Minimizing Data, Maximizing Performance: Generative Examples for Continual Task Learning

Anonymous CVPR submission

Paper ID 75

Abstract

Synthetic data is emerging as a powerful tool in computer vision, offering advantages in privacy and security. As generative AI models advance, they enable the creation of large-scale, diverse datasets that eliminate concerns related to sensitive data sharing and costly data collection processes. However, fundamental questions arise: (1) can synthetic data replace natural data in a continual learning (CL) setting? How much synthetic data is sufficient to achieve a desired performance? How well is the network generalizable when trained on synthetic data.

To address these questions, we propose a sample minimization strategy for CL that enhances efficiency, generalization, and robustness by selectively removing uninformative or redundant samples during the training phase. We apply this method in a sequence of tasks derived from the GenImage dataset [35]. This setting allows us to compare the impact of training early tasks entirely on synthetic data to analyze how well they transfer knowledge for the subsequent tasks or for evaluation on natural images. Furthermore, our method allows us to investigate the impact of removing potentially incorrect, redundant, or harmful training samples.

We aim to maximize CL efficiency by removing uninformative images and enhance robustness through both adversarial training and structured data removal. We experimentally study how the training order of synthetic and natural data, and what generative models are used, significantly impact CL performance maximization and the natural data minimization. Our findings provide key insights into how generative examples can be leveraged for adaptive and efficient CL in evolving environments.

1. Introduction

In recent years, advances in generative artificial intelligence (Gen AI) have led to remarkable performance across a wide range of tasks such as object detection [17, 18], image classification [2, 3], and natural language processing [7, 11]. However, one limitation of such algorithms is their need for extensive real-world data, which can be difficult to access due to privacy concerns or costly data collection. To tackle these problems, synthetic data, which is generated using models such as Generative Adversarial Networks (GANs) [28], Variational Autoencoders (VAEs) [15], and diffusion models [30], has been suggested as an alternative to natural data. The growth of interest in synthetic data has brought an important question in computer vision: Can synthetic data fully or partially substitute for natural data? While several studies have shown that synthetic data can enhance performance or reduce reliance on real data in standard learning settings, to the best of our knowledge, there is no prior work on their impact in more challenging scenarios such as continual learning (CL). CL refers to the setup where the model learns multiple tasks sequentially without access to whole previous task data, with a key challenge of catastrophic forgetting of previously learned tasks [1, 33].

In this paper, we aim to investigate the effective use of synthetic data as a replacement for natural data in the context of CL. Specifically, we aim to enhance the efficiency, robustness, and generalizability of models trained in CL settings by removing uninformative samples. This removal is of particular interest in conjunction with the use of synthetic samples, where (1) there is an increased risk of generative models to make uninformative or erroneous samples compared to the use of natural data; (2) training on synthetic samples is costly. Hence, having a method in place to filter such samples from the training data provides a valuable tool for the efficient and robust CL when used alongside synthetic training samples. To this end, we propose a sample removal framework that identifies and removes uninformative examples. Our experiments further explore the tradeoff between data quantity and robustness across sequential tasks under both natural and adversarial conditions.

Our contribution: In this paper, we (1) study the role of synthetic data in CL and evaluate how well it can substitute for natural data under both standard and adversarial training settings, and (2) propose a loss-based sample removal strategy EpochLoss to identify and eliminate redundant or uninformative examples. Our framework explores two knowledge transfer strategies between tasks: (i) passing the model trained on the entire dataset of the previous task, and (ii) passing the model trained only on the informative subset.

2. Sample Removal Methodology

2.1. Problem Formulation

In this section, we first introduce a brief list of notations, followed by the loss function formula in CL setup.

Table 1. Summary of Notation

Symbol	Description
f	Neural network model (classifier)
T_i	i^{th} task in the sequence of tasks
$\mathcal{D}_i$	Data distribution associated with task T_i
$\bar{\mathcal{D}}_i$	Data distribution associated with the pruned (selected) subset of task i after sample removal
$\mathcal{T}$	Number of epochs used to compute average loss for sample ranking
r	Percentage of samples to be removed
l	Cross entropy loss function
θ	Full set of network parameters
$\mathcal{L}(\theta)$	Loss function parameterized by the model weights
θ_i	Trainable parameters specific to task T_i
$S_{1,\ldots,i}^{Frozen}$	Frozen sub-network for tasks $T_1,\ldots,T_i$
S_i^{Free}	Free sub-network for task T_i
$\bar{s}_i$	Average loss value of sample x_i across $\mathcal{T}$ epochs
$s_i^{(e)}$	Loss value of sample x_i at epoch e
$\mathcal{D}_i^A$	Distribution of adversarially perturbed samples for task T_i
ACC^A	Adversarial test accuracy of the network

Suppose that we are given a sequence of tasks $T_1, T_2, \ldots, T_t$, each associated with dataset $\{(x_i^{(k)}, y_i^{(k)}) | \, i = 1, \ldots, N_k\}$ coming from distribution $\mathcal{D}_k$. let $\bar{\mathcal{D}}_k$ be the distribution of the remaining samples after removing the most uninformative examples from T_k. Let $\mathcal{D}_k^A$ and $\bar{\mathcal{D}}_k^A$ denote the adversarially perturbed versions of the full and pruned datasets, respectively.

In CL setting, at each step, the model is trained on one task at a time without access to the full data from previous tasks. The goal is to minimize the loss on the current task while preserving performance on all previous learned tasks. Let us define the loss function in CL: let f denote a neural network with parameter set θ, trained sequentially on tasks $T_1, \ldots, T_{k-1}$. Following Algorithm 1 (Steps 13 and 14), after training the network on each task, we prune the network and fine-tune it. This approach freezes some important filters and parameters for the current task to not update them during learning a new task. This pruning results in splitting the network into two sub-networks:

- $S_{1,\ldots,k-1}^{frozen}$: The frozen sub-network retaining knowledge from previous tasks, with parameters $\theta_{1,\ldots,k-1}$.
- S_k^{free}: The free trainable sub-network for the next task T_k, with parameters θ_k.

