CS772 Project Report

Bedant Sharma
Roll No: 210260
bedants21@iitk.ac.in

Bhavaj Singla
Roll No: 210265
bhavajs21@iitk.ac.in

Pranjal Bhardwaj
Roll No: 210741
pranjalb21@iitk.ac.in

Sayeedul Islam Sheikh
Roll No: 210953
sayeedul21@iitk.ac.in

Abstract

This paper explores a novel approach to generating high-quality out-of-distribution (OOD) images by combining the strengths of diffusion models and Langevin dynamics. Outlier data generation has emerged as a valuable alternative to manual data collection and cleaning, which are often labor-intensive and costly. Building upon the DREAM-OOD framework[2], we propose improvements to both the sampling strategy and the latent space representation to enhance the diversity and realism of generated OOD samples. Our method leverages inverse stochastic gradient Langevin dynamics (SGLD) to better explore low-density regions of the data distribution. The proposed enhancements aim to make OOD image generation more effective and computationally efficient. The code is publicly available at: https://github.com/sayeed02021/CS772_Course_Project.

1 Literature Review

The paper *"Dream the Impossible: Outlier Imagination with Diffusion Models"* (https://arxiv.org/pdf/2309.13415) presents a method to generate out-of-distribution (OOD) photo-realistic outlier data for classes without the need for manually collected or labeled outlier datasets.

OOD detection is crucial for machine learning models to avoid making confident predictions on unfamiliar inputs. However, collecting such data is labor-intensive, this paper presents a way to do so! There are two phases to this work:

1.1 Phase 1: Learning a Text-Conditioned Latent Space

We learn a class-conditioned embedding of in-distribution (ID) data aligned with the diffusion model's text embeddings, i.e., the CLIP embedding.

The paper uses a ResNet[3] with the last few layers removed and a fully connected layer of 768 dimensions to produce the embeddings. This serves as the feature encoder.

Then, with contrastive loss, we train this encoder to match the class token embeddings $T(y)$ from the diffusion model.

contrastive loss is give by :

$$\mathcal{L} = \mathbb{E}_{(x,y)\sim D}[-\log \frac{\exp(\mathcal{T}(y)^T * z/t}{\sum_{j=1}^{C} \exp(\mathcal{T}(y_j)^T * z/t}] \tag{1}$$

where $z = \frac{h_\theta(x)}{\|h_\theta(x)\|_2}$ is a L_2 normalized image embedding, and t is the temperature, D is ID data, y is class label, $\mathcal{T}(y)$ is m dimensional token embeddings. Specifically, denote $h_\theta : \mathcal{X} \to \mathbb{R}^m$ as a feature encoder that maps an input $x \in \mathcal{X}$ to the image embedding $h_\theta(x)$,

Class conditionals are modeled as von Mises-Fisher distributions.

$$p_m(z; \mu_c, k) = Z_m(k) \exp(k \mu_c^T z), \tag{2}$$

where μ_c is the class centroid with unit norm, $k \geq 0$ controls the extent of class concentration, and $Z_m(k)$ is the normalization. The probability of the feature vector z belonging to class c is:

$$P(y = c \mid \mathbf{z}; \{\kappa, \mu_j\}_{j=1}^C) = \frac{Z_m(\kappa) \exp(\kappa \mu_c^\top \mathbf{z})}{\sum_{j=1}^C Z_m(\kappa) \exp(\kappa \mu_j^\top \mathbf{z})} \tag{3}$$

Alternatively, using a temperature-scaled dot product version:

$$\frac{\exp(\mu_c^\top \mathbf{z}/t)}{\sum_{j=1}^C \exp(\mu_j^\top \mathbf{z}/t)} \tag{4}$$

1.2 Phase 2: Using the Latent Space to Generate Outliers

From the latent space, we use the diffusion model to generate outliers.

The paper samples new embeddings outside the in-distribution regions of the von Mises-Fisher distribution and decodes them into images using the diffusion model.

The paper first identifies the boundary kernel using k-NN distance. We calculate the k-NN distance with respect to Z:

$$d_k(z', Z) = \|z' - z^{(k)}\|_2 \tag{5}$$

where $z^{(k)}$ is the k-th nearest neighbor in Z. Embeddings with large k-NN distances are likely to be on the boundary of the ID data.

