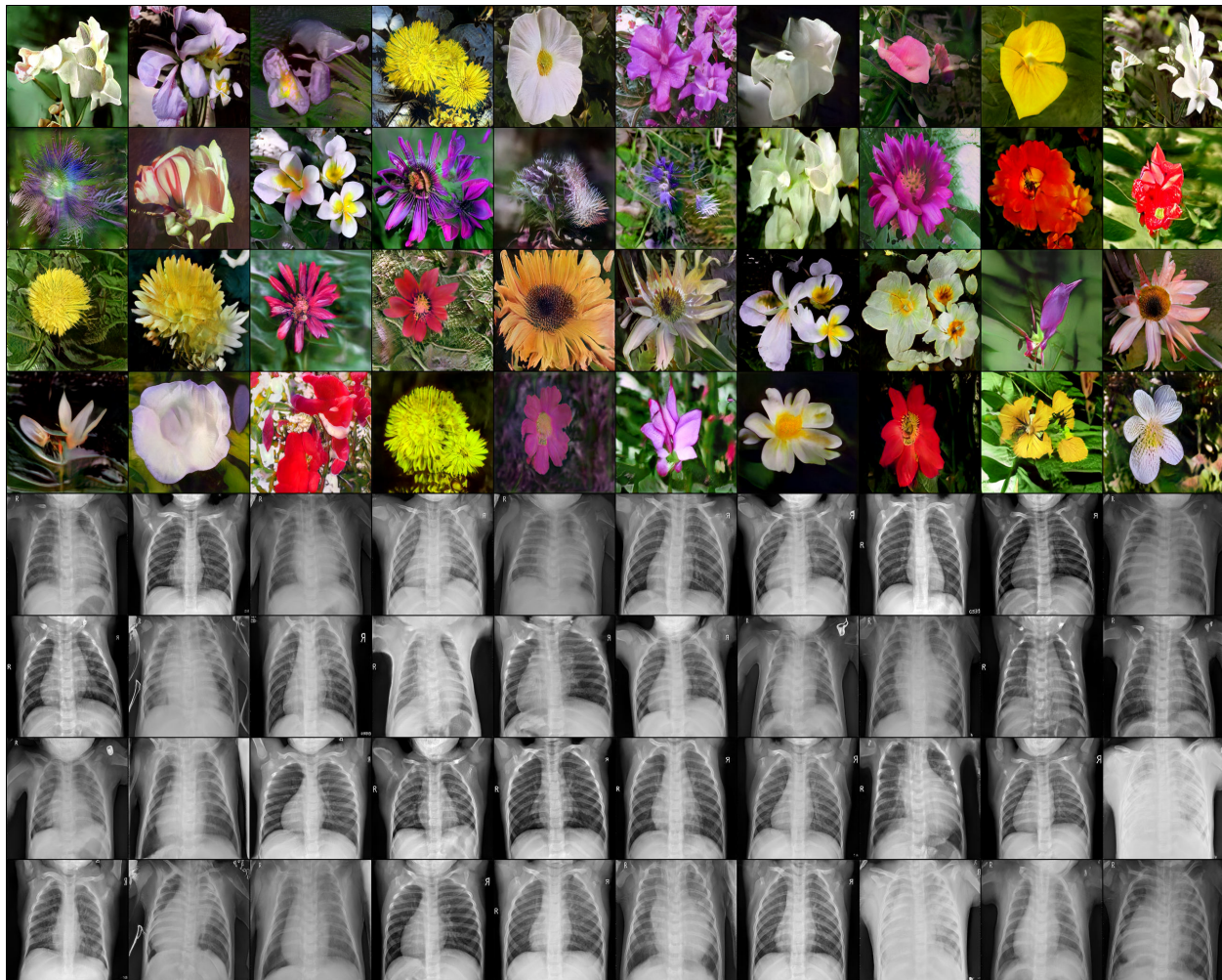


Figure 6: Randomly generated images using StyleCL. **Rows:1-4:** Flowers **Rows:5-8:** Chest X-Ray. The generated images show high similarity to real data with the exception of some finer details and artifacts.



