UtilGen: Utility-Centric Generative Data Augmentation with Dual-Level Task Adaptation

Jiyu Guo[1] **Shuo Yang**[1*] **Yiming Huang**[1] **Yancheng Long**[1] **Xiaobo Xia**[2,3]
Xiu Su[4] **Bo Zhao**[5] **Zeke Xie**[6] **Liqiang Nie**[1]
[1]Harbin Institute of Technology, Shenzhen [2]National University of Singapore
[3]MoE Key Laboratory of Brain-inspired Intelligent Perception and Cognition,
University of Science and Technology of China
[4]Central South University [5]Shanghai Jiao Tong University
[6]Hong Kong University of Science and Technology (Guangzhou)
{ 220110126@stu.hit.edu.cn, shuoyang@hit.edu.cn }

Abstract

Data augmentation using generative models has emerged as a powerful paradigm for enhancing performance in computer vision tasks. However, most existing augmentation approaches primarily focus on optimizing intrinsic data attributes – such as fidelity and diversity – to generate visually high-quality synthetic data, while often neglecting task-specific requirements. Yet, it is essential for data generators to account for the needs of downstream tasks, as training data requirements can vary significantly across different tasks and network architectures. To address these limitations, we propose UTILGEN, a novel utility-centric data augmentation framework that adaptively optimizes the data generation process to produce task-specific, high-utility training data via downstream task feedback. Specifically, we first introduce a weight allocation network to evaluate the task-specific utility of each synthetic sample. Guided by these evaluations, UTILGEN iteratively refines the data generation process using a dual-level optimization strategy to maximize the synthetic data utility: (1) model-level optimization tailors the generative model to the downstream task, and (2) instance-level optimization adjusts generation policies – such as prompt embeddings and initial noise – at each generation round. Extensive experiments on eight benchmark datasets of varying complexity and granularity demonstrate that UTILGEN consistently achieves superior performance, with an average accuracy improvement of 3.87% over previous SOTA. Further analysis of data influence and distribution reveals that UTILGEN produces more impactful and task-relevant synthetic data, validating the effectiveness of the paradigm shift from visual characteristics-centric to task utility-centric data augmentation.

1 Introduction

Recent advances in generative models, particularly diffusion models [1, 2, 3, 4, 5, 6, 7], have significantly advanced data augmentation by enabling the creation of photorealistic images. Such text-to-image systems are capable of generating diverse and high-fidelity samples, and empirical evidence has shown their potential to enhance downstream model performance [8].

Current generative data augmentation approaches can be categorized into two main paradigms: *fidelity preservation* and *diversity enhancement*. The former employs techniques such as LoRA-based fine-tuning [9] to align synthetic data with real-world distributions [10, 11], while the latter employs

*Corresponding author.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

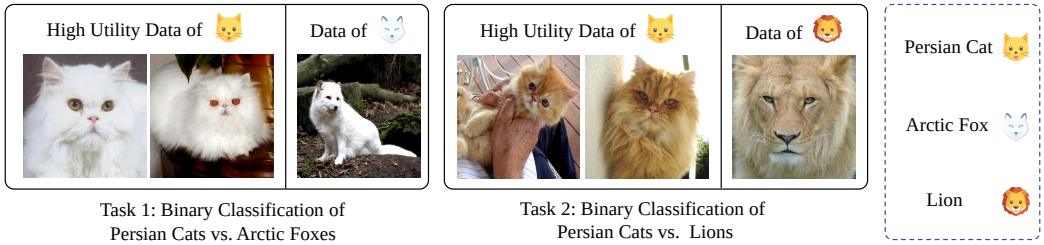


Figure 1: Comparison of high-utility samples within the same category (Persian cats) across two different tasks. White Persian cats (left) are more useful in Task 1, while golden ones (right) are more beneficial in Task 2, highlighting the diverse data requirements in different downstream tasks.

varied prompts or feature perturbations to enhance data diversity [12, 13]. Although effective in generating visually high-quality data, these methods solely optimize the intrinsic data attributes (e.g., fidelity and diversity), often struggling to directly optimize the task-specific utility of synthetic data. In practice, different tasks and model architectures may require distinct training data distributions for optimal performance [14], as exemplified in Figure 1. Despite this, most existing methods lack mechanisms to adapt data generation process based on the needs of specific downstream tasks. This limitation motivates our investigation into utility-centric data augmentation, in which synthetic data is explicitly optimized to enhance task performance rather than merely meet visual standards.

To go beyond the above limitation, an effective mechanism is needed to assess the task-specific utility of synthetic data, thereby providing explicit optimization signals to guide the data augmentation process. However, evaluating utility through full training and testing cycles is computationally prohibitive. Therefore, the core challenges in developing utility-centric data augmentation are: (1) *How to efficiently evaluate the task-specific utility of synthetic data without exhaustive training?* and (2) *How to systematically improve the task-specific utility of synthetic data?*

In this work, we propose UTILGEN, a novel utility-centric data augmentation framework, which can adaptively optimize the data generation process to produce task-specific, high-utility training data based on downstream task feedback. Specifically, we introduce *Task-Oriented Data Valuation*, which quantifies the task-specific utility of synthetic data through a meta-learned weight allocation network [15, 16, 17]. The network is optimized to minimize the classifier's validation loss by adaptively weighting the losses of training samples via estimation of their utility. The trained valuation network serves as an efficient utility predictor, enabling assessment of task-specific utility for newly generated samples without the need for costly retraining cycles. Guided by the utility signals, we employ an integrated dual-level optimization strategy: (1) *Model-Level Generation Capability Optimization* that tailors the data generator to downstream tasks through Direct Preference Optimization (DPO), and (2) *Instance-Level Generation Policy Optimization* that optimizes the generation policies (*i.e.*, prompt embedding and initial noise) to maximize the task-specific utility of synthetic data. Compared to previous advanced data augmentation methods which focus on optimizing intrinsic data characteristics, our proposed method achieves an average accuracy improvement of 3.87% across eight benchmark datasets. To the best of our knowledge, this is the first generative augmentation method where ResNet-50 [18] trained solely on $3\times$ synthetic data surpasses its real-data-trained counterpart on several benchmarks. Before delving into the details, we summarize our contributions as follows:

- Motivated by the observation that training data requirements differ across tasks and network architectures, we introduce a novel paradigm shift in data augmentation. Instead of focusing solely on optimizing intrinsic data attributes, we emphasize enhancing the task-specific utility of synthetic data. This utility-centric approach adaptively optimizes the generation process according to downstream task needs, enabling more targeted and effective data augmentation.
- To efficiently evaluate the task-specific utility of synthetic data, we introduce a meta-learned weight allocation network that measures the utility of synthetic data without requiring costly retraining. These utility signals drive a dual-level optimization framework that enhances both the model generation capability and the generation policies, resulting in high-utility synthetic data tailored to downstream tasks.
- Our method achieves state-of-the-art performance with an average improvement of 3.87% in accuracy across eight benchmarks, while also demonstrating exceptional versatility by delivering

consistent performance gains across diverse architectures (e.g., ResNeXt[19], WideResNet [20], and MobileNet[21]). Through training trajectory analysis, we validate the rationality of task utility measurement based on the weight network. Furthermore, analyses of data influence and distribution reveals that UTILGEN generates data with higher task relevance and stronger positive impact on model performance.

2 Related Work

2.1 Training Data Valuation

Understanding the role of training data in model performance is crucial for data-efficient learning [22, 23, 24, 25, 26] and optimizing model behavior [27, 28, 29, 30, 31]. To better understand how training data affects model behavior, many studies have aimed to quantitatively assess the influence of individual examples on model performance [32, 33, 34, 35, 36, 37, 38, 39]. Existing approaches for evaluating data value can be classified into two categories: (1) Retraining-based approaches, such as Data Shapley [33, 34, 40, 41] and C-score [35], which quantify data influence through expensive model retraining across different training subsets. For the utility evaluation of large-scale synthetic datasets, these methods become computationally prohibitive due to their inherent complexity. (2) Gradient-based methods [32, 42, 43, 44] that estimate data influence by analyzing gradient interactions between training and test points, either through static snapshot analysis or dynamic trajectory examination. Although these approaches avoid model retraining, they still incur significant computational overhead, particularly when performing complex operations such as Hessian matrix inversion [45]. To evaluate the utility of synthetic data, our method employs a weight allocation network [15, 16, 17] to efficiently assess data utility, avoiding the costly retraining or complex computations required by earlier approaches.

2.2 Training Data Augmentation

The availability of high-quality training data has been fundamental to the success of deep learning, enabling models to capture complex patterns, learn meaningful representations, and generate accurate predictions [46, 47, 48, 49, 50]. The methodology of training data augmentation has progressed from traditional techniques to advanced generative approaches [51, 10, 11, 52, 12, 53, 54]. Traditional methods, such as mixup [55, 56], erasing [57, 58], and cropping [59], commonly rely on predefined transformations to augment dataset diversity. However, they are inherently limited to local pixel-level modifications. While Generative Adversarial Networks (GANs) [60] enabled synthetic image generation, they often face challenges in maintaining semantic consistency and distribution alignment [61, 62]. Recent advances in diffusion models, such as Stable Diffusion [4] and GLIDE [63], have demonstrated superior capabilities in generating synthetic data. To enhance diversity, methods such as GIF [12], ALIA [52], and DISEF [13] employ varied prompts or feature perturbations in the latent space. Meanwhile, techniques like RealFake [10], DistDiff [64], and DataDream [11] focus on improving image fidelity by aligning synthetic data distributions with the target domain. Nevertheless, these methods primarily address intra-class distribution alignment without evaluating which types of data better support downstream tasks. In contrast, our approach adaptively tailors the generation process to downstream tasks, producing high-utility data specifically optimized for target applications.