Then, we sample more outlier points using a Gaussian kernel on these boundary points. We select the real outliers among them by using k-NN again. Finally, for those embeddings, we generate synthetic outliers.

To obtain the outlier images in the pixel space, we decode the sampled outlier embeddings v via the diffusion model. In practice, this is done by replacing the original token embedding z_y or $T(y)$ with the sampled new embedding v. Different from the vanilla prompt-based generation,

$$x \sim P(\mathbf{x}|\mathbf{z}_y)$$

our outlier imagination is mathematically reflected by:

$$x_{ood} \sim P(\mathbf{x}|\mathbf{v})$$

The generated synthetic OOD images $\mathbf{x}_{\text{ood}}$ can be used for regularizing the training of the classification model :

$$\mathcal{L}_{\text{ood}} = \mathbb{E}_{\mathbf{x}_{\text{ood}}} \left[-\log \left(\frac{1}{1 + \exp(\phi(E(f_\theta(\mathbf{x}_{\text{ood}}))))} \right) \right] + \mathbb{E}_{\mathbf{x} \sim P_{\text{in}}} \left[-\log \left(\frac{\exp(\phi(E(f_\theta(\mathbf{x}))))}{1 + \exp(\phi(E(f_\theta(\mathbf{x}))))} \right) \right] \tag{6}$$

where $\phi(\cdot)$ is a three-layer nonlinear MLP function with the same architecture as VOS, $E(\cdot)$ denotes the energy function, and $f_\theta(x)$ denotes the logit output of the classification model. In other words, the loss function takes both the ID and generated OOD images, and learns to separate them explicitly. The overall training objective combines the standard cross-entropy loss along with an additional loss for OOD regularization:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{CE}} + \beta \cdot \mathcal{L}_{\text{ood}} \tag{7}$$

where β is the weight of the OOD regularization. $\mathcal{L}_{\text{CE}}$ denotes the cross-entropy loss on the ID training data. During testing, we use the output of the binary logistic classifier for OOD detection.

1.3 Comparison with Prior Work

Unlike prior work, **DREAM-OOD**:

- Requires only in-distribution (ID) data, without needing real outlier datasets.
- Generates interpretable and visible out-of-distribution (OOD) images directly in the pixel space.
- Provides a mechanism to understand what kinds of outliers the model is learning to detect.

2 Problem Description and Motivation

We observed that the feature encoder trained on CIFAR-100 was performing well on the training data in terms of correctly identifying class IDs, but showed significantly degraded performance on the test data. The ResNet paper also reported that ResNet-34 did not perform very well on the CIFAR-100 test set the results show about 78% accuracy, but our results exhibited even more degradation i.e. 60% accuracy due to the loss of classification head.

To address this, we plan to use a more powerful feature encoder, such as deeper ResNet variants or EfficientNet models. According to Papers with Code, EfficientNet[7] achieves the highest test accuracy on CIFAR-100. However, since our method requires a 768-dimensional feature representation, some loss of information is expected during dimensionality alignment.

Furthermore, we noticed that although we have access to the exponential (von Mises-Fisher) distribution for each class, we still rely on k-Nearest Neighbors (k-NN) to generate out-of-distribution (OOD) samples — first to determine boundary regions and then to verify whether a generated sample is truly an outlier.

We believe that sampling methods could be employed to generate outlier samples by twisting the algorithm to give outliers from the exponential class conditionals, potentially eliminating the need for repeated k-NN calculations. This would allow us to:

- Reduce memory overhead by storing fewer points.
- Generate more diverse and representative OOD samples.
- Operate directly with the exponential class conditionals without relying on full data storage for k-NN operations.

3 Novelty

3.1 Changes in Encoder Backbone

To overcome the limitations of ResNet-34, which we found to be insufficiently expressive for generating high-quality feature embeddings, we experimented with alternative encoder backbones. Specifically, we employed ResNet-152 (approximately 60 million parameters) and EfficientNetV2 (ranging from 24 million to over 480 million parameters, depending on the variant). Instead of utilizing pretrained weights, we opted to train these models from scratch to ensure that their feature representations were optimally aligned with the CIFAR-100 dataset. This decision was driven by the domain-specific nature of the data and the need for feature embeddings that are well-suited to our downstream tasks.

