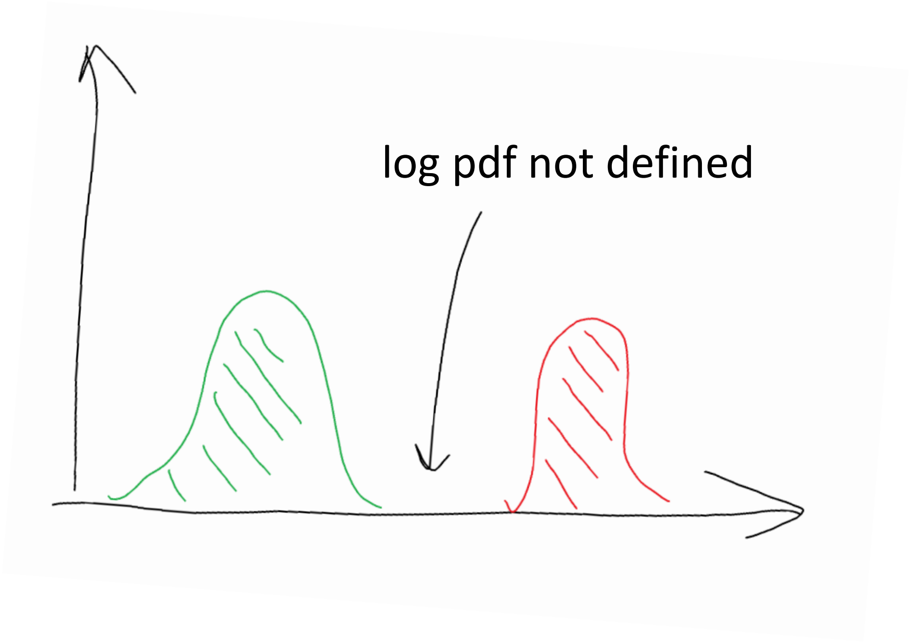
Diffusion Models

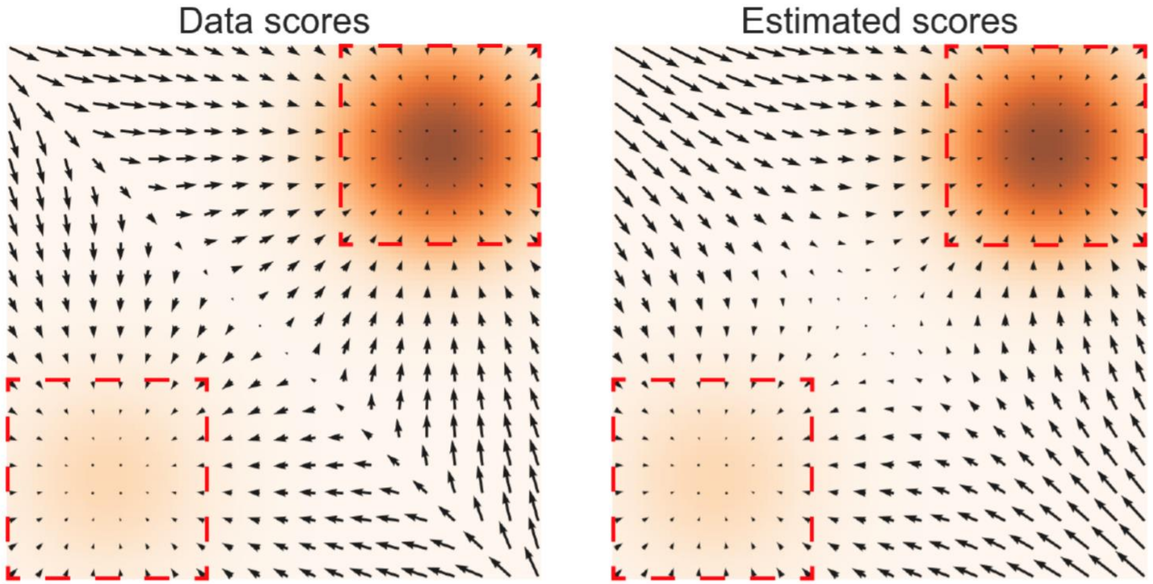
Anthony Baryshnikov

As score matching models

- Score matching estimates the gradient of log prob in data space and generates samples by some variation of gradient ascent.
- It is good but has some downsides.
- Extra backpropagations to estimate trace of score gradient.
- Manifold hypothesis.
- Low data density regions have bad score estimates.
- It's hard to transition between disconnected supports.

