Eberhard Karls Universität Tübingen

Mathematisch-Naturwissenschaftliche Fakultät

Lernbasierte Computer Vision

Masterarbeit

Generative Dataset Distillation: A New Hope?

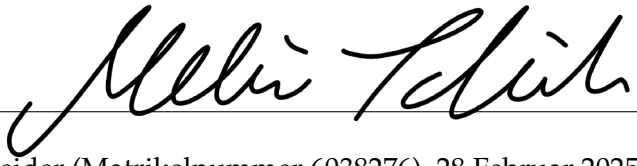
Eberhard Karls Universität Tübingen
Mathematisch-Naturwissenschaftliche Fakultät
Wilhelm-Schickard-Institut für Informatik
Lernbasierte Computer Vision
Melanie Schneider, melanie.schneider@student.uni-tuebingen.de, 2025

Bearbeitungszeitraum: 01.09.2024 - 01.03.2025

Betreuer/Gutachter: Prof. Dr. Andreas Geiger, Universität Tübingen
Zweitgutachter: Prof. Dr. Matthias Hein, Universität Tübingen

Selbstständigkeitserklärung

Hiermit versichere ich, dass ich die vorliegende Masterarbeit selbständig und nur mit den angegebenen Hilfsmitteln angefertigt habe und dass alle Stellen, die dem Wortlaut oder dem Sinne nach anderen Werken entnommen sind, durch Angaben von Quellen als Entlehnung kenntlich gemacht worden sind. Diese Masterarbeit wurde in gleicher oder ähnlicher Form in keinem anderen Studiengang als Prüfungsleistung vorgelegt.



Melanie Schneider (Matrikelnummer 6038276), 28.Februar 2025

Abstract

As datasets grow, managing and processing their large data volumes has become a challenge in machine learning. Dataset distillation (DD) condenses large datasets into smaller, generated subsets without losing their utility, offering a solution to reduce storage and accelerated training, as well as a possibility to adhere to privacy constraints for data. Unlike traditional coreset methods that select representative samples, DD generates synthetic data tailored for efficient model training while maintaining performance. This thesis explores the new approach to distill datasets using generative adversarial networks (GANs), focusing on how they can capture and compress data distributions effectively. Our experiments on CIFAR-100 and TinyImageNet compare GAN-based approaches with classical techniques like Distribution and Training Trajectory Matching. The research demonstrates that GANs are a strong foundation for distillation without the need for specific distillation losses. Problems such as mode collapse and distribution fidelity are also addressed, providing insight into the advantages and limitations of generative distillation. By advancing our understanding of GANs for generative dataset distillation and their underlying mechanics, this research contributes to the refinement of distillation techniques that pave the way for more efficient and scalable data processing strategies in machine learning.

Acknowledgments

I am grateful to Prof. Dr. Andreas Geiger for his invaluable support and insightful feedback throughout this thesis. His guidance has been instrumental in shaping this research. I also extend my heartfelt thanks to Kashyap Chitta and Jovan Cicvaric for their mentorship, contributions, and the many stimulating discussions that enriched this work.

I give special thanks to my family, especially my parents, Claudia and Achim, who have been unwavering role models for me and my sister, Lena, who has always provided steadfast support. To my friends, thank you for your constant encouragement and the emotional support that helped carry me through my studies.

A special thanks goes to my partner Kai, who was by my side through all the lows and long nights, lending his patience and energy to keep me going as I worked on the code and made progress on this thesis. And, of course, to my beloved dogs, whose companionship and joy brought me moments of relief and happiness throughout this journey.

Thank you all for being part of this achievement.

Contents

1 Introduction

As deep neural networks grow in complexity, the amount of data required for training them to reach state-of-the-art performance also increases. In the realm of image classification, large labeled datasets like ImageNet [DDS+09] have propelled the field forward, enabling groundbreaking advances in computer vision. However, the larger these datasets grow, the more daunting the training task becomes - demanding immense memory, computational power, and energy, as shown in Fig. 1.1. Enter Dataset Distillation: a technique that promises to address these growing challenges by compressing large datasets into smaller synthetic samples without sacrificing performance. Imagine squeezing all the valuable knowledge from a massive dataset into a tiny, efficient package that allows us to train faster, use fewer resources, and still achieve remarkable results.

At its core, Dataset Distillation (DD) or Dataset Condensation is similar to the concept of knowledge distillation [GYMT20], where knowledge from a large, complex model (the teacher) is transferred to a simpler model (the student). The aim is to shrink the dataset, yet still preserve enough information that a model trained on this condensed version performs just as well as one trained on the full dataset. This means that we can retain the essence of the original data while drastically reducing the need for storage and computational power, which is an attractive prospect for resource-constrained environments, such as mobile devices, IoT and edge computing.

In the world of image classification, Dataset Distillation creates what is known as "meta-images" or "synthetic samples". These are not just some random or educated picks from the original dataset. Rather, they are generated images optimized to capture the most critical aspects of the dataset, enabling a neural network to generalize the entire distribution of the data. This results in significant reductions in storage and computational requirements.

But the benefits do not stop there. Dataset Distillation can also play a crucial role in privacy preservation. With growing concerns over the collection and sharing of sensitive data, distillation can act as a safeguard - converting private information into distilled representations that can be shared without risking privacy breaches. This has vast potential, particularly in industries like healthcare, where privacy regulations are strict but the need for data-driven insights is urgent [LTOH22].

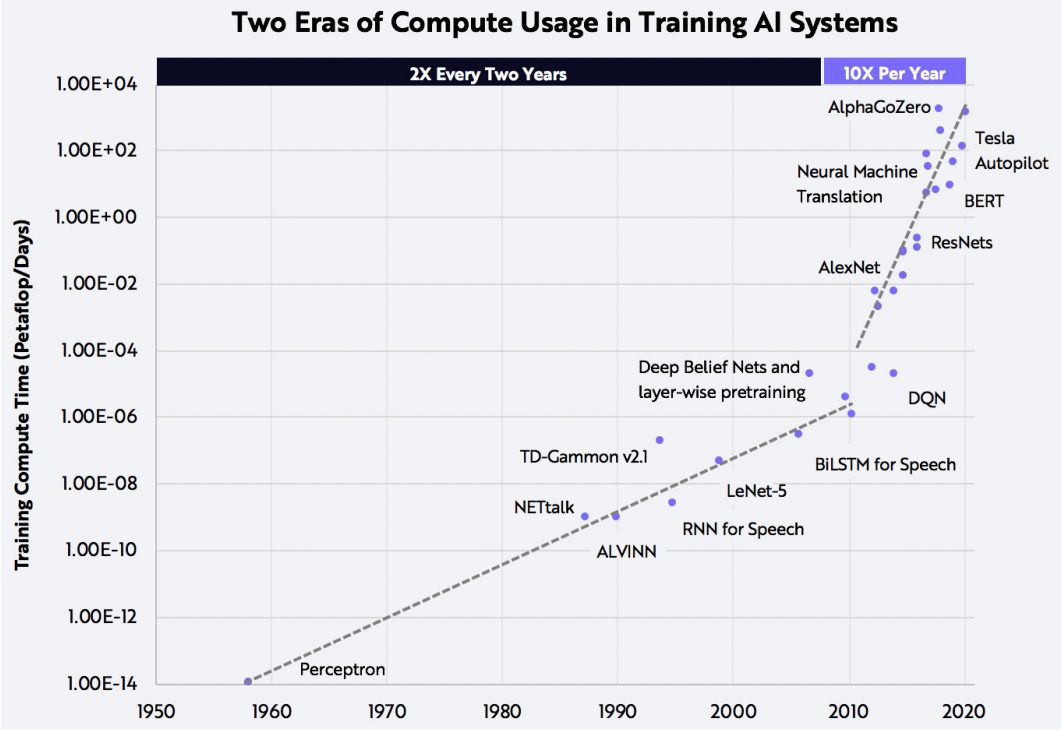


Figure 1.1: **Fast increasing computational cost of training in AI.** This graph by Geifman [Gei20] illustrates how the compute time for machine learning training historically followed Moore's Law, doubling approximately every two years. However, since around 2010, this growth rate has accelerated dramatically, with compute demands increasing nearly tenfold per year. This rapid increase underscores the urgent need for methods that enhance efficiency and reduce computational costs, like Dataset Distillation approaches.

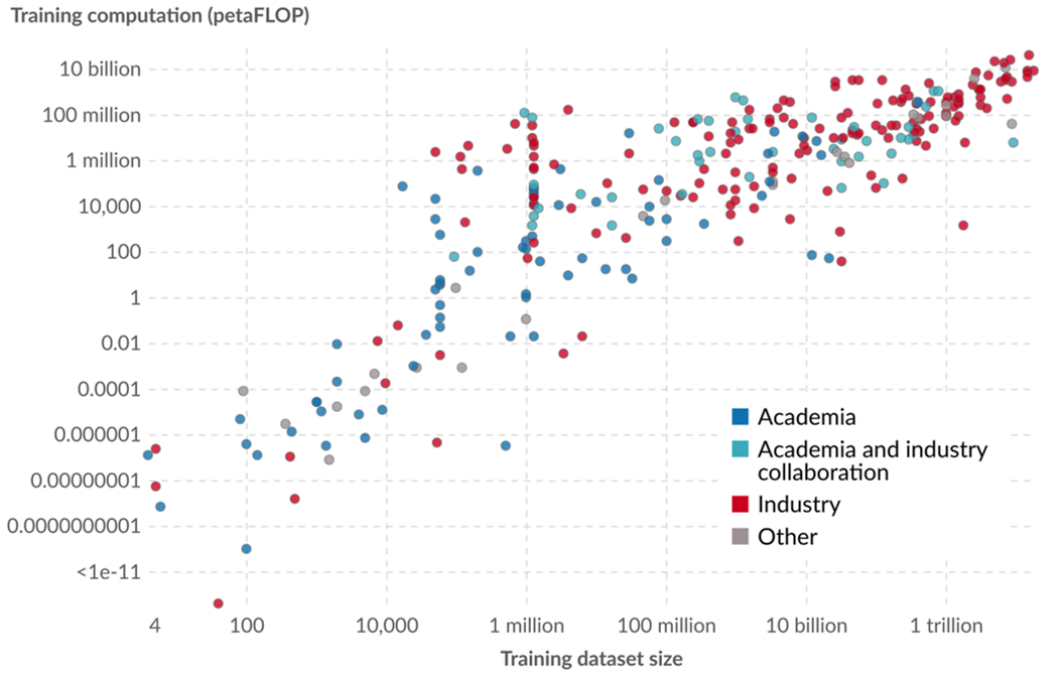


Figure 1.2: **Dataset Distillation and democratization.** The visualization from *Our World in Data* [Epo24] highlights the increasing computational demands for training large neural networks, predominantly driven by industry players. This trend underscores the need for Dataset Distillation to make machine learning development more accessible to academia and smaller organizations, fostering broader participation.

Perhaps one of the most exciting aspects of Dataset Distillation is its potential to democratize machine learning. Fig. 1.2 highlights the increasing computational demands for training large neural networks and illustrates how the current development is predominantly driven by industry players. By lowering the barriers to entry, researchers and organizations with limited access to large datasets and powerful computational resources can still achieve competitive model performance. A smaller, distilled dataset allows smaller-scale institutions to contribute to the field without being overwhelmed by the size of the data or the cost of training.

However, despite the promise, distillation is far from a perfect solution. One of the main challenges is to ensure that the distilled dataset still retains the full diversity of the original dataset's features, especially when it comes to complex, high-dimensional images. Moreover, finding the right balance between compression and representational accuracy is tricky — if the dataset is reduced too much, the distilled set may fail to capture the nuances needed for a robust model in downstream tasks, such as classification, segmentation, or object detection.

Furthermore, the distillation process itself can require substantial computational resources to generate high-quality synthetic samples, partially contradicting the goal of resource efficiency. As a result, the field is still evolving, with ongoing research focused on improving the scalability and efficiency of distillation techniques, especially for high-resolution datasets and real-time applications.

1.1 Problem Statement

Dataset Distillation aims to reduce the size of large datasets while retaining their critical information, allowing faster training and reduced computational requirements. As the field is new and rapidly developing, the comparability of different methods is difficult. Furthermore, the introduction of generative approaches adds complexity to the evaluation setup, as the image generation time must be taken into account. The concept of data set distillation lacks a precise and widely accepted definition, making it difficult to establish a clear evaluation framework. This challenge is further exacerbated by the limited insight into GANs and their effectiveness in solving the distillation task.

1.2 Motivation

Rapid expansion of data sets in machine learning has led to a dramatic increase in the computational resources required to train deep learning models. Datasets like ImageNet, which has more than 1 million images in 1,000 classes [DDS+09], have been widely used in training state-of-the-art models, but their size presents several challenges. As the size of the datasets continues to grow (Fig. 1.1) researchers face increasing difficulties in accessing, processing, and storing these datasets. The growth in the size of the data set also comes with a longer training time, which in turn results in significant resource usage and environmental costs, particularly in terms of carbon emissions from large-scale computing infrastructures [Hao19]. The authors of [Hao19] particularly mention the high environmental effects of the neural architecture search. Neural architecture search explores and evaluates different neural network architectures to find the most optimal design, often requiring extensive computational resources and numerous training iterations. Therefore, this process is one of the main deployments for distilled datasets.

These challenges highlight the need for methods like Dataset Distillation, which compress large datasets into smaller synthetic datasets that preserve key information and reduce the costs associated with training. The idea of using synthetic data for training has gained popularity, and while traditional Dataset Distillation methods have shown promising results, they often rely on fixed small datasets that risk overfitting in the downstream tasks when used with current models that have a large number of parameters. In parallel, the field of generative modeling, which

includes models such as StyleGAN-XL [SSG22] and DALL-E [RPG+21], has made substantial advancements in the creation of high-quality synthetic data.

This thesis aims to create more efficient distillation processes that can generate diverse, high-quality synthetic datasets by combining generative models with Dataset Distillation techniques. By investigating innovative distillation approaches, such as Generative Distribution Matching, this work aims to address these challenges and provide valuable insights into the effective combination of distillation and generative modeling for dataset compression and improved model performance.

1.3 Contributions

This work presents several key contributions to the field of Dataset Distillation, specifically leveraging generative models. The contributions can be summarized as follows:

- General Evaluation Framework: The *First Dataset Distillation Challenge* framework [Cha24] is adapted to compare methods that distill datasets into images versus those that distill datasets into models. This adaptation allows for a more standardized and fair evaluation of the various distillation techniques and enables a direct comparison between different methods.
- Evaluation of Popular Dataset Distillation Methods: Well-known Dataset Distillation methods, such as Training Trajectory Matching and Distribution Matching, are evaluated within the proposed evaluation framework. These methods are tested on two widely used datasets, CIFAR-100 and TinyImageNet, offering a comprehensive comparison of their performance.
- Evaluation of Generative Adversarial Networks (GANs) in the Distillation Setup: The performance of traditional GANs is assessed in the context of Dataset Distillation. The findings provide valuable insight into the limitations and strengths of GAN-based methods to distill effective synthetic datasets.
- Introduction of Generative Distribution Matching (GenDM): GenDM is introduced as a novel distillation method that integrates generative models with traditional Distribution Matching loss to generate synthetic datasets. Evaluated alongside other methods, the results indicate that GenDM does not outperform baselines, suggesting that the combination does not yield the intended improvements.
- Studies on the Influence of Resolution and GAN Truncation: Experiments are conducted to study the impact of image resolution and GAN truncation strategies on Dataset Distillation performance in the evaluation framework. The results provide a deeper understanding of how these factors influence the quality and diversity of the distilled datasets, offering guidance on optimizing distillation methods for different use cases.

Through these contributions, this work advances the understanding of Dataset Distillation, particularly in the context of using generative models for DD. It establishes a comprehensive evaluation framework to compare and improve distillation techniques.

2 Related Work

Dataset Distillation is a relatively new and evolving field within machine learning. There is a growing interest in distillation methods that can scale to handle increasingly large and complex datasets. As there is a need to develop methods that can efficiently distill and compress real world data to enable faster training times and reduce computational costs. The central challenge in this field is what we refer to as the "Dataset Distillation problem"—the task of reducing a large dataset into a smaller, synthetic set of data that retains the essential information necessary for training machine learning models with comparable performance to the original dataset [YLW23].

Dataset Distillation aims to produce a "summary" of the original dataset, capturing its underlying distribution and key characteristics. As datasets continue to grow, training models on full datasets becomes increasingly impractical, both in terms of computational cost and time. This challenge is particularly evident in scenarios that require multiple training runs, such as a neural architecture search. By condensing large datasets into smaller information-rich subsets, Dataset Distillation enables more efficient model training. In this section, existing approaches to Dataset Distillation are reviewed, starting with coreset selection methods which are early attempts at dataset reduction that are not always explicitly classified within the distillation paradigm. Then this section explores classical distillation methods which are summarized as Distillation to Images (DiI). These methods laid the foundation for the field of Dataset Distillation. Finally, more recent methods are discussed that leverage image generation models for the distillation process in the area of Distillation to Models (DiM).