Thus, the full model is represented as $f = S_{1,\ldots,k-1}^{frozen} \cup S_k^{free}$, and only θ_k is updated during training on task T_k. By considering $\ell(\cdot,\cdot)$ as cross-entropy loss, the loss function for task T_k is defined as:

$$\mathcal{L}_k(\theta_k) = \mathbb{E}_{(x,y)\sim\mathcal{D}_k}\left[\ell(f_{\theta_k}(x), y)\right], \tag{1}$$

where the optimal parameters are given by:

$$\theta_k^* = \arg\min_{\theta_k} \mathcal{L}_k(\theta_k). \tag{2}$$

To preserve performance on all tasks, the total loss across tasks $T_1, \ldots, T_t$ is:

$$L(\theta) = \sum_{k=1}^{t} \mathcal{L}_k(\theta_k). \tag{3}$$

In addition to standard training, we incorporate adversarial training (Step 12 in Algorithm 1), which is applied either on the entire adversarially perturbed dataset of the current task k with distribution $\mathcal{D}_k^A$, or a subset of T_k after removing uninformative samples drawn from distribution $\bar{\mathcal{D}}_k^A$. The total loss (adversarial training objectives) over all sequential tasks is defined by:

$$L(\theta) = \sum_{k=1}^{t} \mathbb{E}_{(x,y)\sim\mathcal{D}_k^A}\left[\ell(f_{\theta_k}(x), y)\right]. \tag{4}$$

For the subset adversarial dataset, we have:

$$L(\theta) = \sum_{k=1}^{t} \mathbb{E}_{(x,y)\sim\bar{\mathcal{D}}_k^A}\left[\ell(f_{\theta_k}(x), y)\right]. \tag{5}$$

In the following part, we propose our sample removal framework, which is inspired by CAPER [8]. CAPER proposes a sample removal strategy to improve performance, efficiency, and robustness in a standard setting, by removing samples that are highly susceptible to noises. However, based on our experiments, by extending their strategy to a CL setup, CAPER still under-performs in both accuracy and robustness compared to our loss-based sample removal method. A detailed explanation of CAPER for CL is provided in the supplementary material.

2.2. Epoch-Accuracy Strategy for CL

In this paper, we propose a sample removal approach, designed to identify and remove uninformative samples in a CL setup. Our method uses average loss of samples over a fixed number of training epochs $\mathcal{T}$ to select informative samples and improve model robustness and generalization. Our approach not only enhances learning efficiency but also ensures resilience against adversarial attacks. When training on synthetic data, it additionally provides a filter by which we can remove malformed samples that may be produced by the generative model. Our framework consists of two main components: (1) adversarial baseline training and (2) selective sample removal.

The first part serves as a baseline in which we train the network (adversarially) on the whole dataset of each task, and the second part focuses on the process of identifying and removing redundant nodes.

Baseline Training: The baseline training phase is represented in Algorithm 1 with a sample removal percentage of zero, where Steps 5–14 are skipped, as a result $\bar{\mathcal{D}}_k^A = \mathcal{D}_k^A$. In this setup, the model undergoes adversarial training across a sequence of tasks in a CL fashion. Although Step 15 in Algorithm 1 specifies adversarial training, the same procedure is applied under standard (non-adversarial) training conditions. After training each task, we apply pruning and fine-tuning strategies to preserve useful representations (subnetwork learning). Specifically, a subset of the network's weights is frozen to retain task-specific knowledge, while the remaining weights are updated during subsequent tasks. The Baseline algorithm is used in two different ways:

- By using baseline training, we can directly evaluate the effect of substituting natural images with synthetic (generative) images across tasks.
- It is also used as a reference to compare against models trained after removing uninformative samples to evaluate the effect of removing redundant or harmful samples.

Sample Selection and Removal: In the second stage, we extend the baseline by incorporating selective sample removal. Starting from the fine-tuned model obtained from the previous task (trained on the whole task's dataset or what remains after removal), we train the network non-adversarially on the current task for a specific number of epochs. During this phase, we analyze the model's response to unperturbed training data, aiming to identify susceptible samples—those that may hinder adversarial training due to being vulnerable to attacks. Our methodology supports both standard and adversarial training, making it adaptable to different robustness goals.

To be more specific, in the second stage, starting from the fine-tuned model obtained from the previous tasks $T_1, \ldots, T_{k-1}$ (either trained on the full dataset or a subset after removal), we train the network on the current task T_k for a fixed number of epochs ($\mathcal{T}$), using clean (unperturbed) data in a standard (non-adversarial) training setting. During this $\mathcal{T}$-epoch training phase on unperturbed data, we monitor the per-sample loss at each epoch and compute the average loss, as a measure of sample informativeness, for every training example in $\mathcal{D}_k$. Specifically, for a sample $(x_i, y_i) \in \mathcal{D}_k$, let $s_i^{(e)} = L(f_\theta(x_i), y_i)$, denote its loss at epoch e. We define the informativeness score $\bar{s}_i$ of sample i as its average loss across the $\mathcal{T}$ epochs:

$$\bar{s}_i = \frac{1}{\mathcal{T}} \sum_{e=1}^{\mathcal{T}} s_i^{(e)}. \tag{6}$$

