Blue noise for diffusion models: Supplemental document

XINGCHANG HUANG, MPI Informatics, VIA Center, Germany
CORENTIN SALAÜN, MPI Informatics, Germany
CRISTINA VASCONCELOS, Google DeepMind, UK
CHRISTIAN THEOBALT, MPI Informatics, VIA Center, Germany
CENGIZ ÖZTIRELI, Google Research, University of Cambridge, UK
GURPRIT SINGH, MPI Informatics, VIA Center, Germany

In this document, we present additional details and results.

ACM Reference Format:
Xingchang Huang, Corentin Salaün, Cristina Vasconcelos, Christian Theobalt, Cengiz Öztireli, and Gurprit Singh. 2024. Blue noise for diffusion models: Supplemental document. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers '24 (SIGGRAPH Conference Papers '24), July 27-August 1, 2024, Denver, CO, USA.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3641519.3657435

1 DERIVATION OF OUR BACKWARD PROCESS

The following shows the derivation of Eq. (7) in the main paper. The goal is to compute $\mathbf{x}_{t-1}$ using $\mathbf{x}_t$ based on the definition of the backward process. We start from:

$$\mathbf{x}_{t-1} = \alpha_{t-1}(L_{t-1}\boldsymbol{\epsilon}) + (1-\alpha_{t-1})\mathbf{x}_0 \tag{1}$$

Next, we need to construct $\mathbf{x}_t$ on the right hand side (RHS) as the following:

$$\begin{aligned}\mathbf{x}_{t-1} &= \alpha_{t-1}(L_{t-1}\boldsymbol{\epsilon}) + (1-\alpha_{t-1})\mathbf{x}_0\\ &= (\alpha_{t-1}+\alpha_t-\alpha_t)(L_{t-1}\boldsymbol{\epsilon}) + (1-\alpha_{t-1}+\alpha_t-\alpha_t)\mathbf{x}_0\\ &= \alpha_t(L_{t-1}\boldsymbol{\epsilon}) + (\alpha_{t-1}-\alpha_t)(L_{t-1}\boldsymbol{\epsilon}) + (1-\alpha_t)\mathbf{x}_0 + (\alpha_t-\alpha_{t-1})\mathbf{x}_0\end{aligned} \tag{2}$$

The L_{t-1} term can be expanded as:

$$\begin{aligned}L_{t-1} &= \gamma_{t-1}L_w + (1-\gamma_{t-1})L_b\\ &= L_b + \gamma_{t-1}(L_w - L_b)\\ &= L_b + (\gamma_{t-1}+\gamma_t-\gamma_t)(L_w-L_b)\\ &= L_b + \gamma_t(L_w-L_b) + (\gamma_{t-1}-\gamma_t)(L_w-L_b)\end{aligned} \tag{3}$$

Based on above and $\mathbf{x}_t = \alpha_t(L_t\boldsymbol{\epsilon}) + (1-\alpha_t)\mathbf{x}_0$, we have:

$$\begin{aligned}\mathbf{x}_{t-1} &= \alpha_t(L_t\boldsymbol{\epsilon}) + \alpha_t(\gamma_{t-1}-\gamma_t)(L_w-L_b)\boldsymbol{\epsilon} + (\alpha_{t-1}-\alpha_t)(L_t\boldsymbol{\epsilon})\\ &\quad + (\alpha_{t-1}-\alpha_t)(\gamma_{t-1}-\gamma_t)(L_w-L_b)\boldsymbol{\epsilon} + (1-\alpha_t)\mathbf{x}_0 + (\alpha_t-\alpha_{t-1})\mathbf{x}_0\\ &= \mathbf{x}_t + (\alpha_t-\alpha_{t-1})(\mathbf{x}_0 - L_t\boldsymbol{\epsilon}) + (\gamma_{t-1}-\gamma_t)(\alpha_{t-1})(L_w-L_b)\boldsymbol{\epsilon}\\ &= \mathbf{x}_t + (\alpha_t-\alpha_{t-1})(\mathbf{x}_0 - L_t\boldsymbol{\epsilon}) + (\gamma_t-\gamma_{t-1})(\alpha_{t-1})(L_b-L_w)\boldsymbol{\epsilon}\end{aligned} \tag{4}$$

We ensure that $\alpha_t \geq \alpha_{t-1}$, $\gamma_t \geq \gamma_{t-1}$, $0 \leq \alpha_t \leq 1$, $0 \leq \gamma_t \leq 1$ for different schedulers.

