A. Experimental setup and common design choices

A.1. Additional details on experimental setup

We conduct all our analyses with images of the default resolution, i.e., 224 or 299, on ImageNet models. Here we generate high-resolution images using the cascaded diffusion approach from Dhariwal et al. [10]. We first generate 64×64 size images using the first diffusion model and then upscale them to 256×256 resolution using the second diffusion model.

For feature extraction purposes, we use pretrained networks from the Timm [43] library. We extract features from the last convolutional layer for all networks. We consider five neighbors for AvgkNN computation and twenty neighbors for the local outlier factor. We use the implementation from PyOD [46] to calculate the local outlier factor. In our sampling process, we compute the hardness score at each time step. To calculate the hardness score, we first extract training data features at each timestep. Since the reverse process starts from white noise, we find that features from deep neural networks have extremely small variance at the start of reverse process. This makes the hardness score, thus gradients of the guidance loss, quite unstable at the start of the reverse process. We circumvent this issue by using an identity precision matrix. We use PyTorch [27] with an Nvidia A100 GPU cluster for our experiments.

A.2. Limitations of likelihood estimate from the diffusion model

It is straightforward to obtain an estimate of the likelihood of a given sample using the diffusion model. When choosing a metric to identify low-density regions, it is natural to ask whether the likelihood estimates from diffusion models can serve as this metric. To answer this question we calculate the negative log-likelihood (NLL) of real images from the validation set of the ImageNet dataset. We compare NLL with two commonly used metrics to measure the density of neighborhoods (Figure 11). We find that NLL shows poor correlation with both metrics, suggesting that it is not an effective predictor of neighborhood density.

Limitation of exact likelihood scores. While diffusion models only provide an approximate likelihood score, one can obtain exact likelihood score from autoregressive or flow-based models [14, 31]. We find that the aforementioned limitation of likelihood scores also also extend to *exact* likelihood values. We use DenseFlow [14], which provides *state-of-the-art* likelihood evaluation on ImageNet.

Surprisingly, the model assigns very high likelihood values to our low-density images (Table 2), even higher than highly photorealistic BigGAN images. We find that this observation is not limited to our synthetic samples, but a more *fundamental* characteristic of likelihood scores. To highlight it, we consider low-density real images that are poorly represented in training dataset, such such as sketches [41], renditions [15], and near-distribution images (ImageNet-O [16]).

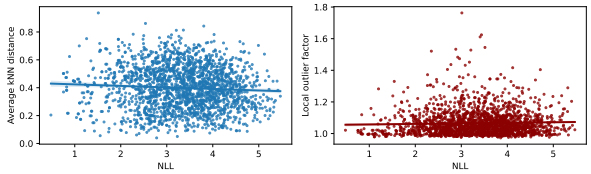


Figure 11. **Is NLL an effective measure of neighborhood density?** We compare the negative log-likelihood (NLL) estimates from the diffusion model with other commonly used metrics to measure data density. We find that NLL is poorly correlated with both of these metrics. Since NLL is computationally expensive to calculate for each image, we use 2K random images from the validation set of the ImageNet dataset for this analysis.

Similar to our low-density samples, DenseFlow assigns very high likelihood scores to all three novel variations of data (Table 2). Such variations (e.g., sketches) are rarely present in training data. Despite that, the model assigns a high likelihood to them. We also provide a qualitative comparison in Figure 12. Our observation is similar to the failure of exact likelihood scores on out-of-distribution data [25].

Table 2. **Quantitative evaluation.** State-of-the-art negative log-likelihood (NLL) evaluation using DenseFlow [14]. Lower value implies higher likelihood.