3 Method

In this section, the framework of our utility-centric generative data augmentation system is presented, as illustrated in Fig. 2. Specifically, the proposed approach comprises three key components: (1) *Task-Oriented Data Valuation (Sec. 3.1)*, which quantitatively assesses the utility of synthetic data for downstream tasks; (2) *Model-Level Generation Capability Optimization (Sec. 3.2)*, which tailors the generative model to align with the training data preferences of the downstream task via DPO; and (3) *Instance-Level Generation Policy Optimization (Sec. 3.3)*, which adapts generation policy to maximize the task-specific utility of the synthetic data.

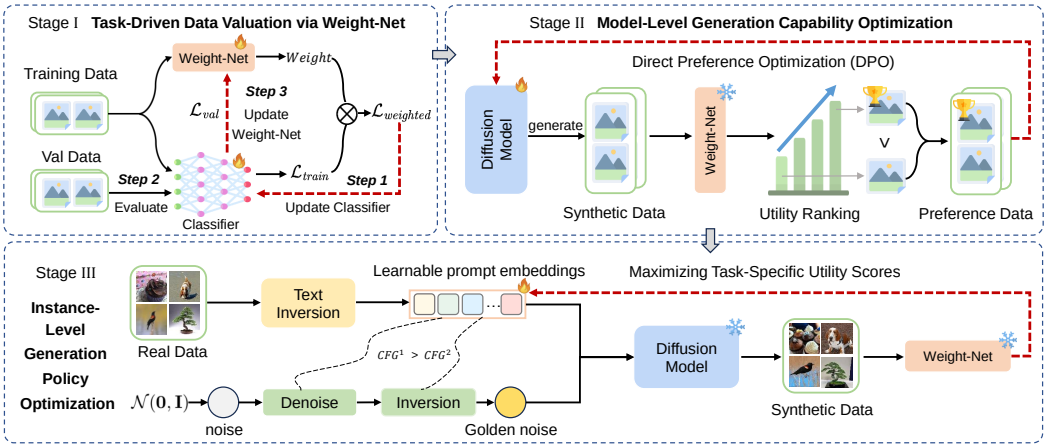


Figure 2: The UTILGEN framework for feedback-driven data augmentation, comprising three key stages: (1) *Task-Oriented Data Valuation* (Sec. 3.1); (2) *Model-Level Generation Capability Optimization* (Sec. 3.2); (3) *Instance-Level Generation Policy Optimization* (Sec. 3.3).

3.1 Task-Oriented Data Valuation (TODV)

It is noted that individual training instances exhibit heterogeneous influences on model performance during the training process [32], with certain data samples potentially introducing negative influences. This observation has motivated a line of research to mitigate model overfitting to data samples with negative influences, such as training sample re-weighting strategies implemented through learnable weight networks[15, 16, 17]. Moreover, it is recognized that these learned weights can be implicitly interpreted as indicators of each sample's utility for downstream tasks [15]. Drawing inspiration from this insight, we propose *Task-Oriented Data Valuation*, which employs a weight network trained via meta-learning to quantitatively assess the utility of synthetic data for downstream tasks.

Specially, given a classifier f_θ, the weight ω_i assigned to each sample x_i is derived through a loss-based process as follows:

$$\omega_i = \mathcal{W}_\phi\left(\mathcal{L}(f(x_i;\theta), y_i)\right), \tag{1}$$

where $\mathcal{W}_\phi$ denotes an MLP network with a single hidden layer. This network is trained to predict normalized weights within the range $[0, 1]$, where higher values reflect samples with greater utility.

To develop such a weight network $\mathcal{W}_\phi$ capable of measuring task-specific data utility, we adopt a bi-level optimization strategy comprising two iterative steps:

Classifier training: To enhance the generalization capability of the weight network and mitigate distribution shift in subsequent generation optimization stages, we first perform textual inversion [65] to learn class-specific identifiers $[I_i]$ from a small set of real images per class. Using these learned identifiers, we generate synthetic data $\mathcal{D}_g$ with class-conditioned prompts c_i of the form "a photo of $[I_i]$". The synthetic data is then combined with the real training data to form a merged dataset $\mathcal{D}_{\text{merge}} = (\mathcal{D}_r \cup \mathcal{D}_g)$, where $\mathcal{D}_r$ denotes the real data. The classifier parameters θ are optimized using a *weighted loss* computed over $\mathcal{D}_{\text{merge}} = \{(x_i, y_i)\}_{i=1}^N$:

$$\theta^*(\phi) = \arg\min_\theta \frac{1}{N}\sum_{i=1}^{N} \omega_i \mathcal{L}(f(x_i;\theta), y_i), \tag{2}$$

Weight network training: Given the classifier's updated parameters, the weight network parameters ϕ are trained to minimize the loss on the validation set $\mathcal{D}_v = \{(x_j, y_j)\}_{j=1}^M$:

$$\phi^*(\theta) = \arg\min_\phi \frac{1}{M}\sum_{j=1}^{M} \mathcal{L}\left(f(x_j;\theta^*(\phi)), y_j\right). \tag{3}$$

This bi-level optimization framework establishes a dynamic feedback loop between data valuation and model training. By quantifying data utility through learned weights, the trained weight network

is subsequently used to guide the optimization of the data generation process. The full procedure for TODV is described in Algorithm 1.

Algorithm 1 Task-Oriented Data Valuation (TODV)

Input: Training data $\mathcal{D}_r \cup \mathcal{D}_g = \{(x_i, y_i)\}_{i=1}^N$; validation data $\mathcal{D}_v = \{(x_j, y_j)\}_{j=1}^M$
Required: Classifier parameters θ ; weight network parameters ϕ; max iteration T.

1: Initialize $\theta^{(0)}$, $\phi^{(0)}$ randomly
2: **for** t = 0 to $T-1$ **do**
3: Sample batch $\{(x_i, y_i)\}_{i=1}^n$ from $\mathcal{D}_r \cup \mathcal{D}_g$
4: Sample batch $\{(x_j, y_j)\}_{j=1}^m$ from $\mathcal{D}_v$
5: Predict weights $\{\omega_i^{(t)}\}_{i=1}^n$ for the batch $\{(x_i, y_i)\}_{i=1}^n$ that reflect their task-specific utility (Eq. 1)
6: Update $\theta^{(t+1)}$ using the loss weighted by $\{\omega_i^{(t)}\}_{i=1}^n$ (Eq. 2)
7: Update $\phi^{(t+1)}$ (Eq. 3)
8: **end for**

Output: Optimized weight network parameter ϕ^T

3.2 Model-Level Generation Capability Optimization (MLCO)

Although standard diffusion models demonstrate remarkable capabilities in generating visually high-quality images, their outputs often fail to meet the specific data requirements of downstream applications. To address this misalignment, we propose an iterative DPO framework that adapts the generative model to downstream task-specific data preferences.

Each optimization cycle begins by prompting the diffusion model with prompts c_i of the form "a photo of $[I_i]$", where class-specific identifiers $[I_i]$ are obtained via textual inversion [65], to generate a synthetic dataset $\mathcal{D}_{\text{syn}} = \{(x_i)\}_{i=1}^M$. The pre-trained weight network $\mathcal{W}_\phi$ subsequently evaluates each sample's utility via the weight score $\omega_i = \mathcal{W}_\phi(\mathcal{L}(f(x_i;\theta), y_i))$, where $\mathcal{L}(f(x_i;\theta), y_i)$ is the loss of the classifier f_θ on sample x_i with its label y_i. Based on these scores, high-utility samples x_i^w and low-utility samples x_i^l are paired to construct a preference dataset:

$$\mathcal{D}_{\text{preference}} = \{(c_i, x_i^w, x_i^l) \mid \mathcal{W}_\phi(\mathcal{L}(f(x_i^w;\theta), y_i^w)) > \mathcal{W}_\phi(\mathcal{L}(f(x_i^l;\theta), y_i^l))\}_{i=1}^N, \tag{4}$$

We then use the preference dataset $\mathcal{D}_{\text{preference}}$ to fine-tune the diffusion model's U-Net ψ using DPO, with the optimization objective formulated according to the Diffusion DPO [66].

$$\begin{aligned}\mathcal{L}_{\text{DPO}}(\psi) = -\,\mathbb{E}_{(x_0^w, x_0^l)\sim\mathcal{D}_{\text{preference}}, t\sim\mathcal{U}(0,T), x_t^w\sim q(x_t^w|x_0^w), x_t^l\sim q(x_t^l|x_0^l)} \\ \left[\log\sigma\left(-\beta T\omega(\lambda_t)\left(\Delta\mathcal{L}_w - \Delta\mathcal{L}_l\right)\right)\right],\end{aligned} \tag{5}$$

$$\begin{aligned}\Delta\mathcal{L}_w &= \|\epsilon^w - \epsilon_\psi(x_t^w, t)\|^2 - \|\epsilon^w - \epsilon_{\text{ref}}(x_t^w, t)\|^2 \\ \Delta\mathcal{L}_l &= \|\epsilon^l - \epsilon_\psi(x_t^l, t)\|^2 - \|\epsilon^l - \epsilon_{\text{ref}}(x_t^l, t)\|^2.\end{aligned} \tag{6}$$