2.1 Coreset Selection

Before the first idea of Dataset Distillation was introduced in 2018 [WZTE18], coreset selection methods were used to reduce the dataset to a smaller sample. These methods are therefore the very basis for all later Dataset Distillation methods. The Dataset Distillation methods first had to show that they can improve the results of the coreset selection methods [CWT+22], [ZMB21], [ZB22].

The main difference between coreset selection and distillation methods is that coreset selection only selects samples from the original full dataset $\mathcal{T}$, making the selected

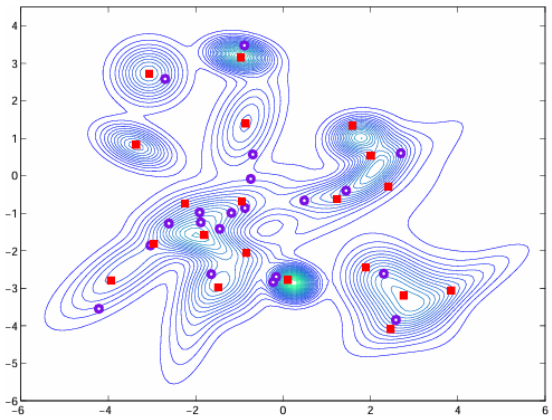


Figure 2.1: **Selection of samples from a density function using coreset methods.** This figure by [CWS12] depicts the first 20 samples from the coreset selection method of herding (red squares) versus a random sampling (purple circles) approach. The contour lines show the underlying probability density function that is sampled.

coreset $\mathcal{C}$ a true subset of the original data $\mathcal{C} \subset \mathcal{T}$. This is also the main limitation of these methods as they do not compress information but instead try to keep only the most informative samples, which limits the upper bound performance in such selected coresets $\mathcal{C}$. The information contained in the images that are not selected is "lost" and cannot be used by later downstream training tasks. Coreset selection "discards a large fraction of training samples, dismissing their contribution to training results and often resulting in sub-optimal performance" [YLW23].
These are some of the popular coreset methods [GZB22]:

- **Random:** This is the most basic approach to reducing the size of a dataset. Selecting a set of samples randomly, without applying any specific criteria or systematic methodology to guide the selection process. Due to it's simplicity this method comes at no extra cost. In a low images per class (IPC) number this method performs very poorly. IPC refers to the number of images available for each class in a dataset. But with increasing IPC, this method can perform on par with many of the more sophisticated coreset methods as shown by [ZB22] and [GZB22]. Our experiments in Section 4.3.1 show that in higher IPC regimes this approach can serve as a competitive baseline, potentially rivaling Dataset Distillation methods.
- **Herding:** The coreset selection method herding [CWS12], [Wel09] is a deterministic process that approximates a probability density function with a collection of samples. It does so by greedily adding samples to the core set so that the mean vector is approaching the mean of the whole dataset. This process is

illustrated in Fig. 2.1, where the herding method (red squares) is compared to random sampling (purple circles). In the figure, one can see how herding is able to correctly sample all modes of the distribution, capturing the key characteristics of the data, while random sampling fails to do so effectively.

- **Forgetting:** The forgetting method [TSdC+18] is inspired by catastrophic forgetting, a phenomenon in which neural networks lose previously learned information when trained on new data. Analysis reveals that some examples are forgotten more frequently than others, suggesting their reduced importance for model retention. Using this frequency as a measurement, a significant fraction of the least forgotten samples from the dataset can be omitted for the coreset. The omitted samples are not harming the downstream task performance, as they are "easier" to learn for the networks compared to the kept samples with higher forgetting frequency.

2.2 Distillation to Images - DiI

The existing distillation approaches based on the initial work of [WZTE18] can be broadly divided into two categories: *Distillation to Images (DiI)* and *Distillation to Models (DiM)*. Although both categories aim to achieve the same goal, distilling large datasets into more compact representations, the outputs differ significantly. In DiI, the final result is a new dataset $\mathcal{S}$, consisting of newly generated synthetic samples. These synthetic samples are then fixed and used for downstream tasks instead of the full dataset. In contrast, DiM methods focus on distilling into a model rather than a dataset, which offers several unique advantages, discussed in Section 2.4.

The core idea of DiI is to generate a synthetic dataset that mimics the distribution of the original dataset, thereby enabling the training of machine learning models with far fewer samples. This is basically the same idea as in coreset selection, but the DiI methods do not select images, they create new images that reflect the distribution. To achieve this, DiI methods typically use various matching strategies during the optimization of synthetic images to ensure alignment between the synthetic and original datasets [WGZ+23].

2.2.1 Matching Strategies

In DiI methods, matching strategies are employed to align the synthetic images with the original dataset, ensuring that the distilled set retains key characteristics. These strategies can be categorized into three areas:

- **Gradient Matching:** This approach aims to match the gradients of a neural network computed on synthetic samples with those of the original dataset. Key works in this area include Dataset Condensation with Gradient Matching (DC) [ZMB21], Differential Siamese Augmentation (DSA) [ZB21], and Dataset

Condensation via Efficient Synthetic-Data Parameterization (IDC) [KKO+22], which focus on gradient matching between synthetic and original data.

- **Training Trajectory Matching:** This category includes methods that align training trajectories (path of gradients) of models trained on synthetic and original datasets. The method of Matching Training Trajectories (MTT) [CWT+22] belongs to this category, as it optimizes loss between training trajectories to improve the effectiveness of synthetic data.
- **Feature/Distribution Matching:** This approach seeks to match the feature distributions between the synthetic and original data. Methods such as Dataset Condensation with Distribution Matching (DM) [ZB22] and Learning to Condense Dataset by Aligning Features (CAFE) [WZP+22] leverage feature matching to reduce bias introduced by large-gradient samples. These methods focus on the resulting distributions rather than on the network training process.

2.2.2 Kernel-Based Methods in Distillation

Kernel-based techniques are closely related to matching strategies in Dataset Distillation, as they aim to efficiently align synthetic data with the original data. These methods leverage kernel functions to map data into higher-dimensional spaces, ensuring key information is preserved while enabling effective classification [ZNB22],[NNXL21]. Their computational efficiency is derived from closed-form solutions for classifiers trained on synthetic data. Some approaches, such as Dataset Distillation using Neural Feature Regression [ZNB22], improve distillation employing truncated backpropagation through time to mitigate challenges associated with bi-level optimization. However, despite their advantages, kernel methods often overfit to specific neural network architectures and face difficulties scaling to larger datasets.

2.2.3 Strategies for Representations

Various strategies have been proposed to enhance the representation of distilled datasets. Dataset Distillation via Factorization (HaBa) [LWY+22] introduces a framework that decomposes the dataset into bases and hallucinators, preserving the diversity in the synthetic samples generated. Meanwhile, Distilling Datasets into Addressable Memories for Neural Networks (MAP) [DR22] employs bases along with addressing matrices to efficiently store and retrieve information from the dataset. These methods collectively aim to maintain the quality and diversity of synthetic data while improving computational efficiency.

2.3 Image Generation

Image Generation is a dynamic field in machine learning, driven by advances in deep neural networks and the demand for high-quality synthetic content in language and visuals. Former classical methods such as restricted Boltzmann machines (RBMs)[HHJ20], and Markov random fields (MRFs)[MR05] have been replaced by the advances in deep learning-based approaches. These leverage neural networks to learn the complex relationships between pixels and high-level image representations [EKV23], [CHW+22], [XGZ+21].

Image generation refers to the process of generating synthetic images that appear to be drawn from the same distribution as a set of real images. It can take several forms, depending on the input type and the generation strategy used. For example, it can involve generating random images from scratch or transforming one image into another [EEAmT22]. Conditional image generation is another key area of research, where models are trained to generate images based on additional input conditions, such as class labels. There are various image generation tasks, such as image denoising [EKV23], inpainting [YLY+18], and super-resolution [CHW+22], as well as the task of generating images from textual input. Recent advances in models such as DALL-E [RPG+21] have demonstrated the ability to generate high-quality images directly from descriptive text.

In particular, two classes of models have emerged as leaders in the field of Image Generation: Generative Adversarial Networks (GANs) and diffusion models [DAS+24], [Pen24]. Each of these approaches has introduced novel mechanisms for generating high-quality images while also presenting unique challenges and solutions.

2.3.1 Generative Adversarial Networks

In 2014 Goodfellow et al. introduced GANs to generative modeling, offering a fresh perspective on how machine learning can produce images with remarkable realism [GPAM+14]. GANs operate by framing the generation task as a game between two neural networks—the generator and the discriminator. The generator learns to create synthetic images, while the discriminator evaluates the authenticity of those images against real data. Over time, this process forces the generator to further and further improve its output, pushing the boundaries of what can be synthesized.

Early advances, such as Deep Convolutional GANs (DCGANs) [Rad15], incorporated convolutional layers to improve image resolution and stabilize training. With the addition of class labels in Conditional GANs (cGANs) [MO14], it enables controlled generation based on additional inputs, such as labels, which can be applied to image-to-image translation tasks like in [IZZE16].

The GAN training process is prone to instability and mode collapse, methods like Wasserstein GAN (WGAN) [ACB17] or the WGAN-GP [GAA+17] introduce new distance measures and gradient penalties to improve training stability and interpretation. To produce higher-fidelity images [KALL17] introduced progressive growing. In this process the image resolution is gradually increased during training, which helps to produce images at higher resolutions while still keeping a good quality. BigGAN [BDS18] scales GANs to handle high-resolution, diverse datasets like ImageNet, enabling the generation of realistic, detailed images at large scales. BigGAN introduced the "Truncation Trick", which reduces the range of latent values during training to improve the quality and variety of generated images. This idea is presented in more detail in Section 3.2.3.

StyleGAN, first introduced by Karras et al. [KLA18], revolutionized GAN-based image synthesis by incorporating a style-based generator architecture that enables fine-grained control over image attributes. StyleGAN2 [KLA+19] improves this by addressing artifacts and improving the overall fidelity of the image. In StyleGAN3 [ZCZG24] the aliasing artifacts are mitigated and the consistency of the generated images improves. The latest iteration, StyleGAN-XL [SSG22], scales the architecture for high-resolution, diverse dataset synthesis, achieving state-of-the-art performance in photorealistic Image Generation.

2.3.2 Diffusion Models

Diffusion models have emerged as a powerful alternative to GANs. These models are grounded in probabilistic principles and work by stepwise introducing noise into an image and then learning to reverse this process, gradually denoising the image to recover the original data.

Early works, such as Score-Based Generative Models [SSK+20], demonstrated the potential of diffusion processes for generative tasks, framing them as denoising score matching. Denoising Diffusion Probabilistic Models (DDPMs) [HJA20] further refine this by modeling data distributions through iterative denoising. This achieves impressive results in high-quality Image Generation.

The use of diffusion models in conditional generation tasks has been explored in Guided Diffusion Models [DN21], which combine diffusion processes with conditional inputs to produce specific image types, expanding the flexibility and utility of diffusion models in various applications. State-of-the-art architectures such as Imagen [SCS+22] and stable diffusion models such as SDXL [PEL+23] leverage text conditioning to generate high-quality images from descriptions in natural language. ControlNet [ZRA23] extends conditional diffusion by integrating structural guidance, such as depth maps and edge detection, to refine the output with user-specified constraints. These advances have significantly improved both the controllability and realism of diffusion-based generative models, enabling diverse applications in art, design, and real-world simulation.

2.4 Distillation to Models - DiM

Distillation to Models initially referred to Knowledge Distillation (KD) which focused on efficiently transferring the knowledge of a large model to a smaller one. Over time, the concept expanded to encompass Dataset Distillation, where the emphasis shifted from simplifying models to condensing the dataset itself, ultimately enhancing learning efficiency.

The foundation of this area is KD [HVD15], which proposed transferring knowledge from a large teacher model to a smaller student model using the soft predictions of the teacher as a target for training the student. This demonstrated that a student model could mimic the behavior of the teacher model while reducing size and complexity. Further research in this area has produced a multitude of distillation approaches that not only match behavior in terms of the output of the teacher model but are also optimized to match "features" such as the activations of intermediate layers of the teacher model [GYMT20].

In the domain of distillation of datasets themselves, the first approach showing that a distillation into a model is feasible is "Distilling Dataset into Generative Models" [WGZ+23]. The authors propose a novel approach where a generative model is distilled from an entire dataset. Unlike knowledge distillation, where the focus is on the model's output or intermediate layers, DiM distills the "essence" of a dataset into a generative model, enabling the model to approximate the distribution of the entire dataset. This technique merges the strengths of both Dataset Distillation and generative modeling, allowing for efficient data representation while leveraging the power of generative models.

More recent advances in Dataset Distillation into models have explored techniques such as GenRep [JPTI21], which aims to compress large-scale datasets into compact generative representations while preserving the diversity and structure of the original data. Methods like GLEAN (Generative Latent Bank) [CWX+20] utilize pre-trained generative models like StyleGAN and their latent codes for the super-resolution task in Image Generation. The idea of using latent codes instead of storing full images is also part of GLaD (Generalizing Dataset Distillation via Deep Generative Prior) [CWT+23]. In GLaD, the Dataset Distillation process is performed in the latent space of a pre-trained generative model. This enables the distilled dataset to capture richer semantic representations with significantly fewer samples while maintaining generalization across different architectures. GLaD learns compact yet expressive latent codes that effectively parameterize synthetic datasets, reducing storage and computation costs while preserving essential data characteristics.

Some works have also investigated approaches using diffusion models as generative models, like in MinMax [GVK+24] where generative diffusion techniques are integrated with a minimax criteria to improve the representativeness and diversity of distilled datasets.

3 Methods

In this section, we first cover Dataset Distillation techniques and then go into the topic of generative models and their related issues. Finally, we look at the Generative Distribution Matching approach that combines both fields.

Dataset Distillation aims to reduce the size of a large dataset while preserving its utility. The first methods used the original images and tried to find the most "important" ones. However, these approaches are limited by the images themselves, so new methods have been developed that are able not only to select images, but also to learn entirely new images that make up the smaller - distilled - dataset. In this new context, this smaller dataset is called the synthetic dataset $\mathcal{S}$. The goal of Dataset Distillation is to generate a synthetic dataset $\mathcal{S} = |(s_i, y_i)|_{i=1}^{|\mathcal{S}|}$ that is significantly smaller than the original dataset $\mathcal{T} = |(x_i, y_i)|_{i=1}^{|\mathcal{T}|}$. Where the generated images s and images $x \in \mathcal{X} \subset \mathbb{R}^d$, and the labels $y \in \{0, ..., C-1\}$ for the number of C classes.

$$\mathcal{S} << \mathcal{T} \tag{3.1}$$

The goal is for a neural network model f trained on $\mathcal{S}$ to achieve a performance comparable to that of a model trained on $\mathcal{T}$. Formally, this can be expressed as:

$$\mathbb{E}_{(x,y)\sim\mathcal{P}_\mathcal{D}}\Big[\mathcal{L}(f_\mathcal{S}(x), y)\Big] \approx \mathbb{E}_{(x,y)\sim\mathcal{P}_\mathcal{D}}\Big[\mathcal{L}(f_\mathcal{T}(x), y)\Big] \tag{3.2}$$

where expectation $\mathbb{E}_{(x,y)\sim\mathcal{P}_\mathcal{D}}$ denotes the average over all possible samples (x, y) drawn from the true data distribution of $\mathcal{P}_\mathcal{D}$. Here, $f_\mathcal{S}$ and $f_\mathcal{T}$ are models trained on $\mathcal{S}$ and $\mathcal{T}$, respectively, and $\mathcal{L}(\cdot, \cdot)$ is a loss function such as cross-entropy.

3.1 Dataset Distillation

The original Dataset Distillation paper [WZTE18] by Wang et al. proposes to solve the distillation objective as formulated in Equation (3.3) as a bi-level optimization problem. The inner objective is to update the model parameters θ on the synthetic dataset, while the outer objective is to refine the distilled dataset $\mathcal{S}$. To optimize the outer parameters one needs to compute this loss as

$$\min_\mathcal{S} \mathcal{L}_o(\theta^*, \mathcal{S}) \text{ s.t. } \theta^* \in \min_\theta \mathcal{L}_i(\theta, \mathcal{S}) \tag{3.3}$$

Where $\mathcal{L}$ are the respective loss functions used for optimization.