Algorithm 1 Removing Uninformative Samples

```
Input: Network $f$ with parameter set $\theta$, Tasks $\{T_1, \ldots, T_t\}$; $E$: Total # of training epochs; $E_1$: Total # of fine-tuning epochs; $\mathcal{T}$: Epochs for loss averaging, r: Sample removal percentage
1  for $T_k \in \{T_1, \ldots, T_t\}$ do
2      Initialization:
3          Initialize the network with a pretrained model on Tiny ImageNet
4          if $k > 1$ then
5              Load the trained model on the previous task with two sub-networks $S_{1,\ldots,k-1}^{frozen}$ and $S_k^{free}$ (The loaded model is either trained on full or pruned dataset of previous task)
6      Loss-Based Scoring:
7          Train the sub-network $S_k^{free}$ non-adversarially on examples from $\mathcal{D}_k$ for $\mathcal{T}$ epochs
8          for each sample $(x_i, y_i) \in \mathcal{D}_k$ do
9              for each epoch $e \in \{1, ..., \mathcal{T}\}$ do
10                 Compute per-sample loss $s_i^{(e)} = \mathcal{L}(f_\theta(x_i), y_i)$
11             Compute the average loss over $\mathcal{T}$ epochs:
                   $\bar{s}_i = \frac{1}{\mathcal{T}} \sum_{e=1}^{\mathcal{T}} s_i^{(e)}$
12     Removal Rank all training samples in descending order based on their average loss $\bar{s}_i$. Identify and remove the top $r\%$ samples with the highest $\bar{s}_i$ values, resulting in the pruned dataset $\bar{\mathcal{D}}_k$.
13     Adv. Attacked For each $(x_i, y_i) \in \bar{\mathcal{D}}_k$, apply an adversarial attack to obtain $\tilde{x}_i = x_i + \epsilon$, resulting in a perturbed dataset with distribution $\bar{\mathcal{D}}_k^A$.
14     Go back to the Initialization step.
15     Training Train the sub-network $S_k^{free}$ adversarially on $\bar{\mathcal{D}}_k^A$ for $E$ epochs, and report $ACC^A$ on adversarial test examples $\mathcal{D}_k^A$.
16     Prune the network and fine-tune it for $E_1$ epochs. This step results in two new sub-networks $S_{1,\ldots,k}^{frozen}$ and $S_{k+1}^{free}$.
```

Once all $\bar{s}_i$ values are computed for samples in $\mathcal{D}_k$, we

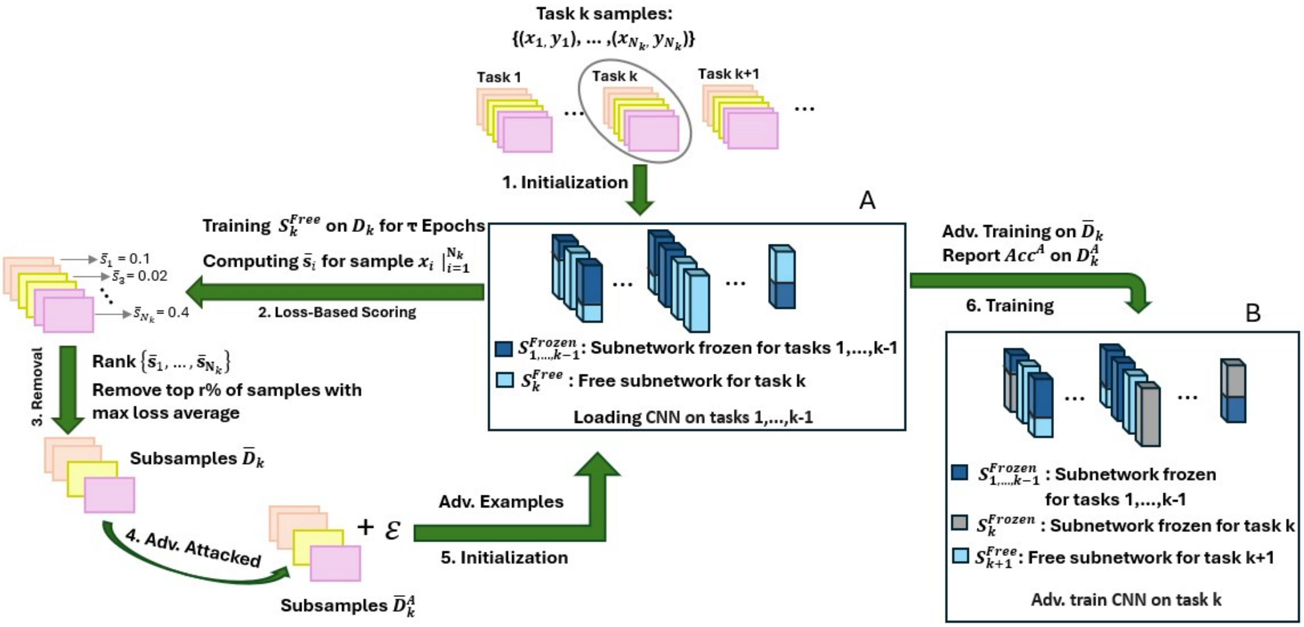
Task k samples:
$\{(x_1, y_1), \ldots, (x_{N_k}, y_{N_k})\}$
Task 1
Task k
Task k+1
1. Initialization
A
Training S_k^{Free} on D_k for τ Epochs
Computing $\bar{s}_i$ for sample $x_i \,|_{i=1}^{N_k}$
2. Loss-Based Scoring
$\bar{s}_1 = 0.1$
$\bar{s}_3 = 0.02$
$\bar{s}_{N_k} = 0.4$
$S_{1,\ldots,k-1}^{Frozen}$: Subnetwork frozen for tasks 1,...,k-1
S_k^{Free} : Free subnetwork for task k
Loading CNN on tasks 1,...,k-1
3. Removal
Rank $\{\bar{s}_1, \ldots, \bar{s}_{N_k}\}$
Remove top r% of samples with max loss average
Subsamples $\bar{D}_k$
4. Adv. Attacked
$+\ \varepsilon$
Adv. Examples
5. Initialization
Subsamples $\bar{D}_k^A$
Adv. Training on $\bar{D}_k$
Report Acc^A on D_k^A
6. Training
B
$S_{1,\ldots,k-1}^{Frozen}$: Subnetwork frozen for tasks 1,...,k-1
S_k^{Frozen} : Subnetwork frozen for task k
S_{k+1}^{Free}: Free subnetwork for task k+1
Adv. train CNN on task k

Figure 1. The proposed framework consists of both baseline training (no removal) and the sample removal phase.

rank them in descending order, and remove the top $r\%$ of high-loss examples as uninformative or potentially detrimental. These samples are then removed from the dataset, resulting in a subset with a distribution $\bar{\mathcal{D}}_k$, which retains only the most informative and stable examples for learning.