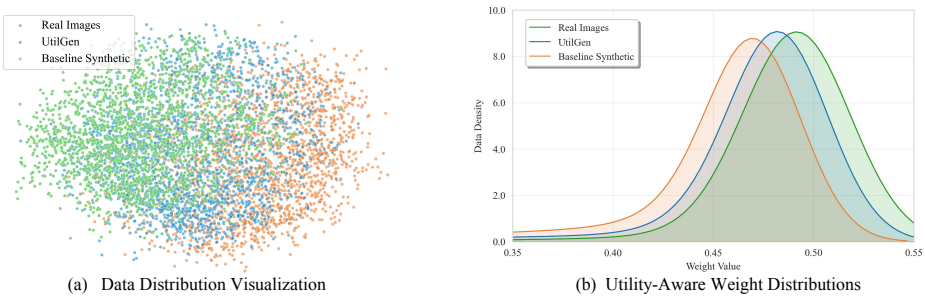
Here, ϵ_ψ and ϵ_{ref} denote the noise predictions from the trainable and reference U-Nets, respectively. The forward diffusion process $q(x_t|x_0)$ adds noise to x_0 at timestep t, where t is sampled from $\mathcal{U}(0,T)$. β balances preference alignment and KL regularization, while $\sigma(\cdot)$ is the sigmoid activation in the loss. λ_t is the signal-to-noise ratio [67] and $\omega(\lambda_t)$ is weighting function [1].

Through iterative DPO fine-tuning, we progressively adapt the diffusion model's generative capability according to the downstream task's preferences for the training data. This process enables better alignment between the model's output distribution and the target application requirements.

3.3 Instance-Level Generation Policy Optimization (ILPO)

While MLPO tailors the generative model to the downstream task at a coarse level, ILPO performs fine-grained refinement of the generation policy by jointly optimizing the prompt embeddings and the initial noise. The overall optimization objective is formulated as:

$$(p^*, \epsilon_T^*) = \arg\max_{p,\epsilon_T} \mathbb{E}\left[\mathcal{W}_\phi(\mathcal{L}(f(g(p,\epsilon_T);\theta), y))\right], \tag{7}$$



(a) Data Distribution Visualization (b) Utility-Aware Weight Distributions

Figure 3: (a) Feature space visualization on the Flower dataset [68]shows that our synthetic data achieves closer alignment with the real data distribution compared to vanilla Stable Diffusion. (b) Utility-aware weight distributions for synthetic and real data on the Flower dataset [68], showing sample utility scores for downstream tasks.

Algorithm 2 Instance-Level Generation Policy Optimization (ILPO)

Input: K classes: $\{c_1, ..., c_K\}$; number of images to generate for each class: $\{N_1, \ldots, N_K\}$, where N_k is the number of samples to generate for class c_k

Required: Diffusion model $g(\cdot, \cdot)$ fine-tuned by MLCO; Textual Inversion technology $TI(\cdot, \cdot)$ using few-shot real images per class $\mathcal{D}_k = \{x_1^k, ..., x_N^k\}$ for each c_k;

for each class $k \in \{1, ..., K\}$ **do**
  // Prompt Embedding Optimization
  Initialize $p_k \leftarrow TI(c_k, \mathcal{D}_k)$
  $p_k^* \leftarrow p_k$ (Eq. 8) ▷ Optimize to maximize data utility
  // Noise Optimization
  for $i = 1$ to N_k **do**
    Sample noise $\epsilon_T \sim \mathcal{N}(0, I)$
    $\epsilon_T' \leftarrow \epsilon_T$ by (Eq. 9) ▷ Inject semantic information representing high-utility data
    $x_i' \leftarrow g(p_k^*, \epsilon_T')$ ▷ Generate high-utility data
    $\mathcal{D}_{\text{synth}} \leftarrow \mathcal{D}_{\text{synth}} \cup \{x_i'\}$
  end for
end for

Output: High-utility synthetic dataset $\mathcal{D}_{\text{synth}}$

where p denotes the prompt embedding, ϵ_T represents the initial noise vector, $g(\cdot)$ is the diffusion model, $f(\cdot; \theta)$ is the downstream classifier, y is the ground-truth label, and $\mathcal{L}(\cdot)$ is the classification loss. The optimization process consists of two synergistic components:

Prompt embedding optimization: Building upon the class-specific identifiers $[I_i]$ learned through textual inversion [65], we optimize the prompt embeddings $p_i = E(\text{“a photo of } [I_i]\text{”})$ using gradient-based optimization, where $E(\cdot)$ denotes the text encoder. This process aims to maximize the utility score predicted by the pre-trained weight network. To preserve semantic alignment during optimization, a CLIP-based regularization term $L_{\text{CLIP}} = -\cos(E(x_i), e_i)$ is applied, where x_i is the generated image, $E(\cdot)$ is the CLIP image encoder and $e_i = \text{avg}(E(\{x_j\})$ is the mean embedding of a set of target-class real images $\{x_j\}$. The complete prompt optimization objective is:

$$p^* = \arg\max_p \left[\mathcal{W}_\phi(\mathcal{L}(f(g(p, \epsilon_T); \theta), y)) - \lambda L_{\text{CLIP}}\right], \tag{8}$$

where λ controls the regularization strength. This joint optimization enhances sample utility while preserving semantic coherence with the original class concept.

Noise optimization: The quality of synthetic images is influenced by both the text prompt and the random Gaussian noise. Yet, since each image generation requires independently sampled noise, directly optimizing noise vectors via gradient ascent to improve utility is computationally prohibitive. Recent studies [69, 70, 71] show that the discrepancy between denoising and inversion Classifier-Free Guidance (CFG) scales can be leveraged to implicitly inject prompt semantic information into the initial noise. Leveraging this observation, we adapt the methodology to optimize the initial noise, enabling it to incorporate semantic information of high-utility data. Formally, the noise optimization

Table 1: Classification performance across eight datasets using ResNet-50 [18] as the backbone classifier. UTILGEN and DataDream [11] generate synthetic data guided by **16-shot real images** per class, while other methods use the full real dataset as guidance for generation. The top block shows results using synthetic data only, while the bottom includes joint training (synthetic + full real dataset). Each method produces synthetic data at 5× the scale of the original real dataset.

Method	Real	Syn	IN-1k-S	IN-100-S	Cal101	DTD	CUB	PETs	Food-S	Flowers	Avg
Training on Synthetic Data Only											
SD v2.1 [4]		✓	24.35	27.96	14.74	7.92	23.43	26.05	24.02	31.41	22.49
GIF [12]		✓	28.95	31.94	20.39	13.19	27.54	29.73	25.94	56.12	29.23
GAP [72]		✓	25.84	30.94	18.70	11.01	29.49	27.46	25.31	53.66	27.80
DataDream [11]		✓	30.35	35.48	23.61	13.24	35.38	34.77	28.41	65.15	33.30
UtilGen		✓	**33.72**	**40.94**	**29.31**	**13.52**	**43.32**	**37.25**	**31.87**	**67.43**	**37.17**
△ *over previous SOTA*			**+3.37**	**+5.46**	**+5.70**	**+0.28**	**+7.94**	**+2.48**	**+3.46**	**+2.28**	**+3.87**
Joint Training with Real Data											
Real Dataset	✓		36.34	38.58	43.55	16.32	21.03	28.78	19.88	73.34	34.73
SD v2.1 [4]	✓	✓	43.26	49.86	59.35	28.82	43.52	52.35	40.32	79.58	49.63
GIF [12]	✓	✓	49.85	54.12	67.60	33.45	43.80	58.21	41.39	84.47	54.11
GAP [72]	✓	✓	46.58	53.14	66.97	32.87	47.46	57.86	43.77	84.84	54.19
DataDream [11]	✓	✓	52.16	57.68	73.38	34.84	53.43	60.83	47.44	89.60	58.67
UtilGen	✓	✓	**54.56**	**61.54**	**75.62**	**36.06**	**57.53**	**64.64**	**52.72**	**93.62**	**62.04**
△ *over real dataset*			**+18.22**	**+22.96**	**+32.07**	**+19.74**	**+36.50**	**+35.86**	**+32.84**	**+20.28**	**+27.31**
△ *over previous SOTA*			**+2.40**	**+3.86**	**+2.24**	**+1.22**	**+4.10**	**+3.81**	**+5.28**	**+4.02**	**+3.37**

process is defined as:

$$\epsilon'_t = \text{DDIM-Inversion}_{\omega_w}(\text{DDIM}_{\omega_l}(\epsilon_t, p^*)). \tag{9}$$

where ω_l and ω_w are the CFG scales for the denoising process $\text{DDIM}(\cdot)$ and the inversion process $\text{DDIM-Inversion}(\cdot)$, respectively. The condition $\omega_l > \omega_w$ enables the implicit injection of semantic information into the initial noise. The entire process of ILPO is outlined in Algorithm 2.