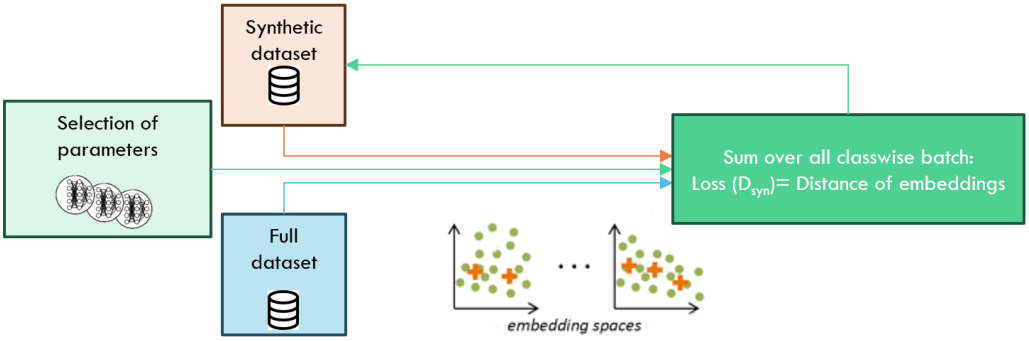


Figure 3.1: **Overview of the Distribution Matching (DM) algorithm.** Inputs are the synthetic and full dataset as well as a set of random initialized neural network parameters. The distillation loss is computed as the difference between the embedding spaces of images from the synthetic dataset and the full dataset, where the embeddings are generated by randomly initialized neural networks for the images.

Bi-level optimization is highly computationally demanding and often infeasible due to the necessity of repeated inner-loop optimizations, rendering it impractical for large-scale datasets and deep models. To address this challenge, recent methods have introduced surrogate objectives for the distillation process, which we discuss in Sections 3.1.1 (Distribution Matching) and 3.1.2 (Matching Training Trajectories).

As a starting point, most distillation methods initialize the smaller synthetic dataset $\mathcal{S}$ with certain datapoints (s_i, y_i). The datapoints contain the images s_i and their respective labels y_i. There exist methods for the so-called soft labels that also distill the labels y_i themselves, but the focus is on methods that keep the standard labeling approach of one single label per sample.

To initialize the synthetic dataset images, one can produce random noise images without prior information about the dataset. Alternatively, the images can be taken from the original dataset. Usually, this initialization is done with randomly selected images from the dataset, but there are also some approaches that select the initial images by a coreset selection algorithm, which we saw in Section 2.1.

3.1.1 Distribution Matching

To generate the synthetic dataset, the surrogate objective in Distribution Matching (DM) proposed by [ZB22] is to compare the resulting embedding spaces between synthetic and full dataset as depicted in Fig. 3.1. The embedding spaces are generated by neural networks. The authors are able to show that these networks do not need to be pre-trained because "a random initialization strategy produces better or

comparable results with the more expensive strategy of using pre-trained networks" [ZB22].

From the difference of the embeddings generated by these random neural networks f_θ parametrized by θ, the distillation loss of DM is then calculated by the maximum mean discrepancy (MMD) [GBR+12]. The MMD is a measurement for the distance between two distributions p and q calculated as the distance between the mean embeddings of features.

$$MMD[\mathcal{F}, p, q] := \sup_{f \in \mathcal{F}} (\mathbb{E}_{x \sim p}[f(x)] - \mathbb{E}_{y \sim q}[f(y)]). \tag{3.4}$$

Where x, y are random variables in the topological space $\mathcal{X}$ and $\mathcal{F}$ is a class of functions $f : \mathcal{X} \longrightarrow \mathbb{R}$. Since this assumes access to the full data distributions, the empirical estimate of MMD is used in the DM loss.

$$\mathcal{L}_{DM} = \sum_{c=0}^{C-1} \| \frac{1}{|B_C^{\mathcal{T}}|} \sum_{(x,y) \in B_C^T} f_\theta(x) - \frac{1}{|B_C^S|} \sum_{(s,y) \in B_C^S} f_\theta(s) \|^2 \tag{3.5}$$

Where $B_C^{\mathcal{T}}$ are the respective mini batches of either images from the real data $\mathcal{T}$ or the synthetic set $\mathcal{S}$ per class C.

The Differentiable Siamese Augmentation (DSA) method $A(\cdot, \omega)$ can be applied to each mini batch to further improve the results of DM. DSA [ZB21] applies the same augmentation strategy to real and synthetic images in a batch. These augmentations typically include rotations, flipping or cropping. By using the same augmentation the loss is not affected, but the learned synthetic datasets improve in their diversity. So DSA increases the data efficiency and thus generalization of the distilled dataset. The augmentations themselves are differentiable so that the synthetic data can be optimized via gradient descent. DSA is used not only in DM, but has also proven successful in other distillation methods and is now used in multiple approaches for distillation [YLW23].

DM is a method that does not depend on bi-level optimizations on network parameters θ and the synthetic data $\mathcal{S}$. However, since this method relies solely on the per-class distribution of images, it is limited to image classification tasks with predefined labels and cannot be applied to other types of image-based tasks.

3.1.2 Matching Training Trajectories

Matching Training Trajectories (MTT) proposed by [CWT+22] is based on the idea of matching the training processes of the networks itself unlike DM which compares the resulting products (images) to each other for generating a distillation loss. The basic idea is that the distilled dataset should support the training of a network in such a way that the parameters of the network are similar. In other words, if we

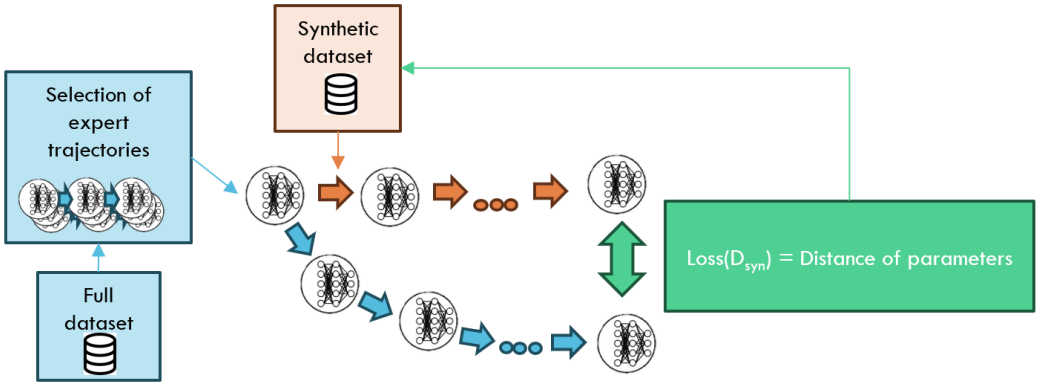


Figure 3.2: **Overview of the matching training trajectories (MTT) algorithm.** The algorithm is based on a set of pretrained expert trajectories using both the full dataset and a synthetic dataset. The distillation loss is defined as the difference between the parameters of the expert model and those of a student model trained on the distilled data for a number of training steps. Notably, the student model undergoes fewer training steps than the expert model.

use the same network to train on the full dataset and on the distilled dataset, the network parameters should be as close to each other as possible.

This idea of matching gradients was already proposed in [ZMB21]. Here, the gradient matching (GM) algorithm is introduced. For gradient matching, the same network f (same network parameters θ) is given a batch of real data or synthetic data. For each real and synthetic batch, the respective loss is computed as: $\mathcal{L}^{\mathcal{T}}(\theta)$ and $\mathcal{L}^{\mathcal{S}}(\theta)$. Then the synthetic set is updated according to the GM loss, which is defined as the distance between the gradients of both parts.

$$D(\nabla_\theta \mathcal{L}^{\mathcal{T}}(\theta), \nabla_\theta \mathcal{L}^{\mathcal{S}}(\theta)) \tag{3.6}$$

As f_θ is a multilayered neural network, the gradients correspond to a set of 2D (out $\times$ in) and 4D (out $\times$ in $\times$ h $\times$ w) weights for each fully connected and convolutional layer. For the distance between these parameters, there are two options to compute this, either by flattening all tensors and then comparing them or by realizing the distance as a layer-wise sum of distances.

$$D(\nabla_\theta \mathcal{L}^{\mathcal{S}}, \nabla_\theta \mathcal{L}^{\mathcal{T}}) = \sum_{l=1}^{L} d(\nabla_{\theta^{(l)}} \mathcal{L}^{\mathcal{S}}, \nabla_{\theta^{(l)}} \mathcal{L}^{\mathcal{T}}) \tag{3.7}$$

where l is the layer index and L is the number of layers with weights.

But instead of focusing on matching single points as in GM, the method MTT is training the synthetic data such that the network parameters match multiple steps of the training on the original dataset. The problem of matching single points, as GM does, is the accumulation of errors through this process.

The matching of multiple gradient update steps in longer trajectories avoids this error accumulation and helps the synthetic dataset to support a full trajectory of the network. To achieve this, MTT generates so-called "expert trajectories" from the full dataset. To train the synthetic set, a certain trajectory is used as a reference, and the network is initialized with the starting parameters of the expert trajectory. Then a certain amount of updates is done using the synthetic dataset, this is shown in the orange arrows on the upper path in Fig. 3.2. After this, the loss is computed as the difference of the resulting parameters of the network trained on the full dataset (blue arrows) versus the network parameters trained on the synthetic data set using the distance of the network parameters as in (3.7). Since we now use trajectories instead of single gradients, the loss is normalized by the distance of the trajectory of the expert.

3.2 Generative Adversarial Networks

Generative Adversarial Networks (GANs) can be used for conditional image generation tasks. Given the label of a certain class, they can produce any number of artificial images that are part of the learned distribution of the original dataset. The experiments in Section 4.4.1 show how these methods that are developed for generative tasks and not specifically for the distillation task are already a good baseline. In this work, we focus on one GAN model, called StyleGAN-XL. Developed by [SSG22], this GAN is better suited to scale to larger data sets than the predecessors of the same family.

The following sub-chapter first provides a brief history of the StyleGAN family of models. Next, StyleGAN-XL is specifically discussed, as it was introduced to scale the generation of realistic images from a given dataset to larger datasets with more classes and higher resolutions, such as the ImageNet dataset [DDS+09]. Finally, Section 3.2.3 covers GAN truncation, a technique used to control the quality and variety of generated images by limiting the range of latent space during generation. This method addresses challenges such as mode collapse, where the generator produces repetitive or limited outputs. The mechanics and impact of truncation on GAN performance are explored in detail.

3.2.1 StyleGAN family

This family of generative models advances image generation by using style transfer techniques. The family began with StyleGAN, followed by StyleGAN2, StyleGAN3

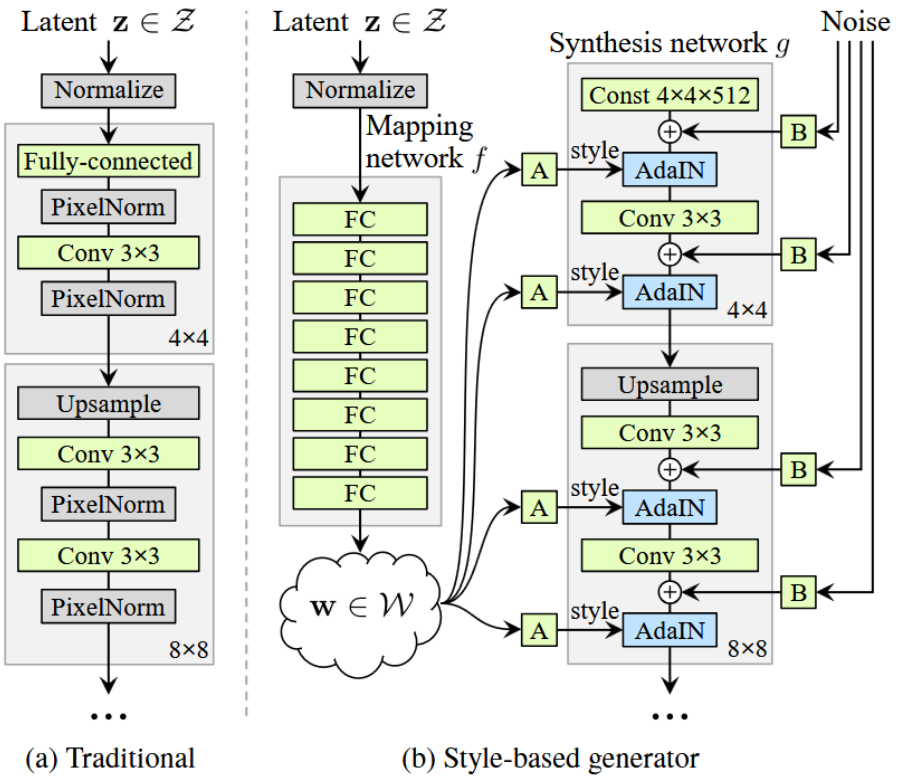


Figure 3.3: **Comparison of StyleGAN and Traditional GAN Architectures.** This figure illustrates the differences between StyleGAN (b) and a conventional GAN (a) in image generation. StyleGAN introduces style-based synthesis and adaptive instance normalization, leading to improved control over image attributes, better disentanglement, and higher-quality image generation compared to traditional GAN architectures [KLA18].

and later the StyleGAN-XL model. As we explore in the following chapters, the improvements from one version to the next not only address previous limitations but also introduce new capabilities that push the boundaries of innovation.

StyleGAN

In 2019 StyleGAN [KLA18] introduced a novel generator architecture inspired by style transfer techniques. Rather than feeding the random latent vector $\mathbf{z}$ directly into the generator as depicted in Fig. 3.3 (a), StyleGAN uses a mapping network f that produces an intermediate latent representation $\mathbf{w}$ given an input latent $\mathbf{z}$. This representation is then transformed through affine transformations into style vectors A that control the Adaptive Instance Normalization (AdaIN) layers in the synthesis network g. To generate the images, Gaussian noise is added to each

convolution layer to enhance the generator performance as shown in Fig. 3.3 (b). These modifications allow StyleGAN to automatically separate high-level attributes and enable intuitive control of image synthesis [KLA18]. As a result, StyleGAN became the state-of-the-art for image generation tasks at the time.

StyleGAN2

StyleGAN2 [KLA+19] aimed to improve StyleGAN by addressing two key issues: blob-like artifacts and problems with progressive growing where so-called "phase" artifacts appear. Phase artifacts are a tendency for the network to generate high-frequency details, like teeth or eyes, that then get "stuck" in certain locations instead of moving smoothly with the image.

The blob-like artifacts trace to the AdaIN normalizations, which are refining the mean and variance across feature maps separately, leading to loss of feature correlation. The solution involves redesigning the normalization to work only on standard deviations and eliminating unnecessary style block operations. For the phase artifacts, the authors trace the issue to each resolution in the training process being temporarily treated as the final output, forcing the network to generate very detailed features too early. The exploration of alternative architectures such as MSG-GAN [KWI19] with residual connections improved artifacts, further advancing StyleGAN2's performance.

StyleGAN3

StyleGAN3 [ZCZG24] focuses on solving the issue of "texture sticking", which is prevalent in previous versions. Texture sticking in StyleGAN occurs when fine details, such as hair or fur, remain fixed to specific pixel coordinates instead of moving naturally with the underlying object. This results in unnatural artifacts during transformations, breaking the illusion of a coherent 3D structure and making interpolations appear unrealistic. To avoid this issue, the synthesis network is redesigned to use Fourier features to define the spatial map for the generated image. This makes the network equivariant to transformations like translation, and there is also a variant of StyleGAN3 that is rotation equivariant.

3.2.2 StyleGAN-XL

The latest iteration, called StyleGAN-XL [SSG22], introduces significant improvements, addressing issues such as poor performance on large unstructured datasets (e.g. ImageNet) and high computational cost at large resolutions. Before the alterations, it was unclear whether StyleGAN is able to generate ImageNet-like data.

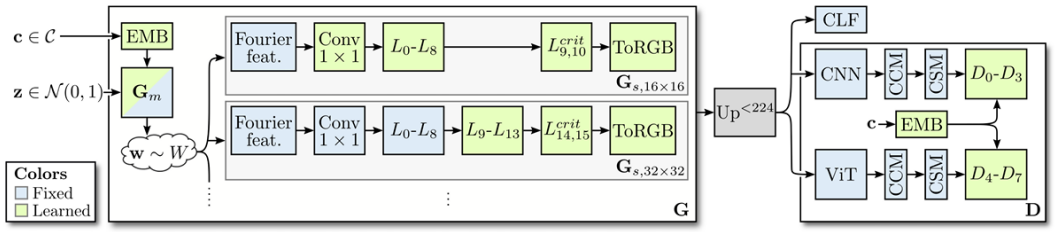


Figure 3.4: **Structure of StyleGAN-XL.** This figure by [SSG22] illustrates the architecture of StyleGAN-XL, highlighting its updated discriminator and the reintroduction of resolution growing in the generator. These enhancements improve image quality, training stability, and scalability for high-resolution image synthesis.