3.2 Improvement in sampling process

3.2.1 Inverse Rejection Sampling Review

To sample outside the distribution (OOD), we invert the logic of rejection sampling. Accept samples with probability inversely proportional to their likelihood under the data distribution $p(x)$

$$A_{\text{inverse}} \propto \frac{1}{\epsilon + p(x)} \tag{8}$$

ϵ is a small constant to aviod dividing by zero. The higher the probability under the data distribution, the less likely we accept it. This encourages sampling from low-density, OOD regions.

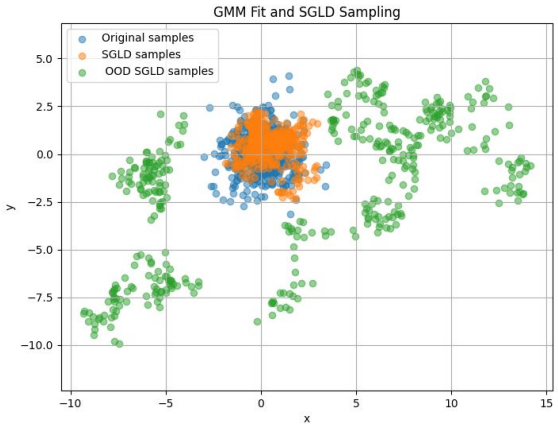


Figure 1: SGLD Results on 2D toy dataset

3.2.2 Inverse SGLD

To sample out-of-distribution data, we reverse the gradient direction. Instead of ascending the log-likelihood, we descend it to explore low-density (OOD) regions. [1]

Inverse SGLD Update Rule:

$$x_{t+1} = x_t - \frac{\eta}{2}\nabla_x \log \hat{p}(x_t) + \sqrt{\eta} * \epsilon \tag{9}$$

We flip the sign of the gradient. This moves the samples away from high-density regions.

Why this works: Inverse SGLD pushes the particle away from those modes—toward the tails or outside the support of $p(x)$, which is where OOD samples live. Think of this as a form of "gradient repulsion" from the data distribution.

Implementation Details: For all text aligned embeddings of images belonging to a particular class we fit a Gaussian mixture model with number of clusters set to 1. This gives us the posterior distribution for the text embeddings: $z \sim \mathcal{N}(z|\mu_{GMM}, \sigma^2_{GMM} I_D)$. We use gradient of this distribution to find the out-of-distribution embeddings using Eq. 9. Figure 1 shows the results on a 2D toy dataset where original samples were randomly generated from Standard Normal Distribution.