The training procedure of our method is based on the derived backward process. For the terms $L_t\boldsymbol{\epsilon}$, $(L_w - L_b)\boldsymbol{\epsilon}$, when L_w represents the identity matrix, we do not need to actually perform the time-consuming matrix-vector multiplication for $L_w\boldsymbol{\epsilon}$, as in this case $\boldsymbol{\epsilon} = L_w\boldsymbol{\epsilon}$. Therefore, this only term that can introduce overhead is $L_b\boldsymbol{\epsilon}$. We experimentally observed that this overhead is negligible for image resolutions at 64^2, 128^2, 256^2.

Extension to DDIM. Here we show that our time-varying noise model can also extend to DDIM, following the procedure shown in Heitz et al. [2023]. First, we define:

$$y_t = (1-\beta_t)x_0 + \beta_t(L_t\epsilon) \tag{5}$$

where y_t, β_t are temporally introduced for easier derivations and will be replaced back by x_t, α_t later. Then, we have:

$$\begin{aligned}y_{t-1} &= (1-\beta_{t-1})x_0 + \beta_{t-1}(L_{t-1}\epsilon)\\ &= (1-\beta_{t-1})\frac{y_t - \beta_t(L_t\epsilon)}{1-\beta_t} + \beta_{t-1}(L_{t-1}\epsilon)\\ &= \frac{1-\beta_{t-1}}{1-\beta_t}y_t - \frac{(1-\beta_{t-1})\beta_t(L_t\epsilon)}{1-\beta_t} + \beta_{t-1}(L_{t-1}\epsilon)\end{aligned} \tag{6}$$

Based on Eq. (3), we can expand $\beta_{t-1}(L_{t-1}\epsilon)$ as two terms:

$$\begin{aligned}\beta_{t-1}(L_{t-1}\epsilon) &= \beta_{t-1}(L_b + \gamma_t(L_w-L_b) + (\gamma_{t-1}-\gamma_t)(L_w-L_b)\epsilon\\ &= \beta_{t-1}(L_t\epsilon) + \beta_{t-1}(\gamma_{t-1}-\gamma_t)(L_w-L_b)\epsilon\end{aligned} \tag{7}$$

By merging the second and third terms on the right-hand side (RHS) for the expanded version of Eq. (6), we can get:

$$\begin{aligned}y_{t-1} &= \frac{1-\beta_{t-1}}{1-\beta_t}y_t - \frac{L_t\epsilon(\beta_t-\beta_{t-1})}{1-\beta_t} + \beta_{t-1}(\gamma_{t-1}-\gamma_t)(L_w-L_b)\epsilon\\ &= \frac{1-\beta_{t-1}}{1-\beta_t}(y_t - L_t\epsilon) + L_t\epsilon + \beta_{t-1}(\gamma_{t-1}-\gamma_t)(L_w-L_b)\epsilon\end{aligned} \tag{8}$$

Next, we let $\beta_t = \frac{\sqrt{1-\overline{\alpha}_t}}{\sqrt{\overline{\alpha}_t}+\sqrt{1-\overline{\alpha}_t}}$ and $y_t = \frac{x_t}{\sqrt{\overline{\alpha}_t}+\sqrt{1-\overline{\alpha}_t}}$ where $\overline{\alpha}_t = \prod_{s=1}^{t}\alpha_s$. Lastly, we can derive the backward process as the following:

$$\begin{aligned}x_{t-1} &= \frac{\sqrt{\overline{\alpha}_{t-1}}}{\sqrt{\overline{\alpha}_t}}(x_t - \frac{\sqrt{\overline{\alpha}_t}\sqrt{1-\overline{\alpha}_{t-1}} - \sqrt{\overline{\alpha}_{t-1}}\sqrt{1-\overline{\alpha}_t}}{\sqrt{\overline{\alpha}_{t-1}}}L_t\epsilon)\\ &\quad + (\gamma_t-\gamma_{t-1})\sqrt{1-\overline{\alpha}_{t-1}}(L_b-L_w)\epsilon\end{aligned} \tag{9}$$

In this case, the network needs to learn $L_t\epsilon$ and $\sqrt{1-\overline{\alpha}_{t-1}}(L_b-L_w)\epsilon$.

