IS SYNTHETIC DATA FROM GENERATIVE MODELS READY FOR IMAGE RECOGNITION?

Ruifei He[1*] **Shuyang Sun**[2] **Xin Yu**[1] **Chuhui Xue**[3] **Wenqing Zhang**[3] **Philip Torr**[2]
Song Bai[3†] **Xiaojuan Qi**[1†]
[1]The University of Hong Kong [2]University of Oxford [3]ByteDance

A MORE ANALYSIS

A.1 LIMITATIONS AND FUTURE DIRECTIONS

A systematic way to study the language enhancement process is needed. In our experiments, we found language enhancement to be an effective way for enlarging synthetic data diversity. In this paper, we adopt an off-the-shelf word-to-sentence language model for this job, but this may bring certain risks and generate a high volume of noisy data. Therefore, effective constraints should be considered for language enhancement, which we leave as future work.

Larger synthetic data sizes for zero/few-shot tasks should be studied. We only generate synthetic data of median size for zero/few-shot tasks due to our limited computational resources; we encourage future works to further explore whether increasing data diversity and data amount could benefit these data-scarce tasks.

Co-training down-stream predictors with generative models. In this paper, we propose a pre-trained off-line image generator to construct our synthetic training set. It would be interesting to investigate how to generate in-domain images by tuning the image generator together with the lateral model for down-stream tasks. This could potentially be useful for domain generalization.

Increasing scales for synthetic data and model capability for pre-training settings. We observe improved performance when the data amount and diversity (label space) increase. However, due to our limited computational resource, we cannot further scale up the investigation, which may take months to train one model. We hope our promising results could inspire future research to explore enlarged scales for synthetic data and model size (larger model capability should be better at benefiting from a larger amount of data) for pre-training settings.

A.2 ANALYSIS OF DIFFERENCES OF PERFORMANCE BOOST ON DIFFERENT DATASETS IN ZERO-SHOT SETTINGS

The main reason is related to GLIDE's training data distribution. The training data distribution of the text-to-image generation model GLIDE would exhibit bias and produce different domain gaps with different datasets. While we do not have direct access to GLIDE's training data, we may use synthesized images from GLIDE to explore. However, how to measure data and its relation to the performance for different tasks in a quantitative manner is still an open problem to our best knowledge as different task may have different levels of difficulties and data will impact the performance in different manners. An empirical thought is related to the synthesized image quality (e.g., synthesized images for DTD dataset is relatively noisy as shown in Figure A.5, where we only achieve 0.96% gains), domain gaps (synthesized images are usually object-centric and exhibit large domain difference with scene-level dataset SUN397, where only 1.56% boost achieved), and task difficulties (e.g., ImageNet is of high difficulty where only 0.45% gains are obtained).

For further improving the results, we suggest that it would be interesting to investigate how to generate in-domain images by tuning the image generator together with the lateral model for downstream tasks. This could potentially better reducing domain gap to the downstream task.

* Part of the work is done during an internship at ByteDance. Email: ruifeihe@eee.hku.hk
† Corresponding authors: songbai.site@gmail.com, xjqi@eee.hku.hk