After this removal step, we adversarially retrain the model on the remaining data from task T_k, with distribution $\bar{\mathcal{D}}_k$ and report the adversarial accuracy of the network on adversarial test examples with distribution $\mathcal{D}_k^A$. Finally, for transferring knowledge to the next task, task $k+1$ in the CL sequence, we explore two strategies: (i) transferring the model trained adversarially on the entire dataset of the current task with distribution $\mathcal{D}_k$, or (ii) transferring the model trained adversarially on the pruned dataset (after removal) with distribution $\bar{\mathcal{D}}_k$.

As shown in Figure 1, we consider two transfer strategies moving from one task to the other. Considering the current task as T_k, in the first strategy, network A is trained adversarially on the full dataset of all previous sequential tasks $T_1, \ldots, T_{k-1}$, followed by pruning and fine-tuning, and then transferred to task T_k. In the second strategy, network B we train only on the remained (informative) subset of previous tasks, denoted by $\bar{\mathcal{D}}_1, \ldots, \bar{\mathcal{D}}_{k-1}$.

3. Experiments

The experimental results section is divided into four parts: (1) Setup, (2) comparing removal methods on natural data, (3) applying substitution of natural tasks with synthetic data, and (4) comparing usefulness of different generative models for substitution of natural images. In the first part, the hyperparameters and all datasets are discussed. The second and third parts discuss the performance and robustness of removing or substitution of natural images with synthetic images in standard and adversarial training scenarios within a CL setup. The last part compares the usefulness of each included generative model to substitute training data and maintain the ability to generalize to natural data.

3.1. Setup

Here, we briefly explain the datasets, types of adversarial attacks, and corruption used through our experiments, with detailed hyper-parameters.

Datasets We conducted experiments using a variety of datasets, including synthetic and real-world benchmarks.

Synthetic Data: For synthetic data, we derive six CL tasks from the generative GenImage [35] dataset. This dataset provides synthetic images of Imagenet classes derived from various types of generative models, including GANs and diffusion models, along with subsets of natural Imagenet images. From these generators, we construct six tasks each consisting of disjoint subsets of 100 classes from Imagenet. For each task, we construct both a synthetic and a natural copy to use for the task's training data. We denote this task sequence as GenImage-Disjoint. For tasks 1-6, the generators used are ADM [6], BigGAN [5], Midjourney [21], Glide [22], Stable Diffusion (v1.4) [27], and VQDM [10], respectively. To compare generator usefulness in data substitution, we additionally create copies of each task consisting only of images generated by one of the generators. In this case, a given task contains the same classes for all

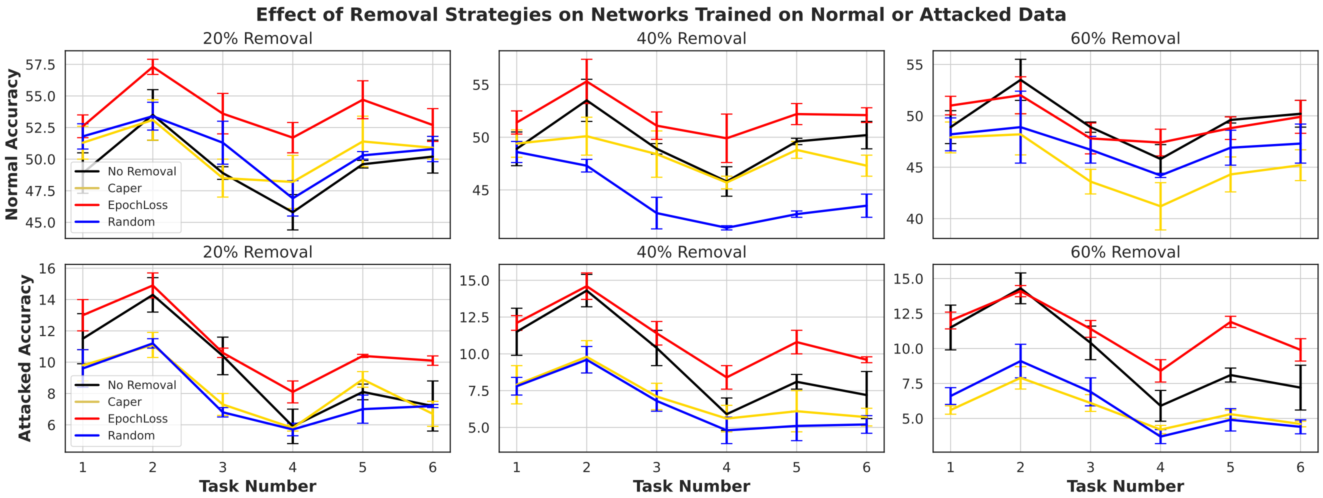

Figure 2. The test accuracy is compared for each task when removing data under different methods for a sequence of natural image tasks. For normally trained networks (top) the normal test accuracy is given, while adversarial accuracy is reported for adversarially trained networks (bottom). Our EpochLoss method frequently and significantly outperforms the case of not removing any training data.

