CAM-GAN: Continual Adaptation Modules for Generative Adversarial Networks

Anonymous Author(s)
Affiliation
Address
email

Abstract

We present a continual learning approach for generative adversarial networks (GANs), by designing and leveraging parameter-efficient feature map transformations. Our approach is based on learning a set of global and task-specific parameters. The global parameters are fixed across tasks whereas the task-specific parameters act as local adapters for each task, and help in efficiently obtaining task-specific feature maps. Moreover, we propose an element-wise addition of residual bias in the transformed feature space, which further helps stabilize GAN training in such settings. Our approach also leverages task similarity information based on the Fisher information matrix. Leveraging this knowledge from previous tasks significantly improves the model performance. In addition, the similarity measure also helps reduce the parameter growth in continual adaptation and helps to learn a compact model. In contrast to the recent approaches for continually-learned GANs, the proposed approach provides a memory-efficient way to perform effective continual data generation. Through extensive experiments on challenging and diverse datasets, we show that the feature-map-transformation approach outperforms state-of-the-art methods for continually-learned GANs, with substantially fewer parameters. The proposed method generates high-quality samples that can also improve the generative-replay-based continual learning for discriminative tasks.

1 Introduction

Lifelong learning is an innate human characteristic; we continuously acquire knowledge and upgrade our skills. We also accumulate our previous experiences to learn a new task efficiently. However, learning new tasks rarely affects our performance on already-learned tasks. For example, after learning one programming language, if we learn a new such language it is rare that our skill on the first deteriorates. In fact, the knowledge of the previous task(s) often speeds up learning of the subsequent task(s). On the other hand, attaining lifelong learning in deep neural networks is still a challenge. Naive implementation of continual learning with deep learning models suffers from catastrophic forgetting [1, 2, 3], which makes lifelong or continual learning (CL) difficult. Catastrophic forgetting refers to the situation when a model exhibits a decline in its performance on previously learned tasks, after it learns a new task.

Several prior works [1, 3, 2, 4, 5, 6, 7, 8] have been proposed to address catastrophic forgetting in deep neural networks, with most of them focusing on classification problems. In contrast, continual learning for deep generative models, such as generative adversarial networks (GAN) [9] and variational auto-encoders (VAE) [10] has not been widely explored. Some recent work [6, 11, 12, 13] has tried to address the catastrophic forgetting problem in GANs. Among these, the generative-replay-based approach [6, 11] usually works well only when the number of tasks is small. As the number of tasks become large, the model starts generating unrealistic samples that are not suitable for generative replay. A regularization-based approach for continual learning in GANs [12] often gives sub-optimal

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

solutions. In particular, instead of learning the optimal weights for each task, it learns a mean weight for all tasks, since a single set of weights tries to generalize across all tasks. Therefore, these approaches mix the sample generation of various tasks and generate blurry and unrealistic samples. Also, continually learning new tasks on a fixed size network is not feasible since, after learning a few tasks, the model capacity exhausts and the model becomes unable to learn the future tasks.

Recently, expansion-based CL models [14, 15, 16, 17, 7, 18, 19, 20, 21, 22] have shown promising results for discriminative (supervised) continual learning settings. These approaches are dynamic in nature and allow the number of network parameters to grow to accommodate new tasks. Moreover, these approaches can also be regularized appropriately by partitioning the previous and new task parameters. Therefore, unlike regularization-based approaches, the model is free to adapt to novel tasks. However, despite their promising performance, the excessive growth in the number of parameters [15, 19, 20, 21, 22] and floating-point (FLOP) requirements are critical concerns.

In this work, we propose a simple expansion-based approach for GANs, which continually adapts to novel tasks without forgetting the previously-learned knowledge. The proposed approach considers a base model with *global parameters* and corresponding *global feature* maps. While learning each novel task, the base model is expanded to consider a task-specific feature transformation which efficiently adapts a global feature map to a task-specific feature map in each layer. Even though the total number of parameters increases due to the additional *local/task-specific parameters*, this feature-map transformation approach allows the leveraging of efficient architecture design choices (e.g., groupwise and pointwise convolution) to obtain compact-sized *task-specific* parameters, which controls the growth and keeps the proposed model compact.

We empirically observe that our feature map transformation is highly efficient, simple and effective, and shows more promising results compared to the weight space transformation based approaches [13]. In addition, our approach enables leveraging the task similarities. In particular, learning a new task that is similar to previous tasks should require less effort compared to learning a very different task; if we already know statistics and linear algebra, learning the subject machine learning is more easy compared to learning a completely different subject, e.g., computer architecture. Most existing CL approaches ignore the task-similarity information during the continual adaptation. We find that learning a novel task by initializing its task-specific parameters with the task-specific parameters of the most similar previous task significantly boosts performance. To this end, we learn a compact embedding for each task using the mean of the Fisher information matrix (FIM) per filter, and use it to measure task similarity. We observe that considering the task similarity information for parameter initialization not only boosts the model performance but can also be useful to reduce the number of *task-specific* parameters. While FIM has been used in regularization-based approaches for continual learning [2], we show how it can be used in expansion based methods like ours.

To show the efficacy of the proposed model, we conduct extensive experiments in various settings on real-world datasets. We show that the proposed approach can sequentially learn a large number of tasks without catastrophic forgetting, while incurring much smaller parameter and FLOP growth compared to the existing continual-learning GAN models. Further, we show that our approach is also applicable to the generative-replay-based discriminative continual learning (e.g., for classification problems). We empirically show that the pseudo-rehearsal provided by the proposed approach shows promising results for generative-replay-based discriminative models. Also, we conduct experiments to demonstrate the effectiveness of considering the task similarity in continual image generation, which we believe can lead to a promising direction for continual learning. Our contribution can be summarized as follows:

- We propose an efficient feature-map-based transformation approach to continually adapt to new tasks in a task sequence, without forgetting the knowledge from previous tasks.
- We propose a parallel combination of groupwise and pointwise 3×3 and 1×1 filters for efficient adaptation to novel tasks, and require significantly fewer parameters for adaptation.
- We propose an approach that leverages the Fisher information matrix to measure task similarities for expansion-based continual learning. We empirically demonstrate its superior performance on several datasets.
- We demonstrate the efficacy of the proposed model for continual learning of GANs and generative-replay-based discriminative tasks. We observe that, with much less parameter and FLOP growth, our approach outperforms existing state-of-the-art methods.