[1] Y. Song, S. Ermon. Generative Modeling by Estimating Gradients of the Data Distribution. arXiv:1907.05600.





[1] Y. Song, S. Ermon. Generative Modeling by Estimating Gradients of the Data Distribution. arXiv:1907.05600.

As score matching models

- Let's perturb data with various levels of Gaussian noise and train a noise conditional network to estimate score.
- High noise fills low density regions and gives a common support.
- Low noise is almost indistinguishable from true data.
- Sample using annealed Langevin dynamics (ALS).
- Our loss becomes:

$$\ell(\boldsymbol{\theta};\sigma) \triangleq \frac{1}{2}\mathbb{E}_{p_{\text{data}}(\mathbf{x})}\mathbb{E}_{\tilde{\mathbf{x}}\sim\mathcal{N}(\mathbf{x},\sigma^2 I)}\left[\left\|\mathbf{s}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}},\sigma)+\frac{\tilde{\mathbf{x}}-\mathbf{x}}{\sigma^2}\right\|_2^2\right] \qquad \mathcal{L}(\boldsymbol{\theta};\{\sigma_i\}_{i=1}^L) \triangleq \frac{1}{L}\sum_{i=1}^{L}\lambda(\sigma_i)\ell(\boldsymbol{\theta};\sigma_i)$$

[1] Y. Song, S. Ermon. Generative Modeling by Estimating Gradients of the Data Distribution. arXiv:1907.05600.

Algorithm 1 Annealed Langevin dynamics.

Require: $\{\sigma_i\}_{i=1}^{L}, \epsilon, T.$

```
Initialize x̃_0
for i ← 1 to L do
    α_i ← ε · σ_i^2 / σ_L^2            ▷ α_i is the step size.
    for t ← 1 to T do
        Draw z_t ~ N(0, I)
        x̃_t ← x̃_{t-1} + (α_i / 2) s_θ(x̃_{t-1}, σ_i) + √α_i z_t
    end for
    x̃_0 ← x̃_T
end for
return x̃_T
```

Initialize $\tilde{\mathbf{x}}_0$

for $i \leftarrow 1$ to L **do**

$\quad\alpha_i \leftarrow \epsilon \cdot \sigma_i^2/\sigma_L^2$ $\quad\triangleright$ α_i is the step size.

$\quad$**for** $t \leftarrow 1$ to T **do**

$\quad\quad$Draw $\mathbf{z}_t \sim \mathcal{N}(0, I)$

$$\tilde{\mathbf{x}}_t \leftarrow \tilde{\mathbf{x}}_{t-1} + \frac{\alpha_i}{2}\mathbf{s}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}_{t-1}, \sigma_i) + \sqrt{\alpha_i}\,\mathbf{z}_t$$

$\quad$**end for**

$\quad\tilde{\mathbf{x}}_0 \leftarrow \tilde{\mathbf{x}}_T$

end for

return $\tilde{\mathbf{x}}_T$

[1] Y. Song, S. Ermon. Generative Modeling by Estimating Gradients of the Data Distribution. arXiv:1907.05600.

Too many hyperparameters

- How can we come up with good noise levels?
- What about sampling step size?
- And the number of steps?

[2] Y. Song, S. Ermon. Improved Techniques for Training Score-Based Generative Models. arXiv:2006.09011.

Noise levels

- Smallest noise level has to be indistinguishable.
- Transition probabilities decay exponentially.
- Choose largest noise level at least as large as the maximum Euclidean distance between all pairs of training data.
- Samples have to cover high density regions of previous noise level.
- Choose a geometric progression with common ratio dependent on data dimensionality.

[2] Y. Song, S. Ermon. Improved Techniques for Training Score-Based Generative Models. arXiv:2006.09011.

ALS parameters

- We want sampling variance to be as close to noise level as possible.
- Can be computed analytically for one data point.
- Choose T as large as possible and optimize the step size making variance ratio as close to 1 as possible.

[2] Y. Song, S. Ermon. Improved Techniques for Training Score-Based Generative Models. arXiv:2006.09011.