2 DATASETS, NETWORK ARCHITECTURE AND TRAINING DETAILS

We use the following datasets for unconditional image generation: CelebA (64^2 and 128^2 resolutions, 30,000 training images) [Lee et al. 2020], AFHQ-Cat (64^2 and 128^2 resolutions, 5,153 training images) [Choi et al. 2020] and LSUN-Church (64^2 resolution, 30,000 out of 126,227 training images) [Yu et al. 2015]. These partitions were set in order to replicate the training conditions of the methods compared here. For conditional image generation, we conduct experiments on image super-resolution using CelebA from 64^2 to 128^2, 32^2 to 128^2 and LSUN-Church from 32^2 to 128^2. For both datasets, we use 25,000 images for training and 5,000 images for evaluation.

We use the diffuser library [von Platen et al. 2022] to build the 2D U-Net [Ronneberger et al. 2015] architecture with 6, 7 down- and up-sampling layers with skip connections for image resolution of 64^2, 128^2, respectively. The number of channels we use are (128, 128, 256, 256, 512, 512) for 6 layers and (128, 128, 128, 256, 256, 512, 512) for 7 layers. Self-attention [Vaswani et al. 2017] module is added in the second last of the down-sampling layer and the second of the up-sampling layer.

For network training details, we show the number of epochs and batch size used for training on different datasets for DDIM, IADB and Ours in Table 1. Table 1 also includes the τ (Eq. (9) in the main paper) values we use for all experiments. For the CelebA (64^2) experiment, we use the exact linear scheduler for γ_t.

Table 1. Network training details and the choices of τ (Eq. (9) in the main paper) for our method. For the CelebA (64^2) experiment, we use the exact linear scheduler for γ_t. We use latent diffusion model (LDM) [Rombach et al. 2022] for the AFHQ-Cat (512^2) experiment.

Dataset	#Epochs	Batchsize	τ
AFHQ-Cat (64^2)	1,000	256	1,000
AFHQ-Cat (128^2)	1,000	128	0.2
AFHQ-Cat (LDM, 512^2)	1,000	256	1,000
CelebA (64^2)	1,000	256	(linear)
CelebA (128^2)	700	128	0.2
LSUN-Church (64^2)	1,000	256	1,000
CelebA ($64^2 \longrightarrow 128^2$)	400	128	0.2
CelebA ($32^2 \longrightarrow 128^2$)	80	64	0.2
LSUN-Church ($32^2 \longrightarrow 128^2$)	80	64	0.2

3 ADDITIONAL RESULTS

Gaussian blue noise at different resolutions. Figure 1 visualizes Gaussian blue noise masks at different resolution using our padding strategy. For the masks at resolution 128^2 and 256^2, the seams between the padded 64^2 tiles are not visible and the property of blue noise is still preserved. In addition, we show in Fig. 2 that padding/tiling with the same 64^2 Gaussian blue noise mask results in repetitive noise patterns, as well as structural artifacts in the frequency spectra.

Table 2. Early stopping tests using Gaussian noise only and Gaussian blue noise only on AFHQ-Cat(128^2). T_e represents the time step we apply early stopping. When $T_e = 0$, the model falls back to the full backward process.

	Ours (Gaussian noise only)			Ours (Gaussian blue noise only)		
T_e	FID (↓)	Precision (↑)	Recall (↑)	FID (↓)	Precision (↑)	Recall (↑)
200	24.10	0.26	0.08	**20.31**	**0.41**	**0.11**
150	**17.39**	**0.47**	**0.13**	22.42	0.33	0.12
0	**10.81**	**0.78**	**0.31**	17.61	0.59	0.18

Table 3. Image super-resolution metrics using IADB and Ours. Our method shows consistent improvement over IADB according to the SSIM and PSNR metrics. 2x and 4x represent super-resolution from 64^2 to 128^2 and 32^2 to 128^2, respectively.