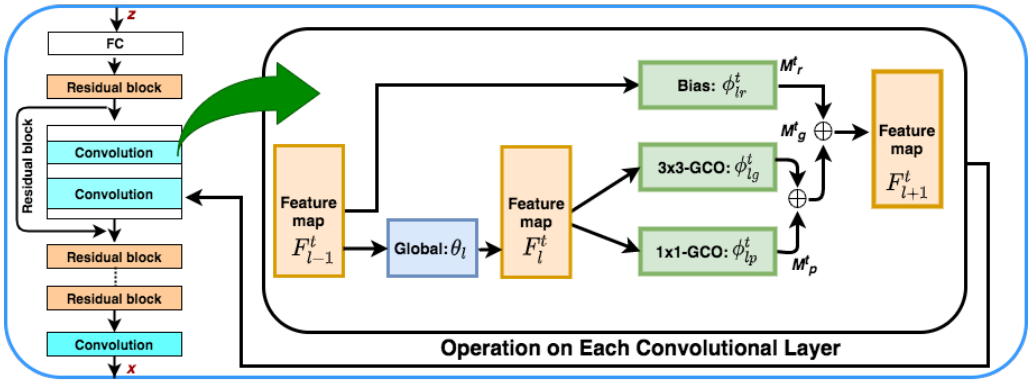

Figure 1: The proposed efficient transformation module over a convolutional layer. Each layer consists of three task-specific parameter ϕ^t_{lp}, ϕ^t_{lg} and ϕ^t_{lr} and one global parameter θ_l. Here ϕ^t_{lr} is the residual bias in the transformed space.

2 Lightweight Continual Adapter

Notation and Problem Setting: We assume a set of tasks $\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_K$ to be learned, and they arrive in a sequence. At a particular time t, data corresponding to the only current task $\mathcal{T}_t$ are available. The data for the tth task are given as $\mathcal{T}_t = \{x_i, y_i\}_{i=1}^{N_t}$ where (x_i,y_i) represents data samples and class labels, respectively. For unconditional data generation, we assume $y_i = 1$, and for the conditional generation $y_i \in \mathcal{Y}$, where $\mathcal{Y}$ is the label set for the task $\mathcal{T}_t$.

The proposed approach consists of a base model $\mathcal{M}$ with parameters θ, which serve as the global parameters. The global parameters at layer l are denoted by θ_l. Each layer of $\mathcal{M}$ also contains the task-specific parameters which perform task-specific feature-map transformation. Let ϕ^t_l denote the task-specific parameter for the lth layer during the training of task t. We use the GAN framework as a base generative model, based on the architecture proposed by [23], although it is applicable to other GAN architectures as well. The base model $\mathcal{M}$ comprises the discriminator (D) and generator (G). Since our main aim is to learn a good generator for each task, our continual learning approach considers only the generator to have task-specific adapters. The discriminator parameters (θ_d) can be modified using each new task's data. We learn the global parameters from a base task using the GAN objective [23] and learn the task specific parameters ϕ^t as we adapt to a new task t:

$$\min_{\phi^t} \max_{\theta_d} \mathbb{E}_{\mathbf{x}\sim p^t_{data}}[log(D(\mathbf{x};\theta_d))] + \mathbb{E}_{\mathbf{z}\sim p_{\mathbf{z}}(\mathbf{z})}[log(1 - D(G(\mathbf{z};\theta,\phi^t);\theta_d))] + R(\theta_d, \phi^t) \quad (1)$$

where $\mathbb{E}$ denotes expectation and $R()$ is the regularization term defined in [23].

We propose an efficient expansion-based approach to mitigate catastrophic forgetting in GANs while learning multiple tasks in a sequential manner. The proposed adapter module is fairly lightweight and easy to use in any standard model for continual adaptation. The added adapter in the standard architecture may, however, destabilize GAN training. To address this issue, we add a residual adapter in the transformed feature space that results in smooth training for the modified architecture. The parameters in efficient adapters and residual bias serve as *local/task-specific* parameters and are specific to a particular task. The details of the proposed adapter and residual bias are provided in Sections 2.1 and 2.2, respectively.

2.1 Efficient Adapter

We propose task-specific adapter modules to continually adapt the model to accommodate novel tasks, without suffering from catastrophic forgetting for the previous tasks. The proposed approach leverages a layerwise transformation of the feature maps of the base model, using the adapter modules to learn each novel task. While training the task sequence, the global parameters corresponding to base model $\mathcal{M}$ remain fixed. Let $F^t_l \in \mathbb{R}^{m\times n\times c}$ be the feature map obtained at the lth layer upon applying the global parameters θ_l of the lth layer during training of task t, i.e.,

$$F^t_l = f_{\theta_l}(F^t_{l-1}) \quad (2)$$

Here f_{θ_l} represents convolution and F^t_{l-1} is the previous layer's feature map. Our objective is to transform the feature map F^t_l that is obtained at the lth layer, but not yet task-specific, to a *task-*

specific feature map using the task-specific adapters. Our task-specific adapters comprise two types of efficient convolutional operators. We combine both in a single operation to reduce layer latency.

