

2025 edition

Deep Learning for Music Analysis and Generation

# Diffusion-based Text-to-music Generation

(text → audio)



**Yi-Hsuan Yang** Ph.D.  
[yhyangtw@ntu.edu.tw](mailto:yhyangtw@ntu.edu.tw)

# References for Diffusion Models

- References
  - Diffusion model tutorial @ ISMIR 2024 (<https://sites.google.com/view/diffusion-tutorial-ismir24/materials>)A thumbnail image for a video titled "FROM WHITE NOISE TO SYMPHONY" featuring a musical note icon. Below the title is the subtitle "DIFFUSION MODELS FOR MUSIC AND SOUND". At the bottom, it says "November 10 @ISMIR 2024".
  - Prof. Hung-Yi Lee's videos (<https://www.youtube.com/watch?v=ifCDXFdeaaM>)

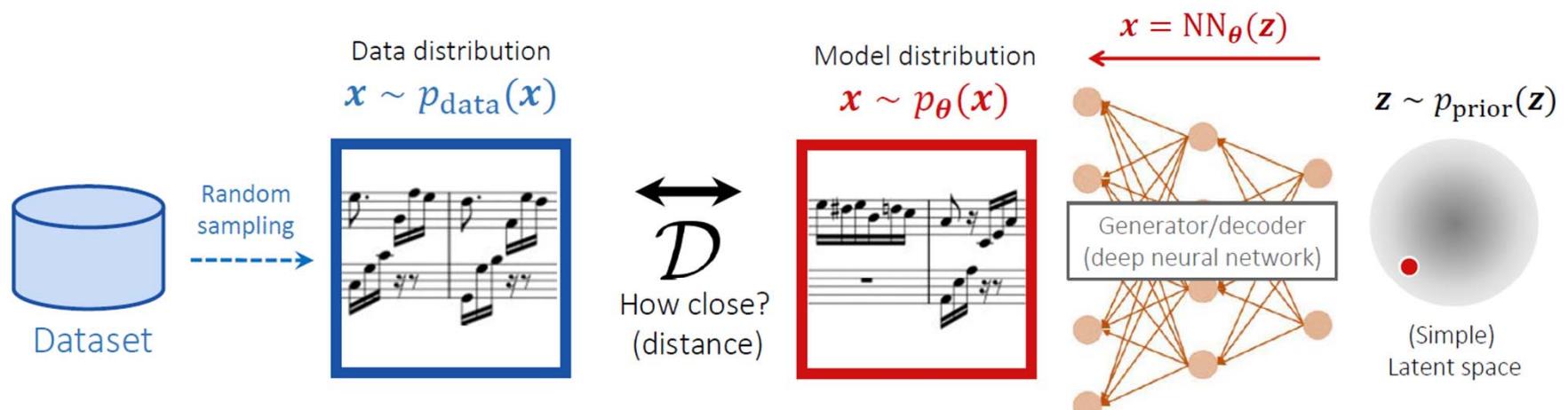


# Outline

- **Diffusion models**
- Diffusion-based Text-to-music Generation
- Music ControlNet & MuseControlLite

# Diffusion Models: Intro

- Learn to fit the real data distribution
- From Gaussian noises (easier to do sampling and generation)

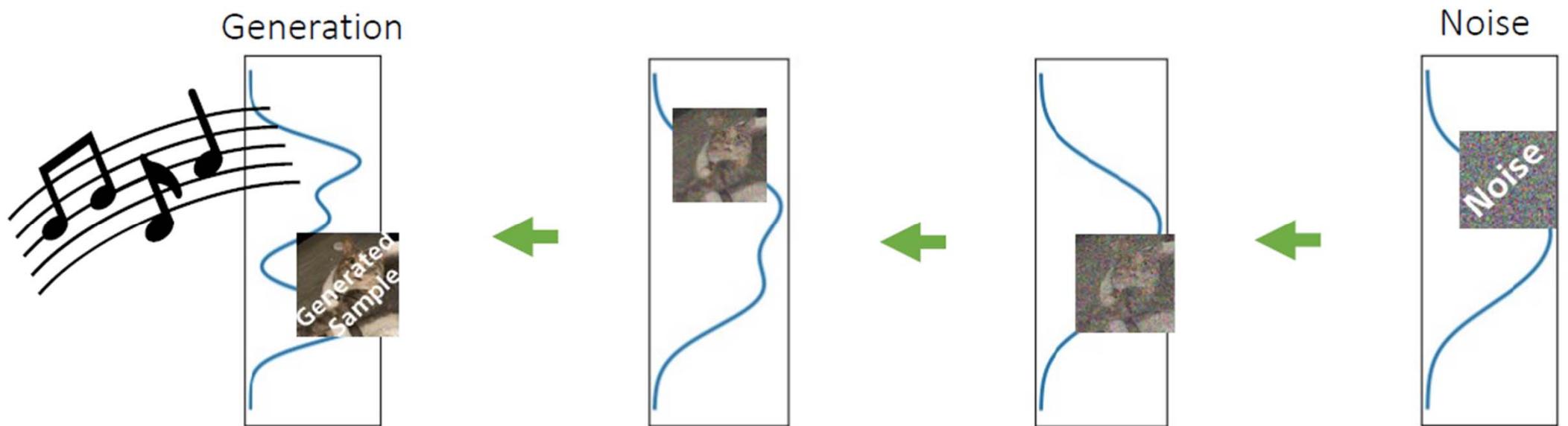


$$\min_{\theta} \mathcal{D}(p_{\theta}(x), p_{\text{data}}(x))$$

(From: [https://github.com/ChiehHsinJesseLai/ISMIR24DiffusionModelTutorial/blob/main/tutorial\\_diffusion\\_model\\_all.pdf](https://github.com/ChiehHsinJesseLai/ISMIR24DiffusionModelTutorial/blob/main/tutorial_diffusion_model_all.pdf))

# Diffusion Models: Intro

- Learn to fit the real data distribution
- From Gaussian noises (easier to do sampling and generation)



(From: [https://github.com/ChiehHsinJesseLai/ISMIR24DiffusionModelTutorial/blob/main/tutorial\\_diffusion\\_model\\_all.pdf](https://github.com/ChiehHsinJesseLai/ISMIR24DiffusionModelTutorial/blob/main/tutorial_diffusion_model_all.pdf))

# Diffusion Models: Learn NN to Gradually Denoise

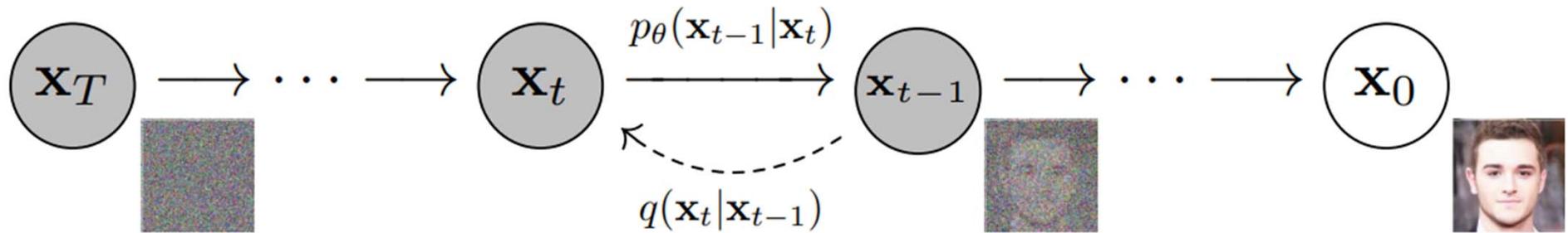
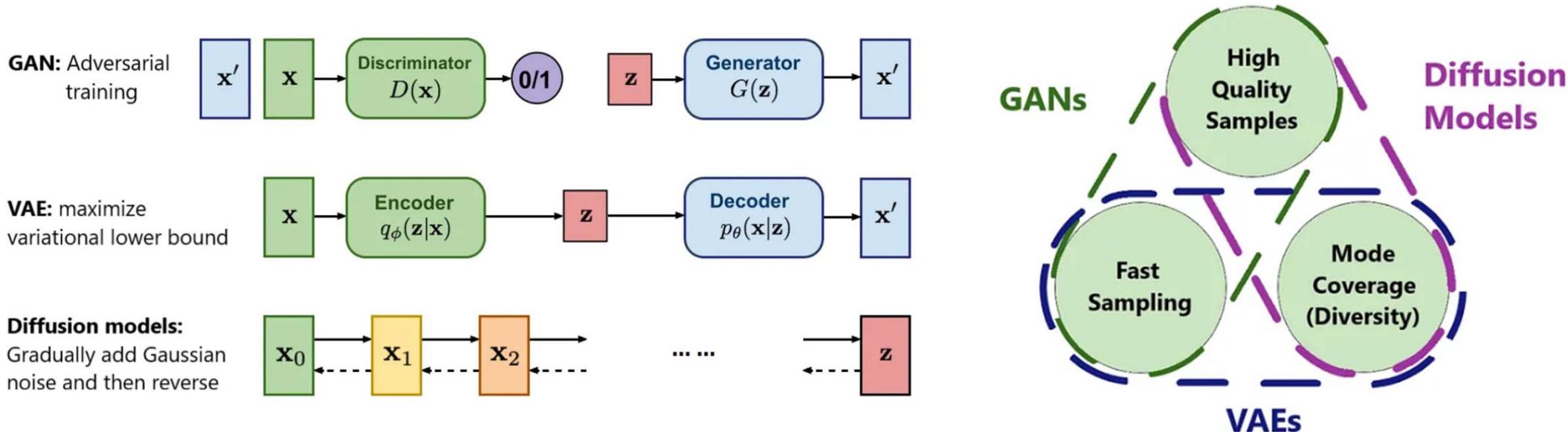


Figure 2: The directed graphical model considered in this work.