StyleGAN-XL uses the changes in the generator structure proposed in StyleGAN3, such as the Fourier features. But it adds ideas from ProjectedGAN [SCMG21] that alter the structure of the discriminator. In StyleGAN-XL, multiple discriminators operate on feature projections. These projections are generated by fixed Vision Transformers (ViTs) and Convolutional Neural Networks (CNNs) combined with feature mixing blocks (the CCM and CSM blocks in Fig. 3.4), which aim to prohibit discriminators from focusing on only a subset of their input feature space. The addition of feature projections leads to a new ProjectedGAN objective:

$$\min_{G} \max_{\{D_l\}} \sum_{l \in \mathcal{L}} \left(\mathbb{E}_x \left[\log D_l(P_l(x)) \right] + \mathbb{E}_z \left[\log \left(1 - D_l(P_l(G(z))) \right) \right] \right) \tag{3.8}$$

with the multiple discriminators D_l and the new feature projections P_l. In addition, a modified progressive growing strategy is introduced. After the progressive growing was eliminated in StyleGAN2, the authors of StyleGAN-XL revisited that idea to be able to scale the model to higher resolutions like the ImageNet dataset. The new progressive growing strategy starts from a low resolution and gradually increases the number of layers while freezing earlier layers to prevent mode collapse. As depicted in Fig. 3.4 the growing starts with a low resolution such as 16×16 pixels where the first eleven layers of the generator are trained. To scale up to the next resolution of 32×32 pixels, the last two layers are dropped and the first eight are frozen, while seven new layers are added. For every new step, this process is repeated with dropping the last two layers and adding seven new layers for the training of the next resolution stage, while keeping the before trained layers frozen. This process allowed StyleGAN-XL to achieve new state-of-the-art results in large-scale image synthesis, generating high-quality images at resolutions as high as 1024×1024 pixels.

3.2.3 GAN Truncation

GAN truncation is a technique used to control the trade-off between sample diversity and quality in generative adversarial networks. It operates by modifying the latent space sampling process, typically by scaling or restricting the range of input vectors. The most common approach, known as the truncation trick [BDS18], adjusts the latent vectors to be closer to the mean of the distribution, reducing variability while increasing the visual quality of the generated images.

In a standard GAN, latent vectors $\mathbf{z}$ are typically sampled from a Gaussian distribution $\mathcal{N}(0, I)$, where I is the identity matrix. Some regions of this distribution correspond to very high-quality outputs, while others may produce lower-quality or unrealistic images. The truncation trick restricts the sampled latent vectors to a smaller, more favorable region of the distribution. Mathematically, this is done by scaling or clipping the latent vectors:

$$\mathbf{z}' = \mu + \psi(\mathbf{z} - \mu) \tag{3.9}$$

with the mean of the latent space distribution μ and the added truncation parameter ψ, which controls how much samples are pulled towards the mean.

A truncation value of $\psi = 1$ retains the full variance, allowing for diverse outputs, while lower values constrain the latent vectors, producing more consistent but potentially less varied results. This technique is particularly useful in high-resolution GANs like StyleGAN, where extreme latent values may generate unrealistic artifacts. Furthermore, in the context of Dataset Distillation, maintaining a balance between sample quality and diversity is crucial. Since the goal is to create a compact yet representative dataset, the truncation trick aligns well with this objective by directly influencing the representativeness and diversity of a generated dataset, as demonstrated by the experimental results described in Section 4.5.

3.3 Mode Collapse

Mode collapse is a common challenge in GANs [KAD22] [Zha21] [LTZQ19], where the generator produces limited diversity, often resulting in similar or identical outputs. This output similarity is visualized for the CIFAR10 dataset in Fig. 3.5. Diversity is crucial for creating a high-quality synthetic dataset, as a lack of variation limits the generalization ability of models trained on it. Although there are methods to mitigate mode collapse, such as those employed in StyleGAN (discussed in Section 3.3.2)—it cannot be entirely eliminated. As a result, training GANs remains a challenging task that requires careful tuning to balance diversity and sample quality.

The mode collapse occurs when the generator prioritizes generating a narrow range of data patterns that successfully fool the discriminator. As a result, it focuses on a few dominant modes in the training data, neglecting the full diversity of the data

Figure 3.5: **Mode Collapse in a StyleGAN-XL Model trained on CIFAR-10.** This figure demonstrates a fully mode-collapsed StyleGAN-XL model trained on the CIFAR-10 dataset [KNHa] at a resolution of 8×8 pixels. Each row contains 10 images generated for the same class label. Due to mode collapse, the conditional GAN fails to capture the full diversity of each class, generating only a single type of image per class instead of varied representations.

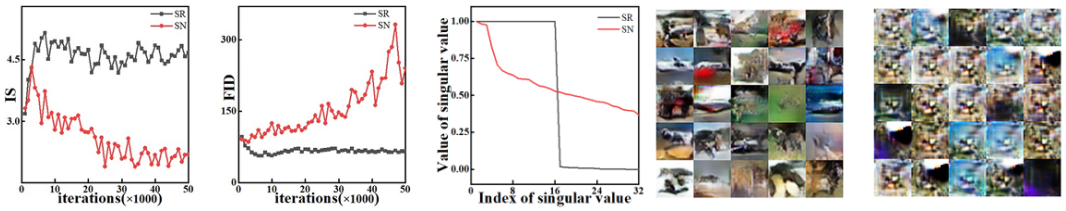
distribution. This phenomenon arises from the dynamic interaction between the generator and the discriminator, which are optimized in response to each other, leading the system to gravitate toward a single dominant solution. This is inherent to the training process of GANs that are trained using stochastic gradient descent (SGD). SGD follows gradients to find minima, and as the loss landscape of GANs is diverse, this process can therefore often find a local and suboptimal minimum [DCLK20]. The following subsections show first how to detect mode collapse, and then in Section 3.3.2 some corrective strategies are presented.

3.3.1 Detection of Mode Collapse

Mode collapse can be detected through various methods, both qualitative and quantitative, which provide insight into whether a GAN has fallen into this issue. By monitoring these indicators, researchers can detect mode collapse early and change parameters to ensure that the generator explores the full range of possibilities in the data distribution.

Visual Inspection of Generated Samples

One of the most straightforward methods for detecting mode collapse is visual inspection of the generated images. If the model is consistently producing identical or very similar images, this is a strong sign that mode collapse has occurred. The absence of diversity in the output is a hallmark of this issue. The generator has



(a) IS (b) FID (c) Spectral Distribution (d) SR (e) SN

Figure 3.6: **Relationship between Mode Collapse Indicators.** This figure presents experiments on the CIFAR-10 dataset conducted by [LTZQ19], illustrating the relationship between different mode collapse indicators like FID for the collapsed model (SN) and the successfully trained model (SR).

become adept at focusing on a limited set of solutions that satisfy the discriminator, rather than exploring the full diversity of the data distribution.

Fréchet Inception Distance

The Fréchet Inception Distance (FID) is a widely used metric to evaluate the quality of generated images, comparing the distribution of real and generated images in the feature space. Their feature representations are extracted from a pre-trained Inception network. With the assumption of Gaussian-like data distribution of synthetic and real-world images, the distance can be measured by the Fréchet distance. This Fréchet distance $d(.,.)$ between the gaussian with mean (m, C) obtained from the real data and the Gaussian with mean (m_w, C_w) from the generated data is called Fréchet Inception Distance (FID) [HRU+17], and given by:

$$d^2((m,C),(m_w,C_w)) = \|m - m_w\|_2^2 + \mathrm{Tr}(C + C_w - 2(CC_w)^{1/2}) \tag{3.10}$$

Lower FID scores indicate that the generated images are more similar to the real images, reflecting both high quality and diversity in the outputs of the generative model. FID improves upon the earlier Inception Score (IS) [SGZ+16] by comparing the mean and covariance of the deep features, providing a more robust measure of both quality and diversity.

If mode collapse occurs, the FID score may increase significantly and/or remain high, indicating that the generated images have little variety and are far from the real data distribution. A consistently high FID suggests that the model has collapsed into a mode that poorly represents the diversity of the original dataset.

Loss Behavior

The behavior of the discriminator and generator losses can also reveal mode collapse. When mode collapse occurs, the discriminator loss becomes very small and stabilizes,

as the discriminator has become too good at identifying the limited set of outputs from the generator [MG21]. Similarly, the generator loss may plateau at a low value, but despite this, the generated images may show little variety, with the same outputs repeating over time [Bro21]. This loss behavior indicates that both the generator and the discriminator are stuck in a suboptimal equilibrium.

Metrics specifically for Mode Coverage

While the Inception Score and FID are commonly used metrics, they focus mainly on the quality of individual samples and may not adequately capture the diversity or modes of the generated data. To address this, several specific metrics have been proposed to assess how well a generator covers the different modes of the data distribution, such as the Geometry Score [KO18] or the Kernel Generalized Empirical Likelihood [RRMD23].

3.3.2 Corrective Strategies for Mode Collapse

Mitigating mode collapse in GANs requires careful manipulation of both the generator and discriminator to encourage the exploration of diverse outputs while still producing high-quality images. Several techniques have been developed to address this challenge and in StyleGAN-XL, a combination of strategies is used to maintain diversity throughout the training.

Freezing Earlier Layers During Progressive Growing

One key method to mitigate mode collapse in StyleGAN-XL is the use of progressive growing, where the network starts by generating low-resolution images and gradually increases the resolution as training progresses. In general, this is already considered a good way to avoid mode collapse [KALL17]. To further stabilize this process and avoid mode collapse, StyleGAN-XL freezes the earlier layers of the network once they have been trained at lower resolutions. This ensures that the generator focuses on refining new features and higher-level patterns as the resolution increases, rather than re-optimizing the same set of low-level features repeatedly. The progressive and stepwise approach reduces the likelihood that the generator overfits to a limited set of outputs, allowing it to better explore the full distribution of the data.

Spectral normalization

Spectral normalization is a technique used in StyleGAN-XL to maintain diversity and stability during training, specifically targeting the discriminator. This normalization indirectly controls the Lipschitz constant of the discriminator, ensuring that the

function remains 1-Lipschitz, which means that its gradient norm is bounded by 1 [MKKY18]. The Lipschitz constant is a measure of how much a function's output can change in response to a change in its input, effectively bounding the function's sensitivity.

By enforcing this constraint, spectral normalization prevents the weight matrices of the discriminator from amplifying the inputs disproportionately. This is crucial for stabilizing training because an overly powerful discriminator can quickly distinguish real from fake samples. This then leads to vanishing gradients and makes it difficult for the generator to learn effectively. If this happens, the generator may collapse and produce only a limited set of outputs.

With spectral normalization, the discriminator's capacity is controlled, enabling the generator to explore the latent space more effectively and resulting in increased diversity in generated samples.

Pre-trained class embeddings

Class conditioning is particularly useful in multi-modal datasets, as it helps to improve the diversity and accuracy of generated outputs across different categories. Instead of training class embeddings from scratch, which has been shown to result in similar samples per class [SSG22], StyleGAN-XL leverages pre-trained embeddings extracted from EfficientNet-Lite0 [TL19].

These pre-trained embeddings are used to condition both the generator and the discriminator, guiding the model toward better class-specific diversity. This prevents the generator from collapsing to a single mode and encourages it to cover the full range of classes in the dataset distribution.

Feature mixing blocks

Feature mixing blocks in StyleGAN-XL also target the discriminator. They help avoid mode collapse by ensuring that the discriminator does not focus on only a small subset of features. A too focused discriminator could lead to the generator exploiting these weaknesses and producing limited repetitive output. Cross-Channel Mixing (CCM) randomizes feature interactions across channels using 1×1 convolutions, while Cross-Scale Mixing (CSM) introduces variation across spatial scales through residual 3×3 convolutions and bilinear upsampling [SSG22]. By enforcing diverse feature representations, these mechanisms prevent the discriminator from becoming overly specialized, ensuring that the generator continues exploring the full data distribution rather than collapsing to a single mode.

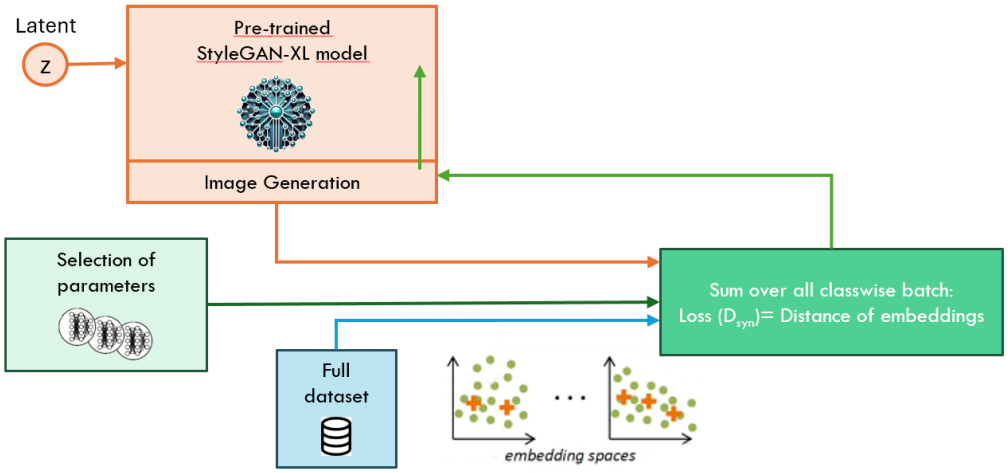


Figure 3.7: **Overview of the Generative Distribution Matching (GenDM) Method.** Instead of using a fixed-size synthetic dataset, the Distribution Matching loss is applied directly to the output of a GAN model (here StyleGAN-XL). This loss is then backpropagated through the model, enhancing the GAN's ability to generate informative images for the distillation task.

3.4 Generative Distribution Matching

Generative Distribution Matching (GenDM) is our novel approach that integrates the ideas of Dataset Distillation and generative modeling to create highly informative synthetic datasets. Using the generative architecture of StyleGAN-XL, a new loss is introduced to improve the images generated by the model not only to look realistic, but also to be informative for the distillation task. An illustration of this method is shown in Fig. 3.7. A pre-trained StyleGAN-XL model is leveraged as a baseline on which fine-tuning training is done using the Distribution Matching loss as in the DiI method described in Section 3.1.1. The final output of GenDM is the trained generator model, which can be used to generate training samples on demand. Therefore, the resulting GenDM method is a distillation to model (DiM) approach.

The detailed training process is described in Algorithm 1. As the Distribution Matching (DM) loss alone is insufficient to train a GAN from scratch a pre-trained StyleGAN-XL is used to initialize the generator G. Instead, the DM loss is applied as a fine-tuning approach to further refine the pre-trained model. This is leveraging the strong prior given by the large-scale pre-training of the GAN model.

In each optimization step, the GAN generates a batch of synthetic images. This is achieved by sampling random latent vectors $\mathbf{z}$ and conditioning them on class labels to obtain a batch of images corresponding to the specified class. This process imposes a higher computational cost compared to classical DM. As in classical DM only a subset of the pre-generated synthetic images needs to be selected in each training

Algorithm 1: Generative Distribution Matching

Data: Training set D
Input: Deep Neural Network F_θ, the distribution/buffer of parameters P_θ, differentiable augmentations $A(\cdot, w)$ parameterized with w, augmentation parameter distribution P_W, number of training iterations K, learning rate η.

Initialize generator G

for $i = 0, ..., K-1$ **do**

- Sample $\theta \sim P_\theta$
- Sample mini-batch of real data $B_c^D \sim D$ and $w \sim P_w$ for every class c
- Sample random latent vectors z
- Generate mini-batch of synthetic images $B_c^S = G(z, c)$ for every class c
- Extract mean feature from training data for every class c
  $EF_D^c = \frac{1}{|B_c^D|} \sum_{x \in B_c^D} F_\theta(A(x, w))$
- Extract mean feature from synthetic data for every class c
  $EF_S^c = \frac{1}{|B_c^S|} \sum_{\hat{x} \in B_c^S} F_\theta(A(\hat{x}, w))$
- Compute loss:
  $L = \sum_{c=0}^{C-1} ||EF_D^c - EF_S^c||^2$
- Update $G \leftarrow G - \eta \nabla_G L$

Result: Generator G

iteration K. However, GANs are the fastest generative models, making them well suited for scenarios requiring continuous image generation throughout training.