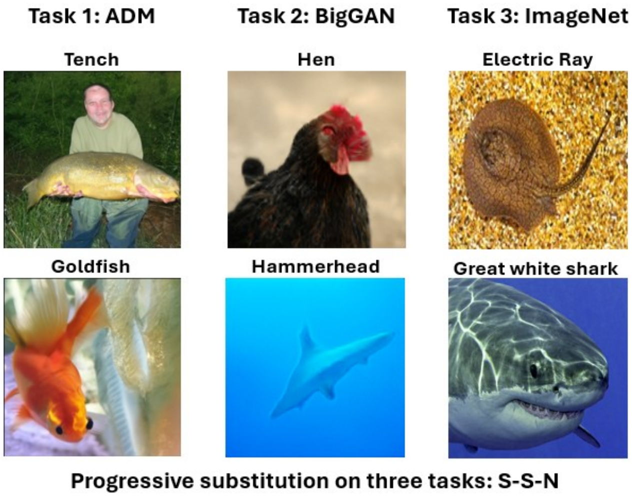

Figure 3. Progressive substitution on three tasks with disjoint classes, where the first task is composed of images generated by ADM, the second is generated by BigGan, and the third one is natural images in ImageNet.

datasets, but those classes are generated by a different generator. We denote these datasets by the generator used (e.g. GenImage-ADM).

The different task configurations investigated in our experiments are as follows:

- No substitution: All tasks are trained on natural images, giving a task string of N-N-N-N-N-N, with 'N' denoting natural image tasks. Training and evaluation are both on natural images.
- Mixed substitution: For the first three tasks of GenImage-Disjoint, we consider different combinations of substituting natural tasks with synthetic ones (N-N-N, N-N-S, N-S-N, etc) as demonstrated in Figure 3 to determine the impact of substitution and the choice of which tasks in the sequence get substituted.
- Progressive Substitution: The first t tasks use synthetic training data from their corresponding generator, while the subsequent tasks use natural images. We consider the impact of gradually increasing the value of t.

We evaluate synthetic tasks on their corresponding natural subset to examine how well the model generalizes when trained on synthetic data.

In addition to this synthetic-vs-natural setting, we evaluate our method in a standard learning setting, using one task including a subset of CIFAR100 [16], to show its effectiveness over CAPER, and random removal. More detailed explanation is provided in the SM.

CNN Architecture We use ResNet-18 to evaluate our approach, as well as VGG16 within the SM.

Adversarial Attacks and Corruption To investigate the effect of sample removal on adversarial robustness, we use the adversarial attack PGD, as well as the corruption effects Gaussian Noise, Gaussian Blur, Saturate and Rotate [2]. We used standard and adversarial accuracies over test samples and perturbed test samples, respectively, in order to measure the performance of our algorithm.

Hyper-Parameters We trained both networks for up to 300 epochs, with an additional 150 epochs for finetuning after freezing task-specific components of the model. The learning rate is set to 0.1 throughout training, except for VGG16 during finetuning, which is reduced to 0.01. For more efficient learning in terms of time and memory, we used early stopping strategy to stop the training whenever it is con-

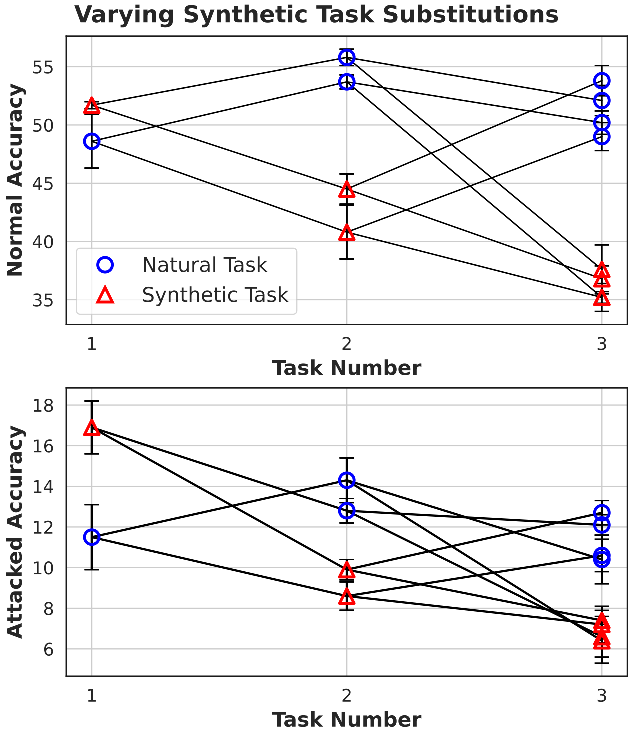

Figure 4. We report the accuracy of the first three tasks in GenImage-Disjoint when substituting tasks with synthetic training data. Choice of synthetic or natural data for each task is denoted, with some sequences (e.g. N-N-N and N-N-S) sharing early task values. For both the non-adversarially trained (top) and adversarial (bottom) settings we observe that the impact of substitution on natural test accuracy varies based on which task is substituted, but is capable of improving accuracy in some cases.

verged. To this end, we decreased the learning rate by a factor of 0.1 if the validation accuracy doesn't improve for 20 epochs. If the learning rate falls below a minimum threshold of 0.0001, the learning process will stop to prevent unnecessary computation. We used a batch size of 128, $\mathcal{T}$ of 50 for both networks, and SGD Optimizer with momentum of 0.9 and weight decay of 0 to prevent frozen weights to be updated. All experiments were averaged over 3 trials.

3.2. Comparison of Data Removal Methods

To investigate the impact of removing training data following Algorithm 1, we initially train on the natural GenImage-Disjoint tasks with no substitution. We either train the network adversarially under PGD attack, or normally. We compare the accuracies when removal is done using CAPER and the proposed loss-based removal method. We compare against no removal and random removal as controls. Figure 2 shows that for both normal and adversarial settings, our EpochLoss strategy significantly outperforms even the case of removing no training data. Although the accuracy eventually deteriorates as the amount of data removed increases, even when removing approximately half of the training data EpochLoss maintains or improves the baseline accuracy. As the process of collecting training done during loss-based scoring is non-adversarial, the overhead it adds to the runtime is offset by the efficiency we gain in the adversarial training when removing this training data.