The synergistic integration of MLCO (Sec.3.2) and ILPO (Sec.3.3) enables UTILGEN to synthesize data that closely aligns with real data feature distributions, as visualized in Fig.3(a). This approach simultaneously achieves higher utility scores (Fig.3(b)), demonstrating enhanced task relevance and superior data utility in downstream applications.

4 Experiments

4.1 Experimental Setup

Benchmarks. We evaluate the effectiveness of UTILGEN across eight datasets spanning three classification tasks: coarse-grained classification (ImageNet-1k-Subset [73], ImageNet-100-Subset [73], and Caltech 101 [74]), fine-grained classification (Oxford Pets [75], Food-S [76], Flowers 102 [68], and CUB-200-2011 [77]), and texture classification (DTD [78]). Specifically, ImageNet-1k-Subset and Food-S are subsets of ImageNet-1K and Food101 [76], respectively, each containing 100 randomly selected images per class. ImageNet-100-Subset is constructed by randomly sampling 100 animal-related classes from the original ImageNet-1K [73], with 100 randomly selected images per class. Further benchmark details are provided in Appendix F.

Baselines. To compare our utility-centric approach with existing methods that focus on optimizing intrinsic data characteristics, we select GIF [12] and DataDream [11] as representative baselines. Specifically, GIF [12] enhances diversity by applying feature perturbations, while DataDream [11] improves fidelity through domain alignment using LoRA fine-tuning of the diffusion model. Additionally, we include GAP [72], which uses feedback from the downstream model to generate adversarial prompts that maximize the model's loss on the generated images. This enables a direct comparison between its loss-based feedback strategy and the utility-based feedback mechanism employed by UTILGEN. For fair comparison, we adopt Stable Diffusion v2.1 [4] (SD v2.1) as the backbone generator across all baseline methods. The implementation details of UTILGEN are in Appendix B.

Figure 4: Comparison of synthetic images generated by SD v2.1, GIF [12], GAP [72], DataDream [11], and UTILGEN.

Table 2: Performance comparison under different synthesis budgets on ImageNet-100. UTILGEN scales more effectively with increasing synthetic data ratios.

Budget	SD v2.1	GIF [12]	GAP [72]	DataDream [11]	UTILGEN
1×	12.18	13.44	15.02	17.12	**18.04**
3×	20.20	21.90	21.16	25.26	**28.52**
5×	27.96	31.94	30.94	35.48	**40.94**

Table 3: Comparison of intra-class diversity of synthetic data on ImageNet-100. Higher values indicate greater diversity.

Method	Mean Intra-Class Diversity ↑
Stable Diffusion v2.1	0.5815
DataDream [11]	0.5238
UTILGEN	**0.6054**

4.2 Evaluation Results

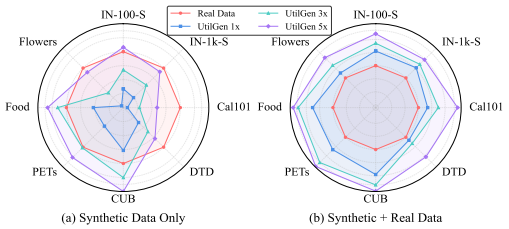
Results on solely synthetic data. As shown in Table 1 (top), UTILGEN achieves the highest average accuracy of 37.17% when trained solely on synthetic data, outperforming the previous best method DataDream (33.30%) by a notable margin of +3.87%. It demonstrates strong performance across both coarse-grained (e.g., 40.94% on IN-100-S [73]) and fine-grained tasks (e.g., 67.43% on Flowers [68]), indicating excellent generalization despite training solely on synthetic data.

Results on real + synthetic Data. In the joint training setting (bottom of Table 1), UTILGEN maintains its lead with an average accuracy of 62.04%, surpassing DataDream (58.67%) by +3.54%. It achieves particularly strong gains on fine-grained datasets (e.g., 93.62% on Flowers [68]), while also delivering consistent improvements on coarse-grained tasks. These results suggest that UTILGEN can effectively complement real data across different task granularities, providing high-utility synthetic samples that enhance model performance and robustness.

Synthetic data scaling effects. UTILGEN exhibits strong scalability and data augmentation efficiency. As shown in Figure 5, scaling synthetic data from 1× to 5× the original set consistently improves ResNet-50 [18] performance across benchmarks under both synthetic-only and joint training settings. When trained with 3× synthetic data alone, models outperform their real-data-trained counterparts on three datasets; at 5× scaling, this advantage extends to four datasets. To further examine how performance scales with the synthesis budget, we compare UTILGEN with four representative baselines under budgets. As shown in Table 2, UTILGEN consistently achieves the best results across all budgets, and the performance gap widens as the synthesis ratio increases, demonstrating superior scalability under large-scale generation. Additional details on augmentation efficiency and computational cost are provided in Appendix D.

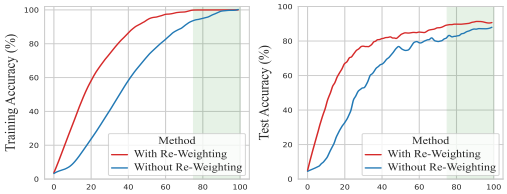
Diversity analysis of synthetic data. We evaluate the diversity of synthetic samples by computing the mean intra-class cosine distance of CLIP (ViT-L/14) features on ImageNet-100. Compared methods include vanilla Stable Diffusion v2.1, DataDream [11], and UTILGEN. Higher values indicate greater intra-class diversity, which typically benefits generalization. As shown in Table 3, UTILGEN achieves the highest intra-class diversity, indicating that its utility-guided optimization preserves sample variety and avoids mode collapse.

Reusability of synthetic data across tasks. We further analyze the reusability of UTILGEN-generated data across different downstream models on the ImageNet-100 dataset. Even when the weight network is trained using ResNet-50 , the resulting synthetic data generalizes effectively to other models such as WideResNet and CLIP. As shown in Table 4, the performance gains remain



(a) Synthetic Data Only (b) Synthetic + Real Data

Figure 5: Synthetic Data Scaling Effects across different training data regimes. (a) Models trained exclusively on synthetic data (Synthetic Data Only). (b) Models trained on combined synthetic and real data (Synthetic + Real Data).



(a) Training Accuracy Comparison (b) Test Accuracy Comparison

Figure 6: Training and test accuracy comparison with and without the weight network using ResNet-50 [18] on the dataset (original Flowers + SD v2.1 augmented data with 1× expansion). (a) Training convergence; (b) Test accuracy.

Table 4: Reusability of synthetic data across downstream models on ImageNet-100.

Classifier used to train weight network	Downstream model using synthetic data	Method	Acc. (%)
–	WideResNet	DataDream [11]	31.76
ResNet-50	WideResNet	UtilGen	**36.40**
–	CLIP	DataDream [11]	71.42
ResNet-50	CLIP	UtilGen	**72.14**

Table 5: Generalization performance across diverse network architectures on ImageNet-100.

Method	ResNeXt-50	WideResNet-50	MobileNetV2
GIF [12]	27.54	27.84	31.24
GAP [72]	27.66	27.76	32.72
DataDream [11]	31.24	31.76	35.48
UtilGen	**37.62**	**37.82**	**40.59**

consistent across architectures, indicating that high-utility samples identified by UtilGen are not tied to a specific model and can be reused for diverse learning objectives.

Generalization across different architectures. To assess the versatility of UtilGen, we evaluate its performance on three architectures: ResNeXt-50 [19], WideResNet-50 [20], and MobileNetV2 [21]. As shown in Table 5, UtilGen consistently achieves the highest accuracy, with 500 images generated per class, confirming the effectiveness of our approach across various network architectures.

Cost-benefit analysis compared to collecting more labeled data. Compared to manual data collection and annotation, UtilGen offers a highly cost-effective solution for dataset expansion. According to the Masterpiece Group[2], manually annotating 10,000 images (e.g., 100 images per class across 100 classes) typically takes about two weeks and costs approximately $800. In contrast, generating the same amount of data using UtilGen requires only about 0.94 hours and $20 on 8 V100 GPUs rented from Google Cloud[3]. On the ImageNet-100-Subset dataset, using 5× synthetic data produced by UtilGen even surpasses the real-data baseline in accuracy, while requiring only ∼4.7 hours and about $100 in compute cost, as shown in Table 6.

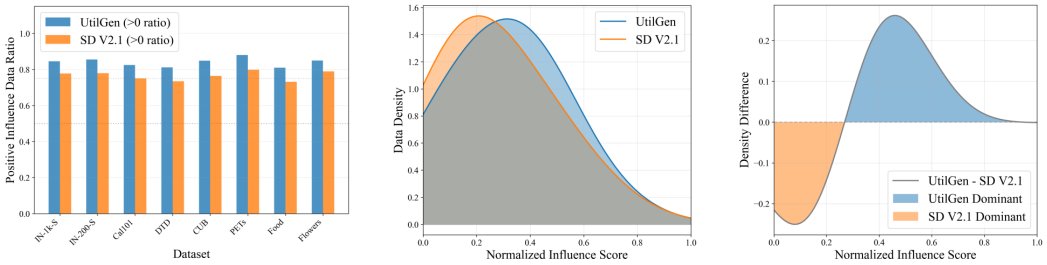
Ablation study. Table 7 presents an ablation study on the IN-100-S dataset using ResNet-50 trained solely on synthetic data (500 aasssper class). We analyze the effects of MLCO and ILPO (including prompt embedding and initial noise optimization). Each component brings improvements over the baseline, and combining all three achieves the best performance, surpassing the baseline by +12.98%, demonstrating the complementary strengths of model-level and instance-level optimizations.