3×3 Groupwise Convolutional Operation Let f^g be the 3×3 groupwise convolutional operation ($3 \times 3-$GCO) [24] with a group size k at a particular layer l, and defined by local parameters ϕ^t_{lg}. Therefore, we have each convolutional filter $c_i \in \mathbb{R}^{3\times3\times k}$ in contrast to the standard convolutional filter $c_s \in \mathbb{R}^{3\times3\times c}$. Here $k \ll c$, and therefore each filter c_i is highly efficient and requires $\frac{c}{k}$ times fewer parameters and FLOPs. We apply the function f_g on the feature map obtained by (2). For the tth task, let the new feature map obtained after applying f^g be M^t_g, which can be written as

$$M^t_g = f^g_{\phi^t_{lg}}(F^t_l) \tag{3}$$

Here $M^t_g \in \mathbb{R}^{m\times n\times c}$ is the feature map of the same dimension as the feature map obtained by (2). Therefore, we can easily feed the obtained feature map to the next layer without any architectural modifications. Note that the standard convolution c_s accumulates the information across all c channels of the feature map, and in each channel its receptive field is 3×3. If we reduce the group size to k, then the new convolution c_i covers the same receptive field but it accumulates the information across only $k \ll c$ channels. The above process is highly efficient but reduces the model's performance, since it suffers from a non-negligible information loss by capturing fewer channels. To mitigate this potential information loss, we further leverage a specialized FLOP- and parameter-efficient convolutional filter, as described below.

1×1 Groupwise Convolutional Operation Pointwise convolution (PWC) is widely used for efficient architecture design [25, 26]. The standard PWC contains filters of size $1 \times 1 \times c$. It is 9 times more efficient as compared to the 3×3 convolution but still contains significant number of parameters. To increase the efficiency of the PWC, we further divide it into z groups. Therefore, each convolutional filter a_i contains the filter of size $1 \times 1 \times z$ and, since $z \ll c$, it reduces the FLOP requirements and number of parameters $\frac{c}{z}\times$ as compared to the standard PWC. Also, since $z > k$, it captures more feature maps and overcomes the drawback of the $3 \times 3-$GCO. Let f^p be the 1×1 groupwise convolutional operational (1×1-GCO) and ϕ^t_{lp} be the lth layer local parameter given by the 1×1 groupwise convolutional operation. Assume that, after applying f^p, we obtain the feature map M^t_p, which can be written as

$$M^t_p = f^p_{\phi^t_{lp}}(F^t_l) \tag{4}$$

Here $M^t_p \in \mathbb{R}^{m\times n\times c}$ also has the same dimension as the incoming feature map, and therefore we can easily apply this input to the next layer without any significant changes to the base architecture.

Parallel Combination Note that local feature map adapters, i.e. (3) and (4), can be applied in a parallel manner and the joint feature map for task t can be written as

$$M^t_l = M^t_g \oplus \beta M^t_p \tag{5}$$

Here $\beta = 1$ when we use 1×1 groupwise convolution operation, otherwise $\beta = 0$, and $\oplus$ is the element-wise addition. Therefore, using β, we can trade-off the model's parameter and FLOP growth with its performance. In this combination, M^t_g is obtained by filters that capture a bigger receptive field and M^t_p is obtained by filters that capture the information across longer feature-map sequences. Therefore, the combination of the two helps increase a model's performance without a significant increase in the model's parameters and FLOPs. Like most of the efficient architecture models [26, 27], we can apply the 3×3 and 1×1 convolutional operation in a sequential manner, denoted as: $M^t_l = f^p_{\phi^t_{lp}}(f^g_{\phi^t_{lg}}(F^t_l))$. Here M^t_l requires two sequential layers in the model architecture and execution of the second layer is followed by the first layer, which limits the parallelization of the model. Therefore, throughout the paper, we follow (5), i.e., the parallel combination. Again, note that the proposed adapter is added only to the generator network since the discriminator model's parameter can be discarded once the model for each task is trained.

2.2 Residual Bias

We append the proposed adapters to the base model; therefore the new architecture is defined by parameters $[\theta, \phi]$. The parameters θ are considered as global parameters and each task t has its own *local/task-specific* parameter ϕ. After adding the adapter layer (discussed in Sec. 2.1) to the base architecture, the model becomes unstable, and after training for a few epochs, the generator loss

diverges and discriminator loss goes to zero. The most probable reason for this behaviour is that, after adding the efficient adapter layer, each residual block doubles the layer depth, and the effect of the residual connection for such a long sequence is not prominent. Therefore the model starts diverging. To overcome this problem, we learn another function f^r using efficient convolutional operations with parameters ϕ^t_{lr}. Here, f^r is also a 3×3 groupwise convolution and, on layer $(l+1)^{th}$, it takes the feature map of layer $l-1$ as input. Therefore, the residual bias can be defined as: $M^t_r = f^r_{\phi^t_{lr}}(F_{l-1})$, where $M^t_r \in \mathbb{R}^{m \times n \times c}$ has the same dimension as M^t_i. Therefore, we can perform element-wise addition of this feature map to (5). We consider this as an element-wise bias, added to the output feature map. Moreover, note that it is like a residual connection from the previous layer, but instead of directly taking the output of the $(l-1)^{th}$ layer, it transforms the $(l-1)^{th}$ layer's feature map. We call this term as *residual bias* and the final feature map after the continual adapter and residual bias is defined as: $F^t_{l+1} = M^t_l \oplus M^t_r$. Therefore, at the lth layer for the tth task, the model has θ_l as the global parameter and $\phi^t_l = [\phi^t_{lg}, \phi^t_{lp}, \phi^t_{lr}]$ as the *local/task-specific* parameters. Here, $|\phi^t_l| \ll |\theta_l|$, i.e., the number of parameters in ϕ^t_l is much smaller than in θ_l, which helps control the growth in model size as it adapts to a novel task.

2.3 Task-similarity based Transfer Learning (TSL)

Transfer learning plays a crucial role in achieving state-of-the-art performance in deep learning models. In continual learning approaches, each novel task's parameters are initialized with the previously learned model parameters. However, this can exhibit high variance in performance [28] depending on the order in which tasks arrive. Also, these approaches may have limited transfer learning ability because we may initialize the novel task's parameters with those of the previous tasks which may be very different. We observe that initializing the novel task's parameters with the most similar task's parameters not only boosts the model performance but also reduces variance in performance. It is easy to learn the task similarity for supervised learning settings and [29] explores the same for the meta-learning. In this work, we explore the task-similarity for more challenging settings, i.e., unsupervised learning connected to GANs.