In the experiments in Section 4.4.3, the objective is to validate if this fine-tuning approach enhances the GAN's ability not only to generate high-quality images but also to produce highly informative samples for downstream training tasks. By incorporating the new objective, GANs trained with this method become a powerful tool for distilling datasets into generative models, effectively capturing and preserving essential data characteristics. Moreover, since they are generative models, they can leverage key benefits for the distillation task, such as on-the-fly sample generation to adapt to new images per class (IPC) settings, making them highly flexible for various downstream training scenarios.

4 Results

This Section presents the experimental results for the distillation approaches. First, we introduce the used datasets, followed by an overview of the evaluation setup. The evaluation is based on the setup of the *First Dataset Distillation Challenge* [Cha24] that was part of the European Conference on Computer Vision (ECCV) 2024. The interpretation of the challenge rules is described, along with the approach used to enable the comparison between traditional and generative methods. This results in a single framework for comparing the different Dataset Distillation methods. In this framework results of some established methods, namely DM [ZB22] and TTM [CWT+22] are compared with generative approaches using out-of-the-box StyleGAN-XL [SSG22] models. Furthermore, the new methods of GenDM, is tested to assess its potential for improvement. However, the results indicate that it does not outperform the baseline methods.

4.1 Datasets

The experiments are carried out on the datasets CIFAR-100 and TinyImageNet since they are part of the challenge setup in ECCV2024. These datasets are already a challenge for some distillation methods, but the general direction of the field is to find methods that can work with larger and higher resolution datasets than these two. We acknowledge this, but given that the experiments are done to compare multiple approaches in a similar setup, we decide to keep working with these two datasets. They are still challenging enough to help to distinguish the methods and their usability, as well as their resulting distillation quality.

4.1.1 CIFAR-100

The CIFAR-100 dataset [KNHb] consists of 100 classes, each containing 600 images (500 training images and 100 testing images). The 100 classes can be ordered by 20 so-called "super classes", where a super class, for example, can be "aquatic mammals" containing the classes of *beaver, dolphin, otter, seal* and *whale*. Only the 100 fine-labeled classes are used in the following experiments. CIFAR-100 images are of low resolution, specifically 32×32 pixels. This dataset is commonly used for evaluating image classification and generative models, offering a diverse set of categories, from animals to vehicles, making it a useful benchmark for testing.

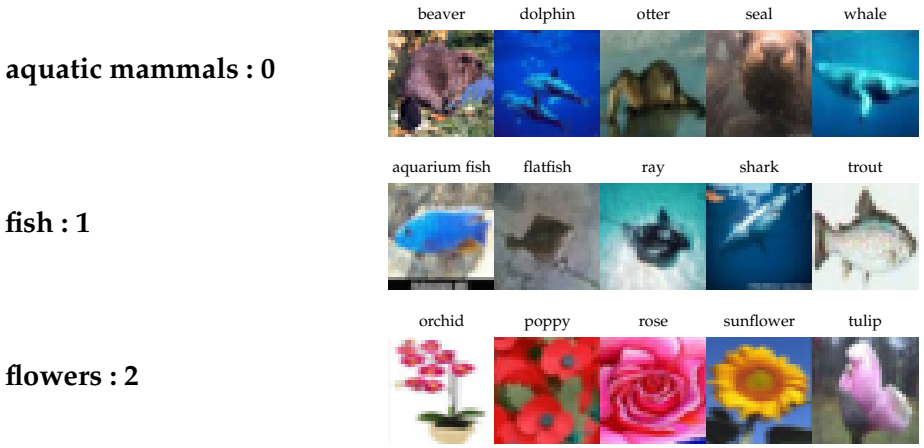


Figure 4.1: **Example visualization of CIFAR-100 classes sorted by superclasses.** This figure shows the first CIFAR-100 superclasses with their respective 5 fine-label classes. All experiments utilize the 100 fine-label classes, rather than the 20 superclass labels.

4.1.2 TinyImageNet

TinyImageNet is a subset of the larger ImageNet dataset, which contains 200 classes out of the 1000 classes, each with 500 training images and 50 validation images. The images in TinyImageNet have a resolution of 64×64 pixels. This dataset serves as a more challenging benchmark than CIFAR-100, providing higher resolution and a larger number of classes, making it ideal for evaluating the scalability and versatility of distillation approaches in producing high-quality and diverse distilled datasets.

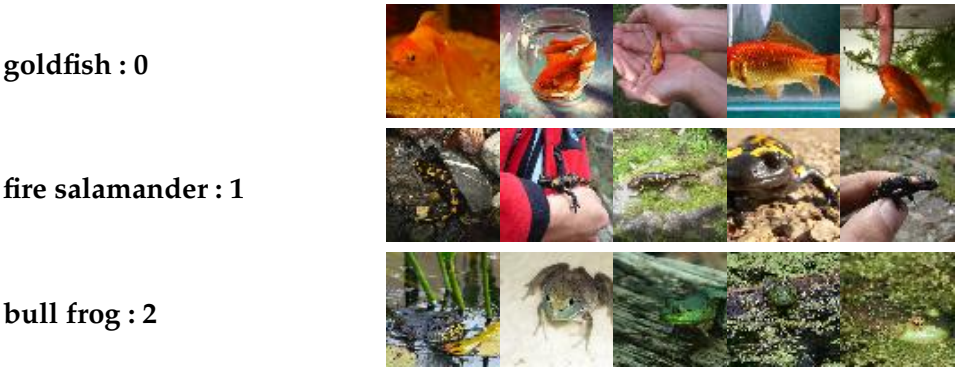


Figure 4.2: **Example visualization of selected TinyImageNet classes.** This figure presents example images from the first three classes of the TinyImageNet dataset.

4.2 Evaluation Setup

The evaluation setup for this thesis is based on the *First Dataset Distillation Challenge* [Cha24] held at ECCV2024. By providing a structured, standardized evaluation platform the challenge is designed to push the development of new and effective Dataset Distillation methods. As this challenge provides a comprehensive and structured way to evaluate a diverse range of Dataset Distillation techniques, this approach is adapted as a basis for evaluation in this thesis.

The challenge is divided into two main tracks, as depicted in Figure 4.3:

1. **Fixed IPC Track:** This track focuses on methods for distilling datasets into a fixed number of images per class (IPC). The chosen 10 IPC setting reduces the dataset to 10 images per class. The training time in this track is fixed to 1000 epochs, providing a controlled environment to assess the effectiveness of the distilled datasets in training models.
2. **Generative Track:** In this track, methods that use generative models to synthesize images are evaluated. The generator may produce any number of images, provided the evaluation time remains consistent, including the time required for image generation. This track allows for a more flexible approach, where the focus is on generative techniques for distillation.

4.2.1 Evaluation Training Time

To ensure the comparability between methods, training time is a key consideration. For the fixed IPC track, the challenge caps training at 1000 epochs. In a fixed IPC setting, this results in the same training times for the downstream task, which is part of the evaluation. The generative track defines that the times should remain consistent. This setup is adapted in this work by modifying epochs relative to the generated images per class ratio to stay within a comparable time window. We found adapting downstream training epochs versus images per class to be more useful than mimicking a specific time measurement. Adjustments to epochs based on training time limits are summarized in Table 4.1.

There is a fixed time for any method that uses a generative model for the image generation of both datasets. The time for CIFAR-100 and TinyImageNet together is 10 min, so 5 min each. This was not clearly stated in the challenge description but the solution this thesis uses as the fixed time period for the generation process during inference. Although comparing the generation time across different hardware setups is challenging, no better reliable measure exists for fairly evaluating different generative models. Wherever this time limit was applied, the Images Per Class (IPC) setting is clearly stated to ensure transparency and consistency in the evaluation. Our setup uses a NVIDIA GeForce RTX 3080 GPU for image generation.

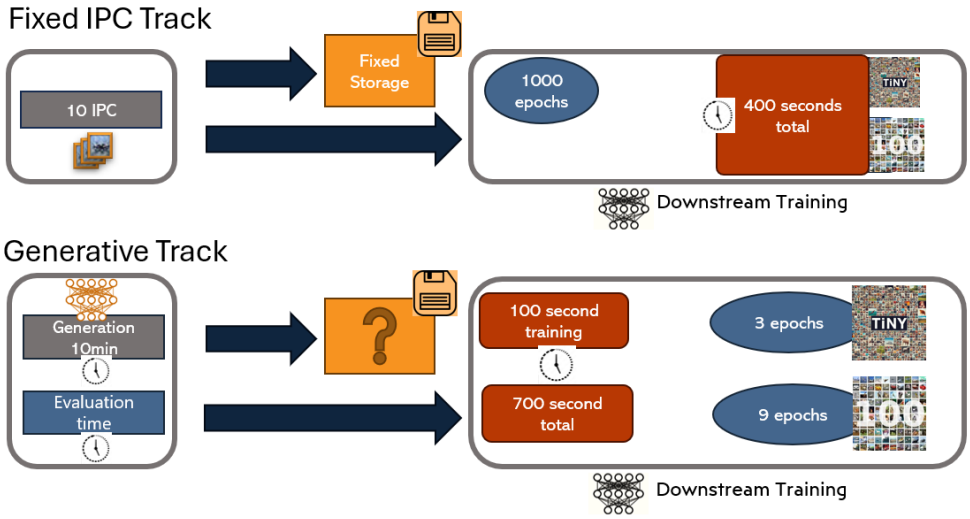


Figure 4.3: **Overview of the Dataset Distillation Challenge tracks.** This figure presents the Dataset Distillation challenge tracks and their associated parameters, highlighting the key factors that enable comparisons between tracks, despite some misalignment in areas such as storage. The left-side boxes illustrate the constraints for each track, such as fixed IPC or generation and evaluation time limits, while the right-hand side translates these constraints into concrete downstream training settings. The red components indicate evaluation time constraints, the orange components represent storage (fixed in the Fixed IPC Track but undefined in the Generative Track), and the blue components denote the number of epochs for evaluation training, which are adjusted in the Generative Track to accommodate training time constraints given higher IPC values.

IPC	No. Epochs in Testing Model Training
10	1000
50	200
100	100
500	20

Table 4.1: **Fixed relationship of IPC to epochs for the testing model training.** By using epochs instead of time in seconds, this approach facilitates more accurate comparisons and makes it easier to recreate experiments on different GPUs.

4.2.2 Distillation Training Time

There are no limitations imposed from the evaluation side on the amount of training that is used for generating the distilled dataset. For comparison of the distillation to images (DiI) approaches with generative approaches, the Distribution Matching (DM) and Matching Training Trajectory (MTT) objectives are setup inside a generative framework. An "identity generator" is implemented to evaluate the traditional DiI objectives without an actual generator part. This dummy generator holds the fixed number of images that the dataset is distilled into. As we still use the generative framework, training times are measured as thousands of images (kimg). So if we trained for 10 kimg that means that we have trained the network with 10.000 images in the training run.

4.3 Baseline Results

To evaluate the effectiveness of the new approach of GenDM, we benchmark it against multiple baseline methods that do not incorporate generative components. Each baseline employs a distinct strategy to select or build distilled datasets, offering a well-rounded performance comparison. The following sections provide a detailed overview of these baselines, beginning with a method based on random selection.

4.3.1 Random

To start with this new experimental setup, as described in Section 4.2, the very basic baseline is created first - a random baseline. Although all methods claim to have some sophisticated way of distilling image information into fewer images as a distilled set, it can often be seen that a primitive baseline using random selection of samples from the full dataset is already quite competitive.

In the case of the Dataset Distillation task, this baseline is created by using a randomly selected subset of images as the "distilled dataset". There is no educated decision about how to choose these images, like in coreset selection, and there is also no part that learns the images themselves. The random baseline gets especially interesting in a setup where the distilled dataset has a reasonable "big" size. This effect of random being a more and more competitive baseline with growing IPC values can be seen in Fig. 4.4.

To complement this baseline, we also present the values for training of the downstream task evaluation setup with the full dataset. This will guide as a theoretically upper limit for all distillation algorithms.

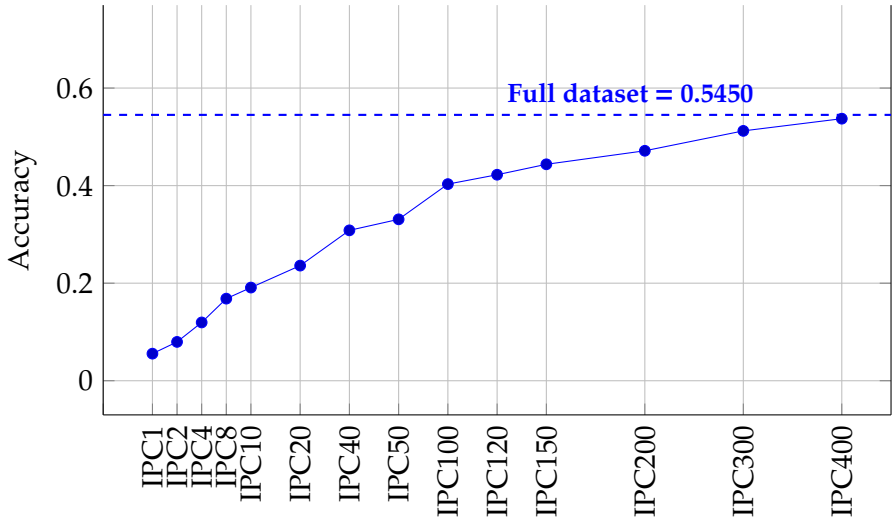


Figure 4.4: **Random Baseline Performance on CIFAR-100.** The random baseline shows the largest accuracy gaps in the low IPC range but quickly converges to the full dataset result. Mean values for 5 runs per IPC are depicted.

4.3.2 Distribution Matching

Experiments using random noise versus taking random images from the respective dataset to initialize the images. The first tests with 1 IPC and 10 IPC settings show that initialization with real images produces faster and better results, so this initialization technique is used for all the following experiments with the other distillation techniques. The initialization effect is illustrated in Fig. 4.5 to facilitate a qualitative assessment.

From the results shown in Table 4.2 we can see that DM is a very good technique for the low IPC regime where it can outperform the random baseline by a margin of 6% over both datasets. This margin shrinks in a higher IPC setting, where it can still outperform the random baseline, but the relative improvement is not as high as in the lower IPC setting.

4.3.3 Matching Training Trajectories

The Matching Training Trajectories (MTT) objective is implemented as an alternative matching method to DM. Examples generated at IPC 10 level for both methods are shown in Table 4.6. Unlike DM, which optimizes for statistical similarity in the resulting feature space of a neural net, MTT directly aligns the training dynamics of the nets on the distilled and real datasets. This approach ensures that distilled data leads to model updates similar to that of training on the entire data set.

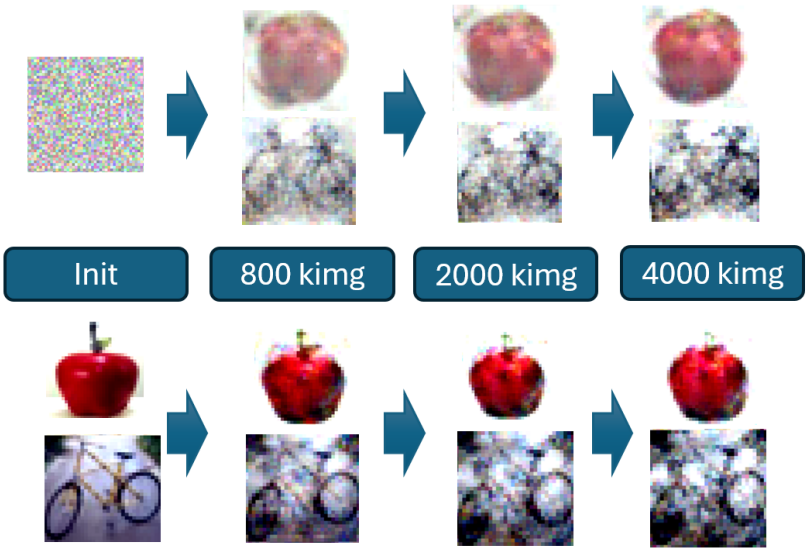


Figure 4.5: **Examples of CIFAR-100 classes "apple" and "bike" with different initialization of synthetic dataset during training.** This figure shows the progression of images for the "apple" and "bike" classes under random and real initialization in the DM 10 IPC setting. Resulting images are displayed at 0, 800, 2000, and 4000 kimg, respectively. The real initialization leads to faster and better results.

To align training dynamics, MTT as a first step requires the generation of buffers containing snapshots of training processes on the full dataset, the expert trajectories. This is an additional computational overhead of this method that also requires significant storage depending on the amount of expert trajectories used for the distillation process.

As shown in Table 4.2, MTT consistently surpasses DM across all IPC settings. For 10 IPC, MTT achieves an improvement of 10% over DM on both TinyImageNet and CIFAR-100, demonstrating its superior ability to capture essential training information. This trend continues in 50 IPC, where MTT further improves and achieves the best performance across all baseline methods.