	IADB		Ours	
	SSIM (↑)	PSNR (↑)	SSIM (↑)	PSNR (↑)
CelebA (2×)	0.91	30.73	**0.92**	**31.56**
CelebA (4×)	0.76	24.74	**0.77**	**25.03**
LSUN-Church (4×)	0.57	19.46	**0.59**	**20.00**

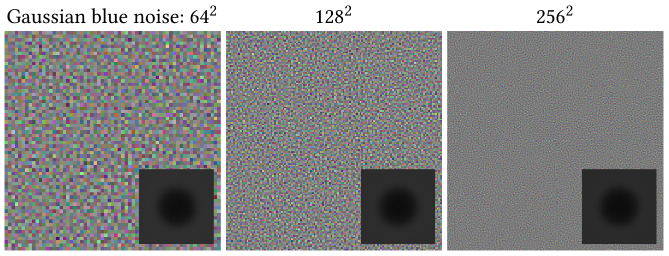

Fig. 1. Gaussian blue noise masks at different resolutions using our padding strategy and the corresponding frequency power spectra. Padding multiple (64^2) tiles does not produce visible artifact and have unnoticeable impact on the frequency distribution.

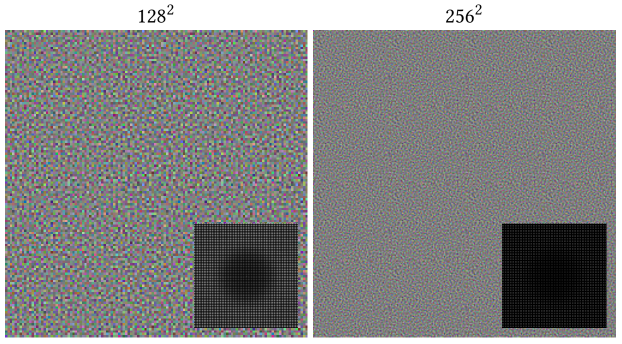

Fig. 2. Padding/tiling with the same 64^2 Gaussian blue noise mask to generate higher-resolution masks results in repetitive noise patterns, as well as structural artifacts in the frequency spectra.

Early stopping tests. Though using Gaussian blue noise only leads to worse results in terms of both quantitative and qualitative evaluations, we observe that the image content appears more clear visually in the early time steps. Based on this observation, we perform a study on early stopping where we stop at a specific early time step T_e and reconstruct the final image with one step. As shown in Table 2, early stopping at $T_e = 200$ gives much better results using Gaussian blue noise only. However, the quality does not improve with more steps but fluctuates in terms of the metrics.

Nearest neighbors test. We conduct a nearest neighbors test on AFHQ-Cat (64^2) using our method to check if there exists an overfitting issue. We test on AFHQ-Cat (64^2) as it is the dataset with smallest training samples and lowest resolution among our tested datasets. As shown in Fig. 4, though the nearest neighbors (second to fifth columns) may be semantically similar to the query on the leftmost column, the generated samples are not identical to the training set samples. This means our method does not suffer overfit the training data.

Image super-resolution. Table 3 provides the SSIM and PSNR metrics for the image super-resolution tasks using IADB and Ours. Our method consistenly improves over IADB.

Image generation. We provide the full Table 7 for quantitative comparisons including FID, Precision and Recall on image generation tasks. More results, comparisons and interactive visualization can be found in the Supplemental HTML.

Extension to DDIM [Song et al. 2021]. Detailed derivations of extending our time-varying noise model to DDIM can be found in Supplemental document Sec. 1. Table 4 shows preliminary results on AHFQ-Cat (64^2) and Ours (DDIM) gets better FID than the original DDIM.

Extension to LDM [Rombach et al. 2022]. Our time-varing noise model can also be incorporated into latent diffusion model (LDM) [Rombach et al. 2022] for high-resolution image generation. We compare IADB and Ours used for latent diffusion in Table 5 and the preliminary results show that Ours gets better FID than IADB. The visual comparisons can be found in the main paper.

Ablation on γ-scheduler. Table 6 compares different parameters and functions of our γ-scheduler. For our sigmoid-based γ-scheduler, $\gamma = 0.02$ gives better results than $\gamma = 1000$, showing the importance of choosing the γ value for our γ-scheduler. Using the cosine-based function proposed by [Nichol and Dhariwal 2021] resulted in worse FID, showing the importance of choosing the function for our γ-scheduler.

Ablation on noise mask size for padding/tiling. Figure 5 shows an ablation study on padding/tiling using different Gaussian blue noise sizes (1^2, 4^2, 16^2, 32^2, 64^2) on the AFHQ-Cat (128^2) dataset. Note that 1^2 size is equivalent to use Gaussian (white) noise. We use 64^2 size as our method for all experiments. The figure shows that increasing the size of Gaussian blue noise mask leads to lower FID than using Gaussian (white) noise in terms of mean values of multiple experiments. However, the standard deviations of using tiled Gaussian blue noise mask can be higher than using Gaussian (white) noise.