We empirically observe that similar tasks share a similar gradient. Therefore, to measure the task correlation, our proposed approach leverages gradient information. In particular, we calculate the Fisher Information Matrix (FIM) of the generator parameters without using explicit label information. Let $\mathcal{L}_g$ be the generator loss w.r.t. parameter θ_g. The diagonal approximation of the FIM for the task $\mathcal{T}_t$ can be defined as:

$$F_t = \text{Diag}\left(\frac{1}{N_t}\sum_{i=1}^{N_t} \nabla\mathcal{L}_g(x_i|\theta_g)\nabla\mathcal{L}_g(x_i|\theta_g)^T\right) \tag{6}$$

Here we consider θ_g as flattened, and consequently it is a vector. We use F_t as an embedding to represent the task; note that it has the same dimension as the generator's parameters. However, directly learning and representing task embeddings in such a manner is expensive, both computationally as well as in terms of storage. We consider the convolutional filters used in the adapter modules for calculating FIM to reduce the number of parameters and we further replace each convolutional filter by its mean value, and ignore the fully connected layers to learn a compact task embedding. We use these learnt task embeddings to calculate task similarities among various tasks. Based on the similarity measure, we initialize adapter modules parameters of the current task with the parameters of the most similar previously seen task.

3 Related work

Catastrophic forgetting [3, 1, 30, 31, 32, 33, 8, 34, 6, 13] is a fundamental problem for deep neural networks, which arises in settings where, while learning from streams of data, the data from previous tasks are no longer available. There is growing interest in mitigating catastrophic forgetting in deep neural networks. Existing methods can be broadly classified into three categories, based on regularization, replay, and expansion. The regularization-based approach [31, 35, 30, 36, 37] regularizes the previous task's model parameters such that we learn to solve the new task with a minimal shift in the previous task's parameters. The replay-based approach [33, 8, 38, 34, 39] stores a subset of the previous tasks' samples in a small memory bank or learns a generative model via VAE [10] or GAN [9] to replay the previous tasks' samples. Dynamic expansion-based approaches [19, 14, 20, 40, 7, 41] increase the model capacity dynamically as each new task arrives, and show promising results compared to fixed-capacity models.

Most of the above-mentioned approaches focus on eliminating catastrophic forgetting in discriminative models (supervised learning, such as classification). Despite the wide popularity of deep generative models, continual learning for such models is relatively under-explored. MeRGAN [6] proposed a generative-replay-based approach to overcome catastrophic forgetting in GANs. For generative replay, MeRGAN stores the immediate copy of the generator to provide the replay of previously learned tasks, while training the new task. While generative-replay-methods succeed in solving catastrophic forgetting to some extent, it is a costly operation as it preserves a copy of the complete model parameters. It also results in blurry samples for previous tasks as the number of tasks increases. Lifelong GAN [12] performs knowledge distillation among multiple realizations of the model to prevent catastrophic forgetting in image-conditioned generation via GANs. These regularization-based models converge to the sub-optimal solution and are unable to model large task sequences. GAN-memory [13] is another recently proposed method based on expansion. It performs style modulation to learn new tasks by performing a transformation over weight parameters. GAN-memory shows promising results for continual learning; however, expansion cost and model complexity may be a bottleneck. In contrast, we propose a simple expansion-based model which is highly parameter and FLOP efficient, by employing efficient feature map transformations. It also requires minimal changes to the base model to adapt to the novel task(s). Therefore, without exploding the model size, we can learn any number of tasks in a sequence. Moreover, unlike existing continual GAN approaches, our approach can leverage task similarities (computed using an FIM based embedding for each task), bringing in an explicit capability of transfer learning [42, 43, 44, 45, 46, 47, 48, 49] within a continual learning setting.

4 Experiments

We perform extensive experiments on several image datasets from, visually diverse domains, to show the efficacy of the proposed approach. We perform experiments for generation of samples for continually steamed datasets. We also demonstrate that the proposed continual generative model can be used for generative-replay-based continual learning for discriminative models (supervised learning). We refer to our approach as CAM-GAN (Continual Adaptation Modules for Generative Adversarial Networks).

4.1 Data Description

Unconditional Generation: We conduct experiments[1] over perceptually distinct datasets to show the continual adaptivity of our model for the data-generation task. For continual data generation, we consider 7 datasets from perceptually distant domains: CelebA ($\mathcal{T}_0$) [50], Flowers ($\mathcal{T}_1$) [51], Cathedrals ($\mathcal{T}_2$) [52], Cat ($\mathcal{T}_3$) [53] Brain-MRI images ($\mathcal{T}_4$) [54], Chest X-ray ($\mathcal{T}_5$) [55] and Anime faces ($\mathcal{T}_6$)[2]. We consider 256×256 resolution images for all the datasets.

Conditional Generation: We also experiment on four task sequences comprising four Imagenet types of data [56]: (i) fish, (ii) bird, (iii) snake and (iv) dog. Each group contains six sub-classes; so we have 24 classes for conditional generation. In our setting, we consider each group as a task. Therefore, we have four tasks: $\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3$ and $\mathcal{T}_4$, corresponding to each group. Each task is formulated as a 6-class classification problem. We selected 256×256 resolution images of selected Imagenet classes.

4.2 Architectural and Evaluation Metric

We utilize the GAN architecture from GP-GAN [23]. On the top of GP-GAN, we add the proposed adapter module. We use Fŕechet Inception Distance (FID) [57] as the evaluation metric to show quantitative results, as it correlates with human evaluation and is efficient to compute. Architecture and evaluation details are discussed in the Supplemental Material.