Furthermore, MTT shows a considerable advantage over random selection, particularly in low IPC settings, where it exceeds the random baseline by more than 12% in both TinyImageNet and CIFAR-100. However, as IPC increases, the performance gap between MTT and random selection narrows, similar to the trend observed for DM.

These results highlight that MTT is the best performing baseline in Dataset Distillation without GANs, making it a strong alternative to DM when optimizing for efficient dataset compression while preserving training dynamics. Nonetheless, these advantages come at an associated cost. MTT requires more computational resources compared to DM due to the need to track and align the training dynamics over

	10 IPC			50 IPC			
Task	**MTT**	**DM**	**Random**	**MTT**	**DM**	**Random**	**Full**
CIFAR100	**39.49**	29.29	17.69	**45.70**	41.26	32.94	54.50
TINY	**19.33**	9.35	6.49	**23.06**	16.12	15.28	34.91
OVERALL	**29.40**	18.35	12.09	**34.38**	29.26	24.11	44.71

Table 4.2: **Comparison of Baseline Methods MTT and DM with the Random Baseline across different IPC settings.** This table shows the mean testing accuracy (in %) for three evaluation runs across two datasets (CIFAR-100 and TinyImageNet) and two IPC settings (10 IPC and 50 IPC). The results are compared with the random baseline and the full dataset performance. The highest accuracy for each task and IPC setting is highlighted in bold.

multiple optimization steps. Additionally, MTT's effectiveness heavily depends on accurate trajectory alignment, making it more sensitive to hyperparameter choices and requiring careful tuning. Although it outperforms other methods in lower IPC settings, its computational overhead increases with the complexity of the data set, limiting its scalability for large-scale distillation tasks.

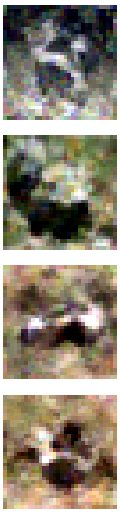
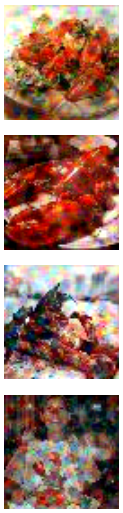
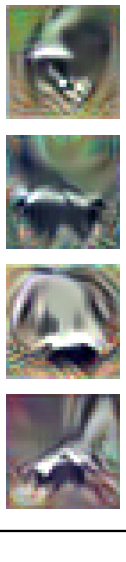
	'skunk' (CIFAR-100)	'chair' (CIFAR-100)	'lobster' (Tiny)	'beach wagon' (Tiny)
DM				
MTT				

Table 4.3: **Generated images from DM and MTT for selected classes.** This figure shows generated images from the DM and MTT methods for selected classes from CIFAR-100 and TinyImageNet, at a distillation rate of 10 IPC.

4.4 Introduction of GANs for Dataset Distillation

The introduction of GANs into the Dataset Distillation process has improved knowledge transfer by leveraging high-quality synthetic data representations. Traditional Dataset Distillation techniques, such as DM and MTT, primarily focus on condensing large datasets into smaller, highly informative subsets. However, GANs provide an alternative route by synthesizing high-fidelity data distributions in the form of generative models that can serve as a rich medium for distillation.

4.4.1 StyleGAN-XL

StyleGAN, a state-of-the-art generative model, is able to generate high-quality images that resemble those from a given image distribution, such as the ImageNet dataset. When used in distillation, the StyleGAN-generated samples act as a synthetic data set used for downstream training tasks. As with all distillation into model approaches, the scaling from one IPC ratio to the next is seamless and does not require any extra training. The generator just needs more generation time to produce more images per class.

In the experiments, the StyleGAN version from [SSG22] StyleGAN-XL (SGAN-XL) is used. As the GAN training process already leads to a compression of the original dataset distribution into the parameters of the generative models, a first evaluation of the SGAN-XL models themselves is done without any additional distillation-specific training.

The available pre-trained ImageNet model from the official repository for the TinyImageNet dataset is used, and the 200 TinyImageNet classes from the full set of 1000 ImageNet classes are selected. For CIFAR-100, as there is no pre-trained version available, two options are explored. The first is to train our own SGAN-XL model for the CIFAR100 dataset. Starting with extensive training in low resolution (resolution 8×8) a few different versions of SGAN-XL were trained. One of the main issues in the experiments was the tendency of the GAN training process to go into mode collapse. Further elaboration on this topic is provided in the next subchapter 4.4.2.

Due to the difficulties with training a very good GAN model, another option to evaluate SGAN-XL as a baseline for the distillation task is to use a mapping. In this mapping, the 100 classes from CIFAR-100 are matched with their "closest" class in ImageNet. Some classes (e.g. "Bee", CIFAR-100 class 6) exist in ImageNet. For other classes there are no exact matches, but only similar classes, like "Bear" in CIFAR-100 which could correspond to the "Brown Bear", "Black Bear" and "Ice Bear" classes of ImageNet. Finally, there are classes in CIFAR-100 like "Baby" that do not have any close correspondence in ImageNet. These were mapped to other classes that might have a spurious correlation, like the ImageNet class "Diaper". This correlation can be seen in the samples of the "diaper" class depicted in 4.6 that show many babies appearing in them as well. By using this mapping, we are able to train a model to

recognize babies from most of the examples, as the images of class "diaper" contain them as well.

Figure 4.6: **Spurious correlation between "Diaper" and "Baby" in ImageNet.** This figure shows samples from the ImageNet class "diaper," illustrating the spurious correlation where the class also contains many images of "babies". This correlation is leveraged in our mapping approach, highlighting how such unintended associations can be used for transferring knowledge between datasets.

Starting with a class mapping generated by ChatGPT, we manually refined the selected corresponding ImageNet classes for cases of CIFAR-100 classes where the base SGAN-XL had very low CIFAR-100 accuracy. We verified the mappings by comparing close mappings against each other as with the example of "Brown Bear", "Black Bear" and "Ice Bear", and we found that using "Brown Bear" as the CIFAR-100 class "Bear" resulted in the highest accuracy. The class mapping used in this work is presented in Table 4.4.

Table 4.4: **Mapping between CIFAR-100 and ImageNet classes.** This table shows the concrete mapping of all 100 CIFAR-100 classes to their respective ImageNet classes. Some mappings are direct, while others leverage spurious correlations or similarities, such as "bear" in CIFAR-100 matching "brown bear" in ImageNet. The mapping illustrates how both direct and indirect associations are used to transfer knowledge between the two datasets.

No.	CIFAR-100 Name	No.	ImageNet Name
0	Apple	957	Pomegranate
1	Aquarium fish	1	Goldfish
2	Baby	529	Diaper
3	Bear	294	Brown bear
4	Beaver	337	Beaver
5	Bed	564	Four-poster
6	Bee	309	Bee
7	Beetle	305	Dung beetle
8	Bicycle	671	Mountain bike
9	Bottle	440	Beer bottle
10	Bowl	659	Mixing bowl
11	Boy	982	Groom
12	Bridge	821	Steel arch bridge
13	Bus	654	Minibus
14	Butterfly	323	Monarch butterfly

Continued on next page

Table 4.4 (continued from previous page)

No.	CIFAR-100 Name	No.	ImageNet Name
15	Camel	354	Arabian camel, Dromedary
16	Can	412	Ashcan, Trash can, Wastebin
17	Castle	483	Castle
18	Caterpillar	329	Sea cucumber
19	Cattle	345	Ox
20	Chair	559	Folding chair
21	Chimpanzee	367	Chimpanzee
22	Clock	409	Analog clock
23	Cloud	980	Volcano
24	Cockroach	314	Cockroach
25	Couch	831	Studio couch
26	Crab	119	Rock crab, Cancer irroratus
27	Crocodile	50	American alligator
28	Cup	968	Cup
29	Dinosaur	51	Triceratops
30	Dolphin	147	Grey whale
31	Elephant	386	African elephant
32	Flatfish	391	Coho salmon
33	Forest	979	Valley, Vale
34	Fox	277	Red fox
35	Girl	445	Bikini, Two-piece
36	Hamster	333	Hamster
37	House	449	Boathouse
38	Kangaroo	104	Wallaby, Bush kangaroo
39	Keyboard	508	Computer keyboard, Keypad
40	Lamp	619	Lampshade, Lamp shade
41	Lawn mower	621	Lawn mower, Mower
42	Leopard	288	Leopard
43	Lion	291	Lion
44	Lizard	44	Alligator lizard
45	Lobster	122	American lobster
46	Man	842	Swimming trunks, Bathing trunks
47	Maple tree	31	Tree frog
48	Motorcycle	665	Moped
49	Mountain	970	Alp
50	Mouse	333	Hamster
51	Mushroom	991	Coral fungus
52	Oak tree	995	Earthstar
53	Orange	951	Orange
54	Orchid	986	Yellow lady's slippers
55	Otter	360	Otter
56	Palm tree	988	Acorn
57	Pear	948	Granny smith
58	Pickup truck	717	Pickup truck
59	Pine tree	955	Jackfruit
60	Plain	718	Pier
61	Plate	923	Plate
62	Poppy	949	Strawberry
63	Porcupine	334	Porcupine
64	Possum	362	Badger

Continued on next page

Table 4.4 (continued from previous page)

No.	CIFAR-100 Name	No.	ImageNet Name
65	Rabbit	332	Angora rabbit
66	Raccoon	387	Lesser panda, Red panda
67	Ray	6	Stingray
68	Road	895	Warplane, Military plane
69	Rocket	812	Space shuttle
70	Rose	986	Yellow lady's slippers
71	Sea	978	Seashore, Coast, Seacoast
72	Seal	150	Sea lion
73	Shark	3	Tiger shark
74	Shrew	357	Mink
75	Skunk	361	Skunk, Polecat
76	Skyscraper	682	Obelisk
77	Snail	113	Snail
78	Snake	52	Thunder snake
79	Spider	72	Black and gold garden spider
80	Squirrel	335	Fox squirrel
81	Streetcar	829	Streetcar
82	Sunflower	985	Daisy
83	Sweet pepper	945	Bell pepper
84	Table	532	Dining table, Board
85	Tank	847	Tank
86	Telephone	487	Cellular telephone
87	Television	851	Television
88	Tiger	292	Tiger
89	Tractor	866	Tractor
90	Train	547	Electric locomotive
91	Trout	389	Barracouta
92	Tulip	985	Daisy
93	Turtle	33	Loggerhead
94	Wardrobe	894	Wardrobe
95	Whale	148	Killer whale
96	Willow tree	979	Valley, Vale
97	Wolf	269	Timber wolf
98	Woman	903	Wig
99	Worm	111	Roundworm

As shown in Table 4.5, the out-of-the-box SGAN-XL model already outperforms random selection for the TinyImageNet dataset, even at the 10 IPC level. However, this is not true for CIFAR-100. As mentioned, there is no pre-trained model for CIFAR-100, so we used two approaches that do not sufficiently cover the dataset for the distillation task. The trained model slightly improves upon the mapped option, but in the 5 min setting, a truncation value of 1 needs to be used for the best result, indicating the model still suffers from mode collapse, reducing dataset diversity. Since the pre-trained SGAN-XL model is more diverse, the mapped SGAN version performs better in the 5 min setting compared to the trained SGAN. The best truncation value of 0.8 for this approach also confirms, that mode collapse is not the factor reducing the accuracy of this model.

Task	10 IPC			5 mins		Full
	Random	SGAN-XL (mapped)	SGAN-XL	SGAN-XL (mapped)	SGAN-XL	
CIFAR100	**17.69**	9.86	11.76	16.41	17.20	54.50
TINY	6.49	9.09	9.09	**14.56**	**14.56**	34.91
OVERALL	12.09	9.48	10.43	15.49	**15.88**	44.71

Table 4.5: **Comparison of GAN baseline methods using StyleGAN-XL (SGAN-XL).** This table presents SGAN-XL results in two configurations: a mapping approach for CIFAR-100 and a fully trained network. Accuracy is shown for 10 IPC (truncation = 0.6) and a 5 min image generation time (IPC=37, truncation=0.8, truncation for custom-trained=1.0). Higher truncation values are needed for the custom-trained to mitigate the mode collapse effects. The highest accuracy for each task across both settings is highlighted in bold for comparability.

	'skunk' (CIFAR-100)	'chair' (CIFAR-100)	'lobster' (Tiny)	'beach wagon' (Tiny)
SGAN-XL				

Table 4.6: **Generated images from SGAN-XL for selected Classes.** This figure displays images generated by SGAN-XL for selected classes from CIFAR-100 and TinyImageNet, using our own trained CIFAR-100 model. As evident in the images, the CIFAR-100 model suffers from a tendency to mode collapse, showing minimal variance within the generated images for each class.

4.4.2 Mode Collapse and Distillation

Even though SGAN-XL implements some methods to mitigate the effects of mode collapse, it is still important to closely monitor the training progress and tune all parameters in a way that helps to avoid mode collapse. Mode collapse occurs when the generator produces a limited variety of outputs that do not represent the full diversity of the target distribution. In Fig. 4.7, we can see an example in which two different stems for the CIFAR-100 dataset are trained at the lowest resolution of 8×8 pixels. The upper model goes into a partial mode collapse where visually the images of the classes in the middle all end up with a light gray background and some blob-like object in the middle. At the same time, the images of the stem on the bottom do look more diverse, and we can clearly see different classes with different backgrounds in the part that was collapsed in the other model. The Fréchet Inception Distance (FID) scores per model also help to identify this issue as the model with collapse is not improving the FID scores as consistently and fast as the other stem does.

For further training in this case we use the stem depicted on the bottom and increase the resolution by a factor of 2 for the next stage and then finally use the best model for the final training of our own SGAN-XL on resolution 32×32 pixels for the CIFAR-100 dataset.

The results of training our own model highlight the crucial role of diversity in the distilled dataset. Even minor instances of mode collapse — sometimes affecting only specific classes — can significantly hinder the quality of the resulting distillation performance. Furthermore, distillation performance itself serves as a valuable indicator of whether a model has undergone partial or complete mode collapse. In some experiments, the accuracy not only for the full dataset but also on a per-class basis was evaluated, sometimes showing large differences. This class-wise accuracy is influenced both by the intrinsic difficulty of the class — especially in relation to similar categories, such as the "tree classes" in CIFAR-100 — and by the extent of mode collapse within the corresponding part of SGAN-XL. Thus, a high variance in class-wise accuracy can serve as a diagnostic tool to detect partial mode collapse in conditional GANs.

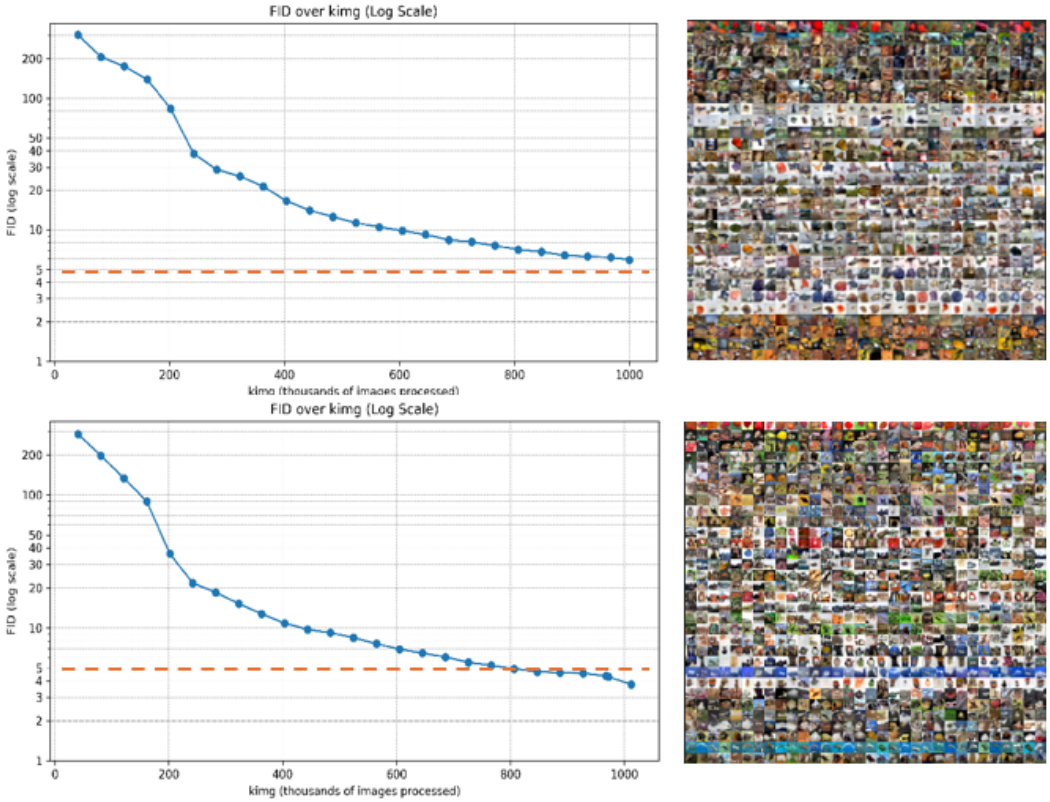


Figure 4.7: **Detecting Mode Collapse in SGAN-XL for CIFAR-100 at 8×8 Resolution.** The right side of the figure shows generated images in an image grid, with each row representing a certain class. On the left, the Fréchet Inception Distance (FID) scores track the model's performance throughout training. The orange-highlighted FID=5 serves as a visual aid in this figure, helping the reader discern differences in FID progression. Both the visual results and the FID scores highlight the onset of mode collapse in the model at the top, where it fails to capture the full diversity of the CIFAR-100 dataset. For this model the images in the middle of the grid all have a similar appearance, while the associated FID score graph shows that the improvement in FID (lower score is better) no longer continues but instead begins to saturate.