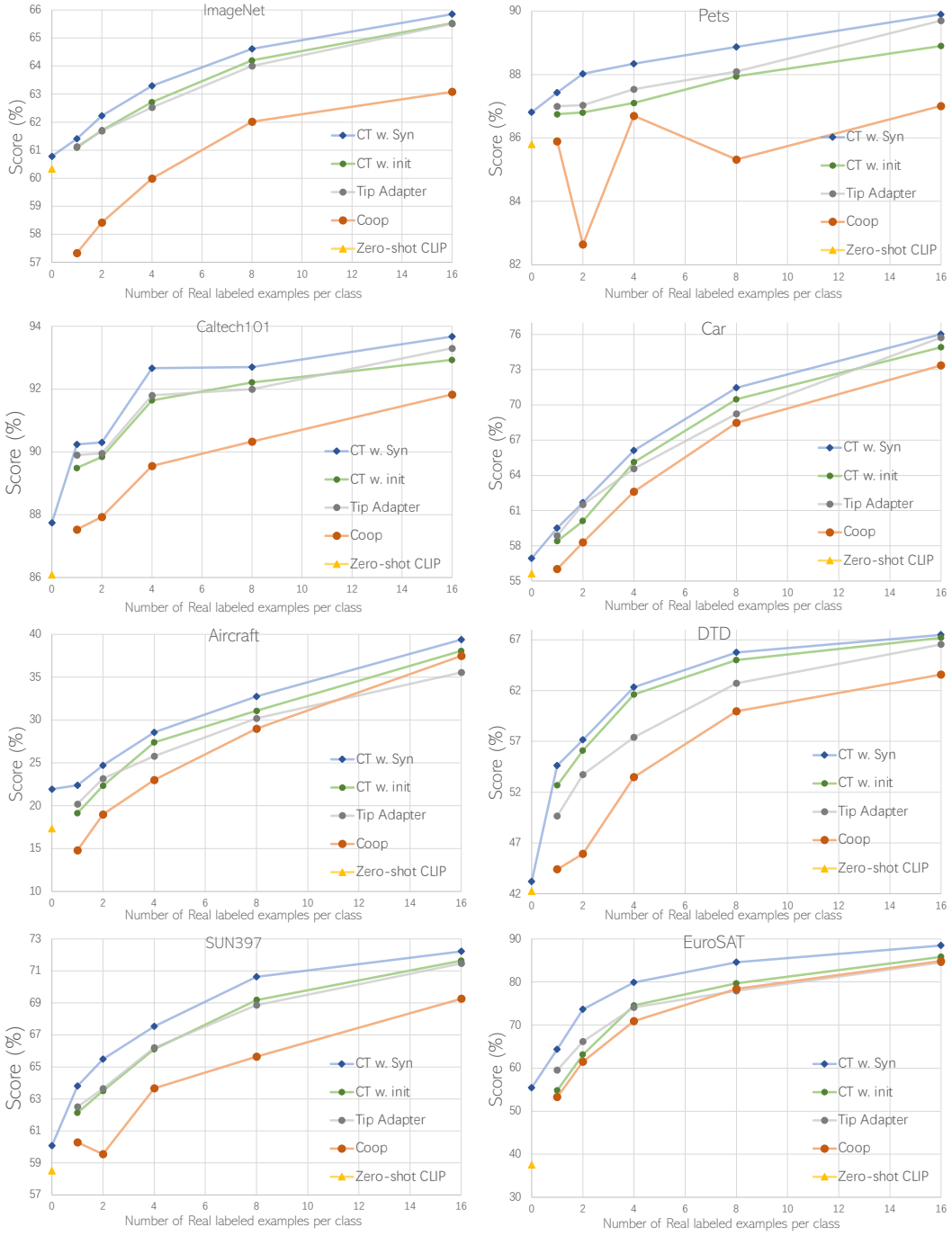

Figure A.1: Main results for Few-shot tasks on all 8 datasets.

B More Experiments

B.1 Main results on Few-shot tasks

Due to the limit of pages and space, we provide the results for few-shot tasks of all 8 datasets in Figure A.1. We observe similar results on various datasets that synthetic data could boost few-shot learning performance and achieve a new state-of-the-art performance.

B.2 REAL-WORLD DATA WITH DOMAIN SHIFT

In order to understand why synthetic data may harm pre-trained image encoder, we experiment with real-world data with domain shifts. Specifically, we conduct experiments on pre-trained CLIP model and study two scenarios (Classifier tuning only, and end-to-end finetune) for the downstream ImageNet classification task. These experiments on conducted on three data sources: 1) in-domain real data: ImageNet data is used for training; 2) real-world data with domain shifts: ImageNet-Sketch data which has the same label space with ImageNet but exhibits a domain shift towards sketch are used for training; and 3) our synthetic data: synthetic data from GLIDE are used for training. The results are shown in the Table A.1. It can be clearly observed that real-world data with domain shifts behave similarly as synthetic data (i.e. end-to-end tuning cause slight harm to the pre-trained encoder) while the performance of in-domain data will benefit from end-to-end tuning. This suggests that domain gap is the main reason for harming the pre-trained image encoder.

We argue that synthetic data might have a better chance to overcome domain shifts in comparison with real-world data since we can customize and keep the label space of the synthetic data in line with the down-stream dataset. First, synthetic data can be made to tailor to a specific label space that the downstream task requires and reduce category shifts, which yet might be very challenging for real-world data. Second, a small amount of real-data can be leveraged to guide the data synthesis process to further alleviate domain shifts (namely Real Guidance in our few-shot settings), which also worth future in-depth explorations.

Train data	ImageNet	ImageNet-Sketch	Synthetic data
Classifier tuning	70.09	**60.50**	**60.78**
End-to-end finetune	**76.17**	60.34	60.35

Table A.1: Classifier tuning v.s. End-to-end finetune on different types of data. Zero-shot performance: 60.33.

B.3 STUDY OF SYNTHETIC IMAGE NUMBER IN ZERO/FEW-SHOT SETTING

Here, we provide the study for the number of synthetic images in the zero-shot and few-shot settings. Firstly, for zero-shot settings, we experiment with 1000/2000/4000 images per class on Cifar10 dataset. As shown in Table A.2, we found 2000 to be a sufficient number while further increasing the number to 4000 only provide limited gains. Second, for few-shot settings, we study upon the settings of Eurosat dataset with 16 shot real images, and vary the number of synthesized images for each class from 400 to 1600. As shown in Table A.3, we found 800 synthetic images for each class to be a good choice between performance and cost.

For both settings, increasing the amount of training data further beyond a certain amount will not bring significant performance gain. The reasons can be attributed to the diversity and quality of data. We found that as the increase of the training data amount, the diversity of the data might not be scaled in a similar manner. Many redundant and similar samples will appear with the increase of data amount. Effective approaches to increase data diversity and quality will help further improve model performance. For the few-shot setting, the performance reach a high value after using a small amount of synthetic data (400-shot). This is because the existence of real-data provide a strong guidance for training the classifier. And, the positive impacts of synthetic data are reduced where a small amount of synthetic data are sufficient to learn a good classifier.

syn shot	0	1000	2000	4000
Acc	70.31	79.21	**80.06**	**80.08**

Table A.2: Study of synthetic image number in the zero-shot setting on CIFAR-10 dataset.

syn shot	0	400	800	1600
Acc	85.83	88.28	**88.47**	**88.49**

Table A.3: Study of synthetic image number in the few-shot setting on EuroSAT dataset with 16 shot real images.

B.4 CLAIFICATION OF ZERO-SHOT CLASSIFICATION PERFORMANCE DIFFERENCE

For all results, we use the official released CLIP model[1]. For our original paper, we conduct experiments using simple prompts "a photo of a [CLASS]" or "a photo of a [CLASS], a type of [dataset type]" (for fine-grained tasks) to better evaluate the performance gains from the data itself by excluding the influence of tuned prompt methods. However, in the original CLIP paper, they use specifically designed prompt ensembles for each dataset. This is the main reason that our baseline is lower than that of the original paper.