Other tricks

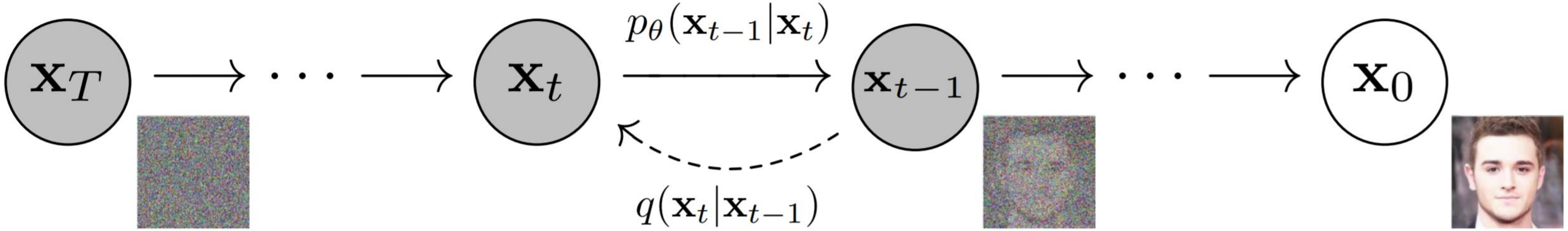
- Hard to condition on noise level (is it really?).
- Make an unconditional score estimation network and divide its output by noise std.
- Samples are empirically very unstable and exhibit artifacts such as common color shift.
- Apply EMA over model weights.

[2] Y. Song, S. Ermon. Improved Techniques for Training Score-Based Generative Models. arXiv:2006.09011.

As nonequilibrium thermodynamics

- Let’s gradually perturb our data with small noise.
- Reverse diffusion process has the same functional form.
- We have to predict mean and variance.
- Train by maximizing variational lower bound.
- Generate by gradually denoising samples from stationary distribution.

[3] J. Sohl-Dickstein et al. Deep Unsupervised Learning using Nonequilibrium Thermodynamics. arXiv:1503.03585.



[3] J. Sohl-Dickstein et al. Deep Unsupervised Learning using Nonequilibrium Thermodynamics. arXiv:1503.03585.

$$
L \geq \int d\mathbf{x}^{(0\cdots T)} q\left(\mathbf{x}^{(0\cdots T)}\right) \cdot
$$

$$
\log \left[p\left(\mathbf{x}^{(T)}\right) \prod_{t=1}^{T} \frac{p\left(\mathbf{x}^{(t-1)}|\mathbf{x}^{(t)}\right)}{q\left(\mathbf{x}^{(t)}|\mathbf{x}^{(t-1)}\right)} \right]
$$

$$
K = -\sum_{t=2}^{T} \int d\mathbf{x}^{(0)} d\mathbf{x}^{(t)} q\left(\mathbf{x}^{(0)}, \mathbf{x}^{(t)}\right) \cdot
$$

$$
D_{KL}\left(q\left(\mathbf{x}^{(t-1)}|\mathbf{x}^{(t)}, \mathbf{x}^{(0)}\right) \middle\| p\left(\mathbf{x}^{(t-1)}|\mathbf{x}^{(t)}\right)\right)
$$

$$
+ H_q\left(\mathbf{X}^{(T)}|\mathbf{X}^{(0)}\right) - H_q\left(\mathbf{X}^{(1)}|\mathbf{X}^{(0)}\right) - H_p\left(\mathbf{X}^{(T)}\right).
$$

[3] J. Sohl-Dickstein et al. Deep Unsupervised Learning using Nonequilibrium Thermodynamics. arXiv:1503.03585.

As nonequilibrium thermodynamics

- We're now working with Gaussian noise only.
- Let's set variance to a time dependent constant.

[4] J. Ho et al. Denoising Diffusion Probabilistic Models.
arXiv:2006.11239.

Loss reparameterization

- Let's rewrite our loss.
- We can remove the factor between estimated noise difference norm to obtain a simplified objective.
- Score matching objective and variance lower bound maximization are very similar.

$$\mathbb{E}_{\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\frac{\beta_t^2}{2\sigma_t^2\alpha_t(1-\bar{\alpha}_t)}\left\|\boldsymbol{\epsilon}-\boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0+\sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon},t)\right\|^2\right]$$

[4] J. Ho et al. Denoising Diffusion Probabilistic Models.
arXiv:2006.11239.