4.3 Mechanism Analysis

Effect of sample re-weighting on classifier training. To assess the effect of the weight allocation network on classifier training, we compare the training trajectories of models with and without dynamic weighting. As shown in Figure 6(a), the weighted model converges faster and achieves higher training accuracy in earlier epochs, suggesting it effectively assigns higher weights to high-utility samples during learning. Figure 6(b) presents the test accuracy curves, where accuracy rises more rapidly and achieves a final improvement of 2.94% over the baseline. This gain is attributed to the network's ability to down-weight low-utility or noisy samples, thereby mitigating their negative

[2] https://mpg-myanmar.com/annotation/

[3] https://cloud.google.com/compute/gpus-pricing



(a) Positive Influence Ratio (b) Data Influence Distribution (c) Data Distribution Difference

Figure 7: Data influence comparison between UTILGEN and SD v2.1, computed using influence functions. (a) Proportion of samples with positive influence scores (influence > 0) across eight datasets. (b) Influence score distributions showing that UTILGEN generates a higher density of samples with stronger influence. (c) Data density difference plot highlighting that UTILGEN dominates in high-influence regions, while SD v2.1 contributes more to low-influence areas.

Table 6: Comparison of the manual annotation paradigm and our synthetic data paradigm in terms of cost, time, and performance.

Method	Images	Time Required	Cost	Acc. (%)
Manual Annotation	10,000	∼2 weeks	$800	38.58
UTILGEN (1×)	10,000	∼0.94 h	$20	18.04
UTILGEN (5×)	50,000	∼4.70 h	$100	**40.94**

Table 7: Ablation study on ImageNet-100.

MLCO	Prompt Optimization	Noise Optimization	Acc. (%)
			27.96
		✓	28.68
✓			36.42
	✓		37.96
✓		✓	32.08
	✓	✓	39.12
✓	✓		39.73
✓	✓	✓	**40.94**

impact. These results confirm that the pre-trained weight allocation network can reliably measure sample utility and serve as a signal to guide synthetic data generation.

Data influence analysis. We measure data influence by jointly training models on synthetic datasets generated by UTILGEN and the baseline SD v2.1, applying Influence Function [32]. Figure 7(a) reveals that UTILGEN consistently produces a higher proportion of positively influential samples (influence > 0) across all benchmarks. Furthermore, the influence score distribution shown in Figure 7(b) shifts noticeably to the right, indicating that UTILGEN generates more samples with stronger positive influence. Complementing this, Figure 7(c) highlights that UTILGEN achieves substantially higher data density within high-influence regions, while simultaneously reducing sample concentration in low-utility areas. Together, these results validate the effectiveness of our method in synthesizing impactful data that better supports downstream model optimization.

5 Conclusion

In this study, we propose UTILGEN, a utility-centric data augmentation framework that shifts the focus from optimizing intrinsic visual properties to enhancing task-specific utility. By incorporating downstream model feedback, UTILGEN adaptively adjusts the data generation process to produce high-utility data tailored for specific downstream tasks, thereby establishing a feedback loop between data generation and model training. Experiments across eight benchmarks demonstrate consistent performance gains, and in certain cases surpassing the performance of models trained solely on real data. These results highlight the superiority of the utility-centric approach over prior methods focusing primarily on intrinsic visual quality. These findings underscore the potential of utility-centric generation and suggest that integrating task-specific utility alongside traditional visual quality considerations offers a more effective paradigm for future data augmentation research.

Acknowledgment

The authors would like to acknowledge the support from the following funding sources. Shuo Yang is supported by the National Natural Science Foundation of China Young Scientists Fund

(No. 62506096). Xiaobo Xia is partially supported by the MoE Key Laboratory of Brain-inspired Intelligent Perception and Cognition at the University of Science and Technology of China (Grant No. 2421002). Bo Zhao is supported by the National Natural Science Foundation of China (Grant No. 62306046).

References

[1] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

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

[3] Zhaoqing Wang, Xiaobo Xia, Runnan Chen, Dongdong Yu, Changhu Wang, Mingming Gong, and Tongliang Liu. Lavin-dit: Large vision diffusion transformer. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 20060–20070, 2025.

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

[5] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

[6] Run Luo, Yunshui Li, Longze Chen, Wanwei He, Ting-En Lin, Ziqiang Liu, Lei Zhang, Zikai Song, Xiaobo Xia, Tongliang Liu, et al. Deem: Diffusion models serve as the eyes of large language models for image perception. *arXiv preprint arXiv:2405.15232*, 2024.

[7] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

[8] Ruifei He, Shuyang Sun, Xin Yu, Chuhui Xue, Wenqing Zhang, Philip Torr, Song Bai, and Xiaojuan Qi. Is synthetic data from generative models ready for image recognition? *arXiv preprint arXiv:2210.07574*, 2022.

[9] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022.

[10] Jianhao Yuan, Jie Zhang, Shuyang Sun, Philip Torr, and Bo Zhao. Real-fake: Effective training data synthesis through distribution matching. *arXiv preprint arXiv:2310.10402*, 2023.

[11] Jae Myung Kim, Jessica Bader, Stephan Alaniz, Cordelia Schmid, and Zeynep Akata. Datadream: Few-shot guided dataset generation. In *European Conference on Computer Vision*, pages 252–268. Springer, 2024.

[12] Yifan Zhang, Daquan Zhou, Bryan Hooi, Kai Wang, and Jiashi Feng. Expanding small-scale datasets with guided imagination. *Advances in neural information processing systems*, 36:76558–76618, 2023.

[13] Victor G Turrisi da Costa, Nicola Dall'Asen, Yiming Wang, Nicu Sebe, and Elisa Ricci. Diversified in-domain synthesis with efficient fine-tuning for few-shot classification. *arXiv preprint arXiv:2312.03046*, 2023.

[14] Daochen Zha, Zaid Pervaiz Bhat, Kwei-Herng Lai, Fan Yang, Zhimeng Jiang, Shaochen Zhong, and Xia Hu. Data-centric artificial intelligence: A survey. *ACM Computing Surveys*, 57(5):1–42, 2025.

[15] Jun Shu, Qi Xie, Lixuan Yi, Qian Zhao, Sanping Zhou, Zongben Xu, and Deyu Meng. Meta-weight-net: Learning an explicit mapping for sample weighting. *Advances in neural information processing systems*, 32, 2019.

[16] Jun Shu, Xiang Yuan, Deyu Meng, and Zongben Xu. Cmw-net: Learning a class-aware sample weighting mapping for robust deep learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(10):11521–11539, 2023.

[17] Zizhao Zhang and Tomas Pfister. Learning fast sample re-weighting without reward data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 725–734, 2021.

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

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

[20] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

[21] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.

[22] Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. Dataset condensation with gradient matching. *arXiv preprint arXiv:2006.05929*, 2020.

[23] Bo Zhao and Hakan Bilen. Dataset condensation with distribution matching. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 6514–6523, 2023.

[24] Xiaofeng Cao, Weiyang Liu, and Ivor W Tsang. Data-efficient learning via minimizing hyperspherical energy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(11):13422–13437, 2023.

[25] Shuo Yang, Zhe Cao, Sheng Guo, Ruiheng Zhang, Ping Luo, Shengping Zhang, and Liqiang Nie. Mind the boundary: Coreset selection via reconstructing the decision boundary. In *Forty-first International Conference on Machine Learning*, 2024.

[26] Shuo Yang, Zhaopan Xu, Kai Wang, Yang You, Hongxun Yao, Tongliang Liu, and Min Xu. Bicro: Noisy correspondence rectification for multi-modality data via bi-directional cross-modal similarity consistency. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19883–19892, 2023.

[27] Ruiheng Zhang, Biwen Yang, Lixin Xu, Yan Huang, Xiaofeng Xu, Qi Zhang, Zhizhuo Jiang, and Yu Liu. A benchmark and frequency compression method for infrared few-shot object detection. *IEEE Transactions on Geoscience and Remote Sensing*, 2025.

[28] Xiaofeng Cao, Yaming Guo, Heng Tao Shen, Ivor W Tsang, and James T Kwok. Mentored learning: Improving generalization and convergence of student learner. *Journal of Machine Learning Research*, 25(325):1–45, 2024.

[29] Rui Huang, Shitong Shao, Zikai Zhou, Pukun Zhao, Hangyu Guo, Tian Ye, Lichen Bai, Shuo Yang, and Zeke Xie. Diffusion dataset condensation: Training your diffusion model faster with less data. *arXiv preprint arXiv:2507.05914*, 2025.