Timing. Here, we present the timing results obtained using a single RTX 2080 NVIDIA GPU for our pipeline. To assess the average inference time for both IADB and our networks, we conducted tests with a batch size of 1 and $T = 250$. The network architectures are identical, except for our network having a 6-channel output instead of 3. The average network inference time is approximately 0.020 seconds for generating a 64^2 image and around 0.023 seconds for a 128^2 image, applicable to both IADB and our method. Regarding noise generation timing, our approach takes roughly 0.0001 seconds to generate a Gaussian blue noise mask at a resolution of 64^2 and about 0.0002 seconds to generate a Gaussian noise or Gaussian blue noise at a resolution of 128^2.

Table 4. Preliminary results on comparing DDIM [Song et al. 2021] and Ours (DDIM) on AFHQ-Cat (64^2).

Method	DDIM	Ours (DDIM)
FID (↓)	9.82	**7.11**

Table 5. Preliminary results on comparing IADB and Ours in latent diffusion model (LDM) [Rombach et al. 2022] on AFHQ-Cat (512^2).

Method for LDM	IADB	Ours
FID (↓)	12.19	**11.45**

Table 6. Comparing the impact of γ-scheduler on CelebA (128^2). Our γ-scheduler with $\gamma = 0.2$ results in lower FID than $\gamma = 1000$ and cosine-based scheduler [Nichol and Dhariwal 2021].

γ-scheduler	cosine-based	$\gamma = 1000$	$\gamma = 0.2$
FID (↓)	37.13	29.70	**16.38**

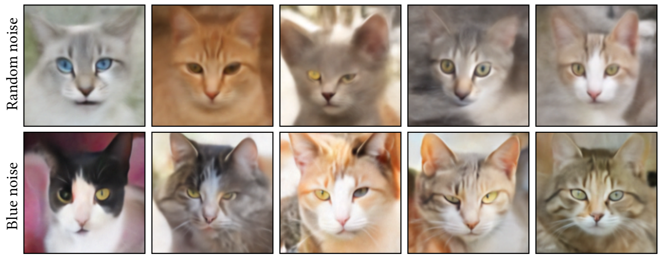

Fig. 3. Early stopping test on AFHQ-Cat (128^2) using single noise during the training process. The backward process is stopped at $t = 200$ step. First row shows generated results using Gaussian noise and the second using Gaussian blue noise. While blue noise alone creates some low frequency artifacts in the eye region, it generates sharper details than random noise.

REFERENCES

Yunjey Choi, Youngjung Uh, Jaejun Yoo, and Jung-Woo Ha. 2020. Stargan v2: Diverse image synthesis for multiple domains. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 8188–8197.

Eric Heitz, Laurent Belcour, and Thomas Chambon. 2023. Iterative α-(de) blending: A minimalist deterministic diffusion model. In *ACM SIGGRAPH 2023 Conference Proceedings*. 1–8.

Cheng-Han Lee, Ziwei Liu, Lingyun Wu, and Ping Luo. 2020. MaskGAN: Towards Diverse and Interactive Facial Image Manipulation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Table 7. Quantitative comparisons between DDIM [Song et al. 2021], IADB [Heitz et al. 2023] and Ours on different datasets, including FID, Precision and Recall metrics. Our method shows consistent improvements over IADB in terms of FID score. Best score for each metric and dataset is shown in bold, second best is underlined.

	DDIM			IADB			Ours		
	FID (↓)	Precision (↑)	Recall (↑)	FID (↓)	Precision (↑)	Recall (↑)	FID (↓)	Precision (↑)	Recall (↑)
AFHQ-Cat (64^2)	9.82	**0.83**	0.32	<u>9.18</u>	0.72	<u>0.37</u>	**7.95**	<u>0.73</u>	**0.50**
AFHQ-Cat (128^2)	<u>10.73</u>	**0.84**	0.28	10.81	<u>0.78</u>	<u>0.31</u>	**9.47**	<u>0.78</u>	**0.34**
CelebA (64^2)	9.26	0.83	0.46	<u>7.53</u>	<u>0.84</u>	**0.53**	**7.05**	**0.85**	<u>0.52</u>
CelebA (128^2)	**11.92**	**0.81**	**0.48**	20.71	0.75	<u>0.45</u>	<u>16.38</u>	**0.81**	0.42
LSUN-Church (64^2)	16.46	<u>0.74</u>	0.41	<u>13.12</u>	0.71	<u>0.47</u>	**10.16**	**0.76**	**0.48**