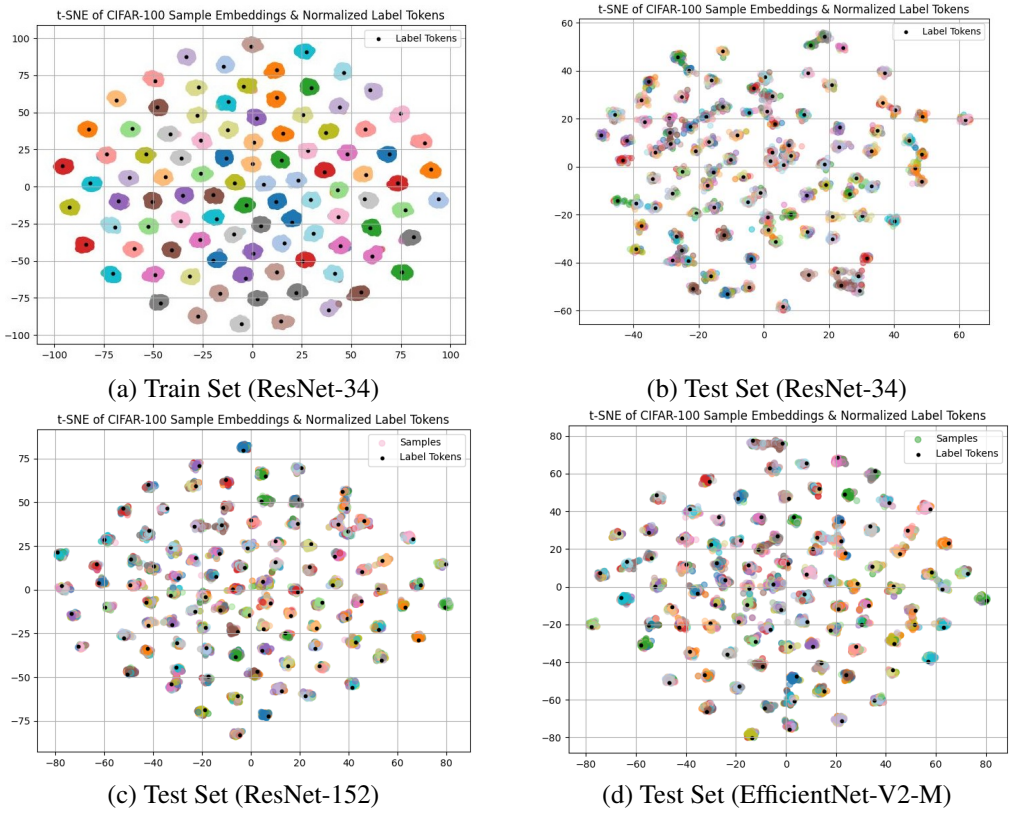
4 Tools Used

PyTorch, Stable Diffusion[6], Dream-OOD framework[2]

5 Lessons Learned

This project provided valuable insights into the challenges and subtleties of out-of-distribution (OOD) image generation using generative models. Several key takeaways emerged from our work:

- **Sensitivity of Sampling Algorithms:** We found that sampling techniques such as inverse SGLD are highly sensitive to hyperparameters like the step size η. Tuning this parameter was crucial for achieving meaningful exploration of low-density regions while maintaining stability.
- **Limitations of k-NN for Outlier Detection:** Although effective, the reliance on k-nearest neighbors (k-NN) for identifying boundary points and validating outliers is computationally expensive and memory-intensive. This motivated us to explore sampling-based alternatives grounded in the underlying probabilistic model.



(a) Train Set (ResNet-34)

(b) Test Set (ResNet-34)

(c) Test Set (ResNet-152)

(d) Test Set (EfficientNet-V2-M)

Figure 2: t-SNE plots of 768-dimensional features. Black dots represent class ID embeddings.

- **Latent Space Structure Matters:** The quality of OOD samples was heavily influenced by the learned latent space. Poorly aligned or insufficiently expressive embeddings led to unrealistic or redundant samples. This emphasized the importance of a well-trained feature encoder and meaningful latent representations.
- **Interpretability Through Image Generation:** Unlike many OOD detection methods that operate solely in the embedding space, generating pixel-level OOD images offered a tangible and interpretable way to understand what "out-of-distribution" means in practice. This added a valuable qualitative dimension to our analysis.
- **Code and Experimentation Infrastructure:** Implementing and testing novel sampling strategies required a modular and reproducible codebase. This project reinforced the importance of clean design, experiment tracking, and open-sourcing code to support future research.