Dataset	Real	BigGAN	DDPM (baseline)	DDPM (ours)	Rendition	Sketch	ImageNet-O
NLL	3.4	3.1	3.3	2.8	2.5	1.2	2.9

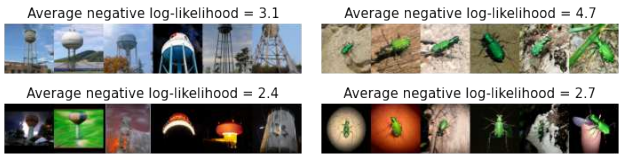
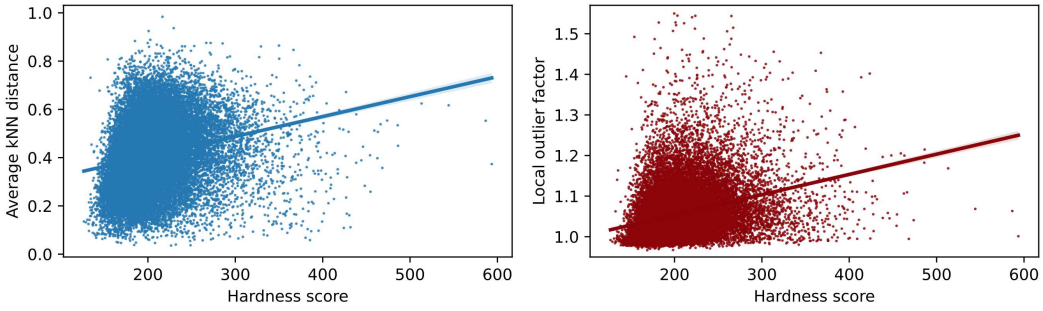


Figure 12. **Qualitative comparison.** *Top* row (baseline sampling) vs *Bottom* row (our sampling). Flow-based model surprisingly assigns much higher likelihood to our novel instances (lower value is higher).

A.3. Higher hardness score implies lower neighborhood density

In our sampling process, we maximize the hardness score of synthetic data. We argued that hardness score is a proxy to neighborhood density, thus maximizing it forces the model to generate low density samples. We provided the

(a) Visualizing images across the hardness scores axis for each class. H_a refers to the a_{th} percentile of hardness score. Classes are: goldfinch, water tower, container ship, hourglass, monarch butterfly, tiger beetle, zebra, and tennis ball.



(b) Correlation of hardness score with other metrics.

Figure 13. **Validating the effectiveness of hardness score.** To validate whether hardness score is a good proxy for neighborhood density, we first visualize images with increasing hardness scores and next show that it correlates with commonly used metrics to measure data density.

validation of its success in Figure 7. Now we delve deeper into why hardness score acts as a proxy to neighborhood density.

First we visualize real images across the spectrum of hardness score. Give a class index in the ImageNet validation set, we visualize its samples with lowest, moderate, and highest hardness scores (Figure 13a). From these images, it is evident that the difficulty of individual instances increases with hardness scores. We also look into the correlation of hardness score with other known density metrics (Figure 13b). We find that hardness score also have a positive correlation with other metrics.

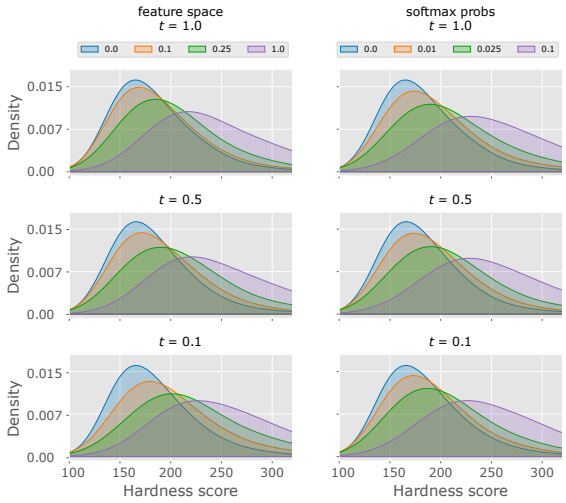


Figure 14. **Choice of loss function.** Loss function in feature space vs. in logits space.

A.4. Motivation to normalize gradients

Slightly different from the classifier guidance approach in Dhariwal et al. [10], we normalize classifier gradients before using them in the sampling process. We do so since it makes the scale of hyperparameters (α and β) independent of the magnitude of gradients of guiding losses (L_{g_1} and L_{g_2}). In particular, we observed that the magnitude of gradients in the diffusion process is often quite small, thus needing a very high scaling parameter. In addition, the magnitude of gradients also fluctuates with timesteps of the sampling process, thus potentially requiring a different scaling parameter at different timesteps. We normalize gradients to have unit ℓ_∞ norm, which ensures a consistent magnitude of gradients across timesteps. Thus normalization isolates the choice of scaling hyperparameters from gradients magnitude, making this choice much simpler.