Algorithm 1 Training

1: **repeat**
2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
3: $t \sim \mathrm{Uniform}(\{1, \ldots, T\})$
4: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5: Take gradient descent step on
$$\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t) \right\|^2$$
6: **until** converged

Algorithm 2 Sampling

1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \ldots, 1$ **do**
3: $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
4: $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right) + \sigma_t \mathbf{z}$
5: **end for**
6: **return** $\mathbf{x}_0$

[4] J. Ho et al. Denoising Diffusion Probabilistic Models.
arXiv:2006.11239.

How to obtain exact log likelihoods?

- Let's set the last term of reversed process to a discrete decoder.
- We can now estimate the conditional probability by calculating an integral.

$$p_\theta(\mathbf{x}_0|\mathbf{x}_1) = \prod_{i=1}^{D} \int_{\delta_-(x_0^i)}^{\delta_+(x_0^i)} \mathcal{N}(x; \mu_\theta^i(\mathbf{x}_1, 1), \sigma_1^2)\, dx$$

$$\delta_+(x) = \begin{cases} \infty & \text{if } x = 1 \\ x + \frac{1}{255} & \text{if } x < 1 \end{cases} \qquad \delta_-(x) = \begin{cases} -\infty & \text{if } x = -1 \\ x - \frac{1}{255} & \text{if } x > -1 \end{cases}$$

[4] J. Ho et al. Denoising Diffusion Probabilistic Models.
arXiv:2006.11239.

Different objectives

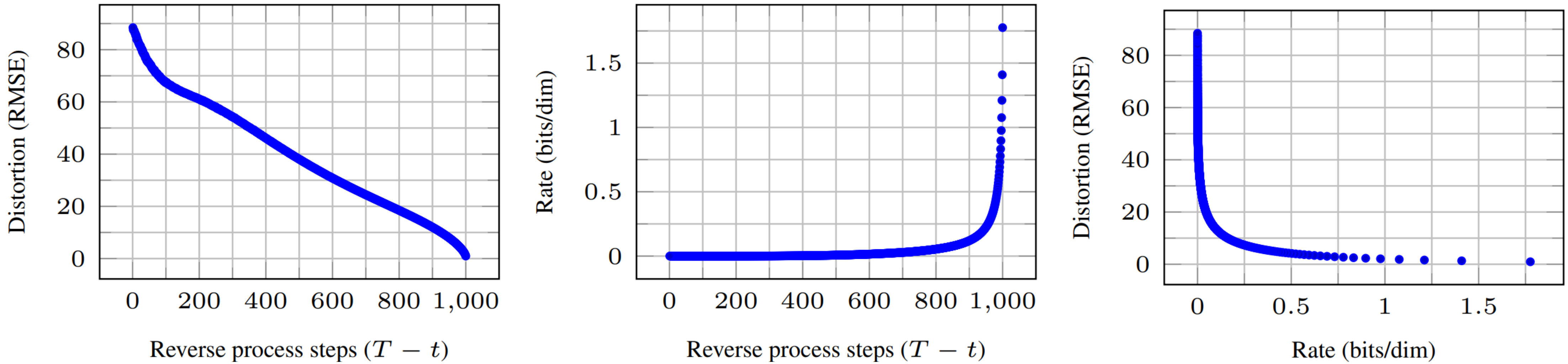
- Simplified objective makes low noise level loss relatively more important and improves sample quality.
- Predicting noise gives similar results to estimating mean of Gaussian.

[4] J. Ho et al. Denoising Diffusion Probabilistic Models.
arXiv:2006.11239.

Lossy compression

- KL divergence corresponds to rate (?).
- RMSE corresponds to distortion.
- Let’s plot them.
- Turns out that the majority of our codelength encodes impeccable details, which is not optimal.
- I’m not sure if I got this right.