3.3. Substitution with Synthetic Training Data

In addition to the full removal of training samples, we consider the impact of replacing them with synthetic samples. For these experiments we limit this investigation to the case where entire tasks are replaced. Figure 4 shows the impact of different sequences of substitution on the first 3 tasks of GenImage-Disjoint under both adversarial and normal training. For these experiments, no removal is performed. We see that the use of synthetic data can match or even improve upon the accuracy of natural task data, particularly in the non-adversarial setting. For adversarial tasks substitution with synthetic images has more mixed results but still often matches or exceeds natural accuracies.

To extend this setting, we consider the combination of both removal and substitution of training data. Here, we progressively substitute more of the initial tasks in the GenImage-Disjoint dataset. We report the average test accuracy over all tasks, using the normal accuracy for normally trained networks and adversarial accuracy for adversarially trained networks. The results in Figure 5 show that there is initially a benefit to substitution with synthetic tasks, however accuracy begins to decrease as more tasks are substituted beyond the first two or three. EpochLoss outperforms the alternative removal methods and often outperforms no removal. This demonstrates a strong potential for reducing the number of natural samples needed either through removal or substitution with generative alternatives.

3.4. Comparison of Generative Models

While we have shown that substitution of natural tasks with synthetic samples can outperform the use of natural data, this was not shown to always be the case, and we observe in Figures 4 and 5 that there are settings where it can significantly worsen performance. This is complicated by the sequential training of CL, making it less clear if this is due to the use of generative task data for multiple sequential tasks, or due to the specific generators chosen for the later tasks. To address this we consider comparisons between versions of GenImage-Disjoint where all six tasks are made by a given generator. Each task uses the same classes as in the original dataset, and is evaluated on the same set of natural images. This allows us to directly compare each generative model's usefulness in substituting natural images. We report the results without removal, and consider the impact

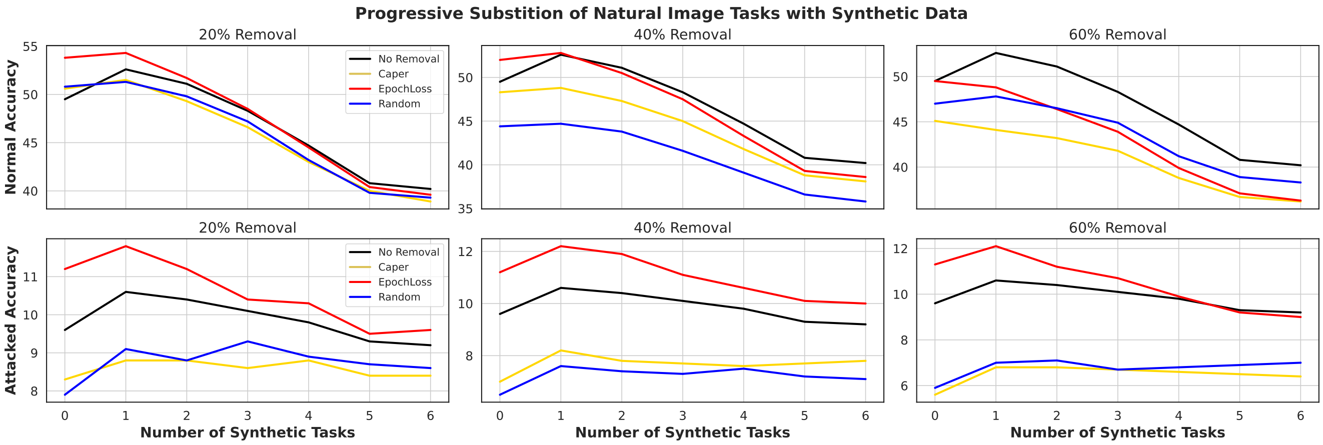

Figure 5. The average accuracy is reported over all tasks when the network is non-adversarially trained (top) or adversarially trained (bottom). We compare between the settings of using no removal against using EpochLoss, CAPER, random removal. EpochLoss frequently matches or outperforms the baseline accuracy when removing up to 40% or more of the training data.

of using EpochLoss for this comparison within the SM. Figure 6 shows that for both non-adversarial training (top) and adversarial training (bottom), there are clear cases where some generators better prepare the network to generalize to the natural data when used for training. We see that ADM consistently matches or exceeds the natural data when used for training, as evaluated on natural test data. Furthermore Midjourney and Glide give the worst accuracies on natural test data. This may partially explain why we observe the accuracy increase when substituting task 1 in Figure 5, but subsequently see it begin to deteriorate after substituting the 3rd and 4th tasks.

4. Related Work

Continual Learning Continual learning (CL) aims to train models on sequential tasks while preventing catastrophic forgetting. Existing approaches to CL fall into three main categories: replay-based methods, which store or regenerate past data [28], regularization-based techniques, which constrain weight updates to preserve previous knowledge [25], and architectural modifications, such as expanding the network dynamically [4]. Among these, freezing a subset of the model's parameters has been widely studied as a way to balance knowledge retention with adaptability [26]. However, while these approaches focus on preserving past knowledge, they do not explicitly address whether all training samples are beneficial, particularly in the presence of adversarial noise, label noise, and data corruption. Notably, many replay methods use generative models to generate samples from previously learned tasks, enabling the network to remember that task [19, 28, 34]. By contrast, we are interested in how well such synthetic samples can help the model learn the task initially.