A.5. Effect of different guiding loss functions

In our sampling process, our objective is to push synthetic images away from high-density neighborhoods. We achieve it by using a softmax-based loss function in the feature space of a pre-trained classifier. However, an equivalent loss function can be derived using softmax probabilities at the logit layer. Though both loss functions require a different scale of hyperparameters, they achieve similar results under properly calibrated scales (Figure 14). We make use of feature space because multiple additional metrics to measure density, such as kNN distance and local outlier factor, can be also easily calculated in the feature space.

A.6. Limitation of class embeddings smoothing

Previously, Li et al. [23] showed that one can manipulate class-embeddings of a pre-trained BigGAN model to improve the diversity of generated images. When approaching the task of low-density sampling, it is natural to test whether it can be achieved by simply controlling class embeddings. To test the effect of class-embeddings, we smooth class embeddings for a diffusion model on the ImageNet dataset. The network is trained with one-hot encoded class embeddings. When sampling, we smooth the embeddings by reducing the correct class probability to y_{max} and distribute the rest of the probability mass equally over all remaining classes. We find that the quality of synthetic images degrade very quickly with a reduction in y_{max} (Figure 15). Given this detrimental effect of smoothing in class embeddings, we chose to modify the sampling process itself, since the latter provides a much better control and quality of synthetic images.

A.7. Integration with fast sampling techniques

In the main paper, we discussed that with fast sampling approaches, our approach enjoys a similar trade-off as baseline sampling process. To support this claim, we provide a comparison of synthetic images sampled using DDIM [37] sampling process from both baseline and our sampling process in figure 16. We integrate the guiding loss in the DDIM sampling process in a similar manner as Dhariwal *et al*. [10].

B. Experimental results

B.1. Neighborhood density with different feature extractors

We use a ResNet-50 classifier, which is pretrained on ImageNet dataset, as feature extractor. Though this is very common choice of deep neural network, we further investigate whether our claims are robust to the choice of the feature extractors. To test it, we consider two more deep neural networks, namely Inception-V3 [40] and VGG [35]. We measure the neighborhood density in the feature space of both classifier and show that both classifier further validate the success of our approach (Figure 17).

B.2. Additional nearest neighbor pairs for visualization

To analyze whether the diffusion model is simply memorizing training data, we visualize the nearest neighbors of each synthetic image from the real images. We synthesize the synthetic images using our sampling process. To complement the top-16 synthetic and real images with the smallest pairwise distance in Figure 8, we present the next 64 pairs in Figure 18. In each pair, the left and right images corresponds to the synthetic and real image, respectively. For completeness, we also analyze nearest neighbors in pixel space (Figure 19). As expected, euclidean distance in pixel space doesn't correspond to semantic similarity between images and it is often highly biased toward