We also provide the results of using CLIP paper's prompts for 13 datasets (only these 13 datasets out of 17 have reported results from CLIP's paper) in Table A.4. After using the same prompts as the CLIP's paper, the averaged performance on 13 datasets increased from 56.14% to 56.33%, closer to the 57.03% of the CLIP's reported results. There are still slight performance differences after using the same prompts, which we suspect to be a small reproduction problem of CLIP since we could match the reported zero-shot results from CoOp (Zhou et al., 2022) and Tip-adapter (Gao et al., 2021). We observe that on some datasets we achieve higher results than CLIP reported results (i.e., CIFAR-100, ImageNet, SUN397, Birdsnap, Flower, Pets) and on others we achieve lower results (i.e., CIFAR-10, Caltech101, Aircraft, Cars, Food, DTD, EuroSAT). The averaged zero-shot performance is similar (56.33% v.s 57.03%) and the performance boost from synthetic data of two different types of prompt is also similar (averaged performance boost on 13 datasets: 3.85% v.s 4.17%). We argue the slight performance differences do not affect the exploration of synthetic data.

	CLIP-RN50*	CLIP-RN50	CLIP-RN50	CLIP-RN50†	CLIP-RN50†+SYN
CIFAR-10	75.6	70.31	80.06 (+9.75)	71.59	80.23 (+8.64)
CIFAR-100	41.6	35.35	45.69 (+10.34)	41.94	48.70 (+6.76)
Caltech101	82.1	86.09	87.74 (+1.65)	79.99	82.34 (+2.35)
ImageNet	59.60	60.33	60.78 (+0.45)	60.33	60.78 (+0.45)
SUN397	59.60	58.51	60.07 (+1.56)	60.23	60.47 (+0.24)
Aircraft	19.3	17.34	21.94 (+4.60)	17.07	21.78 (+4.71)
Birdsnap	32.6	34.33	38.05 (+3.72)	34.33	38.05 (+3.72)
Cars	55.80	55.63	56.93 (+1.30)	55.70	57.65 (+1.95)
Flower	65.90	66.08	67.05 (+0.97)	66.08	67.05 (+0.97)
Food	81.10	80.34	80.35 (+0.01)	80.34	80.35 (+0.01)
Pets	85.40	85.80	86.81 (+1.01)	85.80	86.81 (+1.01)
DTD	41.7	42.23	43.19 (+0.96)	41.31	42.21 (+0.90)
EuroSAT	41.1	37.51	55.37 (+17.86)	37.52	55.35 (+17.83)
Average	57.03	56.14	60.31 (+4.17)	56.33	60.10 (+3.85)

Table A.4: CLIP-RN50*: original CLIP paper results. CLIP-RN50: our results using simple prompt. CLIP-RN50†: our results using the same prompt ensemble as CLIP.

B.5 FID AND CLASSIFICATION ACCURACY WITH A PRE-TRAINED CLASSIFIER FOR SYNTHETIC DATA MEASUREMENT.

Frechet Inception Distance (FID) is a metric that calculates the distance between feature vectors from real and generated images, and lower scores have been shown to correlate well with higher-quality images. Here, we provide the FID scores and classification accuracy with a pre-trained CLIP ViT-B/16 model for measuring diversity and quality of synthesized images. We study with the Eurosat dataset. For the FID score, we do not have access to GLIDE training data, and thus we calculate FID between the downstream task ground-truth data and synthetic data generated by different strategies. As shown in Table A.5, LE could largely reduce FID and provide large diversity while hurting class fidelity. After combing with CF, FID further reduces and we also achieve a higher class fidelity. For few shot settngs, RG could largely reduce FID and yield the best class fidelity.

syn data	z: B	z: LE	z: LE+CF	f: RF	f: RG
FID	84.84	81.15	90.85	79.02	**37.18**
Acc	44.27	35.84	48.03	46.95	**48.76**

Table A.5: FID and classification accuracy with CLIP-ViT-B/16. "z" for zero-shot and "f" for few-shot.

[1] https://github.com/openai/CLIP

Figure A.2: Examples of synthesized images from Language Enhancement strategy.

B.6 Examples of synthesized images from Language Enhancement strategy

Here, we provide successful and failure examples of synthesized images from language enhancement strategy. As shown in Figure A.2 (a) $\sim$ (c), language enhancement could introduce more diversity into the language prompts and lead to more diversified synthesized images for each class, such as introducing "runway" in (a), "pond" in (b), and "red pillow" in (c). However, we also observe failure cases after the language enhancement process. As we can see in Figure A.2 (d) $\sim$ (f), after introducing some other items into the language prompts, the focus of the generated images may move to other objects rather than the target class. In some extreme failure cases we show here, the generated images may even not contain the desired class object.

B.7 Visualization: synthetic data in zero-shot settings

We provide the visual illustration of ground-truth real data and synthesized images by different strategies for the zero-shot settings, *i.e.* basic (**B**), language enhancement (**LE**), and language enhancement and CLIP-based filtering (**LE+CF**). Here, we take the "Highway or road" class in EuroSAT dataset as an example. As shown in Figure A.3, we could see that LE could help increase the diversity but may introduce noisy samples, but LE+CF could further select images with higher class fidelity and yields synthetic data with reduced domain gaps.

B.8 Visualization: synthetic data in few-shot settings

We provide the visual illustration of synthesized images by different strategies for the few-shot settings, *i.e.* basic (**B**), real filtering (**RF**), and real guidance (**RG**) as well as real images of the same class for comparison. Here, we take the "forest" class in EuroSAT dataset as an example. As shown in Figure A.4, both **RF** and **RG** strategies produce images with reduced domain gap from the real images of the target domain. Further, **RG** significantly approaches the real images better than **RF**, demonstrating the effectiveness of the proposed **RG** method.