- **Forward** (or diffusion) process:  $\mathbf{x}_0$  (real data)  $\rightarrow \mathbf{x}_T$  (Gaussian noise)
- **Reverse** process:  $\mathbf{x}_T \rightarrow \mathbf{x}_0$  [denoise]
- The intermediate noisy data serve as latent codes ( $\mathbf{z}$ ) and have the same size as the training data

# Diffusion Models

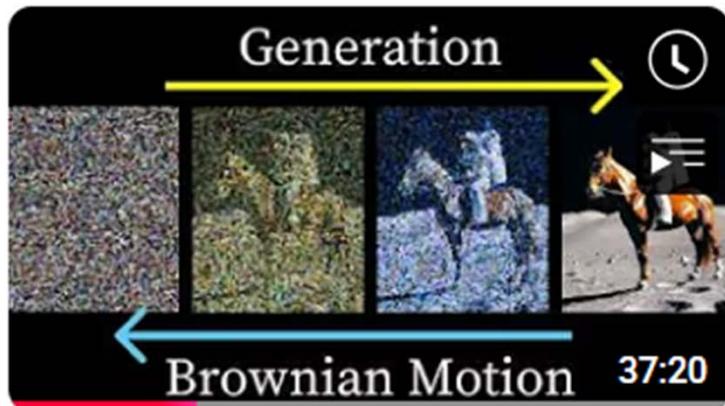
<https://pub.towardsai.net/diffusion-models-vs-gans-vs-vaes-comparison-of-deep-generative-models-67ab93e0d9ae>



- **High-fidelity samples:** due to the nature of gradually removing noise
- **High diversity samples:** likelihood maximization covers all data modes
- **Slow sample generation:** requires multiple runs to gradually generate samples

# Diffusion Models

[https://www.youtube.com/watch?v=iv-5mZ\\_9CPY](https://www.youtube.com/watch?v=iv-5mZ_9CPY)



But how do AI images and videos actually work? | Guest video by...

3Blue1Brown and Welch Labs • 1.1M views

Diffusion models, CLIP, and the math of turning text into images

Welch Labs Book: <https://www.welchlabs.com/resources/imaginar...>

# Denoising Diffusion Probabilistic Models (DDPM)

- Add noises  $\mathcal{N}(\mathbf{0}, \mathbf{I})$  during training ( $\epsilon$ ) and inference ( $\mathbf{z}$ )
  - At training,  $\mathbf{x}_t = a_t \mathbf{x}_0 + b_t \epsilon$
  - **But why at inference time?**

---

## Algorithm 1 Training

---

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

---

---

## Algorithm 2 Sampling

---

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

---

# Denoising Diffusion Probabilistic Models (DDPM)

- The denoisier  $\epsilon_\theta(\cdot, \cdot)$  predicts the total noise added (or, equivalently,  $\mathbf{x}_0$ )
  - Rather than  $\mathbf{x}_{t-1}$ , the previous step; why?
- The denoisier has two inputs:  $\mathbf{x}_t$ , and the timestamp  $t$  (denoising step)

---

## Algorithm 1 Training

---

```
1: repeat
2:    $\mathbf{x}_0 \sim q(\mathbf{x}_0)$ 
3:    $t \sim \text{Uniform}(\{1, \dots, T\})$ 
4:    $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ 
5:   Take gradient descent step on
      
$$\nabla_\theta \|\epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_t} \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t} \epsilon, t)\|^2$$

6: until converged
```

---

---

## Algorithm 2 Sampling

---

```
1:  $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ 
2: for  $t = T, \dots, 1$  do
3:    $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$  if  $t > 1$ , else  $\mathbf{z} = \mathbf{0}$ 
4:   
$$\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left( \mathbf{x}_t - \frac{1 - \alpha_t}{\sqrt{1 - \bar{\alpha}_t}} \epsilon_\theta(\mathbf{x}_t, t) \right) + \sigma_t \mathbf{z}$$

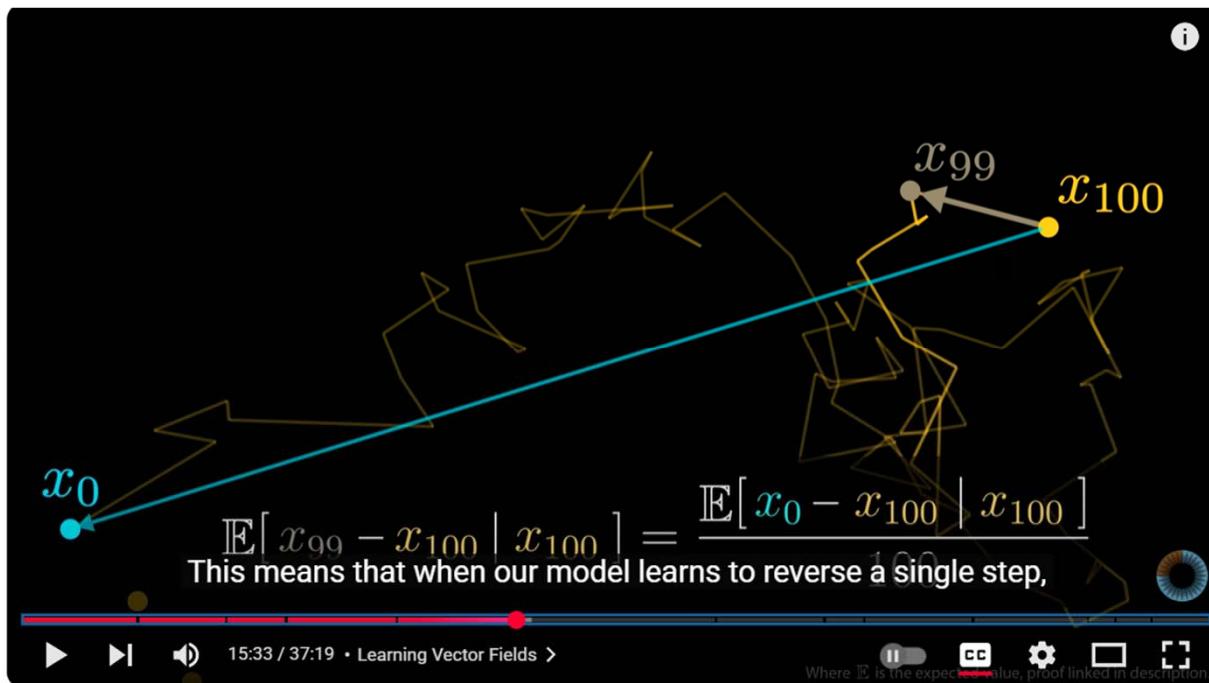
5: end for
6: return  $\mathbf{x}_0$ 
```

---

# Denoising Diffusion Probabilistic Models (DDPM)

[https://www.youtube.com/watch?v=iv-5mZ\\_9CPY](https://www.youtube.com/watch?v=iv-5mZ_9CPY)

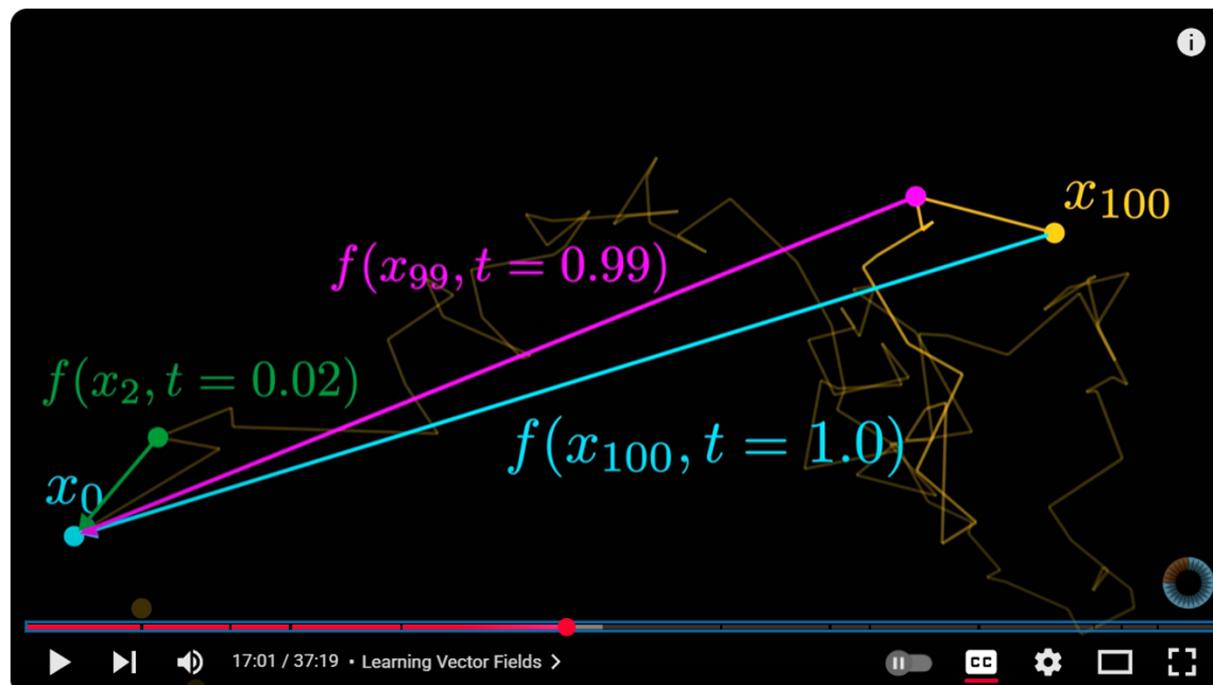
- Predict the original input ( $x_0$ ) rather than the less noisy input ( $x_{t-1}$ )



# Denoising Diffusion Probabilistic Models (DDPM)

[https://www.youtube.com/watch?v=iv-5mZ\\_9CPY](https://www.youtube.com/watch?v=iv-5mZ_9CPY)