4.3 Unconditional Continual Data Generation

The proposed approach shows promising results for unconditional continual data generation without catastrophic forgetting. We perform the experiments over a sequence of diverse datasets to illustrate our approach's capability in the generation of samples belonging to the completely varied domains. For the unconditional generation experiments, we first train a base model $\mathcal{M}$ on the task $\mathcal{T}_0$. We consider the CelebA dataset as the 0th task, to train the base model. The parameters of the model trained on $\mathcal{T}_0$ serve as global parameters. Once we have the global parameters trained on the base model, they are frozen for the subsequent task sequence $\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_6$. Only task-specific parameters

[1] Code will be publicly available upon acceptance.

[2] `https://github.com/jayleicn/animeGAN`

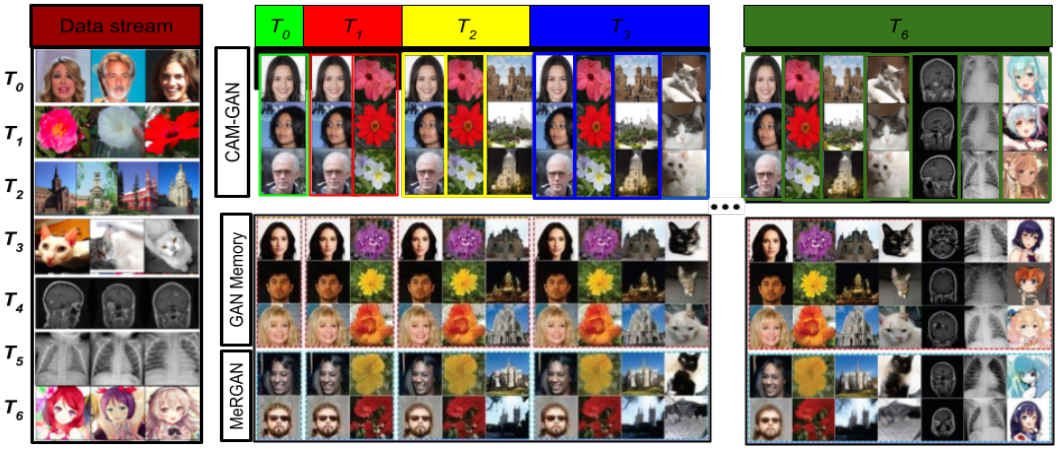

Figure 2: (Left) The real data stream representing tasks in the sequence, (right) Qualitative comparison of samples generated in streamed manner using our approach after training each task ($\mathcal{T} = \mathcal{T}_1, \mathcal{T}_2, \ldots \mathcal{T}_6$) with GAN-memory and MeRGAN.

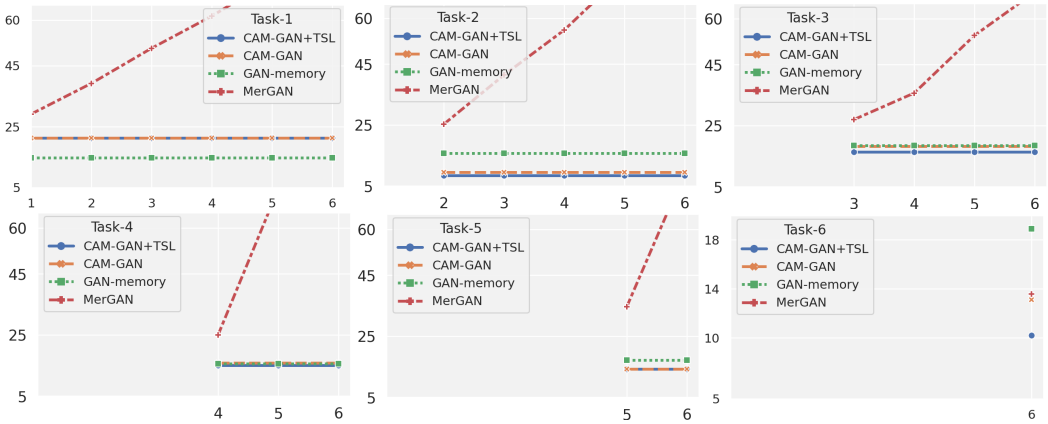

Figure 3: Comparison of FID scores of our approaches (CAM-GAN+TSL,CAM-GAN) with GAN-memory and MeRGAN. Each subgraph represents the FID of current task $\mathcal{T}$ and affects its FID after training subsequent tasks in the sequence. Our approach continually learns new tasks without catastrophic forgetting and, except for the first task, achieves lower/comparable FID scores as compared to GAN-memory and MerGAN, with substantially lower number of parameters.

are free to change for the later tasks. We compare our results with state-of-the-art baselines, both expansion-based [13] (GAN-Memory) and generative-replay-based [6] (MeRGAN). We provide both qualitative and quantitative comparison of our model with these baselines.

Figure 2 shows the qualitative comparison of our method, CAM-GAN, with the baselines. We can observe that MeRGAN starts generating blurry samples for previous tasks with the increased length of the task sequence. In contrast, using our approach, the quality of samples for previous tasks remains unchanged after training subsequent tasks in the sequence. Our simple feature map transformation requires only about $\sim 13\%$ task-specific parameters, while GAN-Memory requires about $\sim 20\%$ parameter growth per task (Table 1; Fig. 4, Left). In addition, as we can observe from Fig. 3 (FID score comparisons), our model generates considerably better samples for all the datasets, except the flower task.

	Parameter (M)	FLOPs (G)	Parameter (% ↑)	FLOPs (% ↑
Original	52.16	14.75	–	–
GAN-Memory	62.69	–	20.18	–
CAM-GAN	58.88	19.69	12.88	35.55

Table 1: The FLOP and parameter growth for each task compared to the base model and GAN-memory [13]. Our approach has much fewer parameter growth while providing better results.