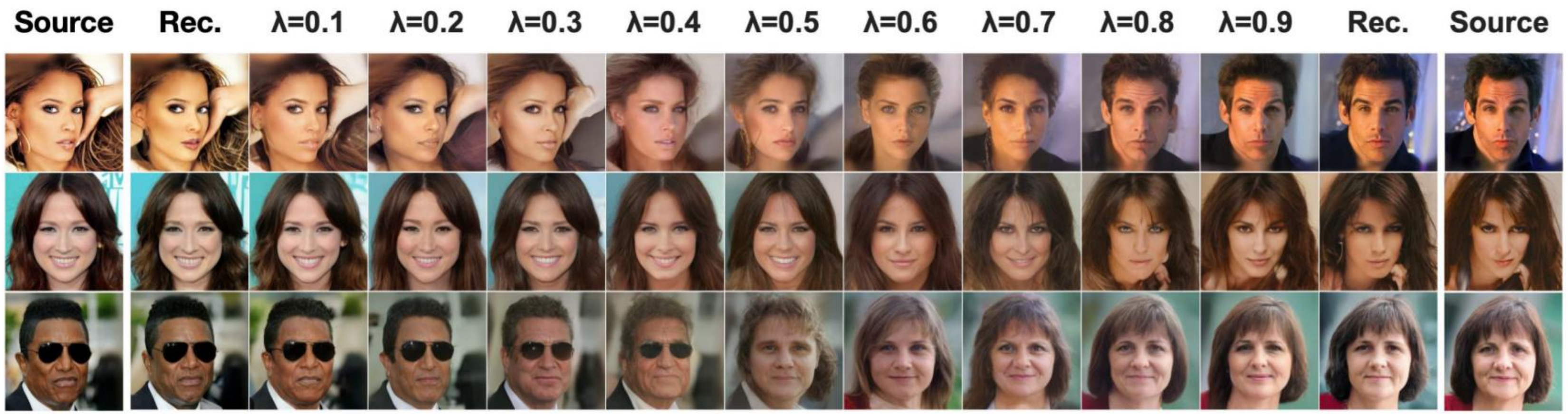
[4] J. Ho et al. Denoising Diffusion Probabilistic Models.
arXiv:2006.11239.



[4] J. Ho et al. Denoising Diffusion Probabilistic Models.
arXiv:2006.11239.

Extra details

- Diffusion process that masks first T pixels corresponds to an autoregressive model.
- Interpolating in latent space works particularly well.



[4] J. Ho et al. Denoising Diffusion Probabilistic Models. arXiv:2006.11239.