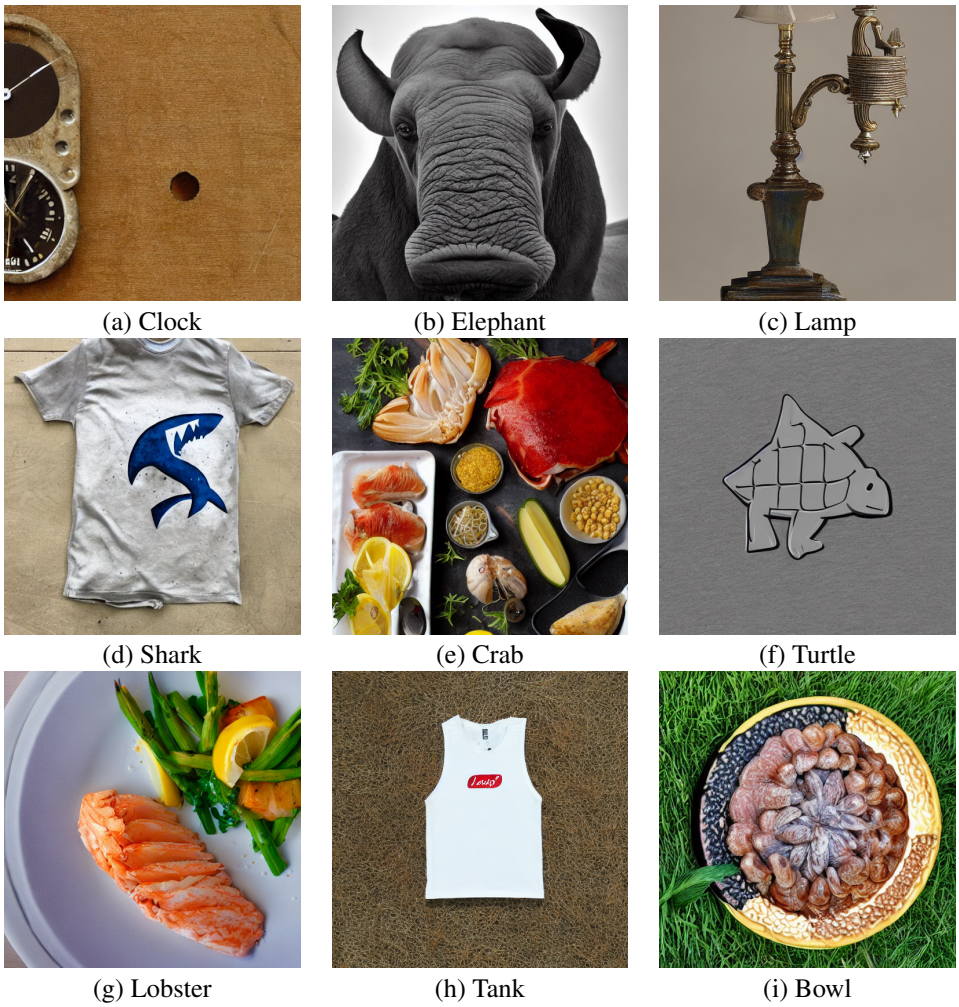
Overall, this work deepened our understanding of generative modeling, OOD detection, and the interplay between sampling dynamics and representation learning. These lessons will inform not only future iterations of this project, but also our broader research in robust and interpretable machine learning.

6 Experiment Results

6.1 Part 1: Improving the Feature Encoder

In this section, we evaluate the effect of using a better feature encoder. The following t-SNE plots visualize the 768-dimensional features extracted by different encoders. Black dots represent the class ID embeddings.

Figure 2 shows four plots:



(a) Clock (b) Elephant (c) Lamp

(d) Shark (e) Crab (f) Turtle

(g) Lobster (h) Tank (i) Bowl

Figure 3: Outlier images generated from inverse SGLD sampling

- **(a)** Train set with ResNet-34: achieves ∼99% accuracy, similar to the results in the original paper.
- **(b)** Test set with ResNet-34: shows significant overlap and poor separation between class features.
- **(c)** Test set with ResNet-152: shows improved separation over ResNet-34.
- **(d)** Test set with EfficientNet-V2-M: surprisingly performs worse than ResNet-152.

We hypothesize that EfficientNet-V2-M underperforms because the entire network is trained from scratch without freezing any layers, in order to align the feature embeddings with the CLIP embeddings. Due to limited GPU resources, we could only train for 200 epochs. Larger models may require a different set of learning rates and hyperparameters, which we did not fine-tune — we used the same hyperparameters as in the original DREAM-OOD paper (optimized for ResNet).Also Larger models will require more training epochs.

Though we see better separation in the later test sets, it is not the best because we were not able to train it further. However, it is better than the original separation in the original model, and the idea of using a better feature encoder is useful.