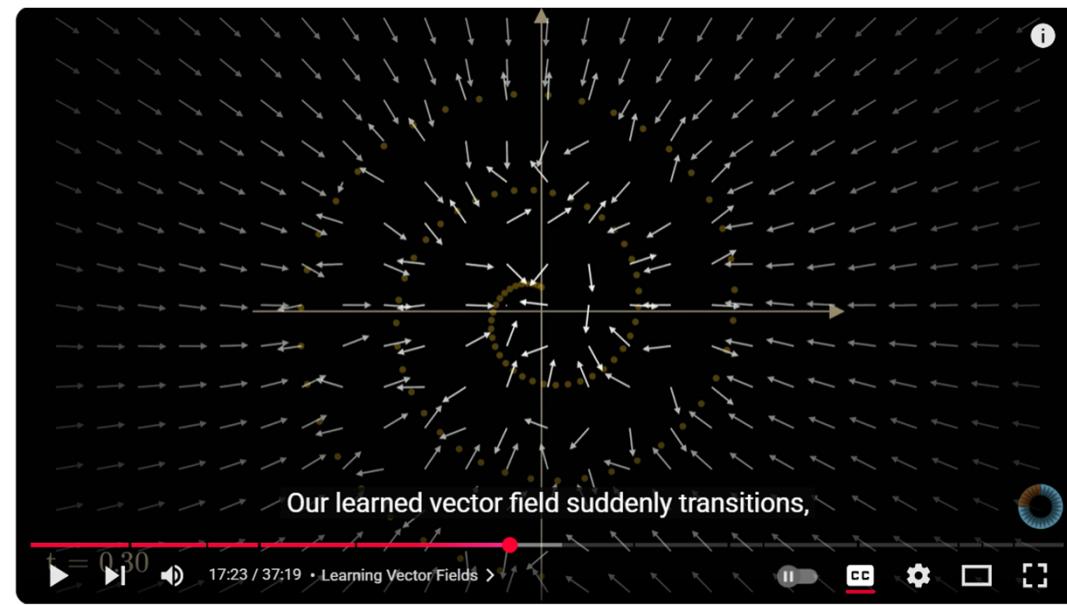
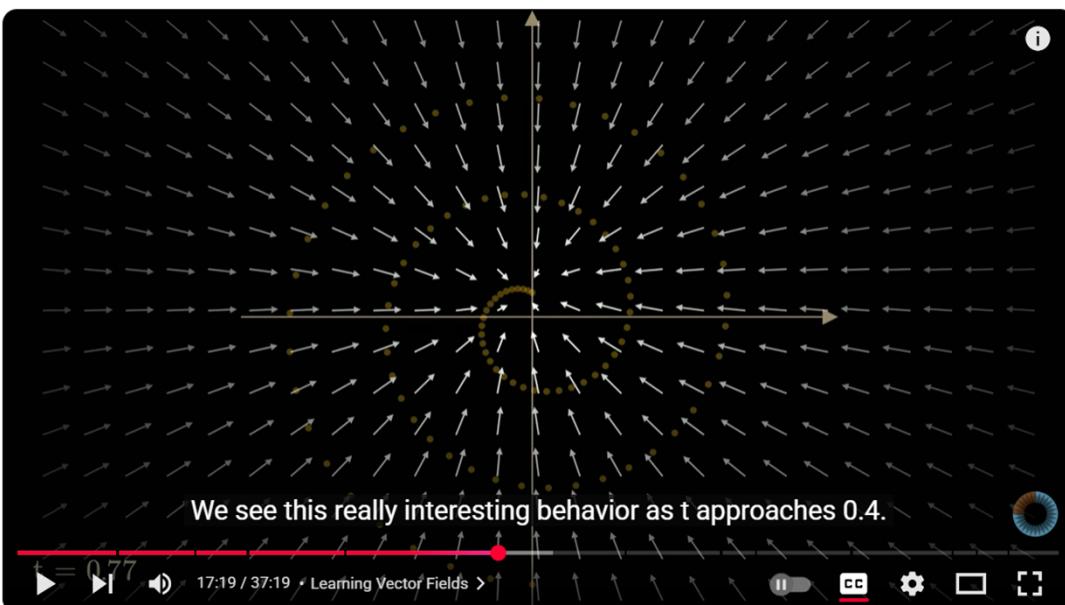
- Add noise  $\mathcal{N}(\mathbf{0}, \mathbf{I})$  during training and inference



# Denoising Diffusion Probabilistic Models (DDPM)

[https://www.youtube.com/watch?v=iv-5mZ\\_9CPY](https://www.youtube.com/watch?v=iv-5mZ_9CPY)

- Add noise  $\mathcal{N}(\mathbf{0}, \mathbf{I})$  during training and inference



# Denoising Diffusion Probabilistic Models (DDPM)

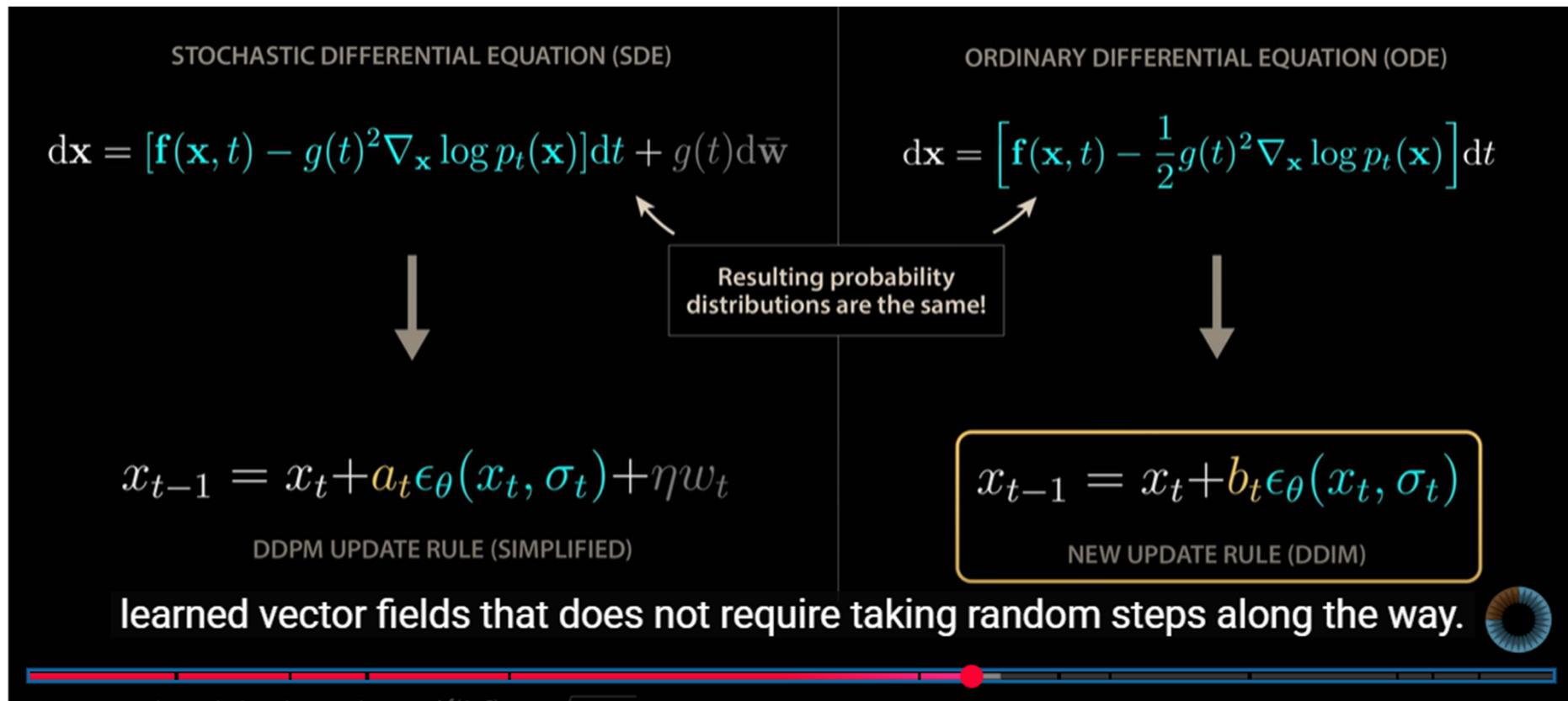
[https://www.youtube.com/watch?v=iv-5mZ\\_9CPY](https://www.youtube.com/watch?v=iv-5mZ_9CPY)

- Add noise  $\mathcal{N}(\mathbf{0}, \mathbf{I})$  during training and inference



# Denoising Diffusion Implicit Models (DDIM)

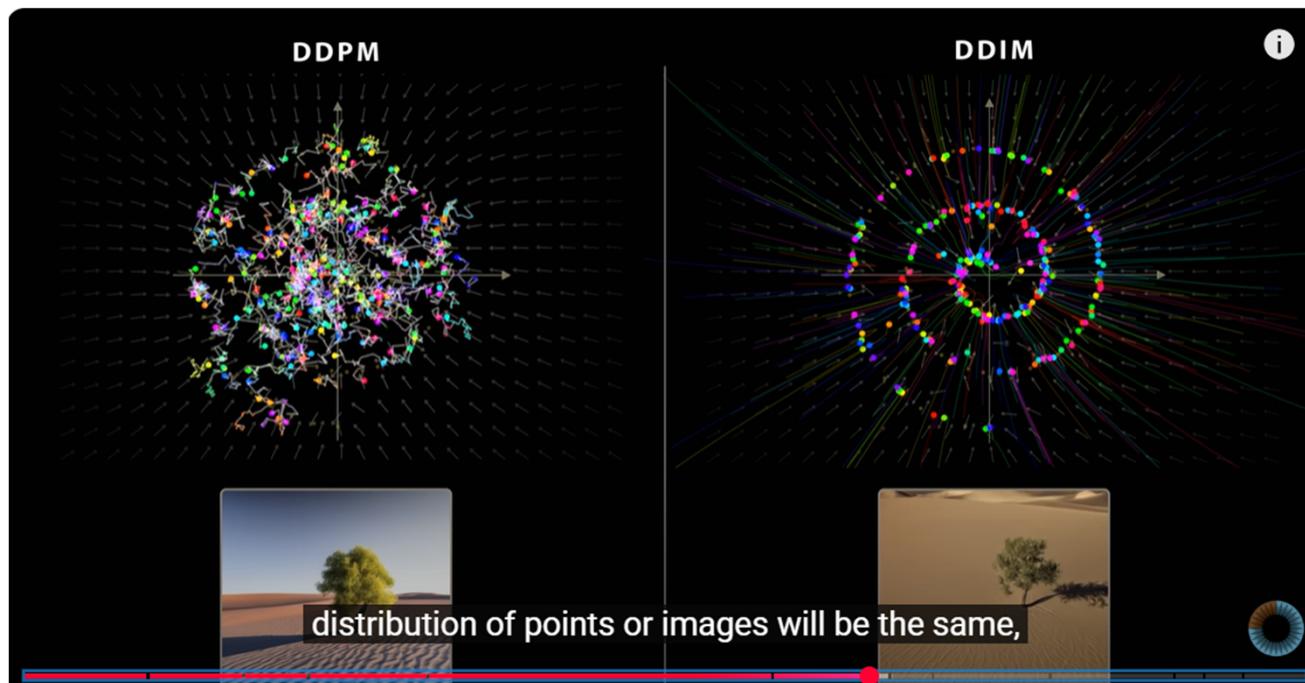
[https://www.youtube.com/watch?v=iv-5mZ\\_9CPY](https://www.youtube.com/watch?v=iv-5mZ_9CPY)