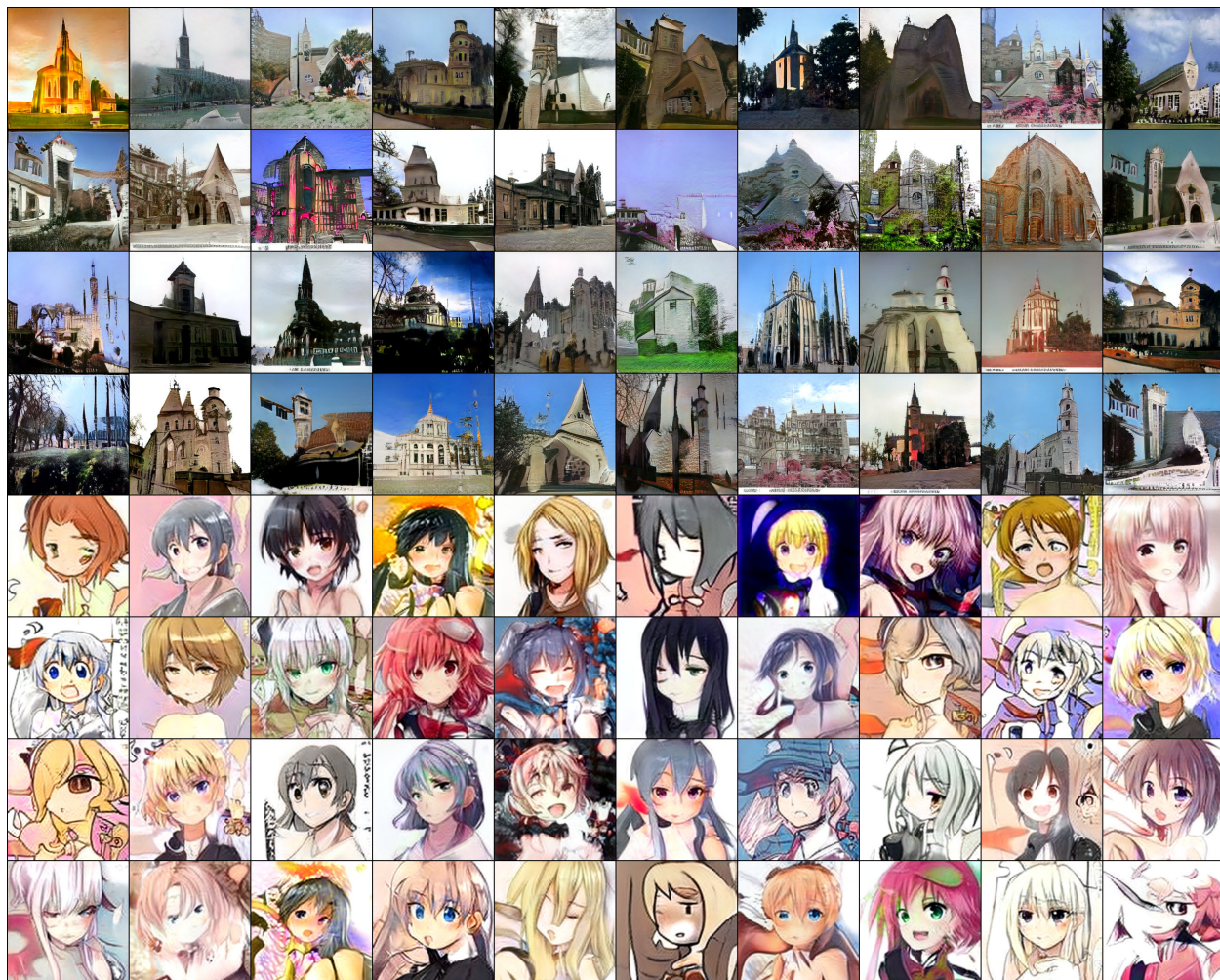


Figure 7: Randomly generated images using StyleCL. **Rows:1-4** LSUN Church **Rows:5-8** Anime. The generated images show high similarity to real data with the exception of some finer details and imperceptible artifacts.