Incorporating Task Similarity: We further incorporate the task-similarity measure to improve the continual adaptation of the model. Figure 6 depicts that generation improves when the parameters are initialized using the task similarity metric to find the most similar previous task. Our qualitative

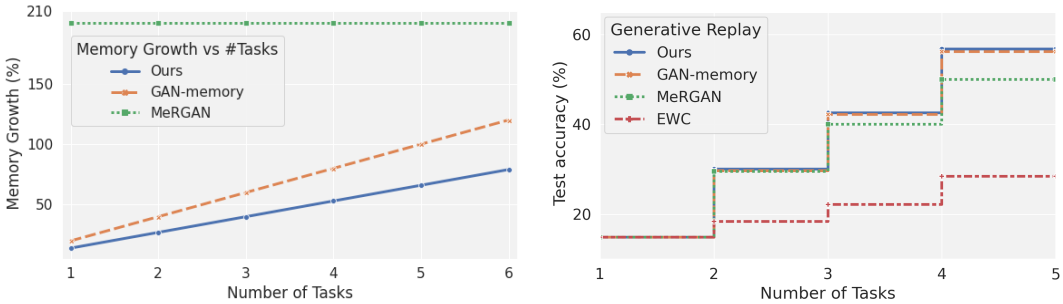

Figure 4: **Left:** Parameter Growth per task. **Right:** Comparison of combined test accuracy while using our approach as generative replay with baseline models on the supervised CL setting.

results (Fig. 2) are supported by FID scores presented in the Fig. 3. The detailed Tables for Fig. 3 are given in the Supplemental Material. The generation of Anime dataset improves significantly when the adapter parameters are initialized using celebA parameters, instead of X-ray parameters.

4.4 Conditional Data Generation as Replay

Our continual GAN approach can also be used as a generative replay [38] in discriminative continual learning. We further test our model's ability in the generative replay paradigm to assist the classifier model in lifelong learning. We consider conditional data generation for generative replay experiments and apply it to the class incremental setup [58] for classification. We perform joint testing [13], i.e., while training task t in the sequence, the classifier must classify all the $6 \times t$ classes accurately. We compare our method with other generative-replay approaches, MeRGAN and GAN-memory, and also compare with the regularization based approach EWC [31].

Figure 4 (right) represents the test accuracy compared to the baseline models. We observe that: (i) EWC does not perform well in the incremental-class setup; its performance for the previously trained tasks declines sharply after training subsequent tasks. (ii) MeRGAN provides good accuracy for the initial tasks in the sequence, but its performance starts degrading as more tasks are added in the sequence. In contrast, our approach maintains the test accuracy for previous tasks by providing good quality samples. (iii) Our approach achieves better/comparable test accuracy than GAN-Memory. However, our approach uses a significantly less number of parameters to attain better/similar accuracy, making it more scalable for generative-replay scenarios with a large number of tasks in the sequence. In the Supplemental Material, we provide model details, explain the training procedure, provide more qualitative results, and interpolation results when moving from one task to the next task.

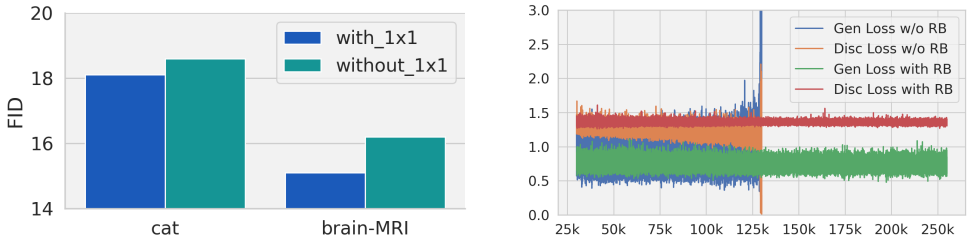

Figure 5: **Left:** Effect of 1×1 covolution on FID scores. **Right:** Effect of the Residual Bias (RB) on the Generator and discriminator loss for the CELEB dataset. The added element-wise residual bias provides highly stable training.

5 Ablation Study

In this section, we disentangled the various components of our model, and we observe that each proposed component plays a significant role in the model's overall performance.

5.1 Effect of 1×1 convolution

We perform an ablation study to demonstrate the effectiveness of 1×1 efficient convolutional operators described in Sec. 2.1 to the adapter modules. We conduct our experiments on datasets ($\mathcal{T}_3$) and brain MRI images ($\mathcal{T}_4$). For the $\mathcal{T}_3$ and $\mathcal{T}_4$ experiment, we select the previous task's model, i.e., $\mathcal{T}_2$ and $\mathcal{T}_3$, respectively, and freeze the 1×1 convolutional layer. Therefore, the $3 \times 3 - GCO$ are only free to adapt to the current task. The results are shown in Fig. 5 (left) which demonstrate that having a small portion of the total parameter ($\sim 1\%$) in the 1×1 adapter improves the performance, and the drop without this adapter is significant.

5.2 Effect of the Residual Bias

Section 2.2 describes the detail of the residual bias. The residual bias plays a crucial role in the model's training stability. Empirically, we observe that after inclusion of adapter modules in the GAN [23] architecture, the model shows highly irregular training, and discriminator and generator losses diverge quickly. To overcome this problem, we learned the residual bias for the feature map space. Residual bias is similar to the residual connection [59] but in the transformed feature space (since two layers are of different dimensions and element-wise addition is not feasible) with the help of learnable parameters. Figure 5 (right) shows that, without residual bias on the CelebA dataset, the discriminator and generator losses have high variance, and after $125K$ iterations, the model diverges. However, residual bias provides highly stable training as the discriminator and generator losses oscillate in a narrow range; therefore, the model is stable and does not diverge as training progresses. We train on the same dataset for $600K$ iterations for an extreme evaluation, and observe the same stability in training and loss curve without any degradation.

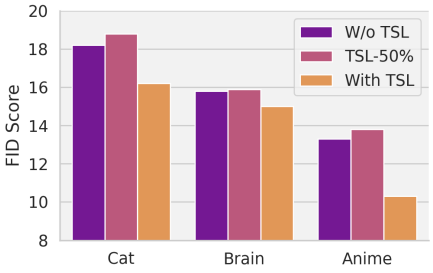

Figure 6: Model's FID score using TSL based initialization, TSL-50% shows if we reduce the model parameters 50% and initialize it with task-similarity.