[30] Ruiheng Zhang, Lixin Xu, Zhengyu Yu, Ye Shi, Chengpo Mu, and Min Xu. Deep-irtarget: An automatic target detector in infrared imagery using dual-domain feature extraction and allocation. *IEEE Transactions on Multimedia*, 24:1735–1749, 2021.

[31] Shuo Yang, Erkun Yang, Bo Han, Yang Liu, Min Xu, Gang Niu, and Tongliang Liu. Estimating instance-dependent bayes-label transition matrix using a deep neural network. In *International Conference on Machine Learning*, pages 25302–25312. PMLR, 2022.

[32] Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International conference on machine learning*, pages 1885–1894. PMLR, 2017.

[33] Amirata Ghorbani and James Zou. Data shapley: Equitable valuation of data for machine learning. In *International conference on machine learning*, pages 2242–2251. PMLR, 2019.

[34] Yongchan Kwon, Manuel A Rivas, and James Zou. Efficient computation and analysis of distributional shapley values. In *International Conference on Artificial Intelligence and Statistics*, pages 793–801. PMLR, 2021.

[35] Ziheng Jiang, Chiyuan Zhang, Kunal Talwar, and Michael C Mozer. Characterizing structural regularities of labeled data in overparameterized models. *arXiv preprint arXiv:2002.03206*, 2020.

[36] Ki Nohyun, Hoyong Choi, and Hye Won Chung. Data valuation without training of a model. In *The Eleventh International Conference on Learning Representations*, 2022.

[37] Zhaoxuan Wu, Yao Shu, and Bryan Kian Hsiang Low. Davinz: Data valuation using deep neural networks at initialization. In *International Conference on Machine Learning*, pages 24150–24176. PMLR, 2022.

[38] Rui Zhang and Shihua Zhang. Rethinking influence functions of neural networks in the over-parameterized regime. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 9082–9090, 2022.

[39] Shuo Yang, Zeke Xie, Hanyu Peng, Min Xu, Mingming Sun, and Ping Li. Dataset pruning: Reducing training data by examining generalization influence. *arXiv preprint arXiv:2205.09329*, 2022.

[40] Michel Grabisch and Marc Roubens. An axiomatic approach to the concept of interaction among players in cooperative games. *International Journal of game theory*, 28:547–565, 1999.

[41] Sebastian Bordt and Ulrike von Luxburg. From shapley values to generalized additive models and back. In *International Conference on Artificial Intelligence and Statistics*, pages 709–745. PMLR, 2023.

[42] Chih-Kuan Yeh, Joon Kim, Ian En-Hsu Yen, and Pradeep K Ravikumar. Representer point selection for explaining deep neural networks. *Advances in neural information processing systems*, 31, 2018.

[43] Garima Pruthi, Frederick Liu, Satyen Kale, and Mukund Sundararajan. Estimating training data influence by tracing gradient descent. *Advances in Neural Information Processing Systems*, 33:19920–19930, 2020.

[44] Yuanyuan Chen, Boyang Li, Han Yu, Pengcheng Wu, and Chunyan Miao. Hydra: Hypergradient data relevance analysis for interpreting deep neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 7081–7089, 2021.

[45] Zayd Hammoudeh and Daniel Lowd. Training data influence analysis and estimation: A survey. *Machine Learning*, 113(5):2351–2403, 2024.

[46] Xiangxin Zhu, Carl Vondrick, Charless C Fowlkes, and Deva Ramanan. Do we need more training data? *International Journal of Computer Vision*, 119(1):76–92, 2016.

[47] Zihui Zhang, Weisheng Dai, Hongtao Wen, and Bo Yang. Logosp: Local-global grouping of superpoints for unsupervised semantic segmentation of 3d point clouds. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 1374–1384, 2025.

[48] Jiahang Liu, Yunpeng Qi, Jiazhao Zhang, Minghan Li, Shaoan Wang, Kui Wu, Hanjing Ye, Hong Zhang, Zhibo Chen, Fangwei Zhong, et al. Trackvla++: Unleashing reasoning and memory capabilities in vla models for embodied visual tracking. *arXiv preprint arXiv:2510.07134*, 2025.

[49] Yiwei Zhou, Xiaobo Xia, Zhiwei Lin, Bo Han, and Tongliang Liu. Few-shot adversarial prompt learning on vision-language models. *Advances in Neural Information Processing Systems*, 37:3122–3156, 2024.

[50] Ruiheng Zhang, Zhe Cao, Yan Huang, Shuo Yang, Lixin Xu, and Min Xu. Visible-infrared person re-identification with real-world label noise. *IEEE Transactions on Circuits and Systems for Video Technology*, 2025.

[51] Zhuo Huang, Xiaobo Xia, Li Shen, Bo Han, Mingming Gong, Chen Gong, and Tongliang Liu. Harnessing out-of-distribution examples via augmenting content and style. *arXiv preprint arXiv:2207.03162*, 2022.

[52] Lisa Dunlap, Alyssa Umino, Han Zhang, Jiezhi Yang, Joseph E Gonzalez, and Trevor Darrell. Diversify your vision datasets with automatic diffusion-based augmentation. *Advances in neural information processing systems*, 36:79024–79034, 2023.

[53] Qilin Liao, Shuo Yang, Bo Zhao, Ping Luo, and Hengshuang Zhao. Bood: Boundary-based out-of-distribution data generation. *arXiv preprint arXiv:2508.00350*, 2025.

[54] Shuo Yang, Lu Liu, and Min Xu. Free lunch for few-shot learning: Distribution calibration. *arXiv preprint arXiv:2101.06395*, 2021.

[55] Dan Hendrycks, Norman Mu, Ekin D Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. Augmix: A simple data processing method to improve robustness and uncertainty. *arXiv preprint arXiv:1912.02781*, 2019.

[56] Linjun Zhang, Zhun Deng, Kenji Kawaguchi, Amirata Ghorbani, and James Zou. How does mixup help with robustness and generalization? *arXiv preprint arXiv:2010.04819*, 2020.

[57] Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random erasing data augmentation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 13001–13008, 2020.

[58] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

[59] Gianluigi Ciocca, Claudio Cusano, Francesca Gasparini, and Raimondo Schettini. Self-adaptive image cropping for small displays. *IEEE Transactions on Consumer Electronics*, 53(4):1622–1627, 2007.

[60] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[61] Maayan Frid-Adar, Eyal Klang, Michal Amitai, Jacob Goldberger, and Hayit Greenspan. Synthetic data augmentation using gan for improved liver lesion classification. In *2018 IEEE 15th international symposium on biomedical imaging (ISBI 2018)*, pages 289–293. IEEE, 2018.

[62] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.

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

[64] Haowei Zhu, Ling Yang, Jun-Hai Yong, Hongzhi Yin, Jiawei Jiang, Meng Xiao, Wentao Zhang, and Bin Wang. Distribution-aware data expansion with diffusion models. *arXiv preprint arXiv:2403.06741*, 2024.

[65] Rinon Gal, Yuval Alaluf, Yuval Atzmon, Or Patashnik, Amit H Bermano, Gal Chechik, and Daniel Cohen-Or. An image is worth one word: Personalizing text-to-image generation using textual inversion. *arXiv preprint arXiv:2208.01618*, 2022.

[66] Bram Wallace, Meihua Dang, Rafael Rafailov, Linqi Zhou, Aaron Lou, Senthil Purushwalkam, Stefano Ermon, Caiming Xiong, Shafiq Joty, and Nikhil Naik. Diffusion model alignment using direct preference optimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8228–8238, 2024.

[67] Diederik Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational diffusion models. *Advances in neural information processing systems*, 34:21696–21707, 2021.

[68] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *2008 Sixth Indian conference on computer vision, graphics & image processing*, pages 722–729. IEEE, 2008.

[69] Zikai Zhou, Shitong Shao, Lichen Bai, Zhiqiang Xu, Bo Han, and Zeke Xie. Golden noise for diffusion models: A learning framework. *arXiv preprint arXiv:2411.09502*, 2024.

[70] Lichen Bai, Shitong Shao, Zikai Zhou, Zipeng Qi, Zhiqiang Xu, Haoyi Xiong, and Zeke Xie. Zigzag diffusion sampling: Diffusion models can self-improve via self-reflection. In *The Thirteenth International Conference on Learning Representations*, volume 2, 2024.

[71] Donghoon Ahn, Jiwon Kang, Sanghyun Lee, Jaewon Min, Minjae Kim, Wooseok Jang, Hyoungwon Cho, Sayak Paul, SeonHwa Kim, Eunju Cha, et al. A noise is worth diffusion guidance. *arXiv preprint arXiv:2412.03895*, 2024.

[72] Teresa Yeo, Andrei Atanov, Harold Benoit, Aleksandr Alekseev, Ruchira Ray, Pooya Esmaeil Akhoondi, and Amir Zamir. Controlled training data generation with diffusion models. *arXiv preprint arXiv:2403.15309*, 2024.

[73] 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. Ieee, 2009.

[74] Li Fei-Fei, Rob Fergus, and Pietro Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In *2004 conference on computer vision and pattern recognition workshop*, pages 178–178. IEEE, 2004.

[75] Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and CV Jawahar. Cats and dogs. In *2012 IEEE conference on computer vision and pattern recognition*, pages 3498–3505. IEEE, 2012.