B.9 Visualization: synthetic data for different datasets

Here, we provide synthesized images for different datasets (i.e., CIFAR10, Caltech101, Cars, ImageNet-Sketch, DTD) in Figure A.5. All images are randomly chosen rather than human-picked. Each row consists of images of the same class. We observe that for most datasets, synthesized im-

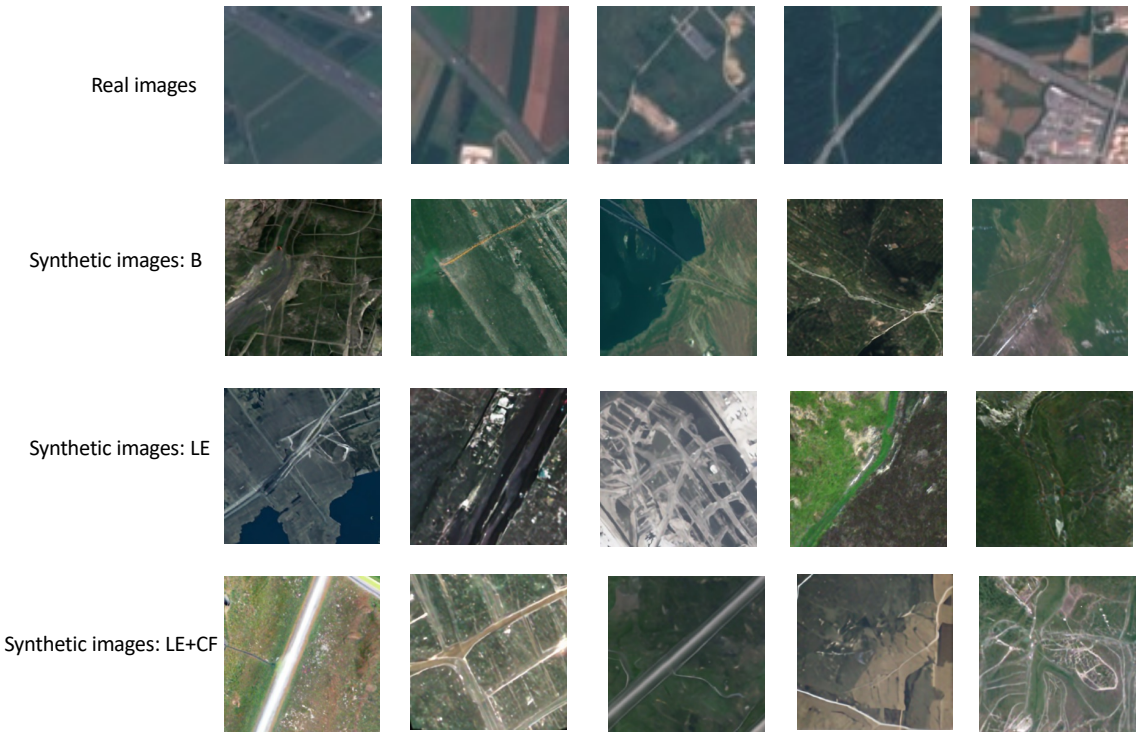

Figure A.3: Visualization of different strategies of synthetic data in zero-shot settings.

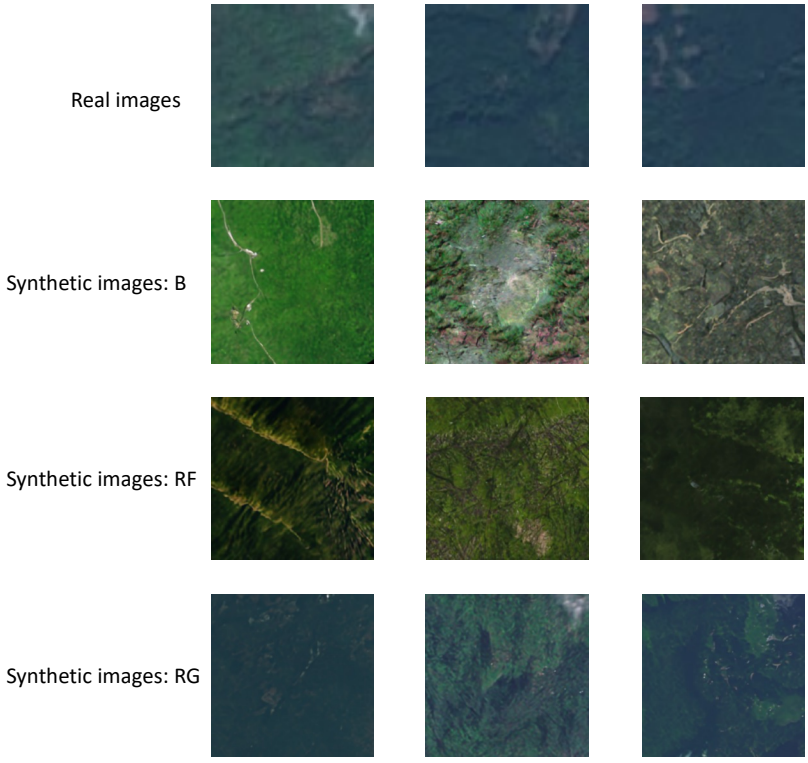

Figure A.4: Visualization of different strategies of synthetic data in few-shot settings.

ages from the GLIDE model are of high quality, but there also exist cases that many unsatisfactory examples are generated, such as the DTD datasets.