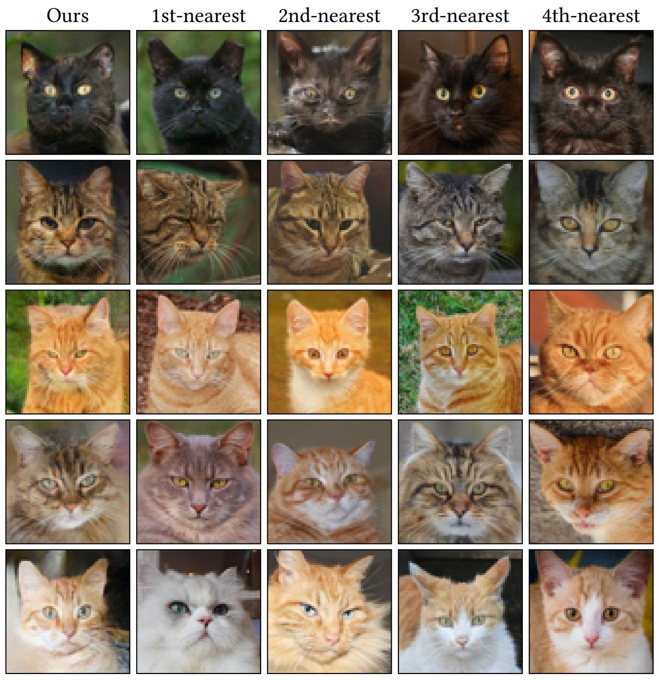

Fig. 4. We conduct a nearest neighbors test showing our method does not overfit the training data. Generated samples of our method trained on AFHQ-Cat (64^2) are shown in the leftmost column. Training set nearest neighbors are in the remaining columns, ordered from the 1st-nearest neighbor to the 4th-nearest neighbor (based on pixel-wise mean squared distance) from left to right.

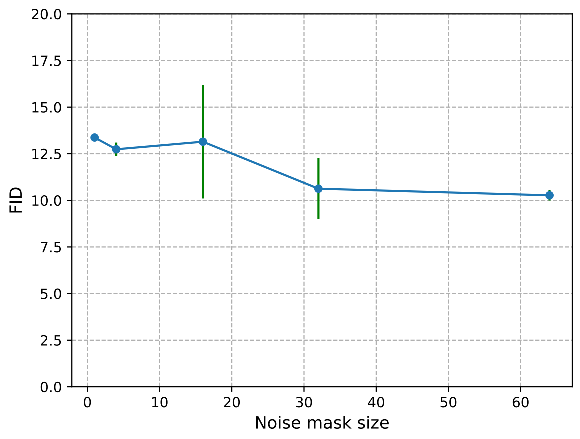

Fig. 5. Ablation study on different Gaussian blue noise sizes used for padding/tiling (1^2, 4^2, 16^2, 32^2, 64^2) experimented on the AFHQ-Cat (128^2) dataset. Note that 1^2 size is equivalent to use Gaussian (white) noise only. We use 64^2 size as our method in all experiments.

Alexander Quinn Nichol and Prafulla Dhariwal. 2021. Improved denoising diffusion probabilistic models. In *International conference on machine learning*. PMLR, 8162–8171.

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

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*. Springer, 234–241.

Jiaming Song, Chenlin Meng, and Stefano Ermon. 2021. Denoising Diffusion Implicit Models. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net. https://openreview.net/forum?id=St1giarCHLP

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

Patrick von Platen, Suraj Patil, Anton Lozhkov, Pedro Cuenca, Nathan Lambert, Kashif Rasul, Mishig Davaadorj, and Thomas Wolf. 2022. Diffusers: State-of-the-art diffusion models. https://github.com/huggingface/diffusers.

Fisher Yu, Yinda Zhang, Shuran Song, Ari Seff, and Jianxiong Xiao. 2015. LSUN: Construction of a Large-scale Image Dataset using Deep Learning with Humans in the Loop. *arXiv preprint arXiv:1506.03365* (2015).