[76] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101–mining discriminative components with random forests. In *Computer vision–ECCV 2014: 13th European conference, zurich, Switzerland, September 6-12, 2014, proceedings, part VI 13*, pages 446–461. Springer, 2014.

[77] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011.

[78] Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3606–3613, 2014.

[79] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We confirm that the abstract and introduction accurately reflects the paper's contributions, as it clearly outlines the proposed UTILGEN framework, its adaptive optimization approach, and experimental validation across diverse datasets.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Discussed in Appendix C

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: This paper does not include theoretical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide comprehensive implementation details in Appendix B and dataset specifications in Appendix F, including methodological implementations and dataset characteristics, to ensure full reproducibility of our work.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: Our study uses exclusively publicly accessible datasets and includes the complete implementation source code in the supplementary materials (provided as a ZIP archive).

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: We provide comprehensive implementation details, including data partitioning, hyperparameter settings, and additional relevant information in Appendix B and Appendix F.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: We opted not to report error bars because the results demonstrated consistent stability across multiple runs, reducing the necessity for such reporting. To ensure robustness, each experiment was conducted using three different random seeds with results averaged accordingly. Additionally, extensive evaluations across diverse datasets further strengthen the reliability of our findings.

Guidelines:

- The answer NA means that the paper does not include experiments.

- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide detailed information on the computational resources used in this work, including the types and numbers of GPUs, GPU memory usage during execution, and overall running time. For more details, please refer to Appendix D.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have carefully examined the NeurIPS Code of Ethics and affirm that all aspects of this research comply with its guidelines.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

Justification: We provide a discussion of both positive and negative societal implications in Appendix E.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: This research does not involve releasing new models or datasets, thus the discussion of release safeguards is not applicable to our work.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: All datasets (e.g., ImageNet [73]) and foundation models (e.g., Stable Diffusion [4]) used in this work are appropriately cited the paper.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.

- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The code documentation is provided in the supplementary materials.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: LLMs were used for language polishing (grammar and spelling corrections).

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

A Additional Visualizations of Synthetic Data

In this section, we present further visualizations of synthetic data generated by the UTILGEN. As illustrated in Figure 8, we compare synthetic samples produced by different augmentation strategies, including GIF (prioritizing diversity in visual features) and DataDream (emphasizing fidelity in visual features). Figure 8 presents representative samples from UTILGEN, demonstrating semantically consistent, visually realistic generations that preserve class-discriminative features while maintaining diversity.

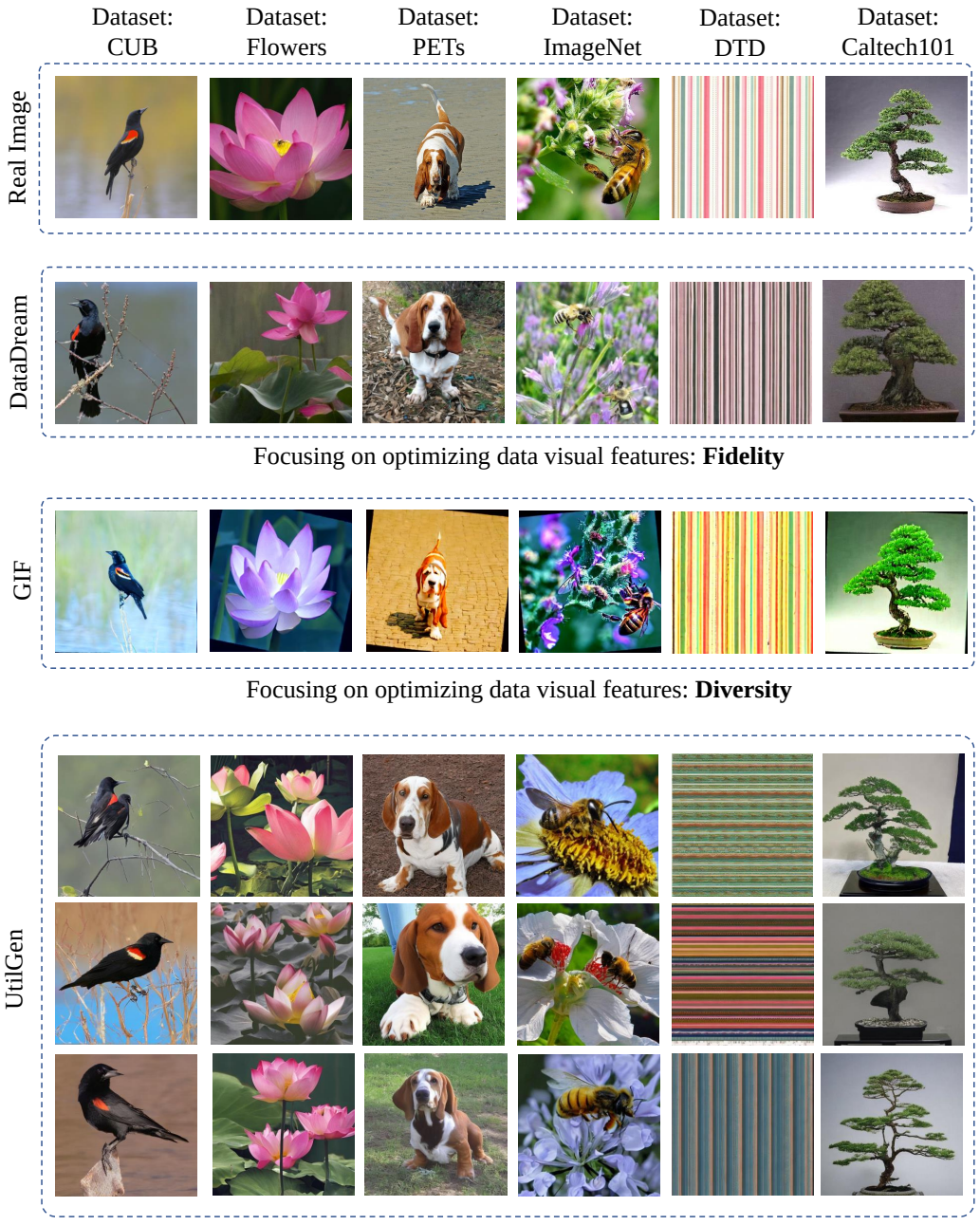


Figure 8: Comparison of synthetic samples generated by GIF, DataDream, and UTILGEN.

B More Implementation Details

B.1 Implementation Details of Weight Network

The weight network $\mathcal{W}_\phi$ is designed as a lightweight MLP with a single hidden layer. It takes the per-sample classification loss $\ell_i = \mathcal{L}(f(x_i;\theta), y_i)$ as input, where $f(\cdot;\theta)$ is the downstream classifier, and predicts normalized sample weights $\omega_i \in [0,1]$ via a sigmoid activation. These weights are used to re-weight the corresponding per-sample losses during classifier training, thereby prioritizing examples deemed to have higher utility by the meta-learned network. The network architecture is defined as:

$$\mathcal{W}_\phi(\ell_i) = \sigma(W_2\,\mathrm{ReLU}(W_1\ell_i + b_1) + b_2) \tag{10}$$

where $\ell_i \in \mathbb{R}$ is the scalar loss value for the i-th sample, $W_1 \in \mathbb{R}^{1\times 100}$ and $W_2 \in \mathbb{R}^{100\times 1}$ are weight matrices, and b_1, b_2 are bias terms. The ReLU activation introduces non-linearity, and the final sigmoid ensures the output lies in $[0,1]$. Albeit simple, this network is a universal approximator for continuous functions and can fit a wide range of weighting functions. The training settings for both the classifier and the weight network, including optimizers, learning rates, and batch sizes, are summarized in Table 8.

Table 8: Training settings for the classifier and weight network.

Component	Optimizer	Learning Rate	Batch Size
Classifier	SGD (momentum=0.9)	0.01	128
Weight Network	Adam	10^{-3}	128

B.2 Implementation Details of MLCO

Algorithm 3 Model-Level Generation Capability Optimization (MLCO)

Input: Initial diffusion model g_ψ; class prompts $\{c_1, ..., c_K\}$.
Required: Batch size B; DPO learning rate η; reference model g_{ref}; weight network $\mathcal{W}_\phi$; downstream classifier f_θ; loss function $\mathcal{L}$; max iteration I; selection ratio ρ..

for iteration = 1 to I **do**
  // Generation & Evaluation
  for each class $k \in \{1, ..., K\}$ **do**
    for $i = 1$ to B **do**
      Sample $\epsilon_T \sim \mathcal{N}(0, I)$
      $x_i^k \leftarrow g_\psi(c_k, \epsilon_T)$ ▷ Generate with current model
      $\omega_i^k \leftarrow \mathcal{W}_\phi(\mathcal{L}(f(x_i^k;\theta), y_k))$ ▷ Evaluate utility using classification loss as input
    end for
  end for
  // Preference Construction
  for each class k **do**
    Sort $\{x_i^k\}$ by ω_i^k descending
    Select top ρ proportion of samples as $\mathcal{D}_k^w$, bottom ρ proportion of samples as $\mathcal{D}_k^l$
    $\mathcal{D}_{\text{pref}} \leftarrow \mathcal{D}_{\text{pref}} \cup \{(c_k, x^w, x^l) | x^w \in \mathcal{D}_k^w, x^l \in \mathcal{D}_k^l\}$
  end for
  // Model Optimization
  for each (c_k, x^w, x^l) in $\mathcal{D}_{\text{pref}}$ **do**
    $\psi \leftarrow \psi - \eta\nabla_\psi \mathcal{L}_{\text{DPO}}(g_\psi, g_{\text{ref}}, c_k, x^w, x^l)$ ▷ Update via Eq. 5
  end for
end for

Output: Optimized diffusion model g_ψ^*

The proposed Model-Level Generation Capability Optimization (MLCO) framework iteratively improves the diffusion model's generative ability based on utility-guided preferences. In each iteration, the process proceeds in three stages, and he full process is illustrated in Algorithm 3.:

- **Generation & Evaluation:** The current diffusion model g_ψ generates synthetic images for each class prompt. Each generated sample is then evaluated by the trained weight network $\mathcal{W}_\phi$ to obtain utility scores ω.
- **Preference Construction:** Samples within each class are ranked by their utility scores. The top ρ and bottom ρ proportions are selected to form preference pairs $\mathcal{D}_{\text{pref}}$.
- **Model Optimization:** Using Direct Preference Optimization (DPO), the model is updated to customize its generation capability based on utility-guided preferences over training data.