# Denoising Diffusion Implicit Models (DDIM)

[https://www.youtube.com/watch?v=iv-5mZ\\_9CPY](https://www.youtube.com/watch?v=iv-5mZ_9CPY)

- **Faster inference:** solving ODEs instead of SDEs



## DDPM vs DDIM

<https://www.youtube.com/watch?v=i2qSxMVeVLI>

- DDPM outperforms DDIM with a large number of denoising steps (but slow)
- DDIM works better with fewer denoising steps
- Don't retrain: we can take a model trained with the DDPM objective and accelerate it with the DDIM sampler

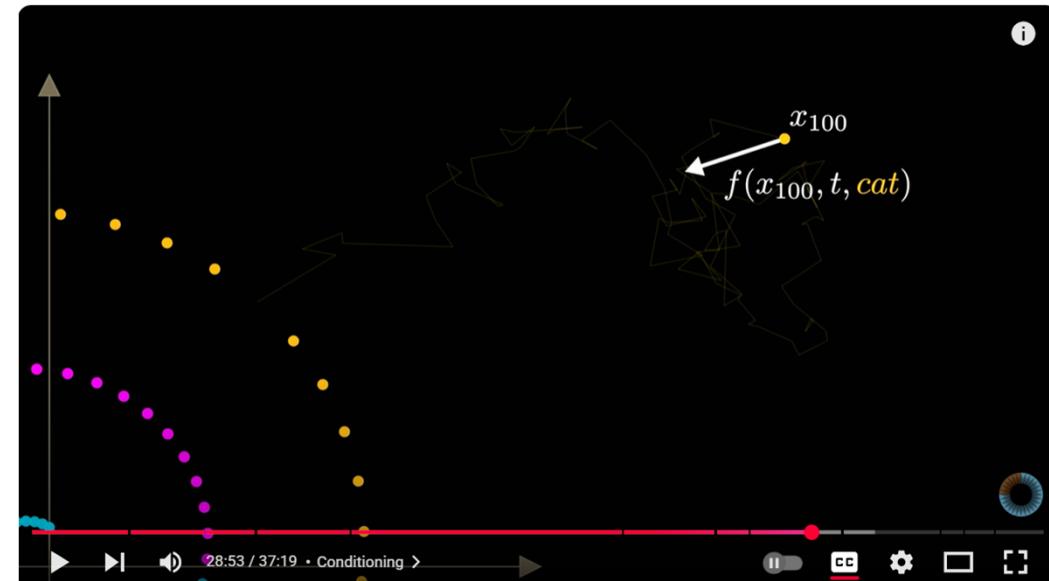
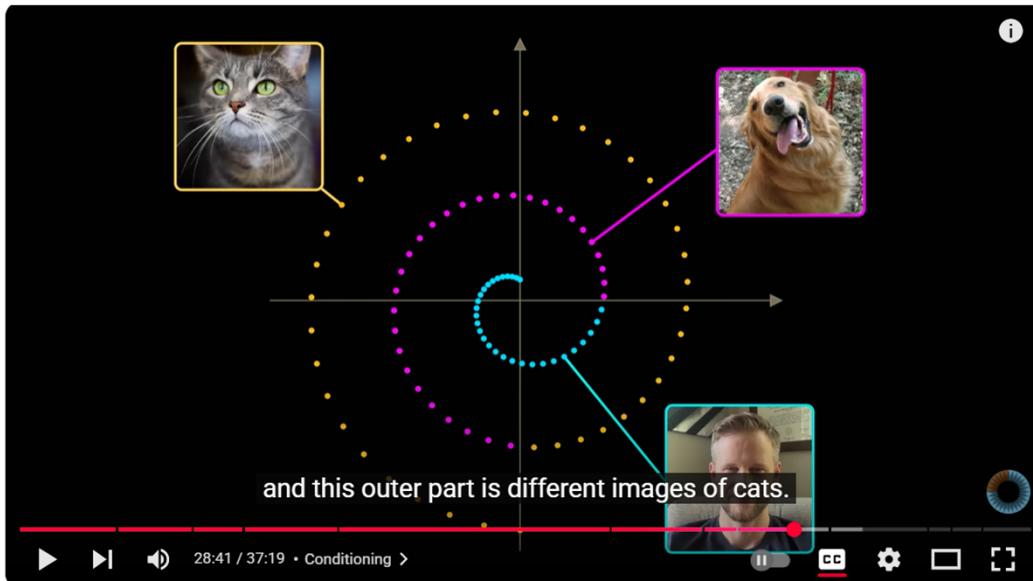
	CIFAR10 (32 × 32)					CelebA (64 × 64)				
	10	20	50	100	1000	10	20	50	100	1000
DDIM	<b>13.36</b>	<b>6.84</b>	<b>4.67</b>	<b>4.16</b>	4.04	<b>17.33</b>	<b>13.73</b>	<b>9.17</b>	<b>6.53</b>	3.51
DDPM	367.43	133.37	32.72	9.99	<b>3.17</b>	299.71	183.83	71.71	45.20	<b>3.26</b>

(the value here (FID) is the lower the better)

# Classifier-Free Guidance (CFG)

[https://www.youtube.com/watch?v=iv-5mZ\\_9CPY](https://www.youtube.com/watch?v=iv-5mZ_9CPY)

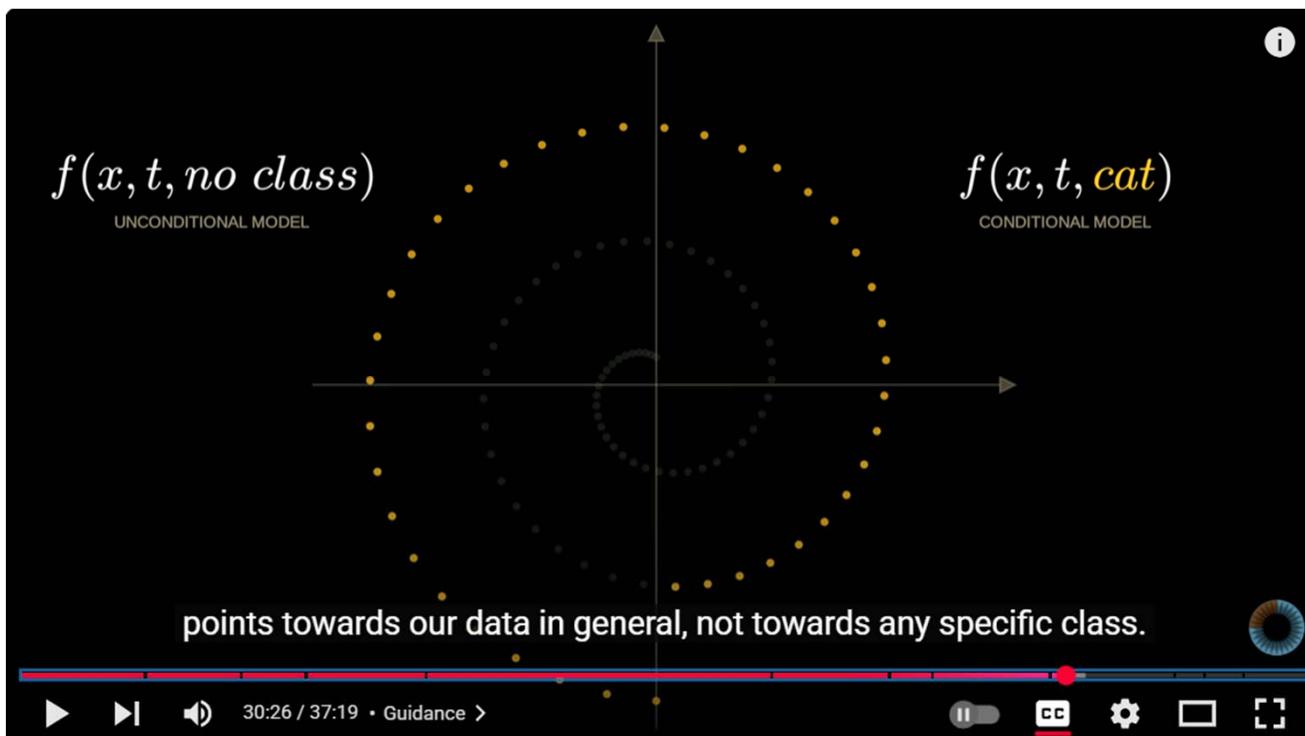
- Going from unconditional generation to **conditional** generation



# Classifier-Free Guidance (CFG)

[https://www.youtube.com/watch?v=iv-5mZ\\_9CPY](https://www.youtube.com/watch?v=iv-5mZ_9CPY)