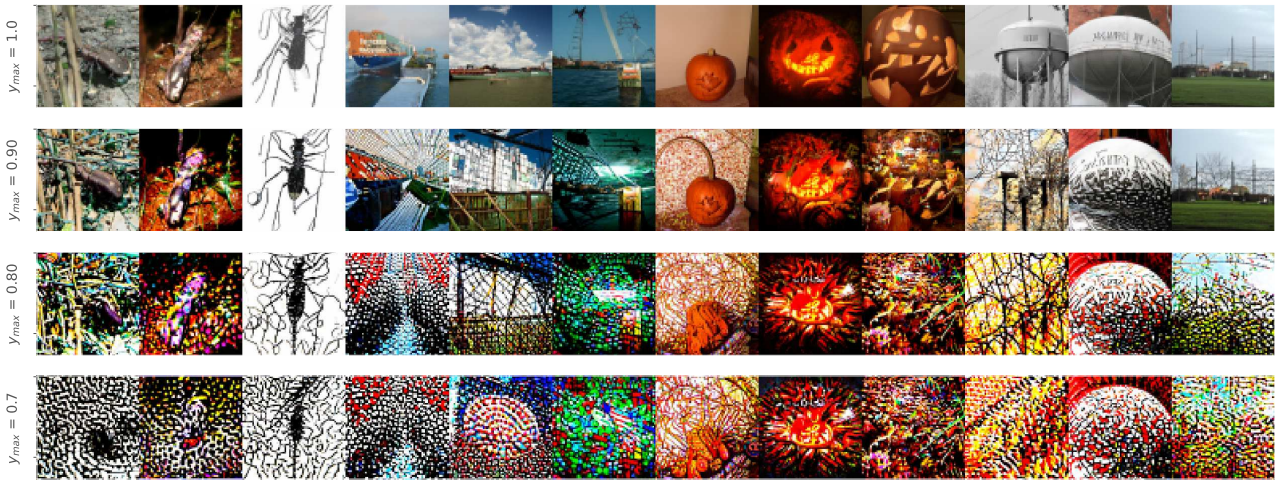


Figure 15. **Smoothing of class embeddings.** Demonstrating how smoothing of class embeddings leads to poor quality synthetic images with diffusion models.

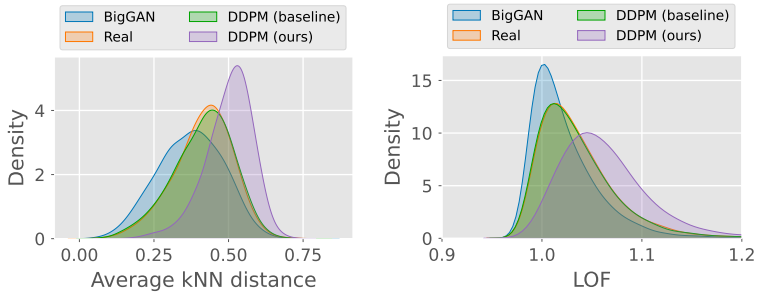
(a) $T = 10$ (b) $T = 20$ (c) $T = 50$

Figure 16. **Fast sampling.** We integrate our guiding objective in the fast DDIM sampling process [37]. Top two rows correspond to the baseline DDIM sampling approach while bottom two correspond to our approach. We use the identical starting latent vectors for both approaches and across the three choices of the number of sampling steps.

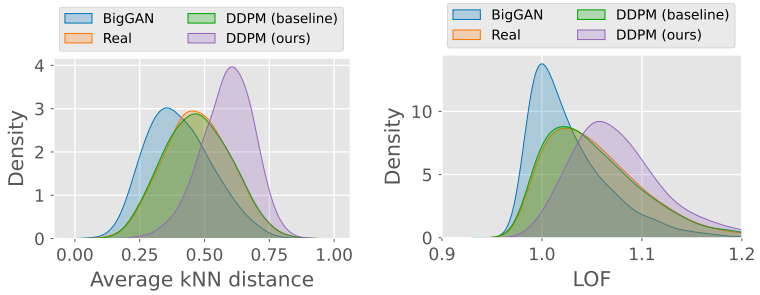
background similarities between synthetic and real images.

B.3. Comparing our samples with baseline sampling process

We present additional images to compare the baseline and our sampling process in figure 20 and 21.



(a) Inception-v3



(b) VGG19

Figure 17. **Comparing neighborhood density across different choices of feature extractors.** We use two additional feature extractors, namely Inception-v3 and VGG19.