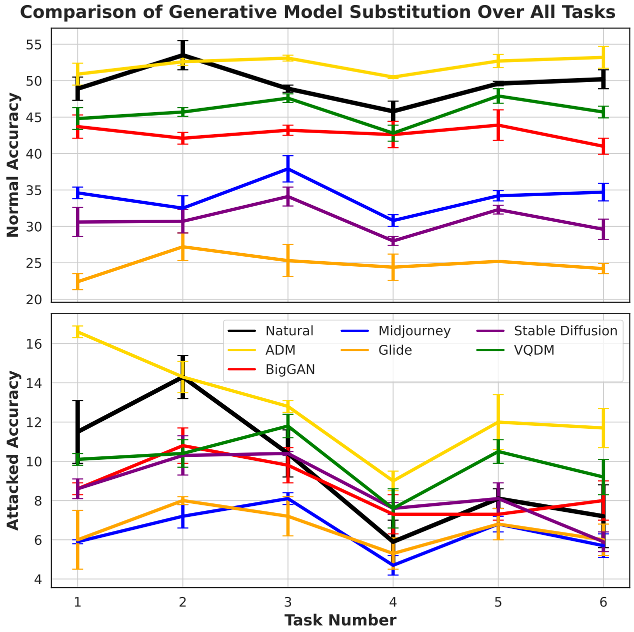

Figure 6. For each single-generator variant of GenImage-Disjoint, we compare the natural test accuracy. We compare training on synthetic data for all tasks against training on the natural tasks of GenImage-Disjoint. In both the non-adversarial (top) and adversarial (bottom) settings, certain generators such as ADM perform significantly better than others in enabling the network to generalize to natural images, even outperforming natural training data.

Synthetic Data The use of synthetic data has gained much interest in computer vision and deep learning, where real-world data is scarce, sensitive, or costly to collect. Generative models such as GANs [28], VAEs [15], and diffusion models [30] have been widely used to create synthetic datasets that supplement or replace real-world data in training deep networks. These methods have demonstrated effectiveness in various tasks such as classification, detection, and segmentation. For instance, [9] showed that synthetic data could bridge the domain gap in object detection tasks, while [32] demonstrated that models trained on synthetic data can generalize well when designed with sufficient diversity and realism. We instead look to answer how effective synthetic data is for learning tasks in the CL setting.

Sample Selection and Sample Removal Beyond data generation, sample selection plays a critical role in robust learning under data noise, adversarial attacks, and natural corruption. Recent work in adversarial robustness [20] demonstrates that deep networks are highly sensitive to small perturbations, leading to misclassifications. Natural corruptions, such as blur, noise, and contrast shifts, significantly degrade model performance. [12] systematically benchmarked these effects and demonstrated that deep networks struggle under such perturbations.

Recent advancements have also explored intelligent sample selection and removal techniques to further enhance model robustness under adversarial attacks and natural corruptions. Such techniques may aim to retain samples that provide easy and diverse training data [13], are fair and low-loss [23], or are otherwise deemed important for the task [14, 24, 26]. Others have considered removal of easily-forgotten data [31] or trivial samples that don't contribute to challenging or informing the gradient updates in the network [29]. These approaches demonstrate that careful sample selection—even in synthetic CL—can reduce computational cost, enhance robustness, and maintain accuracy in the face of noise and distributional shifts. Additional works have aimed to remove samples most susceptible to noise

In the adversarial setting, Q-TART [8] notably introduces a fast and robust training pipeline by selecting high-quality adversarial examples that improve both robustness and transferability across tasks. By excluding low-quality adversarial examples, Q-TART achieves faster convergence and better performance under adversarial conditions.

In contrast to these works, our study proposes a systematic sample removal framework specifically designed for adversarial CL using synthetic datasets. Unlike prior methods focusing on label noise or reweighting heuristics, we directly address the core challenge of identifying and removing harmful samples during sequential training.

5. Discussion and Conclusion

In this study, we investigate the effectiveness of substituting natural images with synthetic data in CL, and we introduce a sample removal framework for CL designed to improve efficiency, generalization, and robustness by removing uninformative samples during the training phase.

While previous works have demonstrated the use of synthetic data instead of natural images in terms of improving performance and reducing reliance on real data in standard learning settings, there remains a gap in understanding how such data performs in CL scenarios and how much data is enough. Some contexts such as replay methods have made use of synthetic images, such as those generated by GAN models trained on a given task, to avoid forgetting. By contrast we show here the ability of such generative training data to allow a network to learn a new task as well.

Our experimental findings clearly illustrate that regardless of using natural images or synthetic images as subsequent tasks, our proposed EpochLoss strategy outperforms other removal methods and often even the baseline scenario, where no data is removed, under either normal and adversarial training. In addition, in the absence of sample removal, we observe that, in some cases, substituting natural tasks with synthetic samples can lead to better performance than using only natural images, particularly in the non-adversarial setting. These results highlight the potential to reduce the number of natural samples needed either through removal or substitution with generative alternatives.

The ability to attain better natural test accuracy when generalizing from synthetic data may initially be unintuitive, however there are some potential causes which may explain this result. If we consider that the features present in the synthetic images are those that a generative model strongly associates with a class, then we can view the substitution process as a form of knowledge transfer from an expert model. Here the generator is an expert on the features of each class, and is ideally passing images that were generated to contain important information for the model to learn those classes. In this way, a useful generator could avoid incidental features present in natural images which may be unassociated with the class label. Furthermore, by applying our removal approach EpochLoss, we can help mitigate the cases where the generator produces erroneous samples, such as by misinterpreting the prompt used to generate an image and effectively creating a mislabeled sample.

We observe in our experiments that the combination of removal and substitution of training data, as in Figure 5 can improve upon the accuracy obtained by only substituting the data. As we also see improvements when removing in the natural image tasks, it is difficult to discern how much of this improvement may be due to this filtering role, but it is a relationship between this substitution and removal settings that remains an avenue of interest for further investigation. There is also a potential that the removal of incidental features through substitution, or the removal of challenging samples through loss-based removal may lead to issues with the network in terms of overfitting or generalization as we may be removing rare or challenging features from the data. Despite this, we do not observe significant cases of such issues in our experiments. It still remains an important point to investigate in subsequent works to better consider the impact of synthetic data on CL.