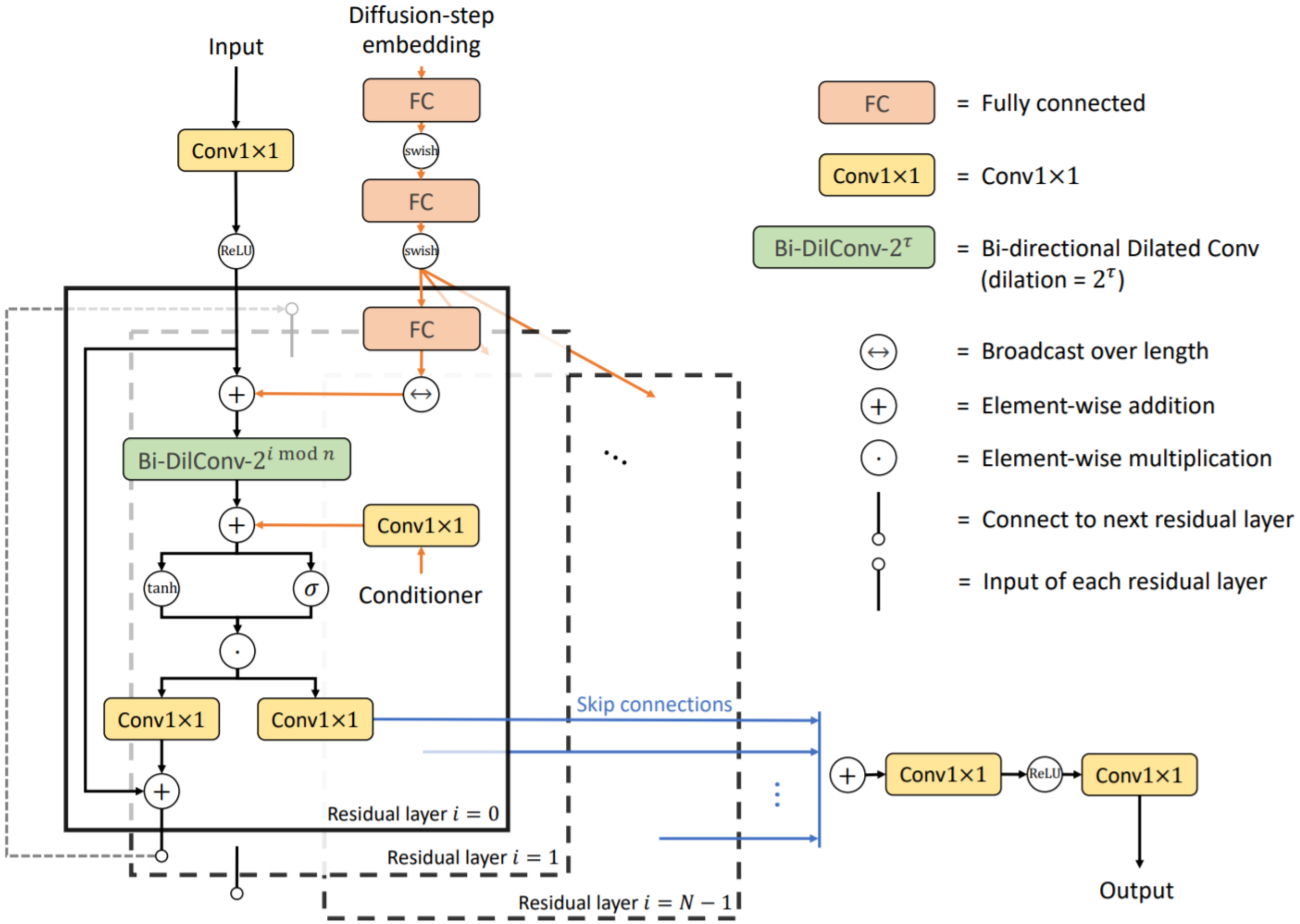
Advantages in audio generation

- Diffusion models avoid mode/posterior collapse.
- Non-autoregressive which means faster parallel synthesis.
- Very flexible architecture.
- Does not require auxiliary losses (e.g. Mel spectrogram loss).
- Provides intuitive speed/quality tradeoff by varying number of denoising steps.

DiffWave

- Bidirectional dilated convolutions.
- N residual layers divided into m blocks.
- Dilation is doubled at each layer within each block.
- Big receptive fields because of multiple denoising steps.
- Positional embeddings of timesteps from transformers.
- Let’s upsample local conditioning to the same length.
- We can then add both local and global conditioning as bias terms after dilated convolutions by using 1x1 conv.
- We can use faster noise schedules at inference to increase speed.

[5] Z. Kong et al. DiffWave: A Versatile Diffusion Model for Audio Synthesis. arXiv:2009.09761.



[5] Z. Kong et al. DiffWave: A Versatile Diffusion Model for Audio Synthesis. arXiv:2009.09761.

DiffWave results

- Performs well at neural vocoding.
- Better MOS than WaveNet at 6.91M vs 4.57M parameters.
- Real-time generation but still much slower than flow based models (1.1-5.6x vs 40+x).
- Completely destroys everybody at unconditional and class-conditional generation.
- Zero-shot speech denoising and latent space interpolation is also available.

[5] Z. Kong et al. DiffWave: A Versatile Diffusion Model for Audio Synthesis. arXiv:2009.09761.

Table 1: The model hyperparameters, model footprint, and 5-scale Mean Opinion Score (MOS) with 95% confidence intervals for WaveNet, ClariNet, WaveFlow, WaveGlow and the proposed DiffWave on the **neural vocoding** task. ↑ means the number is the higher the better, and ↓ means the number is the lower the better.

Model	T	T_{infer}	**layers**	**res. channels**	**#param(↓)**	**MOS(↑)**
WaveNet	—	—	30	128	4.57M	**4.43** $\pm$ 0.10
ClariNet	—	—	60	64	2.17M	4.27 $\pm$ 0.09
WaveGlow	—	—	96	256	87.88M	4.33 $\pm$ 0.12
WaveFlow	—	—	64	64	5.91M	4.30 $\pm$ 0.11
WaveFlow	—	—	64	128	22.25M	4.40 $\pm$ 0.07
DiffWave $_{\text{BASE}}$	20	20	30	64	2.64M	4.31 $\pm$ 0.09
DiffWave $_{\text{BASE}}$	40	40	30	64	2.64M	4.35 $\pm$ 0.10
DiffWave $_{\text{BASE}}$	50	50	30	64	2.64M	**4.38** $\pm$ 0.08
DiffWave $_{\text{LARGE}}$	200	200	30	128	6.91M	**4.44** $\pm$ 0.07
DiffWave $_{\text{BASE}}$ (Fast)	50	6	30	64	2.64M	*4.37* $\pm$ 0.07
DiffWave $_{\text{LARGE}}$ (Fast)	200	6	30	128	6.91M	*4.42* $\pm$ 0.09
Ground-truth	—	—	—	—	—	4.52 $\pm$ 0.06

[5] Z. Kong et al. DiffWave: A Versatile Diffusion Model for Audio Synthesis. arXiv:2009.09761.