We state that this is a limitation of the current text-image generation model that it may produce images of low quality for certain tasks, which mainly due to the domain gap between the training data of the generation model and the task. However, with the study of future text-image generation models, the quality of synthesized images is potentially growing higher constantly. Besides, the relatively lower quality images could be used for pre-training tasks for better improving performance of different down-stream tasks.

C Additional Details

C.1 Denoising diffusion probabilistic model

Denoising diffusion probabilistic model (DDPM) learns the data distribution through introducing a series of latent variables and matching the joint distribution. Formally, given a sample from the data distribution $x_0 \sim q(\mathbf{x}_0)$, a forward process $q\left(\mathbf{x}_{1:T} \mid \mathbf{x}_0\right)=\prod_{t=1}^{T} q\left(\mathbf{x}_t \mid \mathbf{x}_{t-1}\right)$ progressively perturbs the data with Gaussian kernels $q\left(\mathbf{x}_t \mid \mathbf{x}_{t-1}\right):=\mathcal{N}\left(\sqrt{1-\beta_t} \mathbf{x}_{t-1}, \beta_t \mathbf{I}\right)$, producing increasingly noisy latent variables $\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_T$. Notably, x_t can be directly sampled from x_0 thanks to the closed form:

$$q\left(\mathbf{x}_t \mid \mathbf{x}_0\right)=\mathcal{N}\left(\mathbf{x}_t ; \sqrt{\bar{\alpha}_t} \mathbf{x}_0,\left(1-\bar{\alpha}_t\right) \mathbf{I}\right) \tag{1}$$

where $\alpha_t:=1-\beta_t$ and $\bar{\alpha}_t:=\prod_{s=1}^{t} \alpha_s$. In general, the forward process variances β_t are fixed and increased linearly from $\beta_1=10^{-4}$ to $\beta_T=0.02$. Besides, T should be large (*e.g.*, 1000) enough to ensure $q\left(\mathbf{x}_T \mid \mathbf{x}_0\right) \approx \mathcal{N}(0, \mathbf{I})$. Diffusion model aims to model the joint distribution $q\left(\mathbf{x}_{0: T}\right)$ which naturally involves a tractable sampling path for the marginal distribution $q\left(\mathbf{x}_0\right)$.

Specifically, the candidate distribution is formulated as a Markov chain with parameterized transition kernels:

$$p_\theta\left(\mathbf{x}_{0: T}\right):=p\left(\mathbf{x}_T\right) \prod_{t=1}^{T} p_\theta\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t\right), p_\theta\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t\right):=\mathcal{N}\left(\mathbf{x}_{t-1} ; \boldsymbol{\mu}_\theta\left(\mathbf{x}_t, t\right), \boldsymbol{\Sigma}_\theta\left(\mathbf{x}_t, t\right)\right) \tag{2}$$

The training is thus achieved by optimizing a variational bound of negative log likelihood:

$$\mathrm{E}_{q\left(\mathbf{x}_0\right)}\left[-\log p_\theta\left(\mathbf{x}_0\right)\right] \leq \mathrm{E}_{q\left(\mathbf{x}_{0: T}\right)}\left[-\log \frac{p_\theta\left(\mathbf{x}_{0: T}\right)}{q\left(\mathbf{x}_{1: T} \mid \mathbf{x}_0\right)}\right]=: L \tag{3}$$

The loss term L can be rewritten as:

$$\mathbb{E}_q[\underbrace{D_{\mathrm{KL}}\left(q\left(\mathbf{x}_T \mid \mathbf{x}_0\right) \| p\left(\mathbf{x}_T\right)\right)}_{L_T}+\sum_{t>1} \underbrace{D_{\mathrm{KL}}\left(q\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0\right) \| p_\theta\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t\right)\right)}_{L_{t-1}} \underbrace{-\log p_\theta\left(\mathbf{x}_0 \mid \mathbf{x}_1\right)}_{L_0}] \tag{4}$$

In practice, the core optimization terms are $L_{t-1}(t>1)$ that can be analytically calculated since both two terms compared in the KL divergence are Gaussians, *i.e.*,:

$$q\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0\right)=\mathcal{N}\left(\mathbf{x}_{t-1} ; \tilde{\boldsymbol{\mu}}_t\left(\mathbf{x}_t, \mathbf{x}_0\right), \tilde{\beta}_t \mathbf{I}\right), p_\theta\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t\right):=\mathcal{N}\left(\mathbf{x}_{t-1} ; \boldsymbol{\mu}_\theta\left(\mathbf{x}_t, t\right), \boldsymbol{\Sigma}_\theta\left(\mathbf{x}_t, t\right)\right) \tag{5}$$

where $\tilde{\boldsymbol{\mu}}_t\left(\mathbf{x}_t, \mathbf{x}_0\right):=\frac{\sqrt{\bar{\alpha}_{t-1}} \beta_t}{1-\bar{\alpha}_t} \mathbf{x}_0+\frac{\sqrt{\alpha_t}\left(1-\bar{\alpha}_{t-1}\right)}{1-\bar{\alpha}_t} \mathbf{x}_t \quad$ and $\quad \tilde{\beta}_t:=\frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t} \beta_t$. Ho et al. (2020) fix $\boldsymbol{\Sigma}_\theta\left(\mathbf{x}_t, t\right)=\sigma_t^2 \mathbf{I}$ during training, where σ_t^2 is set to be β_t or $\tilde{\beta}_t$. Through reparameterization trick (Kingma & Welling (2013)) and empirical simplification (Ho et al. (2020)), the final training term is performed as follows:

$$L_{\text {simple }}(\theta):=\mathbb{E}_{t, \mathbf{x}_0, \epsilon}\left[\left\|\boldsymbol{\epsilon}-\boldsymbol{\epsilon}_\theta\left(\sqrt{\bar{\alpha}_t} \mathbf{x}_0+\sqrt{1-\bar{\alpha}_t} \boldsymbol{\epsilon}, t\right)\right\|^2\right] \tag{6}$$

where $\epsilon \sim \mathcal{N}(0, \mathbf{I})$ and t is uniformly sampled between 1 and T.