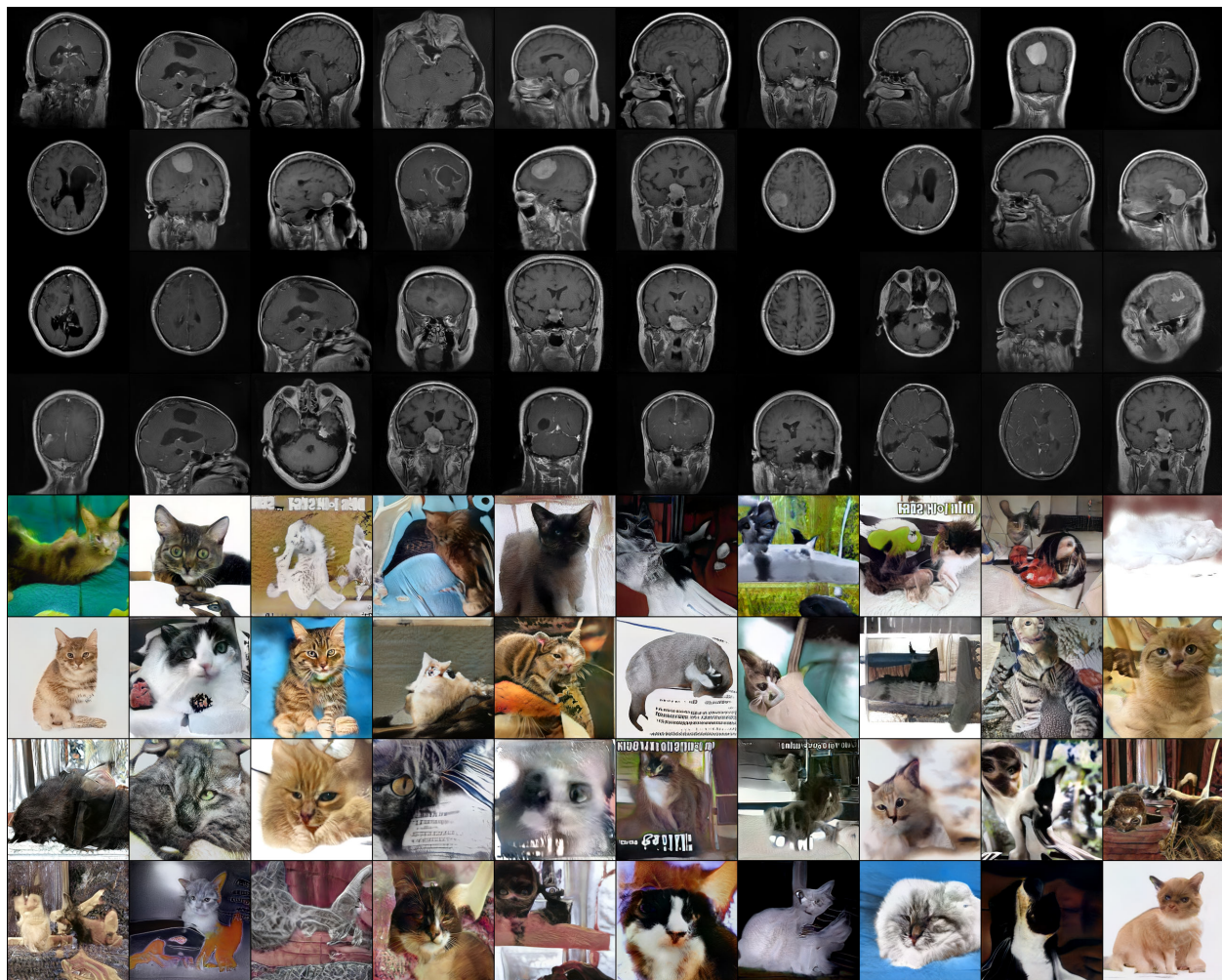


Figure 8: Randomly generated images using StyleCL. **Rows : 1-4: Brain MRI Rows : 5-8: LSUN-Cats.** The generated images show high similarity to real data with the exception of some finer details and imperceptible artifacts.





Figure 9: We perform linear interpolation ( $stepsize = 0.1$ ) between two randomly sampled vectors from the latent subspace learned by StyleCL. Each column represents a linear interpolation where the first and last image corresponds to vectors sampled from the latent subspace of Anime.





Figure 10: We perform linear interpolation ( $stepsize = 0.1$ ) between two randomly sampled vectors from the latent subspace learned by StyleCL. Each column represents a linear interpolation where the first and last image corresponds to vectors sampled from the latent subspace of Flowers.