Table 2: The automatic evaluation metrics (FID, IS, mIS, AM, and NDB/K), and 5-scale MOS with 95% confidence intervals for WaveNet, WaveGAN, and DiffWave on the **unconditional** generation task. ↑ means the number is the higher the better, and ↓ means the number is the lower the better.

Model	**FID**(↓)	**IS**(↑)	**mIS**(↑)	**AM**(↓)	**NDB**/K(↓)	**MOS**(↑)
WaveNet-128	3.279	2.54	7.6	1.368	0.86	1.34 ± 0.29
WaveNet-256	2.947	2.84	10.0	1.260	0.86	1.43 ± 0.30
WaveGAN	1.349	4.53	36.6	0.796	0.78	2.03 ± 0.33
DiffWave	**1.287**	**5.30**	**59.4**	**0.636**	**0.74**	$\mathbf{3.39} \pm 0.32$
Trainset	0.000	8.48	281.4	0.164	0.00	—
Testset	0.011	8.47	275.2	0.166	0.10	3.72 ± 0.28

Table 3: The automatic evaluation metrics (Accuracy, FID-class, IS, mIS), and 5-scale MOS with 95% confidence intervals for WaveNet and DiffWave on the **class-conditional** generation task.

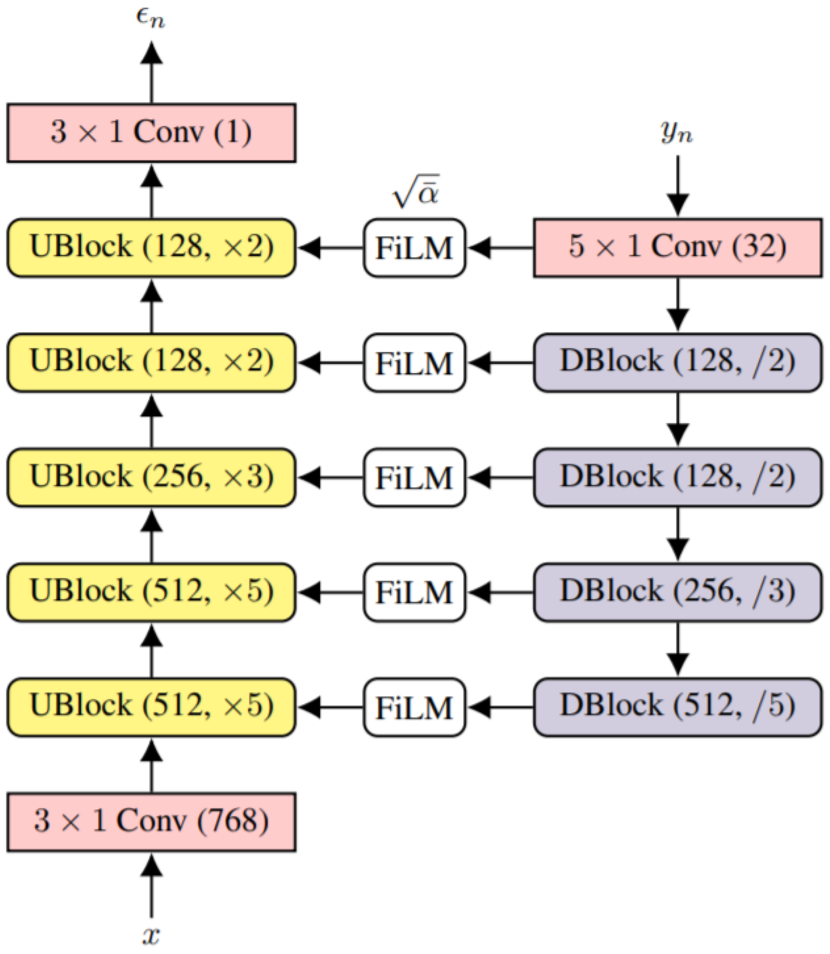
Model	**Accuracy**(↑)	**FID-class**(↓)	**IS**(↑)	**mIS**(↑)	**MOS**(↑)
WaveNet-128	56.20%	7.876±2.469	3.29	15.8	1.46 ± 0.30
WaveNet-256	60.70%	6.954±2.114	3.46	18.9	1.58 ± 0.36
DiffWave	91.20%	1.113±0.569	6.63	117.4	$\mathbf{3.50} \pm 0.31$
DiffWave (deep & thin)	**94.00%**	**0.932**±0.450	**6.92**	**133.8**	3.44 ± 0.36
Trainset	99.06%	0.000±0.000	8.48	281.4	—
Testset	98.76%	0.044±0.016	8.47	275.2	3.72 ± 0.28