After training, started from an initial noise map $x_T \sim p(\mathbf{x}_T)=\mathcal{N}(0, \mathbf{I})$, new images can be then generated via iteratively sampling from $p_\theta\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t\right)$ using the following equation:

$$\mathbf{x}_{t-1}=\frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t-\frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}} \boldsymbol{\epsilon}_\theta\left(\mathbf{x}_t, t\right)\right)+\sigma_t \mathbf{z}, \text { where } \mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \tag{7}$$

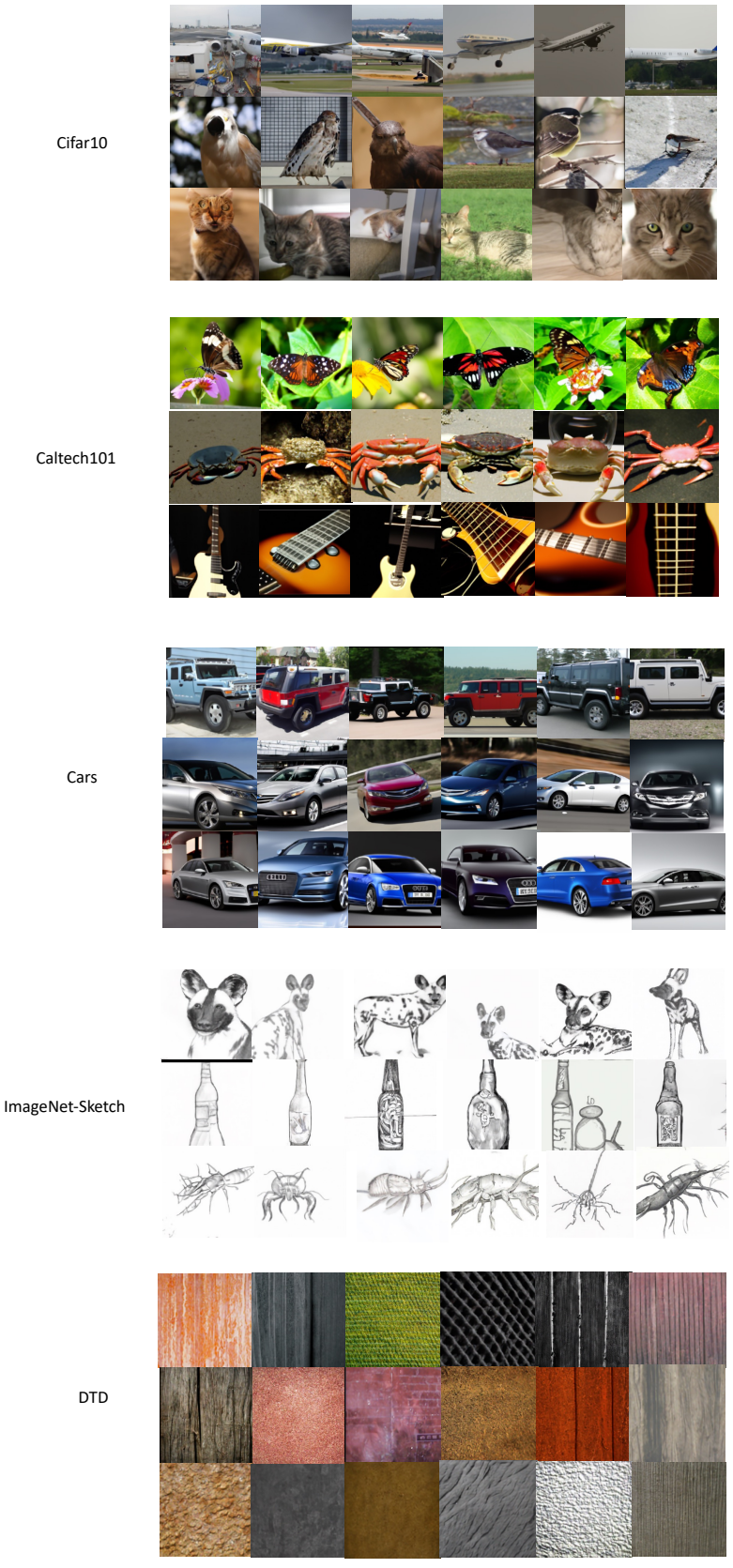

Figure A.5: Visualization of different synthetic datasets.

C.2 TEXT-TO-IMAGE GENERATION

The text-to-image diffusion model extends the basic unconditional diffusion model by changing the target distribution $q(\mathbf{x_0})$ into a conditional one $q(\mathbf{x_0} \mid \mathbf{c})$, where $\mathbf{c}$ is a natural language description. The derivation of the training terms and sampling procedure are similar to Sec. C.1, except that a conditioning signal $\mathbf{c}$ is included. Besides, following the improved DDPM (Nichol & Dhariwal (2021)), $\mathbf{\Sigma}_\theta$ is also estimated in GLIDE (Nichol et al. (2021)).