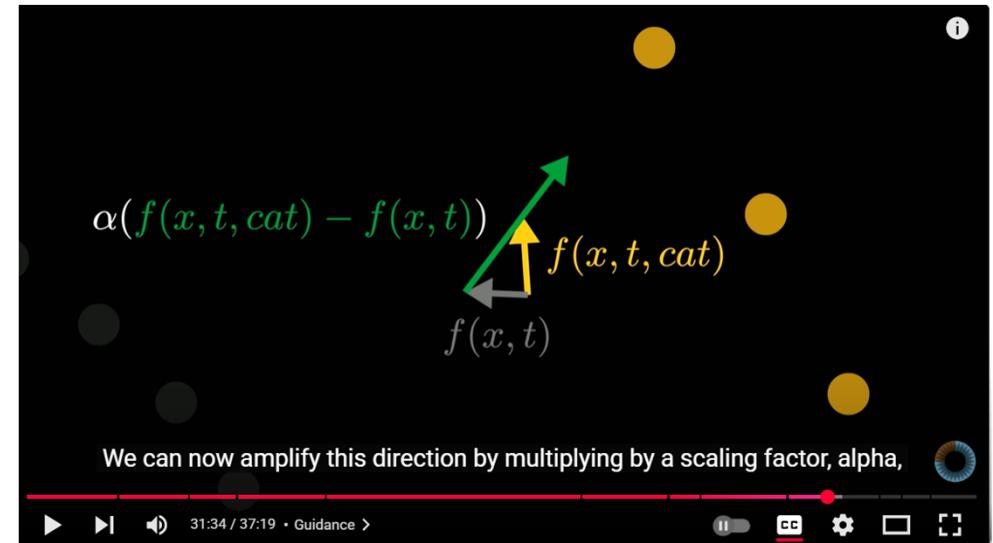
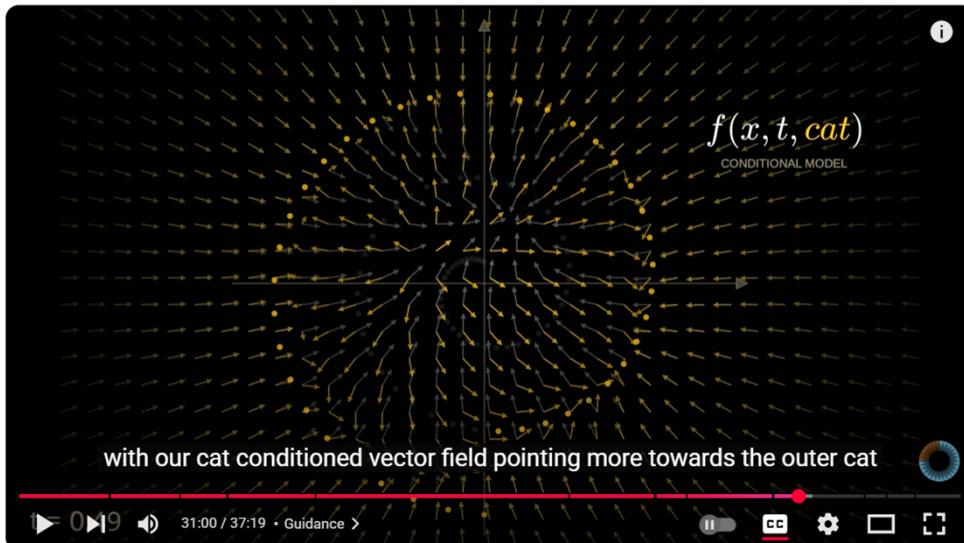
- One model is sufficient: use “no class” as the dummy condition



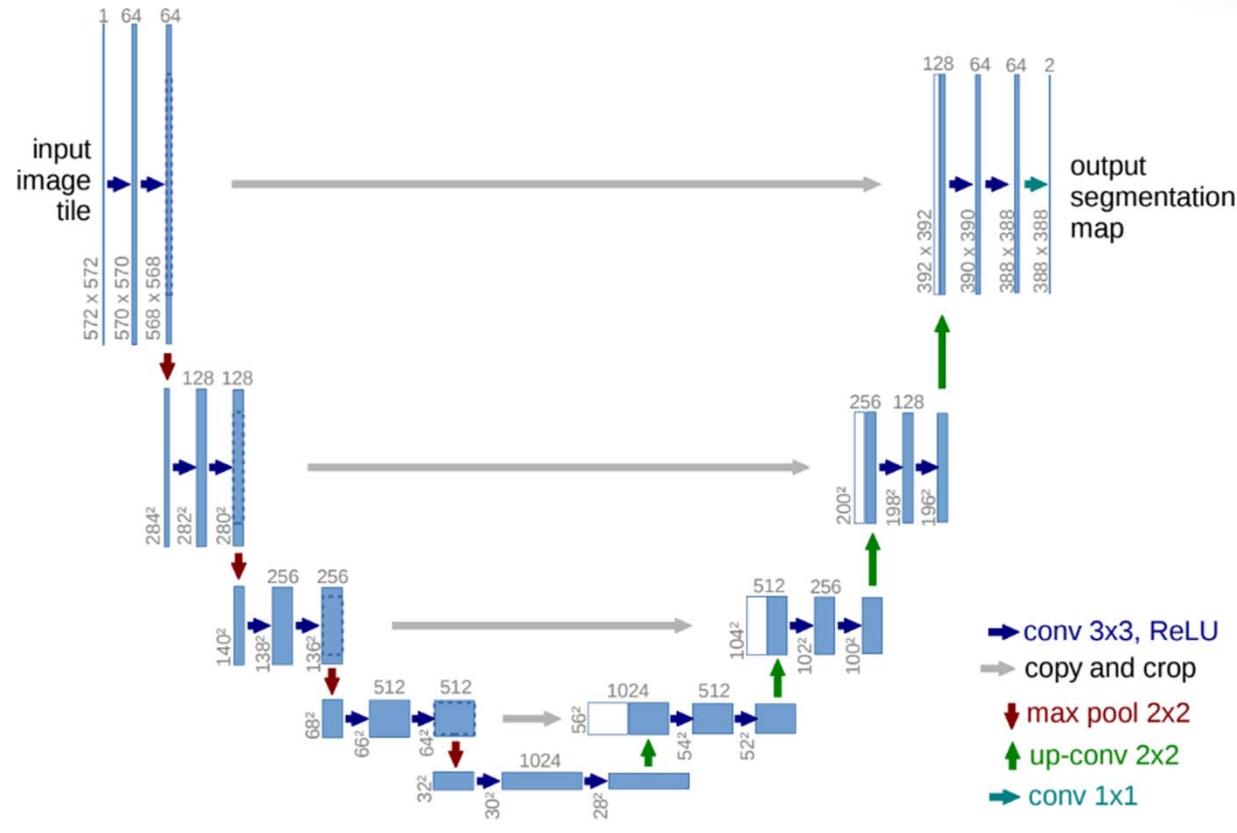
# Classifier-Free Guidance (CFG)

[https://www.youtube.com/watch?v=iv-5mZ\\_9CPY](https://www.youtube.com/watch?v=iv-5mZ_9CPY)

- The vector fields of the unconditional and conditional models may diverge
- Taking their difference provides a simple solution
- Moreover, we can use the “guidance scale”  $\alpha$  as a flexible control



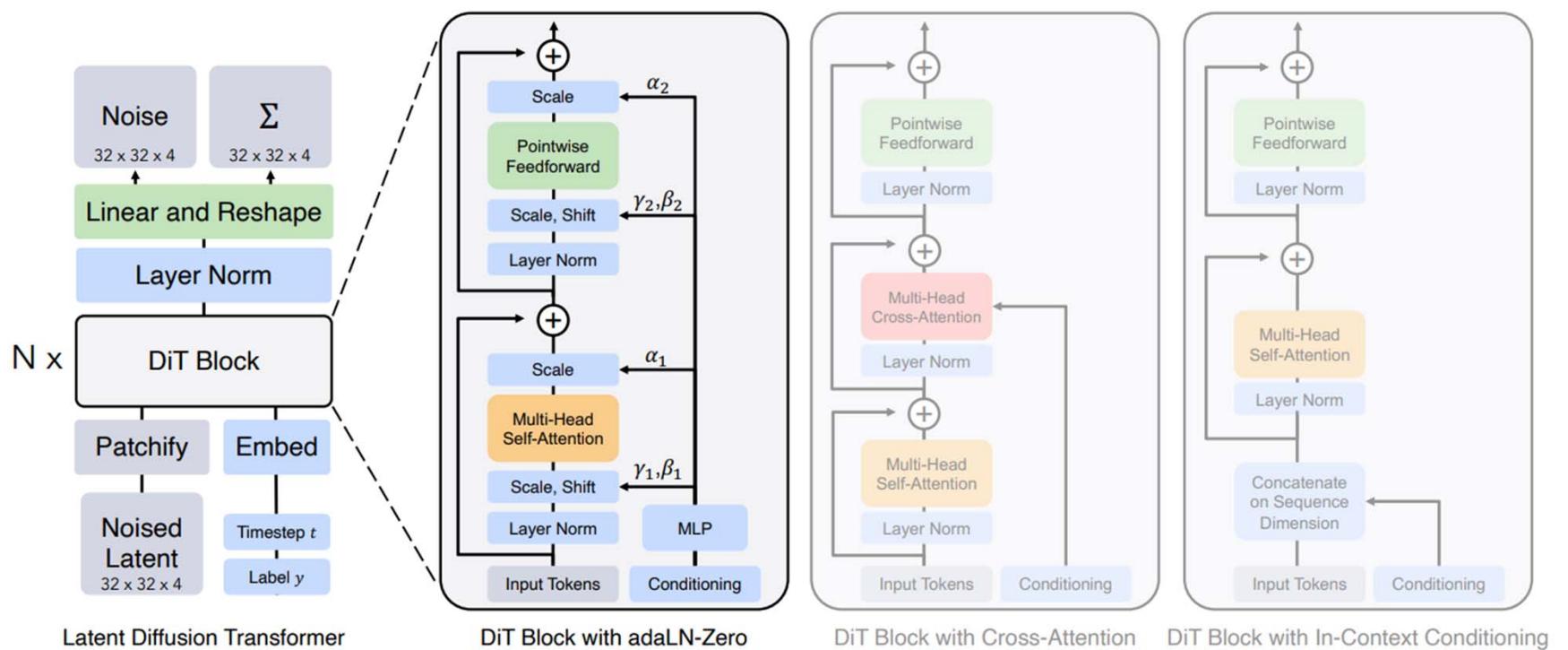
# Denoiser: U-Net



- U-net are often used in tasks where the input and output are of similar shape
  - image denoising
  - image segmentation
  - denoiser in diffusion models
- “Per-pixel” classification
- Components
  - downsampling encoder
  - upsampling decoder
  - skip connections