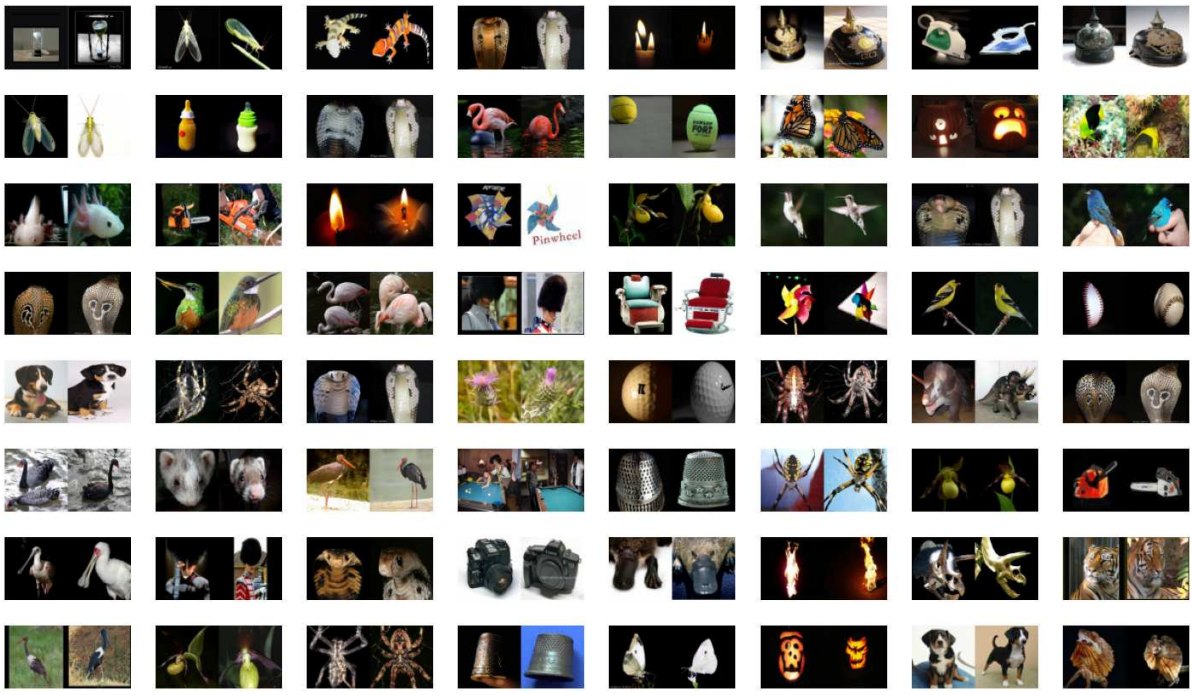


Figure 18. Nearest neighbour pairs of real and synthetic data with lowest pairwise distance. In each pair, the left and right image correspond to the synthetic and real image, respectively.