4.4.3 GenDM - combining DM-style loss with GAN

The Generative Distribution Matching method (GenDM) is based on a pre-trained GAN model. As shown in Section 4.4.1, the SGAN-XL model serves as a strong baseline for the distillation task, which is why we use it in this work. For the TinyImageNet dataset, a pre-trained SGAN-XL model is leveraged that is trained for the full ImageNet dataset on a resolution of 64×64 pixels. For the CIFAR-100 dataset, we use the approach with an ImageNet model with special mapping to the 100 classes, as described in Section 4.4.1. For SGAN-XL the results of the trained version are used in Table 4.7 as it performed better than the mapped version on the CIFAR-100 dataset, although it is not as diverse as it could be compared to the available pre-trained versions for other datasets.

With each of the models, the fine-tuning training of GenDM is initiated, replacing the standard GAN loss with the Distribution Matching (DM) loss. The goal is to determine whether this loss, specifically designed for the distillation task, can enhance the GAN's ability to generate images that preserve more relevant information for downstream tasks.

However, as shown in Table 4.7, the results do not support our hypothesis that DM loss can improve GAN-generated images for the Dataset Distillation task. In our experimental setup, the introduction of DM loss as a singular loss to train the SGAN-XL model results in a decline in the distillation accuracy. As we can see in the graph in Fig. 4.9, the trained GAN consistently reduces DM loss during the fine-tuning stages. After about 40 batches (of size 128 images), so around 5 kimg the decrease in loss starts to slow down to then saturate against a level of 300. However, this reduction in loss in the training process does not result in a better distilled dataset for the downstream task.

In Fig. 4.8, we visualize three image grids, each row corresponding to a label class on TinyImageNet. The first one for 1 kimg stage, the next at 5 kimg and then last one after 50 kimg. We can see how the shapes of animals and other objects are gradually lost and repetitive structures are introduced. This structure is similar to what is observed in the baseline of pure DM, as shown in Table 4.6.

Although these structures in the DM baseline improve the images for the distillation task, the same does not apply to the GAN-generated images. A possible explanation for this discrepancy is that the loss function performs well in direct pixel space but does not translate effectively to the latent space of GANs. This may be due to fundamental differences in the representations learned by GANs compared to direct pixel-based reconstructions. However, this hypothesis requires further validation through systematic experiments, analyzing the impact of loss formulation on both domains and assessing whether alternative regularization techniques can bridge this gap.

(a) (b)

(c)

Figure 4.8: **Example images produced by GenDM at different stages of fine-tuning.** This figure shows generated images from the GenDM approach on TinyImageNet dataset at three different fine-tuning stages: 1 kimg (a), 5 kimg (b), and 50 kimg (c). As fine-tuning progresses, repetitive structures begin to emerge in the images, causing a decline in image quality. This degradation negatively impacts the performance in downstream tasks, making the approach less effective as a Dataset Distillation method.

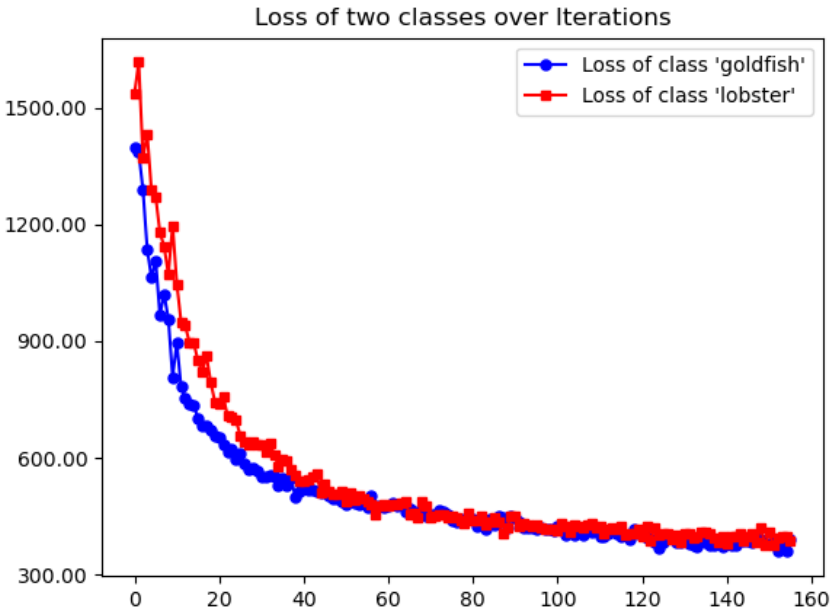


Figure 4.9: **Training GenDM reduces the DM Loss.** This figure shows the loss curves for two exemplary classes, illustrating how the GenDM approach reduces the DM loss over the course of training. The y-axis represents the DM loss value, while the x-axis denotes the number of batches of images during the training process (128 images per batch), covering a total of 20 kimg.

	10 IPC			5 mins		
Task	**GenDM**	**SGAN-XL**	**MTT**	**GenDM**	**SGAN-XL**	**Full**
CIFAR100	8.86	11.76	**39.48**	13.47	17.20	54.50
TINY	8.89	9.09	**19.33**	14.08	14.56	34.91
OVERALL	8.88	10.43	**29.40**	13.78	15.88	44.71

Table 4.7: **Comparison of Baseline Methods and GenDM Results.** This figure presents the results of both generative and non-generative baselines compared to the GenDM approach. For StyleGAN, the results of the trained version are shown. Notably, the MTT method consistently outperforms GenDM across all tasks, demonstrating its superior effectiveness in comparison to the other methods.

4.5 Enhancing Generative Dataset Distillation

In this section, the impact of various modifications to generative Dataset Distillation is analyzed through a series of ablation studies. These studies focus on understanding how different design choices and hyperparameter configurations affect the overall performance of generative approaches. To enhance the generative approach, we look at the truncation parameter of the GAN, the trade-off between resolution and images per class and the use of multiple resolutions in a synthetic dataset.

Truncation

Figure 4.10: **Examples of TinyImageNet class 'goldfish' at different truncation levels.** This figure shows generated images for class 'goldfish' from TinyImageNet at three truncation levels: 0.3 (top), 0.6 (middle), and 1.0 (bottom). As truncation decreases, the diversity of the generated images is limited, while higher values can introduce artifacts. Based on our experiments, the optimal truncation value for this model is 0.6 at 10 IPC distillation rate.

The experiments are conducted using the pre-trained SGAN-XL models to find optimal truncation values for the GAN setups. The truncation between a value of 0 and 1 influences the variance of the underlying normal function as described in Section 3.2.3. We can either preserve the full variance, corresponding to a truncation value of 1, or truncate the input variance to obtain values closer to the mean of the image distribution - truncation values close to 0.

The results suggest that, while higher truncation values yield more diverse samples, they can introduce artifacts, as shown in Fig. 4.10, while lower truncation values improve the fidelity of the samples at the cost of reduced diversity. For the Dataset Distillation task, balancing diversity and representativeness via this parameter is a crucial factor.

The 10 IPC setting is representative of a scenario where only a few images are available for distillation, and thus a truncation value of 0.6 achieves the best balance

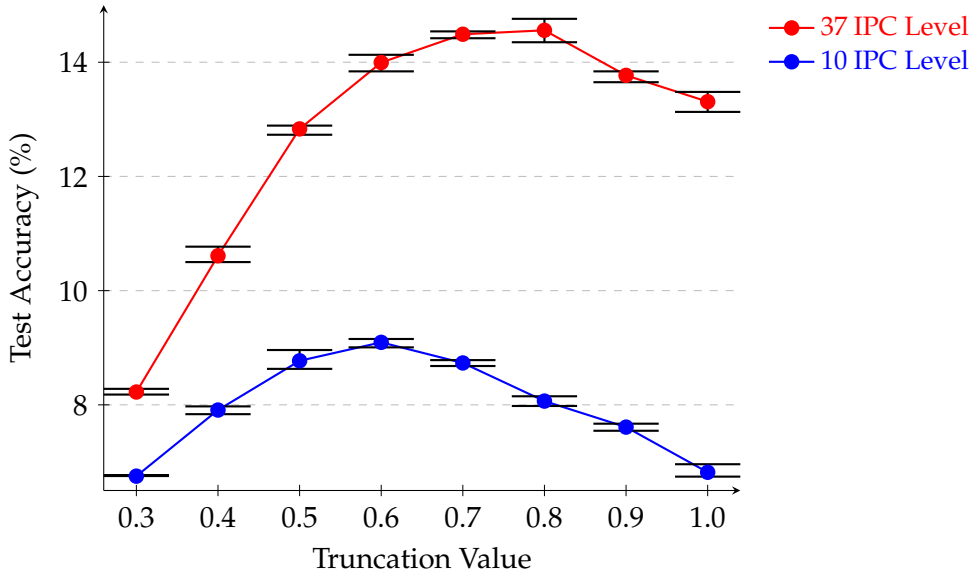


Figure 4.11: **Effect of Truncation Levels on StyleGAN-Generated Images for Tiny-ImageNet at IPC 10 and IPC 37.** This figure illustrates the impact of different truncation levels on the performance of StyleGAN-generated images at two IPC settings: 10 IPC and 37 IPC (equivalent to 5-minute generation time). In the IPC 10 setting (indicated by the blue line), the optimal truncation value is 0.6, while in the IPC 37 setting, a higher truncation value of 0.8 is found to be optimal. Overall, mid-range to high truncation values yield the best results, with a significant performance gap observed between the best and worst truncation levels.

between diversity and fidelity. In this context, the best truncation value improves accuracy by more than 2%. Using the optimal truncation value with the same underlying model results in an improvement of greater than 25% compared to the suboptimal truncation settings.

In contrast, the 37 IPC setting, which corresponds to a 5-minute generation time per image, introduces a different challenge. With higher IPC levels, the model benefits from a more diverse dataset, so the optimal truncation value shifts to 0.8. The 37 IPC setting (red line in Fig. 4.11) demonstrates this shift, where higher truncation values allow the model to capture more variation and diversity in the data, essential for tasks requiring more images. This increased diversity comes at the cost of some loss in fidelity, but leads to a significant 6% more accuracy when the optimal truncation value is used instead of the suboptimal 0.3 setting.

These results illustrate that the optimal truncation value is not fixed but depends on the number of images used in the distilled dataset. As the IPC level increases, the optimal truncation value moves from intermediate to higher values, as a good

Generation Time	Resolution	IPC	Accuracy
1 min	16×16	33	**7.99**
	32×32	13	7.44
	64×64	7	6.83
5 min	16×16	170	11.59
	32×32	68	13.08
	64×64	37	**14.31**

Table 4.8: **Trade-off between resolution and IPC under different generation time constraints.** The results are based on SGAN-XL for the TinyImageNet dataset. With a fixed generation time, increasing image resolution reduces the number of images per class (IPC). At short generation times (1 min), it is beneficial to generate a large number of low-resolution images. However, with longer generation times (5 min), the best strategy shifts toward generating fewer high-resolution images, which achieve better accuracy.

distilled dataset needs more diversity in higher IPC settings than it does at the 10 IPC level. We show our results and this effect of high influence of truncation on accuracy, as well as the shift in the optimal value, in Fig. 4.11.

Resolution-IPC-Tradeoff

As the Dataset Distillation test setup includes a fixed time period for image generation for the generative approaches, there is a coupling between the image resolution and the number of images per class (IPC). The generation time for images of lower resolution is lower as these models have fewer layers and therefore parameters. This allows the generation of 100 images with resolution 16×16 pixels in 2.91 minutes, while generating the same amount of images at resolution of 64×64 takes 13.26 minutes, which is 4.5 times longer. By the demand for a fixed overall generation time, we can either use a large number of low-resolution images and upscale them or work with a smaller set of images at the target resolution.

Table 4.8 presents the results for the TinyImageNet dataset, highlighting the optimal tradeoff between resolution and IPC at two different image generation time settings. In the 1-minute setting, the lowest resolution yields the best accuracy, whereas in the 5-minute setting, the best performance is achieved with a smaller quantity of high-resolution images. So, the tradeoff becomes more significant with shorter generation times, as longer generation periods allow for the creation of sufficient high-resolution images, diminishing the benefits of larger amounts of low-resolution data.

Synthetic Datasets with Multi-Scale Resolutions

A hybrid approach that leverages multiple resolutions within the synthetic dataset offers the potential to maximize the resolution-IPC tradeoff. Instead of exclusively generating either low- or high-resolution images, both can be combined: producing a large number of low-resolution images (which are later scaled up) while simultaneously generating a smaller subset of high-resolution images to capture finer details. This ensures that the model benefits from both the diversity of a high IPC and the structural richness of high-resolution images.

By incorporating high-resolution samples alongside upscaled low-resolution images, we aim to improve the overall training effectiveness of downstream models. The high-resolution images serve as reference points, helping the model learn finer textures and details that are otherwise lost during up-scaling. Meanwhile, the large number of lower-resolution images improves the sample diversity, making the distilled dataset more representative.

The approach is particularly beneficial under strict generation time constraints, as it balances computational efficiency with the need for high-quality training samples. An exploration of optimal low- and high-resolution ratios in the distilled dataset for the TinyImageNet dataset is presented in Table 4.9.

The results demonstrate that the use of a combination of low- and high-resolution images provides a notable improvement in classification accuracy compared to the use of a single resolution. Specifically, mixed resolution settings with a combination of 16×16 and 64×64, in a $2:1$ IPC ratio, achieved the highest accuracy of 15.39%, outperforming configurations with only lower or higher resolution. Using only low-resolution images at 16×16 resulted in an accuracy of only 11.59%, while using high-resolution images at 64×64, reached 14.31%.

This suggests that while higher resolutions offer some accuracy advantages over lower resolutions, the most effective approach is a balanced combination of both. The mixed-resolution strategy achieves an optimal tradeoff between computational efficiency and accuracy. In particular, the best mixed resolution setting outperforms the single resolution top approach (64×64) by 1.08%, emphasizing the benefit of incorporating high and low resolution images in the distilled dataset. This approach is especially valuable when time and computational resources are limited, making mixed resolutions a compelling choice for efficient yet high-quality model training.

Resolutions	IPC Ratios	Accuracy
16×16 with 32×32	1:1	14.04
	2:1	14.10
	3:1	13.97
16×16 with 64×64	1:1	15.00
	2:1	**15.39**
	3:1	14.94
32×32 with 64×64	1:1	15.33
	2:1	15.25
	3:1	15.37

Table 4.9: **Accuracy results for combinations of different resolutions and IPC ratios.** This table presents the results for combining different image resolutions at different IPC ratios for SGAN-XL-generated images for the TinyImageNet dataset. Three different resolution combinations are tested: 16×16 with 32×32, 16×16 with 64×64, and 32×32 with 64×64. For each setting the IPC ratios of 1:1, 2:1, and 3:1 are evaluated, that means first we mix same amount of lower resolution with higher resolution images while in the next setting we double the amount of low resolution in relationship to the high resolution images. The highest accuracy over all settings is highlighted. Overall, the combination of different resolutions leads to a higher IPC and consistently improves performance compared to using only the higher resolution. The best result is achieved when using high resolution with double the amount of the lowest resolution.