5.3 Effect of Task Similarity

If the next task tends to be very similar to the previous task, adaptation is expected to be easy and only a few *task-specific* parameters would be sufficient for adaptation and training should converge quickly. Previous approaches [6, 11, 12, 13], while learning a novel task, initialize the model from the previously learned task's parameters. The previous task may or may not be similar to the novel task, and therefore the model's performance is not optimal; in this setting one also requires a significant number of new learnable parameters and many epochs for convergence. In our approach, we measure the task similarity (discussed in Section 2.3) of the new task with all the previous tasks, and the model is initialized with the parameters of the most similar previous task. Results are shown in Fig. 6 which shows that using similarity information significantly improves performance. Compared to the previous task's parameter initialization, if we reduce the parameter by $\sim 50\%$, similarity-based initialization still achieves similar performance. Further details of training and hyperparameters are discussed in the Supplemental Material.

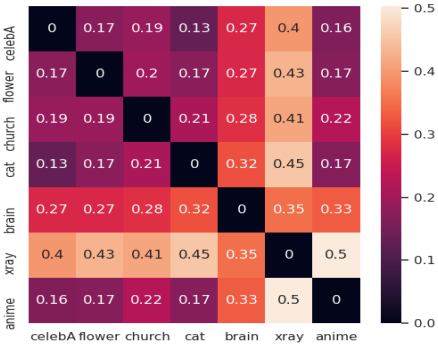

Figure 7: Task-similarity measure over various dataset (lower if better) using the model described in Sec-2.3

6 Conclusions

We propose a simple and efficient, dynamically expandable model to overcome catastrophic forgetting in continual learning of GANs. The proposed model contains a set of *global* and *local/task-specific* parameters. The global parameters are obtained from the base task dataset and remain fixed for the remaining tasks. Each task's specific parameters act as local adapters and efficiently learn *task-specific* feature maps. The local parameters leverage parallel combinations of efficient filters that minimize the per-task model's FLOP and parameter growth. Also, to overcome the unstable GAN training, we propose a residual bias in the transformed space, which provides highly stable training. The task-similarity-based initialization significantly boosts the model's performance and provides room to further reduce the model parameters. Our approach mitigates catastrophic forgetting by design; while learning the novel task, the previous model's parameters are unchanged. Therefore, it does not show any catastrophic forgetting. We demonstrate the effectiveness of our approach through qualitative and quantitative results. The proposed model requires substantially fewer parameters ($\sim 36\%$ fewer parameters) for continual learning in GAN, while showing a better or similar result compared to the current state-of-the-art models. We also show that the proposed model can be used for generative replay, and it offers a promising way to overcome catastrophic forgetting in discriminative (supervised) models. It requires a reasonable base task to learn the global parameters. Although our experiments show that our model is fairly robust to the choice of base model, some care should be taken since a considerably weak base task may degrade the model performance.

References

[1] Gail A Carpenter and Stephen Grossberg. A massively parallel architecture for a self-organizing neural pattern recognition machine. *Computer vision, graphics, and image processing*, pages 54–115, 1987.

[2] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks, 2017.

[3] Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier, 1989.

[4] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. volume 70 of *Proceedings of Machine Learning Research*, pages 3987–3995, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR.

[5] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H. Lampert. iCaRL: incremental classifier and representation learning. 2017.

[6] Chenshen Wu, Luis Herranz, Xialei Liu, Joost van de Weijer, Bogdan Raducanu, et al. Memory replay gans: Learning to generate new categories without forgetting. *Advances in Neural Information Processing Systems*, 31:5962–5972, 2018.

[7] Pravendra Singh, Vinay Kumar Verma, Pratik Mazumder, Lawrence Carin, and Piyush Rai. Calibrating cnns for lifelong learning. *Advances in Neural Information Processing Systems*, 33, 2020.

[8] Jathushan Rajasegaran, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, and Mubarak Shah. itaml: An incremental task-agnostic meta-learning approach. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13588–13597, 2020.

[9] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *NIPS*, pages 2672–2680, 2014.

[10] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *ICLR*, 2014.

[11] Amanda Rios and Laurent Itti. Closed-loop memory gan for continual learning. *IJCAI*, 2019.

[12] Mengyao Zhai, Lei Chen, Frederick Tung, Jiawei He, Megha Nawhal, and Greg Mori. Lifelong gan: Continual learning for conditional image generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2759–2768, 2019.

[13] Yulai Cong, Miaoyun Zhao, Jianqiao Li, Sijia Wang, and Lawrence Carin. GAN Memory with no forgetting. *Advances in Neural Information Processing Systems*, 33, 2020.

[14] Marc Masana, Tinne Tuytelaars, and Joost van de Weijer. Ternary feature masks: continual learning without any forgetting. *arXiv preprint arXiv:2001.08714*, 2020.

[15] M Zhai, L. Chen, J. He, M. Nawhal, F. Tung, and G. Mori. Piggyback GAN: Efficient lifelong learning for image conditioned generation.

[16] Arun Mallya and Svetlana Lazebnik. Packnet: Adding multiple tasks to a single network by iterative pruning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7765–7773, 2018.

[17] Vinay Kumar Verma, Kevin J Liang, Nikhil Mehta, Piyush Rai, and Lawrence Carin. Efficient feature transformations for discriminative and generative continual learning. *CVPR*, 2021.

[18] Mitchell Wortsman, Vivek Ramanujan, Rosanne Liu, Aniruddha Kembhavi, Mohammad Rastegari, Jason Yosinski, and Ali Farhadi. Supermasks in superposition. *Advances in Neural Information Processing Systems*, 33, 2020.

[19] Jathushan Rajasegaran, Munawar Hayat, Salman H Khan, Fahad Shahbaz Khan, and Ling Shao. Random Path Selection for Continual Learning. *Neural Information Processing Systems*, 2019.