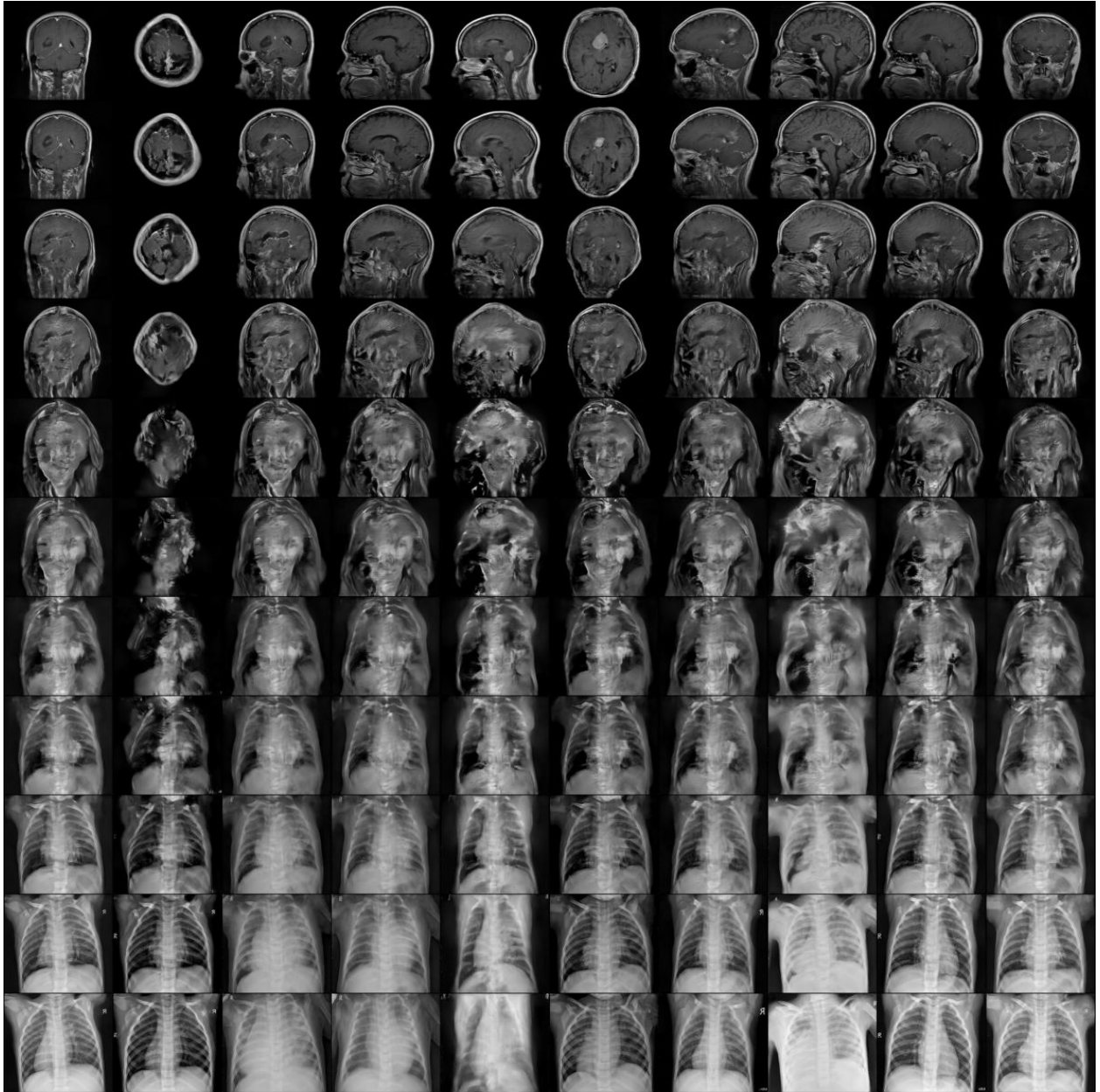


Figure 11: We perform linear interpolation ( $stepsize = 0.1$ ) between two randomly sampled vectors from the latent subspace learned by StyleCL. Each column represents a linear interpolation where the first and last image corresponds to vectors sampled from the latent subspace of Brain MRI and Chest X-Ray respectively.





Figure 12: We perform linear interpolation ( $stepsize = 0.1$ ) between two randomly sampled vectors from the latent subspace learned by StyleCL. Each column represents a linear interpolation where the first and last image corresponds to vectors sampled from the latent subspace of Flowers and Anime respectively.