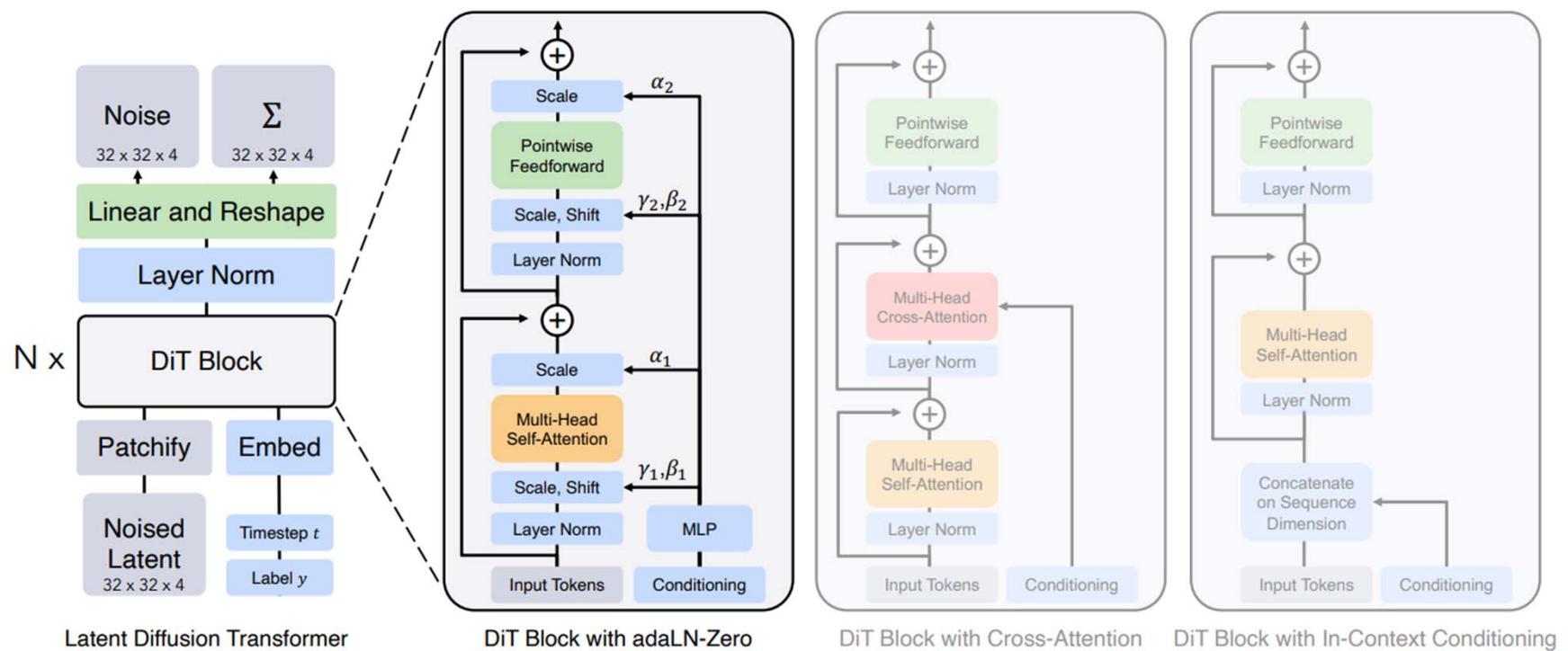
# Denoiser: Diffusion Transformer (DiT)

- DiT is based on the Vision Transformer (ViT) architecture which operates on sequences of patches



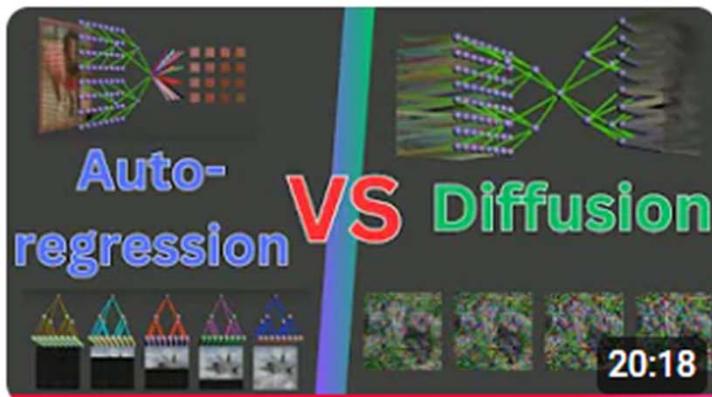
# Denoiser: Diffusion Transformer (DiT)

- Self-attention layers instead of convolution layers
- Use adaptive layer norm to incorporate conditioning (e.g., timestamps)



# Diffusion vs. Transformers

<https://www.youtube.com/watch?v=zc5NTeJbk-k>



## Why Does Diffusion Work Better than Auto-Regression?

Algorithmic Simplicity • 374K views

Have you ever wondered how generative AI actually works? Well the short answer is, in exactly the same way as regular AI! In this...



# Diffusion Models (from Prof. Jia-Bin Huang)

<https://www.youtube.com/watch?v=i2qSxMVeVLI>



## How I Understand Diffusion Models

Jia-Bin Huang • 55K views

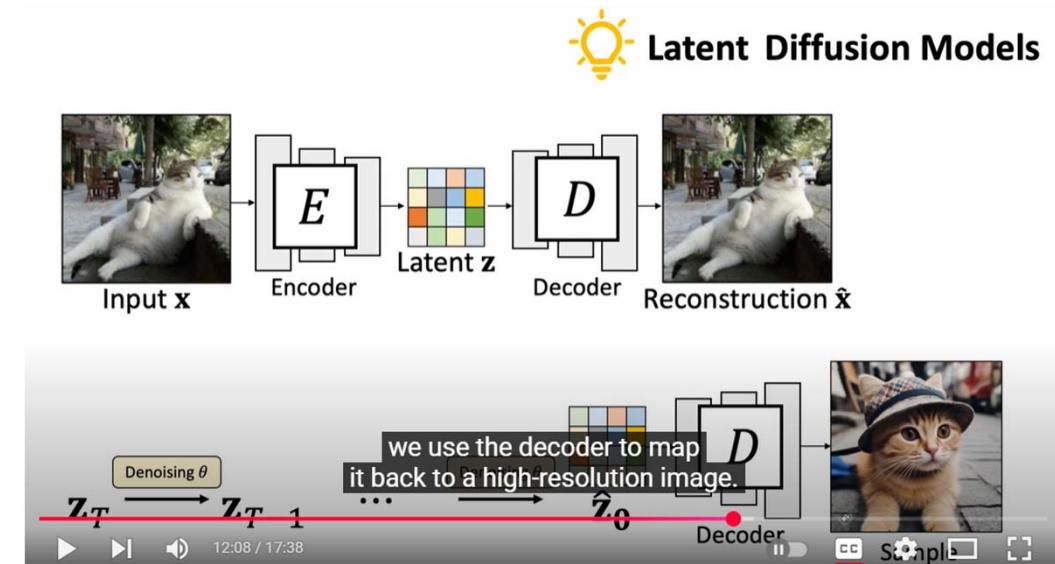
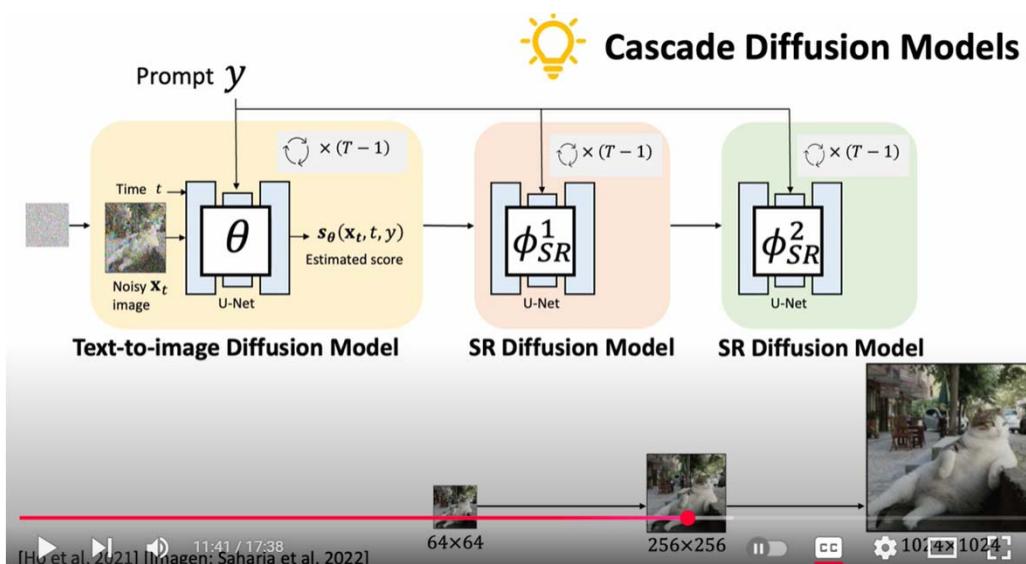
Diffusion models are powerful generative models that enable many successful applications like image, video, and 3D generation from...