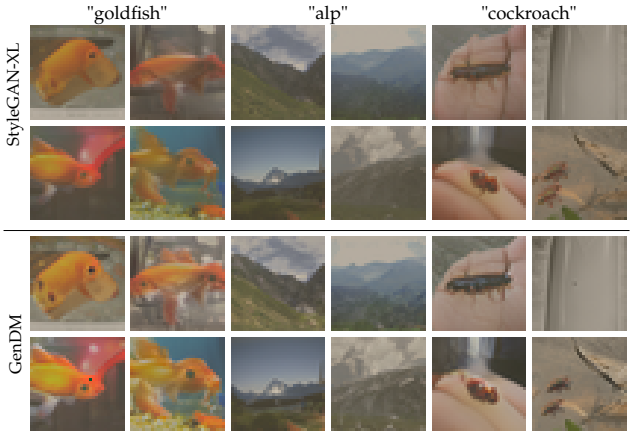


Figure 4.12: **Generative models synthesize informative training samples.** We visualize TinyImageNet data generated by two different models, (1) an off-the-shelf StyleGAN-XL [SSG22], and (2) GenDM, a StyleGAN-XL generator fine-tuned with Distribution Matching [ZB22].

4.6 The First Dataset Distillation Challenge at ECCV2024

We submitted an initial GenDM version to the generative track of the first dataset distillation challenge [Cha24]. The challenge benchmarked different dataset distillation techniques under constrained computational budgets in two different tracks. Our method, GenDM, based on a pre-trained StyleGAN-XL model, was fine-tuned using the Distribution Matching objective to enhance the informativeness of generated samples. Despite limited fine-tuning iterations, GenDM demonstrated competitive performance, ranking second in the generative track. The results of the challenge are shown in Table 4.10. In Fig. 4.12, samples from the StyleGAN base, as well as this version of GenDM, show strong similarities due to limited fine-tuning.

Our second place finish in the challenge validates GenDM as a promising approach to distilling data sets. In this thesis, we explored this first approach further and gathered more data on extended fine-tuning, optimized hyperparameters, and multiple resolution generation to enhance the performance of GenDM.

Rank	Team	CIFAR100	Tiny ImageNet	Overall
1	zheli	37.65	23.53	30.54
2	Mel (Ours)	14.59	11.84	13.21
3	Guang	1.00	4.60	2.80
-	*Full Dataset*	*54.03*	*29.69*	*41.86*
*1**	*Zhou_chuhao*	*40.61*	*18.31*	*29.45*
-	*Random 10 IPC*	*18.58*	*5.34*	*11.95*

Table 4.10: **Dataset Distillation Challenge.** Our approach placed second on the generative track. *top entry of the fixed IPC track.

5 Conclusion

Generative Dataset Distillation presents an innovative direction in the field of dataset distillation, leveraging the power of generative models to enhance knowledge transfer and improve data efficiency. This thesis explores generative approaches, particularly focusing on StyleGAN-XL and the new GenDM method that integrates the loss of traditional Distribution Matching in GAN training. These generative approaches demonstrate the potential to outperform traditional dataset distillation techniques under specific conditions, particularly when dealing with tasks that require a variable number of images per class (IPC).

However, despite the promising aspects, the results of GenDM show that the direct use of the Distribution Matching loss in a GAN does not improve the results and even pushes the GAN into some sub-optimal local minima. One of the key limitations observed is the performance gap in scenarios where memory constraints are extremely tight. Traditional methods such as Distribution Matching (DM) and Training Trajectory Matching (TTM) still maintain advantages in these settings. They are more computationally efficient and better suited for tasks requiring strict data compression without excessive computational overhead.

However, generative approaches are particularly effective in tasks where adaptability to varying IPC is crucial. Using generative models, it becomes possible to create diverse and informative synthetic datasets, which can be advantageous in applications where generalization across multiple domains is required. Nevertheless, challenges such as mode collapse, computational cost, and fidelity of the generated samples remain barriers to widespread adoption.

An important contribution of this thesis is the establishment of a new evaluation protocol, supported by the first Dataset Distillation Challenge. This protocol provides a common framework to compare generative and non-generative approaches, ensuring a fair and structured evaluation of different methodologies. Although this has helped ground the comparisons, there remains room for discussion on how such evaluations should be conducted. Specifically, factors such as memory usage, computational efficiency, and the practical deployment of distilled datasets should be carefully considered when designing future benchmarks. Addressing these aspects will be essential to refine the evaluation standards and improve understanding of the trade-offs between different distillation strategies.

The conducted experiments provide valuable insights into how the number and resolution of generated images impact the effectiveness of downstream task training. A well-balanced dataset plays a crucial role in supporting efficient model learning. The results highlight the tradeoff between image resolution and generation time in GAN-based dataset distillation. Here, a certain number of high-resolution images combined with some larger number of low-resolution samples works best for the later tasks.

Additionally, the experiments highlight the importance of truncation as a valuable parameter for tuning a well-distilled model. Although not originally designed for dataset distillation, truncation aligns closely with its objectives by controlling the tradeoff between diversity and sample quality. This makes it particularly useful for small distilled datasets, where balancing these factors is crucial. While truncation is generally ineffective for large synthetic datasets, it enhances the quality of smaller ones.

In conclusion, while generative dataset distillation approaches offer a promising new avenue for dataset condensation, they are not a replacement for traditional methods. Instead, they should be seen as a complementary approach, particularly valuable in settings where flexibility and adaptability are more important than extreme memory efficiency. As for the use of Distribution Matching loss as a guidance for fine-tuning, our results indicate that applying it solely is not sufficient to improve GAN-based dataset distillation. This suggests that while Distribution Matching provides useful constraints, it may need to be combined with other objectives to effectively guide the generative process.

Future work could explore alternative losses for the distillation task and investigate how they can be integrated with GAN loss as an additional component rather than a sole objective for fine-tuning. Another direction for further research should evaluate the potential of diffusion models as generative models for dataset distillation, assessing their effectiveness and suitability for this task.

Bibliography

[ACB17] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan, 2017.

[BDS18] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. *CoRR*, abs/1809.11096, 2018.

[Bro21] Jason Brownlee. How to identify and diagnose gan failure modes, Jan 2021.

[Cha24] Dataset Distillation Challenge. The first dataset distillation challenge. `https://dd-challenge-main.vercel.app/#/`, 2024. Accessed: 2025-01-28.

[CHW+22] Darius Chira, Ilian Haralampiev, Ole Winther, Andrea Dittadi, and Valentin Liévin. Image super-resolution with deep variational autoencoders, 2022.

[CWS12] Yutian Chen, Max Welling, and Alexander J. Smola. Super-samples from kernel herding. *CoRR*, abs/1203.3472, 2012.

[CWT+22] George Cazenavette, Tongzhou Wang, Antonio Torralba, Alexei A. Efros, and Jun-Yan Zhu. Dataset distillation by matching training trajectories, 2022.

[CWT+23] George Cazenavette, Tongzhou Wang, Antonio Torralba, Alexei A. Efros, and Jun-Yan Zhu. Generalizing dataset distillation via deep generative prior, 2023.

[CWX+20] Kelvin C. K. Chan, Xintao Wang, Xiangyu Xu, Jinwei Gu, and Chen Change Loy. GLEAN: generative latent bank for large-factor image super-resolution. *CoRR*, abs/2012.00739, 2020.

[DAS+24] Priyanshu Deshmukh, Pranav Ambulkar, Pranoti Sarjoshi, Harshal Dabhade, and Saurabh A. Shah. Advancements in generative modeling: A comprehensive survey of gans and diffusion models for text-to-image synthesis and manipulation. In *2024 IEEE International Students' Conference on Electrical, Electronics and Computer Science (SCEECS)*, pages 1–8, 2024.

[DCLK20] Ricard Durall, Avraam Chatzimichailidis, Peter Labus, and Janis Keuper. Combating mode collapse in GAN training: An empirical analysis using hessian eigenvalues. *CoRR*, abs/2012.09673, 2020.

[DDS+09] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[DN21] Prafulla Dhariwal and Alex Nichol. Diffusion models beat gans on image synthesis. *CoRR*, abs/2105.05233, 2021.

[DR22] Zhiwei Deng and Olga Russakovsky. Remember the past: Distilling datasets into addressable memories for neural networks, 2022.

[EEAmT22] Mohamed Elasri, Omar Elharrouss, Somaya Al-ma'adeed, and Hamid Tairi. Image generation: A review. *Neural Processing Letters*, 54, 03 2022.

[EKV23] Michael Elad, Bahjat Kawar, and Gregory Vaksman. Image denoising: The deep learning revolution and beyond – a survey paper –, 2023.

[Epo24] Epoch. Training computation (petaflop) [dataset]. `https://ourworldindata.org/grapher/training-computation-vs-dataset-size-in-notable-ai-systems-by-researcher-affiliation`, 2024. With major processing by Our World in Data. Original data from "Parameter, Compute and Data Trends in Machine Learning".

[GAA+17] Ishaan Gulrajani, Faruk Ahmed, Martín Arjovsky, Vincent Dumoulin, and Aaron C. Courville. Improved training of wasserstein gans. *CoRR*, abs/1704.00028, 2017.

[GBR+12] Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *The Journal of Machine Learning Research*, 13(1):723–773, 2012.

[Gei20] Amnon Geifman. Efficient Inference in Deep Learning—Where is the Problem? — medium.com. `https://medium.com/towards-data-science/efficient-inference-in-deep-learning-where-is-the-problem-4ad59434fe36`, 2020. [Accessed 02-02-2025].

[GPAM+14] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014.

[GVK+24] Jianyang Gu, Saeed Vahidian, Vyacheslav Kungurtsev, Haonan Wang, Wei Jiang, Yang You, and Yiran Chen. Efficient dataset distillation via minimax diffusion, 2024.

[GYMT20] Jianping Gou, Baosheng Yu, Stephen John Maybank, and Dacheng Tao. Knowledge distillation: A survey. *CoRR*, abs/2006.05525, 2020.

[GZB22] Chengcheng Guo, Bo Zhao, and Yanbing Bai. Deepcore: A comprehensive library for coreset selection in deep learning, 2022.

[Hao19] Karen Hao. Training a single AI model can emit as much carbon as five cars in their lifetimes — technologyreview.com. https://www.technologyreview.com/2019/06/06/239031/training-a-single-ai-model-can-emit-as-much-carbon-as-five-cars-in-their-lifetimes, 2019. [Accessed 02-02-2025].

[HHJ20] Juno Hwang, Wonseok Hwang, and Junghyo Jo. Tractable loss function and color image generation of multinary restricted boltzmann machine. *CoRR*, abs/2011.13509, 2020.

[HJA20] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *CoRR*, abs/2006.11239, 2020.

[HRU+17] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, Günter Klambauer, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a nash equilibrium. *CoRR*, abs/1706.08500, 2017.

[HVD15] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network, 2015.

[IZZE16] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks. *CoRR*, abs/1611.07004, 2016.

[JPTI21] Ali Jahanian, Xavier Puig, Yonglong Tian, and Phillip Isola. Generative models as a data source for multiview representation learning. *CoRR*, abs/2106.05258, 2021.

[KAD22] Youssef Kossale, Mohammed Airaj, and Aziz Darouichi. Mode collapse in generative adversarial networks: An overview. In *2022 8th International Conference on Optimization and Applications (ICOA)*, pages 1–6. IEEE, 2022.

[KALL17] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *CoRR*, abs/1710.10196, 2017.

[KKO+22] Jang-Hyun Kim, Jinuk Kim, Seong Joon Oh, Sangdoo Yun, Hwanjun Song, Joonhyun Jeong, Jung-Woo Ha, and Hyun Oh Song. Dataset condensation via efficient synthetic-data parameterization, 2022.

[KLA18] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. *CoRR*, abs/1812.04948, 2018.

[KLA+19] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. *CoRR*, abs/1912.04958, 2019.

[KNHa] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (canadian institute for advanced research).

[KNHb] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-100 (canadian institute for advanced research).

[KO18] Valentin Khrulkov and Ivan V. Oseledets. Geometry score: A method for comparing generative adversarial networks. *CoRR*, abs/1802.02664, 2018.

[KWI19] Animesh Karnewar, Oliver Wang, and Raghu Sesha Iyengar. MSG-GAN: multi-scale gradient GAN for stable image synthesis. *CoRR*, abs/1903.06048, 2019.

[LTOH22] Guang Li, Ren Togo, Takahiro Ogawa, and Miki Haseyama. Dataset distillation for medical dataset sharing, 2022.

[LTZQ19] Kanglin Liu, Wenming Tang, Fei Zhou, and Guoping Qiu. Spectral regularization for combating mode collapse in gans. *CoRR*, abs/1908.10999, 2019.

[LWY+22] Songhua Liu, Kai Wang, Xingyi Yang, Jingwen Ye, and Xinchao Wang. Dataset distillation via factorization, 2022.

[MG21] Karttikeya Mangalam and Rohin Garg. Overcoming mode collapse with adaptive multi adversarial training. *CoRR*, abs/2112.14406, 2021.

[MKKY18] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. *CoRR*, abs/1802.05957, 2018.

[MO14] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *CoRR*, abs/1411.1784, 2014.

[MR05] Sarvesh Makthal and Arun Ross. Synthesis of iris images using markov random fields. In *2005 13th European Signal Processing Conference*, pages 1–4, 2005.

[NNXL21] Timothy Nguyen, Roman Novak, Lechao Xiao, and Jaehoon Lee. Dataset distillation with infinitely wide convolutional networks. *CoRR*, abs/2107.13034, 2021.

[PEL+23] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis, 2023.

[Pen24] Yingying Peng. A comparative analysis between gan and diffusion models in image generation. *Transactions on Computer Science and Intelligent Systems Research*, 2024.

[Rad15] Alec Radford. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[RPG+21] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. *CoRR*, abs/2102.12092, 2021.

[RRMD23] Suman Ravuri, Mélanie Rey, Shakir Mohamed, and Marc Deisenroth. Understanding deep generative models with generalized empirical likelihoods, 2023.

[SCMG21] Axel Sauer, Kashyap Chitta, Jens Müller, and Andreas Geiger. Projected gans converge faster. *CoRR*, abs/2111.01007, 2021.

[SCS+22] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in neural information processing systems*, 35:36479–36494, 2022.

[SGZ+16] Tim Salimans, Ian J. Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *CoRR*, abs/1606.03498, 2016.

[SSG22] Axel Sauer, Katja Schwarz, and Andreas Geiger. Stylegan-xl: Scaling stylegan to large diverse datasets. *CoRR*, abs/2202.00273, 2022.

[SSK+20] Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *CoRR*, abs/2011.13456, 2020.

[TL19] Mingxing Tan and Quoc Le. EfficientNet: Rethinking model scaling for convolutional neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 6105–6114. PMLR, 09–15 Jun 2019.

[TSdC+18] 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. *CoRR*, abs/1812.05159, 2018.

[Wel09] Max Welling. Herding dynamical weights to learn. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, page 1121–1128, New York, NY, USA, 2009. Association for Computing Machinery.

[WGZ+23] Kai Wang, Jianyang Gu, Daquan Zhou, Zheng Zhu, Wei Jiang, and Yang You. Dim: Distilling dataset into generative model, 2023.

[WZP+22] Kai Wang, Bo Zhao, Xiangyu Peng, Zheng Zhu, Shuo Yang, Shuo Wang, Guan Huang, Hakan Bilen, Xinchao Wang, and Yang You. Cafe: Learning to condense dataset by aligning features, 2022.

[WZTE18] Tongzhou Wang, Jun-Yan Zhu, Antonio Torralba, and Alexei A. Efros. Dataset distillation. *CoRR*, abs/1811.10959, 2018.

[XGZ+21] Yuan Xue, Yuan-Chen Guo, Han Zhang, Tao Xu, Song-Hai Zhang, and Xiaolei Huang. Deep image synthesis from intuitive user input: A review and perspectives. *CoRR*, abs/2107.04240, 2021.

[YLW23] Ruonan Yu, Songhua Liu, and Xinchao Wang. Dataset distillation: A comprehensive review, 2023.

[YLY+18] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S. Huang. Generative image inpainting with contextual attention. *CoRR*, abs/1801.07892, 2018.

[ZB21] Bo Zhao and Hakan Bilen. Dataset condensation with differentiable siamese augmentation. *CoRR*, abs/2102.08259, 2021.

[ZB22] Bo Zhao and Hakan Bilen. Dataset condensation with distribution matching, 2022.

[ZCZG24] Tianlei Zhu, Junqi Chen, Renzhe Zhu, and Gaurav Gupta. Stylegan3: Generative networks for improving the equivariance of translation and rotation, 2024.

[Zha21] Kaifeng Zhang. On mode collapse in generative adversarial networks. In *Artificial Neural Networks and Machine Learning–ICANN 2021: 30th International Conference on Artificial Neural Networks, Bratislava, Slovakia, September 14–17, 2021, Proceedings, Part II 30*, pages 563–574. Springer, 2021.

[ZMB21] Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. Dataset condensation with gradient matching, 2021.

[ZNB22] Yongchao Zhou, Ehsan Nezhadarya, and Jimmy Ba. Dataset distillation using neural feature regression, 2022.

[ZRA23] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models, 2023.