6.2 Generated OOD images

Figure 3 shows a few generated outlier images from the Stable Diffusion[6] model. Although the code for generating outlier images in the pixel space is not present in the github repository of Du et al. we follow the instructions given in the paper. We generate 500, 768 dimensional samples from each class, and then randomly select 3 samples for generating image using the Stable Diffusion model. These samples serve as the token embedding for that class. We replace the original token embedding in the CLIP[5] text embedding table with the sampled embeddings, and then generate text embeddings of shape 77×768 where 77 is the context length of the CLIP text encoder model. The text embeddings are then supplied down the pipeline to generate final pixel space images. For this work we make use of the `diffusers`[8] library.

7 Possible future work

7.1 Changing the distance kernel in the loss function

In the original paper, the loss function employs a Euclidean distance kernel, which captures both magnitude and direction between vectors. However, the provided implementation uses a cosine similarity kernel, which only accounts for directional alignment. Although we adapted the implementation to incorporate the Euclidean kernel, this modification resulted in increased error rates. Future work could focus on designing or identifying a more effective distance metric or hybrid kernel that better balances directionality and magnitude, potentially leading to improved performance.

7.2 Hyperparameter Sensitivity in Inverse SGLD/ Different Proposal

The performance and behavior of our sampling algorithm, particularly in the context of inverse SGLD, is highly sensitive to the step size hyperparameter η. The update rule is given by:

$$x_{t+1} = x_t - \frac{\eta}{2}\nabla_x \log \hat{p}(x_t) + \sqrt{\eta} \cdot \epsilon \tag{10}$$

We noticed that although above algorithm works smoothly for toy datasets, the image embeddings have very low covariance values($\sim 10^{-4} - 10^{-5}$), and hence η of the order of `1e-3` does not produce meaningful images. Hence for actual sampling we use very low η(comparable to covariance values), but using very low η slows down the sampling process, as we need to exclude more data first before arriving sampling from low pdf regions. For future work, instead of using a constant η, one should change the value of η after every iteration so that it is high in high probability density regions but small in low probability density regions.

We can also skip SGLD and instead use a different proposal that is more stable than it.

Barker's Proposal[4] instead of moving the mean of the Normal proposal in the direction of the gradient, skews the normal proposal in the direction of the gradient.

$$q_B(x, y) = \frac{2}{1 + e^{-(y-x)}} \nabla \log f(x)\, \mu_\sigma(y - x) \tag{11}$$

and sample from low lying probability areas to get x_{t+1}

8 Contribution of Team Members

- **Bedant Sharma:** Handled diffusion model integration and image generation from outlier embeddings. Also worked on visualizations and formatting the final report.
- **Bhavaj Singla:** Focused on training the feature encoder and analyzing the latent space representations. Contributed to data preprocessing and experimental setup.
- **Pranjal Bhardwaj:** Handled diffusion model integration and image generation from outlier embeddings. Also worked on visualizations and formatting the final report.
- **Sayeedul Islam Sheikh:** Worked on implementing inverse SGLD and hyperparameter tuning for sampling and used stable diffusion to get images from embeddings. Also contributed

to drafting technical sections of the report and reviewing related literature. Also maintained code repository and reproducibility scripts.

All members participated in discussions, ideation, debugging, and final presentation of the project.

References

[1] Giovanni Bussi and Michele Parrinello. Accurate sampling using langevin dynamics. *Physical Review E—Statistical, Nonlinear, and Soft Matter Physics*, 75(5):056707, 2007.

[2] Xuefeng Du, Yiyou Sun, Jerry Zhu, and Yixuan Li. Dream the impossible: Outlier imagination with diffusion models. *Advances in Neural Information Processing Systems*, 36:60878–60901, 2023.

[3] 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.

[4] Samuel Livingstone and Giacomo Zanella. The barker proposal: Combining robustness and efficiency in gradient-based mcmc. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 84(2):496–523, 2022.

[5] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021.

[6] 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.

[7] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019.

[8] Patrick von Platen, Suraj Patil, Anton Lozhkov, Pedro Cuenca, Nathan Lambert, Kashif Rasul, Mishig Davaadorj, Dhruv Nair, Sayak Paul, William Berman, Yiyi Xu, Steven Liu, and Thomas Wolf. Diffusers: State-of-the-art diffusion models. `https://github.com/huggingface/diffusers`, 2022.