# Diffusion Models: Generation High-Resolution Output

<https://www.youtube.com/watch?v=i2qSxMVeVLI>

- Cascade diffusion models
- Latent diffusion models

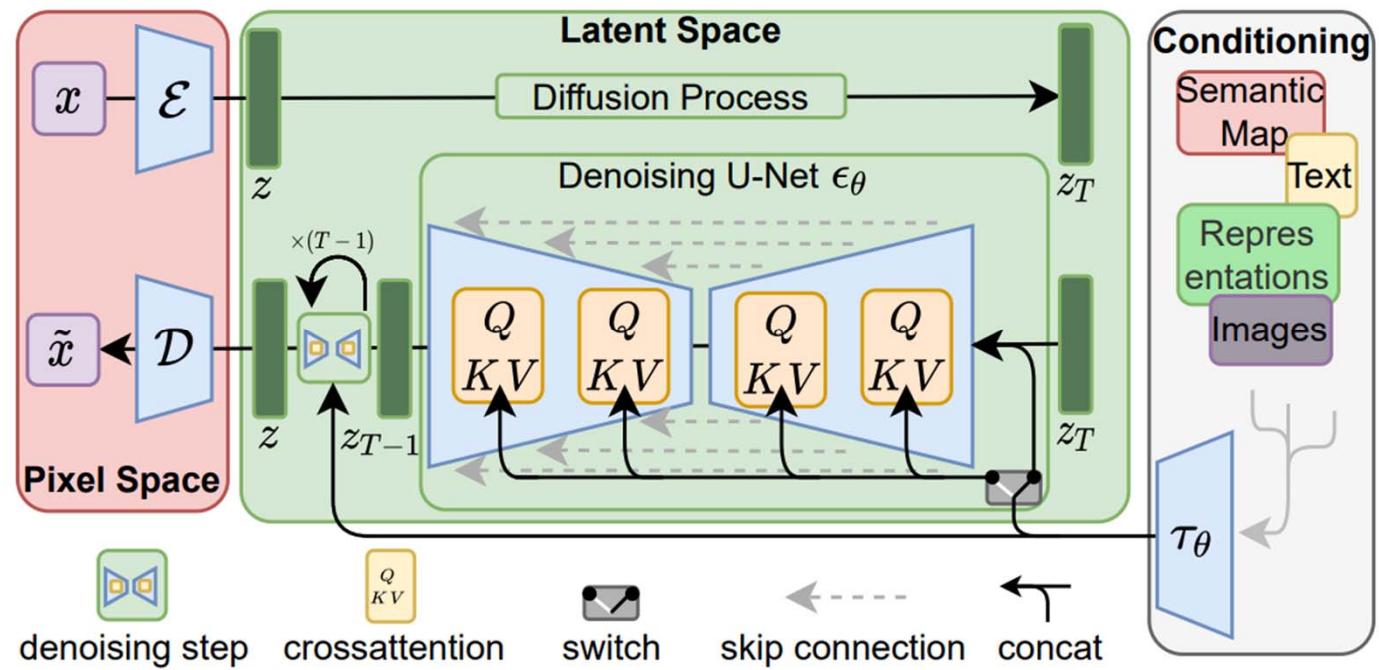


# Latent Diffusion Models

- Work on a **latent** space instead of pixel space
  - noisy latents  $\rightarrow$  clean latents
  - focus on the important, semantic bits of data
  - computational efficiency
- Latent space regularization
  - KL-reg
  - VQ-reg

$$x \in \mathbb{R}^{H \times W \times 3}$$

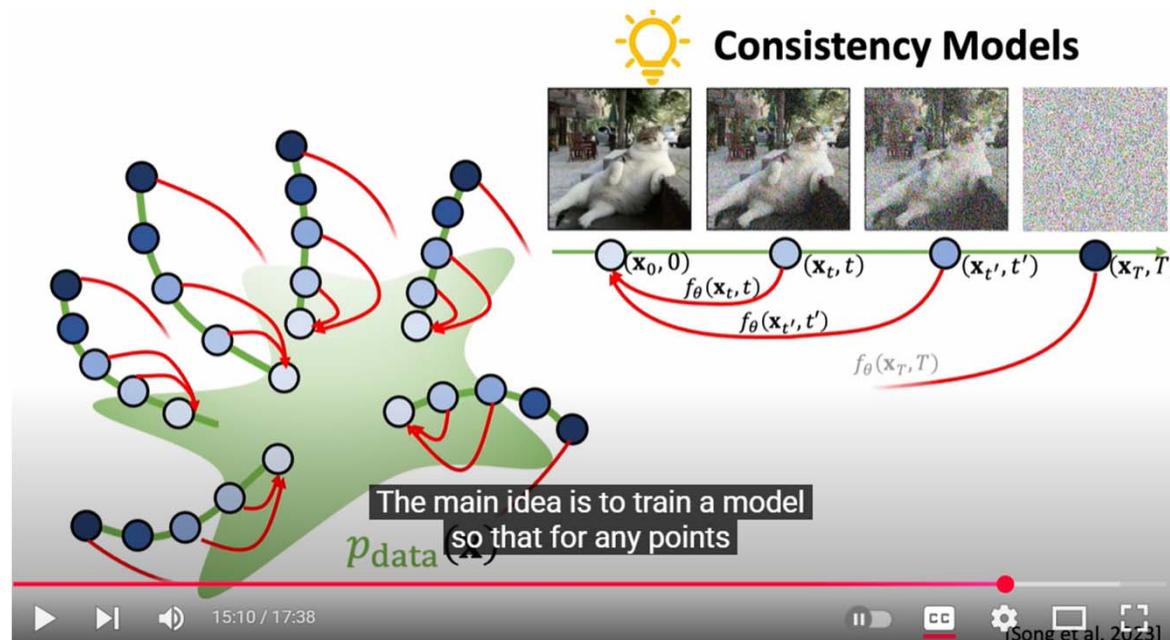
$$z \in \mathbb{R}^{h \times w \times c}$$



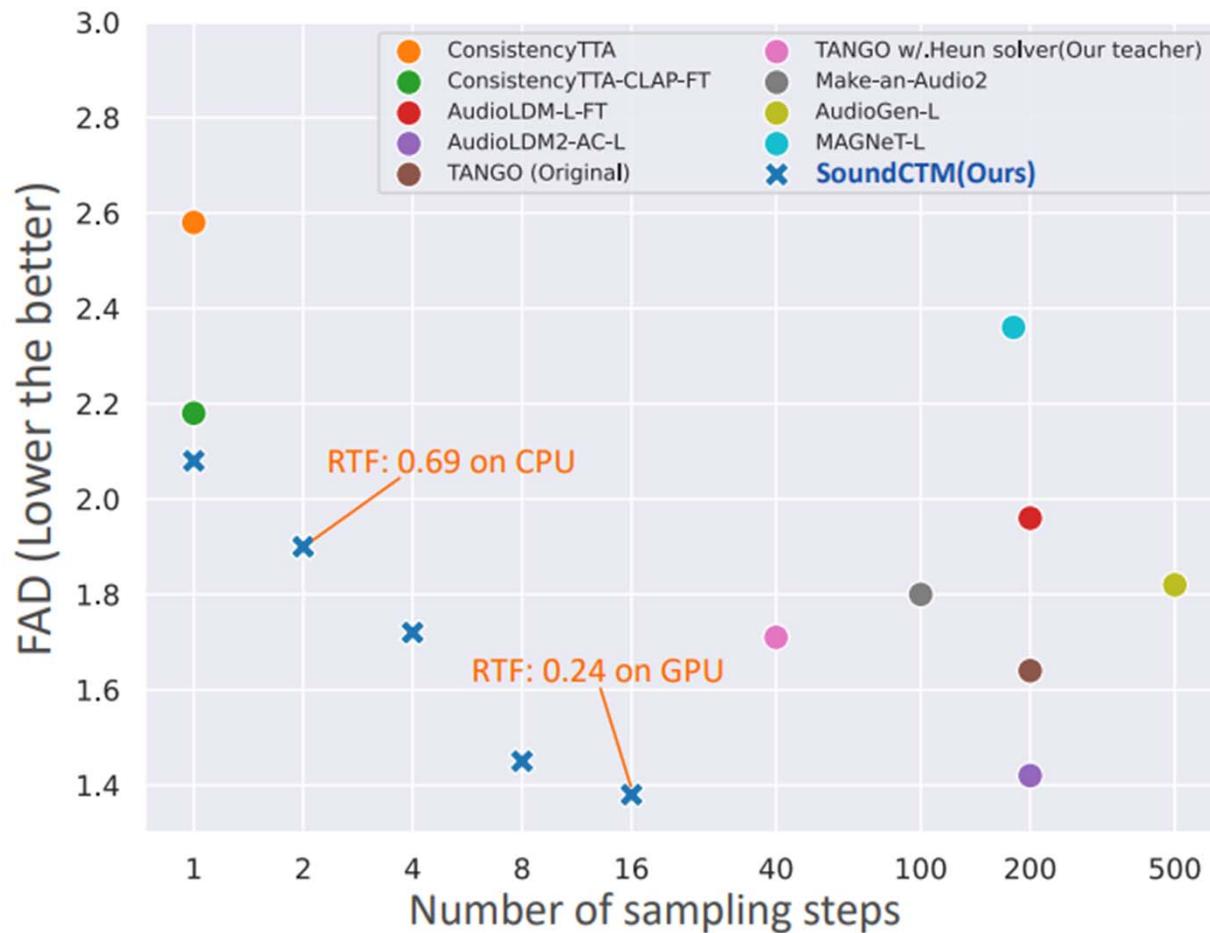
# Diffusion Models: Faster Inference

<https://www.youtube.com/watch?v=i2qSxMVeVLI>

- Progressive distillation / guided distillation
- Consistency models / latent consistency models



# Consistency Model: One-Step Generation



Ref: Saito et al, "SoundCTM: Uniting score-based and consistency models for text-to-sound generation," arXiv 2024

# Consistency Model: One-Step Generation

<https://sites.google.com/view/diffusion-tutorial-ismir24/materials>

## MEET OUR SPEAKERS



Chieh-Hsin (Jesse) Lai



Bac Nguyen



Koichi Saito



Yuki Mitsufuji

# Mean Flows (from Prof. Jia-Bin Huang)

<https://www.youtube.com/watch?v=swKdn-qT47Q>



The Rise of Single-Step  
Generative Models

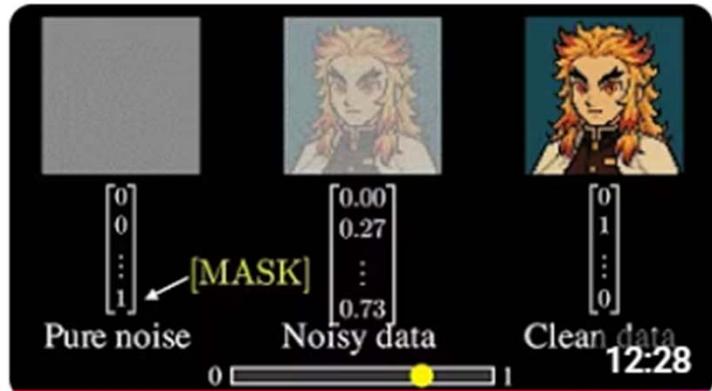
Jia-Bin Huang • 7.5K views

Diffusion and flow-matching models are key techniques for the  
current generative AI boom. However, their fundamental limitation i...