Especially, GLIDE employs a coarse-to-fine two-stage generation framework (Nichol & Dhariwal (2021); Saharia et al. (2022)) with two guidance techniques for balancing mode coverage and sample fidelity, namely classifier guidance (Dhariwal & Nichol (2021)) and classifier-free guidance (Ho & Salimans (2022)). Classifier guidance mainly relies on an extra trained noise CLIP model to provide feedback at intermediate sampling steps. Classifier-free guidance, on the other hand, randomly drops the text prompt with a fixed probability p during the training, which can be viewed as a joint training of an unconditional model $\boldsymbol{\epsilon}_\theta\left(\mathbf{x}_t \mid \emptyset\right)$ (*i.e.*, $\boldsymbol{\epsilon}_\theta\left(\mathbf{x}_t\right)$) and a conditional model $\boldsymbol{\epsilon}_\theta\left(\mathbf{x}_t \mid \mathbf{c}\right)$. At each sampling step, the model's output is actually performed using an extrapolation as follows:

$$\hat{\boldsymbol{\epsilon}}_\theta\left(\mathbf{x}_t \mid \mathbf{c}\right) = \boldsymbol{\epsilon}_\theta\left(\mathbf{x}_t \mid \emptyset\right) + s \cdot \left(\boldsymbol{\epsilon}_\theta\left(\mathbf{x}_t \mid \mathbf{c}\right) - \boldsymbol{\epsilon}_\theta\left(\mathbf{x}_t \mid \emptyset\right)\right) \tag{8}$$

where s is a guidance scale that can trade off sampling quality and diversity. In our work, we use classifier-free guidance with default setting $s = 3$ for all experiments since it achieves better results than CLIP guidance. To speed up the sampling process, DDIM (Song et al. (2020)) is utilized which allows the model to produce high-quality images within few seconds. We follow the default settings in GLIDE and set $T = 100$ in the coarse stage and $T = 27$ in the upsampler stage.

C.3 REAL GUIDANCE (**RG**) STRATEGY

We elaborate how we use few-shot in-domain real images to guide the generation process for few-shot settings. In a normal text-to-image generation process, a pure noisy image $x_T \sim \mathcal{N}(0, \mathbf{I})$ would be sampled first as the initialization of the reverse path. Then, the pretrained GLIDE model iteratively predicts a less noisy image x_{t-1} $(t = T, T-1, ..., 1)$ using the given text prompt c and the noisy latent image x_t as inputs. In our case, we add noise to a reference image x_0^{ref} such that the noise level corresponds to a certain time-step $t_\star$:

$$x_{t_\star}^{ref} = \sqrt{\bar{\alpha}_{t_\star}} x_0^{ref} + \sqrt{1 - \bar{\alpha}_{t_\star}} \epsilon \tag{9}$$

Then, rather than sampling from time-step T, we initialize the noisy latent variable as $x_{t_\star}^{ref}$ and begin our denoising process from time-step $t_\star$, as illustrated in Algorithm 1. Note that the GLIDE model adopts a coarse-to-fine two-stage generation framework and involves classifier-free guidance. However, we omit them in Algorithm 1 for simplicity since our image-guidance strategy only modifies the start point and leaves the other settings unchanged. In this way, the generated images can share similar in-domain properties, and thus helping to close the domain gap. While small $t_\star$ could synthesis images which are more similar to the reference image, it results in low diversity, which harms the classifier's learning. In the case of a large $t_\star$, $x_{t_\star}^{ref}$ retains too little information from x_0^{ref}, causing the generated image to deviate from the domain. In our experiments, we conduct different trade-offs considering different few-shot settings. Empirically, we set $t_\star$ as 15, 20, 35, 40, and 50 for shot 16, 8, 4, 2, and 1, respectively.

Algorithm 1 Real Guidance (**RG**) Strategy

Input: Reference image x_0^{ref}, text prompt c and GLIDE model $(\mu_\theta, \Sigma_\theta)$.
Output: Generated image x_0

```
# Noisy variable initialization
Select a time-step t_⋆ ~ 1, 2, 3, ..., T and random noise ε ~ N(0, I)
Obtain initial noisy image x_{t_⋆} := x_{t_⋆}^{ref} according to Eq. 9
# Random Sampling (could be replaced by DDIM for speed-up)
for s from t_⋆ to 1 do
    μ, Σ ← μ_θ(x_s, s, c), Σ_θ(x_s, s, c)
    x_{s-1} ← sample from N(μ, Σ)
end for
return x_0
```

C.4 SOFT-TARGET CROSS-ENTROPY LOSS

Example code for soft-target cross-entropy loss is shown below.

```
def soft_target_cross_entropy(logits, target, labels, T=2):
    # T: temperature for soft targets.
    loss_func_CE = torch.nn.CrossEntropyLoss()
    CE = loss_func_CE(logits, labels)
    soft_targets = torch.softmax(target/T, dim=1)
    SCE = torch.sum(-soft_labels * F.log_softmax(x, dim=-1), dim=-1)
    loss = 0.5 * CE + 0.5 * SCE
    return loss
```

C.5 IMPLEMENTATION DETAILS

C.5.1 ZERO-SHOT SETTING

For text-to-image generation process, we adopt the default hyperparameters from the official GLIDE text-to-image code. The input text of the basic strategy is "a photo of a [CLASS]", and the input text of language enhancement strategy is "a photo of a [SENTENCE]". For language enhancement, we adopt an off-the-shelf word-to-sentence T5 model pre-trained on "Colossal Clean Crawled Corpus" dataset (Raffel et al., 2020) and finetuned on CommonGen dataset (Lin et al., 2019). We generate 2000 synthetic images for each class in **B** and **LE**, and use a threshold of 1/N in **CF** where N is the number of classes. For **LE**, we generate 200 sentences for each class name.