Our implementation of DPO is adapted from Diffusion-DPO [66], which enables preference-based training of diffusion models for guiding generative outputs with utility-informed preferences. The loss function Eq. 5 used in Line 19 of Algorithm 3 follows this framework. For the detailed mathematical derivation of Eq. 5, please refer to the original Diffusion-DPO paper [66]. The key hyperparameters used in our DPO training process are summarized in Table 9.

Table 9: Hyperparameters used in DPO training

Batch Size	Max Steps Per Class	Learning Rate	Gradient Accumulation Steps	Beta DPO
1	400	1×10^{-8}	1	5000

B.3 Implementation Details of ILPO

The prompt-noise optimization process consists of two primary components: the optimization of prompt embeddings and the initial noise used during generation. The goal is to maximize the utility of each synthetic sample while ensuring semantic alignment with the target domain.

Prompt embedding optimization: The optimization process begins with textual inversion [65] to establish class-specific prompt embeddings that align with target concepts. Specifically, we use DeepSeek-R1-Distill-Qwen-1.5B [79] to select an initializer token for textual inversion [65] for each class. After obtaining the class-specific prompt embeddings aligned with the target labels, these embeddings are refined through gradient-based optimization to maximize the utility score of the generated samples. The learning rate for prompt optimization is set to 0.001, and the optimization process runs for 400 epochs to ensure convergence and meaningful results.

Table 10: Hyperparameters used in textual inversion [65]

Batch Size	Learning Rate	Training Steps	Instance Images per Class
1	1×10^{-4}	400	16

Noise optimization: For noise optimization, we leverage the discrepancy between denoising and inversion Classifier-Free Guidance (CFG) scales, which allows us to implicitly inject prompt semantic information into the noise vector. The denoising guidance scale is set to 5.5, while the inversion guidance scale is set to 0.

The specific hyperparameters used for this optimization process are summarized in Table 11.

Table 11: Hyperparameters used in the prompt-noise optimization process.

Prompt Learning Rate	Prompt Learning Epochs	Guidance Strength (Denoise)	Guidance Strength (Inversion)
0.001	400	5.5	0

B.4 Implementation Details of Image Generation

The image generation process is guided by the hyperparameters listed in Table 12. Instead of using fixed prompts and random noise, both the class-specific text prompts and the initial noise vectors

are optimized via our proposed ILPO strategy to better align with high-utility regions in the data distribution. The Stable Diffusion v2.1 model, fine-tuned with MLCO, is employed for the generation process, using a sampling method with 50 steps, the DDIM scheduler, and a guidance scale of 2.0. The images are generated at a resolution of 512×512 pixels.

Table 12: Hyperparameters for Training Data Synthesis

Base Model	Sampling Steps	Scheduler	Guidance Scale	Image Size
Stable Diffusion v2.1 (MLCO-fine-tuned)	50	DDIM	2.0	512×512

B.5 Implementation details of model training

The downstream classifiers are trained on four standard architectures: ResNet-50[18], ResNeXt-50 [19], WideResNet-50 [20], and MobileNetV2 [21]. All models are trained with identical hyperparameters. The training configuration uses SGD optimizer with momentum 0.9 and weight decay 5e-4. The learning rate starts at 0.01 with cosine decay schedule over 100 epochs. A fixed batch size of 256 is used for all experiments, with standard data augmentation including random horizontal flips and crops. Each experiment is repeated three times with different random seeds to ensure reliability.

Table 13: Hyperparameters for Downstream Classifier Training

Optimizer	Weight decay	Initial LR	Epochs	Batch size
SGD (momentum=0.9)	5e-4	0.01	100	256

C Limitations

While demonstrating strong performance in enhancing synthetic data utility for downstream tasks, UTILGEN presents two noteworthy considerations: (1) The dual-level optimization framework incurs modest computational overhead compared to conventional augmentation methods; (2) Although effectively improving the utility of synthetic data for downstream tasks, the approach remains contingent upon the base generative model's capability to produce viable initial samples. These considerations do not substantially compromise overall performance but indicate potential avenues for future enhancement.

D Efficiency and Cost of Data Augmentation

To evaluate the computational efficiency of UTILGEN, we compare it against three representative generative augmentation methods: GIF [12], GAP [72], and DataDream [11]. The comparison is conducted under a unified setup using the ImageNet-1K dataset, where each method generates 1000 synthetic images per class. We decompose the computational pipeline into three stages: (1) *Model Optimization*, (2) *Policy Optimization*, and (3) *Image Generation*. UTILGEN incorporates feedback-driven optimization components at both the model and instance levels, introducing moderate overhead that remains manageable.

Table 14 presents the runtime taken for each stage, while Table 15 shows the peak GPU memory usage during the execution of each stage. All methods are evaluated on a multi-GPU server equipped with 8×A100 GPUs. Despite the feedback-based optimization, our framework remains computationally efficient. Overall, UTILGEN strikes a favorable balance between computational cost and data utility, demonstrating its scalability and practicality for real-world deployments.

E Broader Impact

Our utility-driven augmentation approach facilitates more efficient model training while reducing reliance on real data, especially benefiting domains with limited or private datasets. By generating

Table 14: Computational cost (in hours) comparison on ImageNet-1K for generating 1000 images per class. Values are estimated averages.

Method	Model Optimization	Policy Optimization	Image Generation
GIF [12]	–	–	∼229.1h
GAP [72]	–	∼45.1h	∼93.7h
DataDream [11]	∼10.8h	–	∼40.2h
UTILGEN (Ours)	∼7.7h	∼25.1h	∼41.6h

Table 15: GPU memory usage (peak usage in GB) per stage. All values are measured during maximum workload per module on each GPU.

Method	Model Optimization	Policy Optimization	Image Generation
GIF [12]	–	–	∼25.9G
GAP [72]	–	∼5.2G	∼4.5G
DataDream [11]	∼19.5G	–	∼15.8G
UTILGEN (Ours)	∼20.5G	∼4.4G	∼4.3G

task-specific synthetic training data, it enhances learning efficiency and lowers dependence on large-scale real datasets. Nonetheless, since the synthesis process is guided by a small set of real images, the generated data may inadvertently inherit and amplify biases present in the original samples.

F Dataset Details

To evaluate UTILGEN's performance, we utilize eight benchmark datasets spanning a variety of classification tasks: coarse-grained classification, fine-grained classification, and texture classification.

The coarse-grained datasets include ImageNet-1k-Subset [73], ImageNet-100-Subset [73] and Caltech 101 [74]. ImageNet-1k-Subset [73] and ImageNet-100-Subset [73], both randomly sampled with 100 images per class. ImageNet-100-Subset is a subset of 100 animal-related classes from ImageNet-1K. Caltech 101 [74] consists of 101 object categories. For fine-grained classification, we use Oxford Pets [75], Food101-Subset [76], Flowers 102 [68], and CUB-200-2011 [77], with Food101-Subset [76] being a curated subset of Food101 [76] containing 101 food categories. Other datasets follow their original training and validation setups. Texture classification is evaluated using the DTD [78] dataset, which contains 47 texture categories. Detailed dataset statistics are provided in Table 16, summarizing the number of classes, training samples, and test samples for each dataset. It is important to note that datasets with a higher number of classes or fewer average samples per class present greater challenges in terms of classification and generalization.

Table 16: Statistics of the benchmark datasets

Dataset	Task Type	Classes	Training Data	Test Data
ImageNet-1k-Subset [73]	Coarse-grained object classification	1000	100,000	50,000
ImageNet-100-Subset [73]	Coarse-grained object classification	100	10,000	5,000
Caltech 101 [74]	Coarse-grained object classification	101	3060	6084
Oxford Pets [75]	Fine-grained object classification	37	3680	3669
Food101-Subset [76]	Fine-grained object classification	101	10100	25,250
Flowers 102 [68]	Fine-grained object classification	102	6,552	818
CUB-200-2011 [77]	Fine-grained object classification	200	5,994	5,794
DTD [78]	Texture classification	47	1880	1,880