[5] Z. Kong et al. DiffWave: A Versatile Diffusion Model for Audio Synthesis. arXiv:2009.09761.

WaveGrad

- Let’s use diffusion models in TTS.
- Network similar to a feature pyramid.
- Uses spatial feature-wise linear modulation for conditioning.
- Proposes conditioning on the fraction of true signal instead of on the timestamp which provides better generalization between noise schedules.
- Authors also suggest Fibonacci and manual noise schedules.

[6] N. Chen et al. WaveGrad: Estimating Gradients for Waveform Generation. arXiv:2009.00713.



[6] N. Chen et al. WaveGrad: Estimating Gradients for Waveform Generation. arXiv:2009.00713.

WaveGrad results

- Large model with 1000 iterations achieves a MOS of 4.51 (4.58 GT).
- Base model with 6 iterations achieves a MOS of 4.41 with good real-time factors (0.2 on NVIDIA V100 and 1.5 on CPU).

[6] N. Chen et al. WaveGrad: Estimating Gradients for Waveform Generation. arXiv:2009.00713.

Model	**MOS (↑)**
WaveRNN	4.49 ± 0.04
Parallel WaveGAN	3.92 ± 0.05
MelGAN	3.95 ± 0.06
Multi-band MelGAN	4.10 ± 0.05
GAN-TTS	4.34 ± 0.04
WaveGrad	
Base (6 iterations, continuous noise levels)	4.41 ± 0.03
Base (1,000 iterations, discrete indices)	4.47 ± 0.04
Large (1,000 iterations, discrete indices)	4.51 ± 0.04
Ground Truth	4.58 ± 0.05

Iterations (schedule)	**LS-MSE (↓)**	**MCD (↓)**	**FFE (↓)**	**MOS (↑)**
WaveGrad conditioned on discrete index				
25 (Fibonacci)	283	3.93	3.3%	3.86 ± 0.05
50 (Linear $(1 \times 10^{-4}, 0.05)$)	181	3.13	3.1%	4.42 ± 0.04
1,000 (Linear $(1 \times 10^{-4}, 0.005)$)	116	2.85	3.2%	4.47 ± 0.04
WaveGrad conditioned on continuous noise level				
6 (Manual)	217	3.38	2.8%	4.41 ± 0.04
25 (Fibonacci)	185	3.33	2.8%	4.44 ± 0.04
50 (Linear $(1 \times 10^{-4}, 0.05)$)	177	3.23	2.7%	4.43 ± 0.04
1,000 (Linear $(1 \times 10^{-4}, 0.005)$)	106	2.85	3.0%	4.46 ± 0.03

References

- [1] Y. Song, S. Ermon. Generative Modeling by Estimating Gradients of the Data Distribution. arXiv:1907.05600.
- [2] Y. Song, S. Ermon. Improved Techniques for Training Score-Based Generative Models. arXiv:2006.09011.
- [3] J. Sohl-Dickstein et al. Deep Unsupervised Learning using Nonequilibrium Thermodynamics. arXiv:1503.03585.
- [4] J. Ho et al. Denoising Diffusion Probabilistic Models. arXiv:2006.11239.
- [5] Z. Kong et al. DiffWave: A Versatile Diffusion Model for Audio Synthesis. arXiv:2009.09761.
- [6] N. Chen et al. WaveGrad: Estimating Gradients for Waveform Generation. arXiv:2009.00713.

Questions?