References

[1] Joshua Andle and Salimeh Yasaei Sekeh. Theoretical understanding of the information flow on continual learning performance. In *European Conference on Computer Vision*, pages 86–101. Springer, 2022. 1

[2] Shekoofeh Azizi, Simon Kornblith, Chitwan Saharia, Mohammad Norouzi, and David J Fleet. Synthetic data from diffusion models improves imagenet classification. *arXiv preprint arXiv:2304.08466*, 2023. 1, 5

[3] Jordan J Bird and Ahmad Lotfi. Cifake: Image classification and explainable identification of ai-generated synthetic images. *IEEE Access*, 12:15642–15650, 2024. 1

[4] Vladimir Braverman, Avinatan Hassidim, Yossi Matias, Mariano Schain, Sandeep Silwal, and Samson Zhou. Adversarial robustness of streaming algorithms through importance sampling. *Advances in Neural Information Processing Systems*, 34:3544–3557, 2021. 7

[5] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018. 4

[6] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021. 4

[7] Peter Eigenschink, Thomas Reutterer, Stefan Vamosi, Ralf Vamosi, Chang Sun, and Klaudius Kalcher. Deep generative models for synthetic data: A survey. *IEEE Access*, 11: 47304–47320, 2023. 1

[8] Madan Ravi Ganesh, Salimeh Yasaei Sekeh, and Jason J Corso. Q-tart: Quickly training for adversarial robustness and in-transferability. *arXiv preprint arXiv:2204.07024*, 2022. 2, 8

[9] Georgios Georgakis, Arsalan Mousavian, Alexander C Berg, and Jana Kosecka. Synthesizing training data for object detection in indoor scenes. *arXiv preprint arXiv:1702.07836*, 2017. 8

[10] Shuyang Gu, Dong Chen, Jianmin Bao, Fang Wen, Bo Zhang, Dongdong Chen, Lu Yuan, and Baining Guo. Vector quantized diffusion model for text-to-image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10696–10706, 2022. 4

[11] Xuanli He, Islam Nassar, Jamie Kiros, Gholamreza Haffari, and Mohammad Norouzi. Generate, annotate, and learn: Nlp with synthetic text. *Transactions of the Association for Computational Linguistics*, 10:826–842, 2022. 1

[12] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019. 8

[13] Lu Jiang, Deyu Meng, Shoou-I Yu, Zhenzhong Lan, Shiguang Shan, and Alexander G Hauptmann. Self-paced learning with diversity. *Advances in neural information processing systems*, 27, 2014. 8

[14] Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *International conference on machine learning*, pages 2304–2313. PMLR, 2018. 8

[15] Diederik P Kingma, Max Welling, et al. Auto-encoding variational bayes, 2013. 1, 7

[16] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 5

[17] Shaobo Lin, Kun Wang, Xingyu Zeng, and Rui Zhao. Explore the power of synthetic data on few-shot object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 638–647, 2023. 1

[18] Lanlan Liu, Michael Muelly, Jia Deng, Tomas Pfister, and Li-Jia Li. Generative modeling for small-data object detection. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6073–6081, 2019. 1

[19] Xialei Liu, Chenshen Wu, Mikel Menta, Luis Herranz, Bogdan Raducanu, Andrew D Bagdanov, Shangling Jui, and Joost van de Weijer. Generative feature replay for class-incremental learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 226–227, 2020. 7

[20] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017. 8

[21] Midjourney. Midjourney website, 2022. 4

[22] Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741*, 2021. 4

[23] Deep Patel and PS Sastry. Adaptive sample selection for robust learning under label noise. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3932–3942, 2023. 8

[24] Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. Learning to reweight examples for robust deep learning. In *International conference on machine learning*, pages 4334–4343. PMLR, 2018. 8

[25] Leslie Rice, Eric Wong, and Zico Kolter. Overfitting in adversarially robust deep learning. In *International conference on machine learning*, pages 8093–8104. PMLR, 2020. 7

[26] Yuji Roh, Kangwook Lee, Steven Whang, and Changho Suh. Sample selection for fair and robust training. *Advances in Neural Information Processing Systems*, 34:815–827, 2021. 7, 8

[27] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022. 4

[28] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. *Advances in neural information processing systems*, 30, 2017. 1, 7

[29] Abhinav Shrivastava, Abhinav Gupta, and Ross Girshick. Training region-based object detectors with online hard example mining. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 761–769, 2016. 8

[30] Chawin Sitawarin, Supriyo Chakraborty, and David Wagner. Sat: Improving adversarial training via curriculum-based loss smoothing. In *Proceedings of the 14th ACM Workshop on Artificial Intelligence and Security*, pages 25–36, 2021. 1, 7

[31] Mariya Toneva, Alessandro Sordoni, Remi Tachet des Combes, Adam Trischler, Yoshua Bengio, and Geoffrey J Gordon. An empirical study of example forgetting during deep neural network learning. *arXiv preprint arXiv:1812.05159*, 2018. 8

[32] Gul Varol, Javier Romero, Xavier Martin, Naureen Mahmood, Michael J Black, Ivan Laptev, and Cordelia Schmid. Learning from synthetic humans. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 109–117, 2017. 8

[33] Liyuan Wang, Xingxing Zhang, Hang Su, and Jun Zhu. A comprehensive survey of continual learning: Theory, method and application. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024. 1

[34] 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. 7

[35] Mingjian Zhu, Hanting Chen, Qiangyu Yan, Xudong Huang, Guanyu Lin, Wei Li, Zhijun Tu, Hailin Hu, Jie Hu, and Yunhe Wang. Genimage: A million-scale benchmark for detecting ai-generated image, 2023. 1, 4