For training on synthetic data for zero-shot recognition, we use AdamW (Loshchilov & Hutter, 2017) optimizer and an initial learning rate of 0.002 that is decayed by the cosine annealing rule. We train for 30 epochs, and use weight decay of 0.1 and batch size of 512. For image preprocessing, we resize the image's short side to 224 while keeping the original aspect ratio.

For datasets in the zero-shot settings, we follow previous works (Zhou et al., 2022; Gao et al., 2021) that use 11 datasets, and we excludes UCF101 since GLIDE exclude generating 'person' related content for privacy issues. Besides, we add another 7 popular datasets for more comprehensive evaluation. We do not conduct on all CLIP's 27 datasets since our computing resources are limited, and we believe our 17 datasets are already enough to study the effectiveness of synthetic data for zero-shot settings.

C.5.2 FEW-SHOT SETTING

For text-to-image generation process in few-shot settings, our basic strategy and Real filtering strategy both apply the same process as in the zero-shot settings; and for Real guidance strategy, the generation process is illustrated in Sec. C.3. For synthetic image number, we generate 800 images per class for **RG** method to approximately match the number of images in **B** and **RF**.

For training methods in the few-shot settings, we provide the implementation details of phase-wise training and mix training. For phase-wise training, we utilize synthetic data and real data in two different phases, and the order of using synthetic data and real data also yields two variants. For syn→real, we first tune the classifier on synthetic data for 30 epochs, and then tune on real data for 30 epochs; and change the order for real→syn. For mix training, in each training iteration, we get a batch of real data input into the model and obtain the loss value of real part data, and also get a batch of synthetic data input into the model and get a loss value of synthetic part data, and then add two loss values as the final loss to do back propagation.

For training in the few-shot settings, we again use AdamW optimizer, weight decay of 0.1 and the cosine annealing rule. We use batch size of 32 for few-shot real images and 512 for synthetic images. For phase-wise training, we train for 30 epochs for each stage and use an initial learning rate of 0.002. For mix training, we train for 30 epochs and use an initial learning rate of 0.001, where the loss value from real data and synthetic data are added with a 1:1 ratio in each iteration. For image preprocessing, we adopt the same strategy as zero-shot settings.

C.5.3 PRE-TRAINING SETTING

For pre-training settings, we adopt the **LE** strategy in zero-shot settings for generating massive amount of diversified synthetic pre-training data. For downstream-aware settings, we generate synthetic data by language prompts constructed from CIFAR-100 label space through the word-to-sentence model, and we generate synthetic data of 1.2M, 2.4M, 3.6M. For downstream-agnostic settings, we generate from a generic label space of ImageNet-1K or ImageNet-2K, and data amount of 1.2M, 2.4M, 4M.

For training details of downstream-aware synthetic pre-training on CIFAR-100 label space, we use AdamW optimizer, weight decay of 0.9, batch size of 512, training epochs of 90, and the cosine annealing rule for adjusting learning rate. For the initial learning rate, we use $1e$-4 when pre-training from random initialization, and $1e$-5 when pre-training from ImageNet pre-trained weights. For data augmentation, we adopt random cropping, resizing, and random horizontal flip. For transfering on CIFAR-100 dataset, we train for 200 epochs and use a SGD optimizer with an initial learning rate of 0.003, which is multiplied by 0.2 at 60, 120, 160 epochs. We use batch size of 128 and weight decay of $5e$-4.

For downstream-agnostic synthetic supervised pre-training with ResNet50 backbone, we train for 90 epochs and use a SGD optimizer with an initial learning rate of 0.2 for training from random initialization and 0.001 for training from ImageNet pre-trained weights. We multiply the initial learning rate by 0.1 every 30 epochs. We use batch size of 512 and weight decay of 1e-4. For data augmentation, we also adopt random cropping, resizing, and random horizontal flip.

For downstream-agnostic synthetic supervised pre-training with DeiT-S backbone, we follow the training scripts for ImageNet pre-training and CIFAR-100 transfer learning in the official DeiT codebase[2] but only replace the pre-training data with our synthetic dataset. We do not change any hyper-parameters.

For downstream-agnostic synthetic self-supervised pre-training, *i.e*. Moco v2, we follow the hyper-parameters of the original implementation of the paper when training from random initialization. When initialized from ImageNet pre-trained weights, we use a small initial learning rate of 0.003 and keep other hyperparameters the same.

For transfer evaluation of object detection on PASCAL VOC 2012, we use the Faster R-CNN (Ren et al., 2015) detector and backbones are initialized by the pre-trained weights. All the setups follow the evaluation protocols in Moco (He et al., 2020).

REFERENCES

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 34:8780–8794, 2021.

Peng Gao, Shijie Geng, Renrui Zhang, Teli Ma, Rongyao Fang, Yongfeng Zhang, Hongsheng Li, and Yu Qiao. Clip-adapter: Better vision-language models with feature adapters. *arXiv preprint arXiv:2110.04544*, 2021.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, 2020.

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

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Bill Yuchen Lin, Wangchunshu Zhou, Ming Shen, Pei Zhou, Chandra Bhagavatula, Yejin Choi, and Xiang Ren. Commongen: A constrained text generation challenge for generative commonsense reasoning. *arXiv preprint arXiv:1911.03705*, 2019.

[2]https://github.com/facebookresearch/deit

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

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.

Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, pp. 8162–8171. PMLR, 2021.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J Liu, et al. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(140):1–67, 2020.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *NIPS*, 2015.

Chitwan Saharia, Jonathan Ho, William Chan, Tim Salimans, David J Fleet, and Mohammad Norouzi. Image super-resolution via iterative refinement. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

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

Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-language models. *International Journal of Computer Vision*, pp. 1–12, 2022.