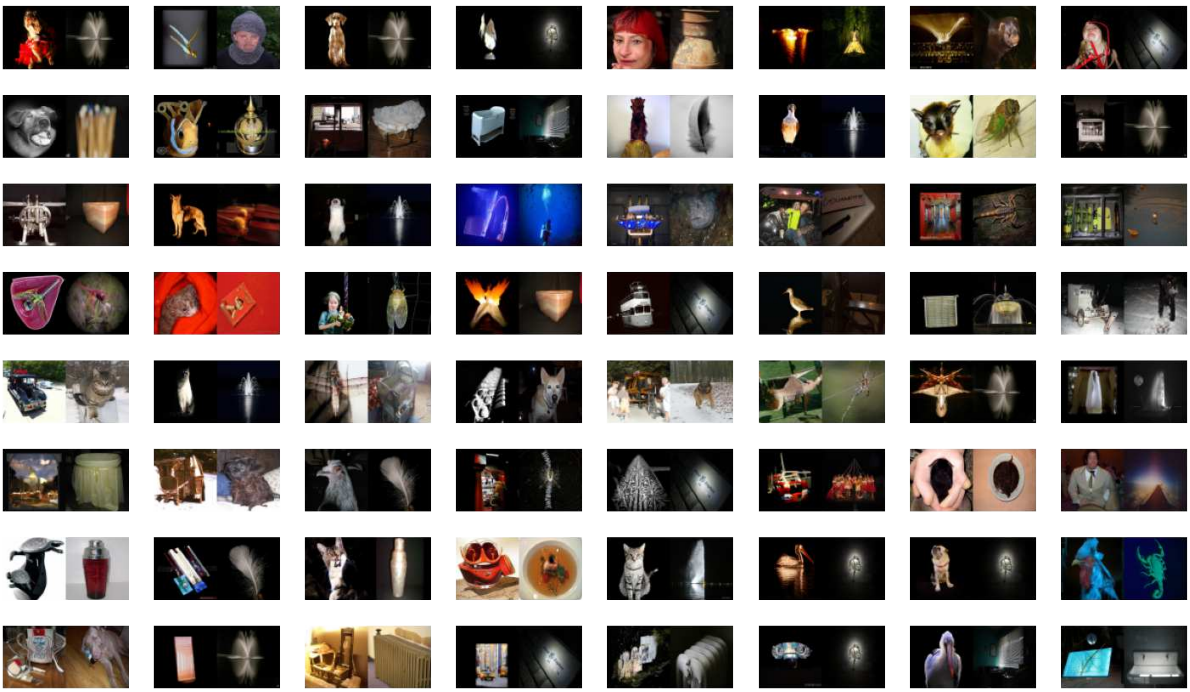
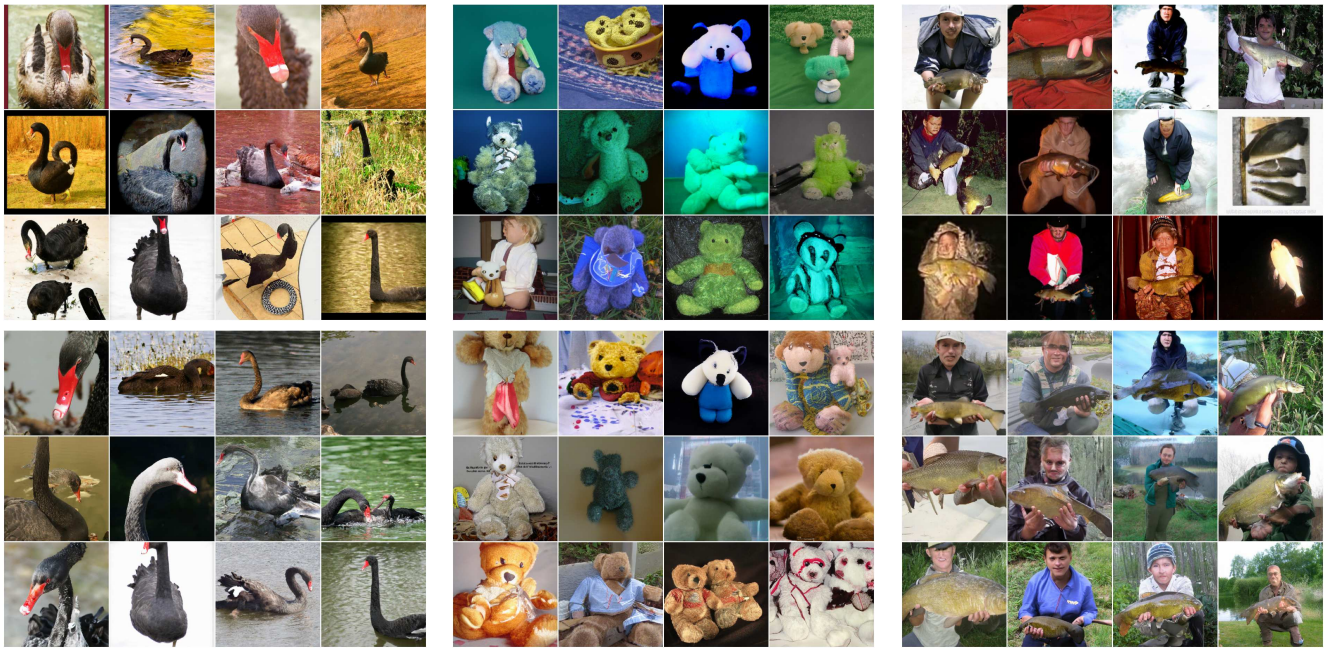


Figure 19. Nearest neighbour pairs of real and synthetic data with lowest pairwise distance in pixel space. In each pair, the left and right image correspond to the synthetic and real image, respectively.

Figure 20. Synthetic images from the baseline sampling process (left) and our approach (right) for each class on the CIFAR-10 dataset. We use identical random seed for both approaches.



(a) ImageNet (class 0 and 7) (b) ImageNet (class 73) and CIFAR-10 (c) ImageNet (class 73) and CIFAR-10

Figure 21. Synthetic images from the baseline sampling process (bottom) and our approach (top) for few classes on the ImageNet dataset. We use identical random seed for both approaches.