[20] Ju Xu and Zhanxing Zhu. Reinforced continual learning. In *Advances in Neural Information Processing Systems*, pages 899–908, 2018.

[21] Jaehong Yoon, Eunho Yang, Jeongtae Lee, and Sung Ju Hwang. Lifelong Learning with Dynamically Expandable Networks. *International Conference on Learning Representations*, 2018.

[22] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive Neural Networks. *arXiv preprint arXiv:1606.04671*, 2016.

[23] Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. Which training methods for gans do actually converge? In *International conference on machine learning*, pages 3481–3490. PMLR, 2018.

[24] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500, 2017.

[25] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[26] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[27] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6848–6856, 2018.

[28] Jaehong Yoon, Saehoon Kim, Eunho Yang, and Sung Ju Hwang. Scalable and Order-robust Continual Learning with Additive Parameter Decomposition. *International Conference on Learning Representations*, abs/1902.09432, 2020.

[29] Alessandro Achille, Michael Lam, Rahul Tewari, Avinash Ravichandran, Subhransu Maji, Charless C Fowlkes, Stefano Soatto, and Pietro Perona. Task2vec: Task embedding for meta-learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6430–6439, 2019.

[30] David Lopez-Paz, Ranzato, and Marc'Aurelio. Gradient episodic memory for continual learning. In *Advances in neural information processing systems*, pages 6467–6476, 2017.

[31] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming Catastrophic Forgetting in Neural Networks. *Proceedings of the National Academy of Sciences*, 2017.

[32] Heechul Jung, Jeongwoo Ju, Minju Jung, and Junmo Kim. Less-forgetting learning in deep neural networks. *arXiv preprint arXiv:1607.00122*, 2016.

[33] S. Rebuffi, A. Kolesnikov, G. Sperl, and C. Lampert. iCaRL: Incremental classifier and representation learning. In *CVPR*, pages 2001–2010, 2017.

[34] Joseph KJ and Vineeth Nallure Balasubramanian. Meta-Consolidation for Continual Learning. *Advances in Neural Information Processing Systems*, 33, 2020.

[35] Lu Yu, Bartlomiej Twardowski, Xialei Liu, Luis Herranz, Kai Wang, Yongmei Cheng, Shangling Jui, and Joost van de Weijer. Semantic drift compensation for class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6982–6991, 2020.

[36] Z. Li and D. Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017.

[37] Ziyang Wu, Christina Baek, Chong You, and Yi Ma. Incremental learning via rate reduction. *arXiv preprint arXiv:2011.14593*, 2020.

[38] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual Learning with Deep Generative Replay. *Neural Information Processing Systems*, 2017.

[39] G. van de Ven and A. Tolias. Generative replay with feedback connections as a general strategy for continual learning. *arXiv preprint arXiv:1809.10635*, 2018.

[40] Arun Mallya, Dillon Davis, and Svetlana Lazebnik. Piggyback: Adapting a single network to multiple tasks by learning to mask weights. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 67–82, 2018.

[41] Nikhil Mehta, Kevin J Liang, Vinay Kumar Verma, and Lawrence Carin. Continual Learning using a Bayesian Nonparametric Dictionary of Weight Factors. *Artificial Intelligence and Statistics*, 2021.

[42] Y. Bengio. Deep learning of representations for unsupervised and transfer learning. In *Proceedings of ICML workshop on unsupervised and transfer learning*, pages 17–36, 2012.

[43] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. *arXiv preprint arXiv:1312.6229*, 2013.

[44] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. How transferable are features in deep neural networks? In *NIPS*, pages 3320–3328, 2014.

[45] M. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *ECCV*, pages 818–833. Springer, 2014.

[46] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, pages 580–587, 2014.

[47] M. Long, Y. Cao, J. Wang, and M. Jordan. Learning transferable features with deep adaptation networks. *arXiv preprint arXiv:1502.02791*, 2015.

[48] Z. Luo, Y. Zou, J. Hoffman, and L. Fei-Fei. Label efficient learning of transferable representations acrosss domains and tasks. In *NIPS*, pages 165–177, 2017.

[49] Q. Sun, B. Schiele, and M. Fritz. A domain based approach to social relation recognition. In *CVPR*, pages 3481–3490, 2017.

[50] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *ICCV*, pages 3730–3738, 2015.

[51] M. Nilsback and A. Zisserman. Automated flower classification over a large number of classes. In *ICCV, Graphics & Image Processing*, pages 722–729. IEEE, 2008.

[52] F. Yu, Y. Zhang, S. Song, A. Seff, and J. Xiao. LSUN: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015.

[53] W. Zhang, J. Sun, and X. Tang. Cat head detection-how to effectively exploit shape and texture features. In *ECCV*, pages 802–816. Springer, 2008.

[54] J. Cheng, W. Yang, M. Huang, W. Huang, J. Jiang, Y. Zhou, R. Yang, J. Zhao, Y. Feng, and Q. Feng. Retrieval of brain tumors by adaptive spatial pooling and fisher vector representation. *PloS one*, 11(6), 2016.

[55] D. S Kermany, M. Goldbaum, W. Cai, C. CS Valentim, H. Liang, S. L Baxter, A. McKeown, G. Yang, X. Wu, F. Yan, et al. Identifying medical diagnoses and treatable diseases by image-based deep learning. *Cell*, 172(5):1122–1131, 2018.

[56] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV*, 115(3):211–252, 2015.

[57] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In *NIPS*, pages 6626–6637, 2017.

[58] G. van de Ven and A. Tolias. Three scenarios for continual learning. *arXiv preprint arXiv:1904.07734*, 2019.

[59] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] proposed approach may be used to generate fake data

(d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]

2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyper-parameters, how they were chosen)? [Yes] We have mentioned some of the details in the Sec. 4, remaining details we will provide in the supplementary document
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [N/A] We are modelling image generation, so it is not applicable
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] We will provide details in supplementary document
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] See Sec. 4
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] We will provide the code in the supplementary/URL
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A] data is publicly available
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]