# Diffusion LM (from Prof. Jia-Bin Huang)

<https://www.youtube.com/watch?v=8BT0oc0yDVA>



But how do Diffusion Language  
Models actually work?

Jia-Bin Huang • 13K views

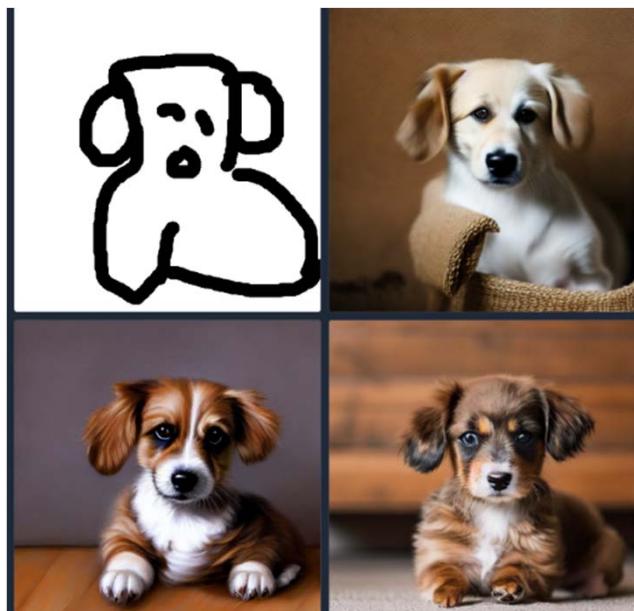
Most Large Language Models (LLMs) today are based on  
Autoregressive models (i.e., they predict texts in a left-to-right order...)

X :

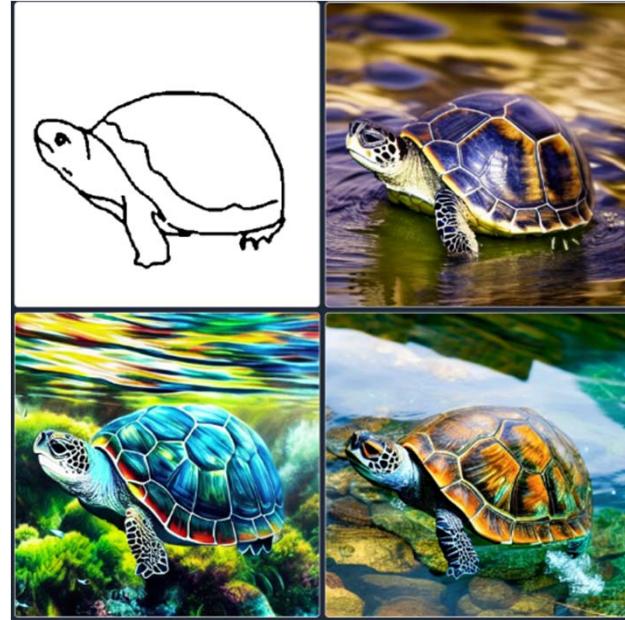
# Not Just Text: “ControlNet” for Adding More Controls

<https://github.com/lIlyasviel/ControlNet>

scribble map



scribble map



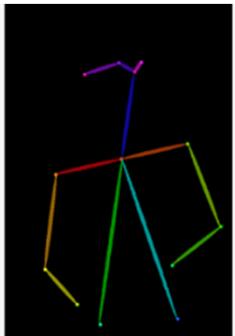
anime line drawing



# Not Just Text: “ControlNet” for Adding More Controls

<https://github.com/lIlyasviel/ControlNet>

human pose



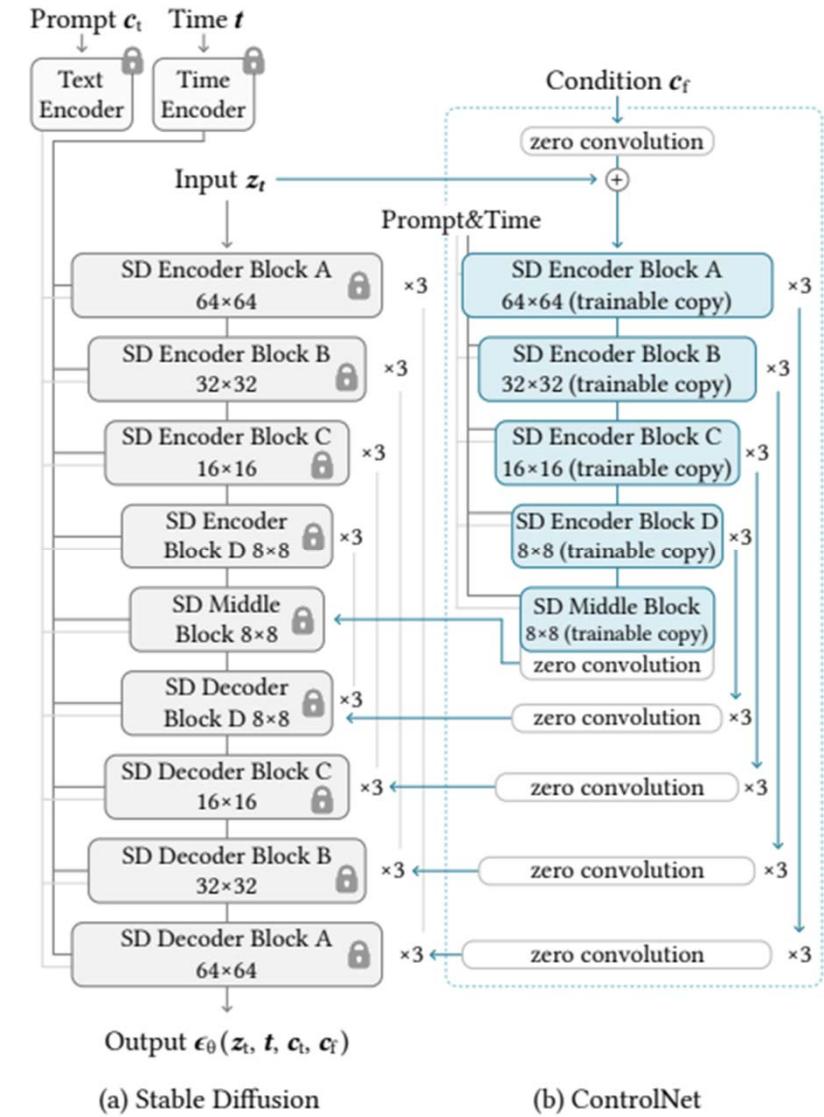
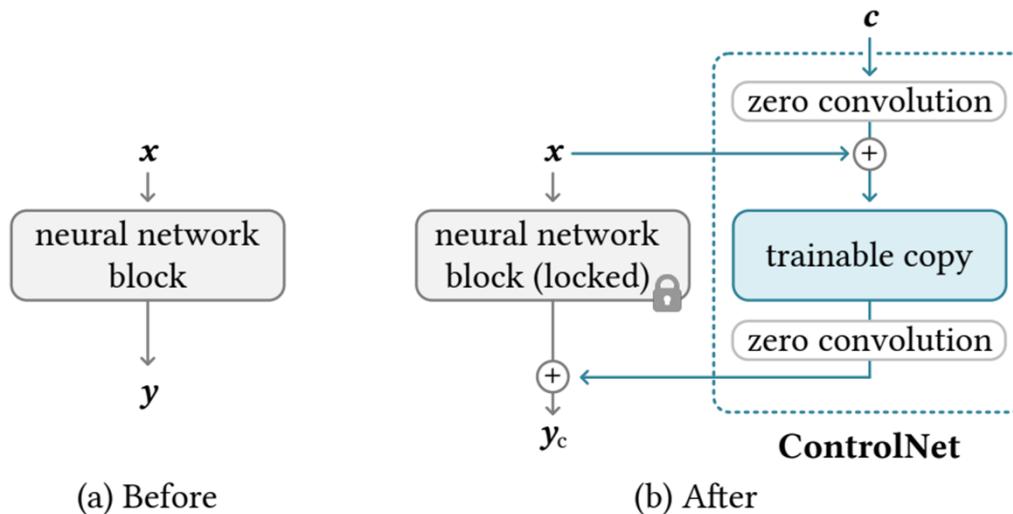
segmentation map



depth map



# ControlNet



Ref: Zhang et al, "Adding conditional control to text-to-image diffusion models," ICCV 2023

# ControlNet: Zero-init Convolutions

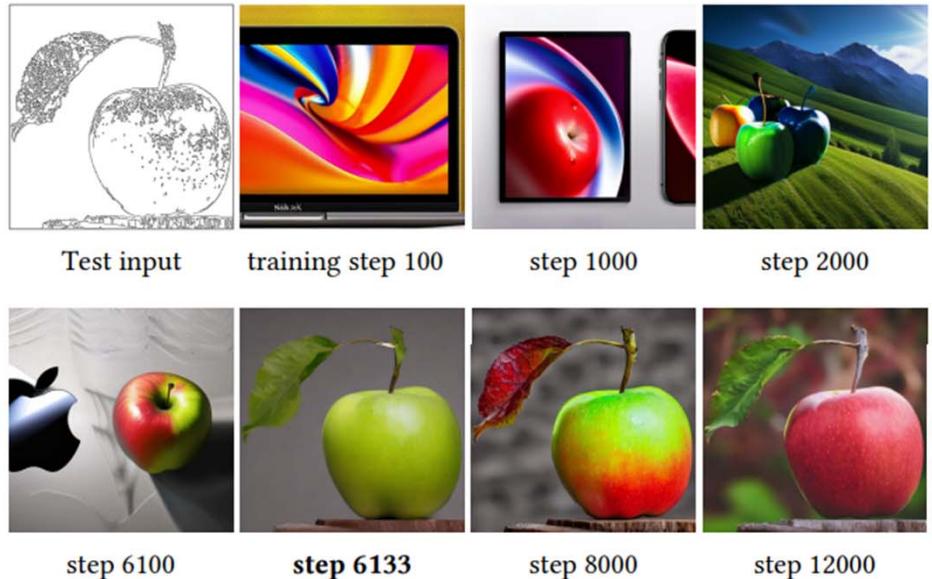
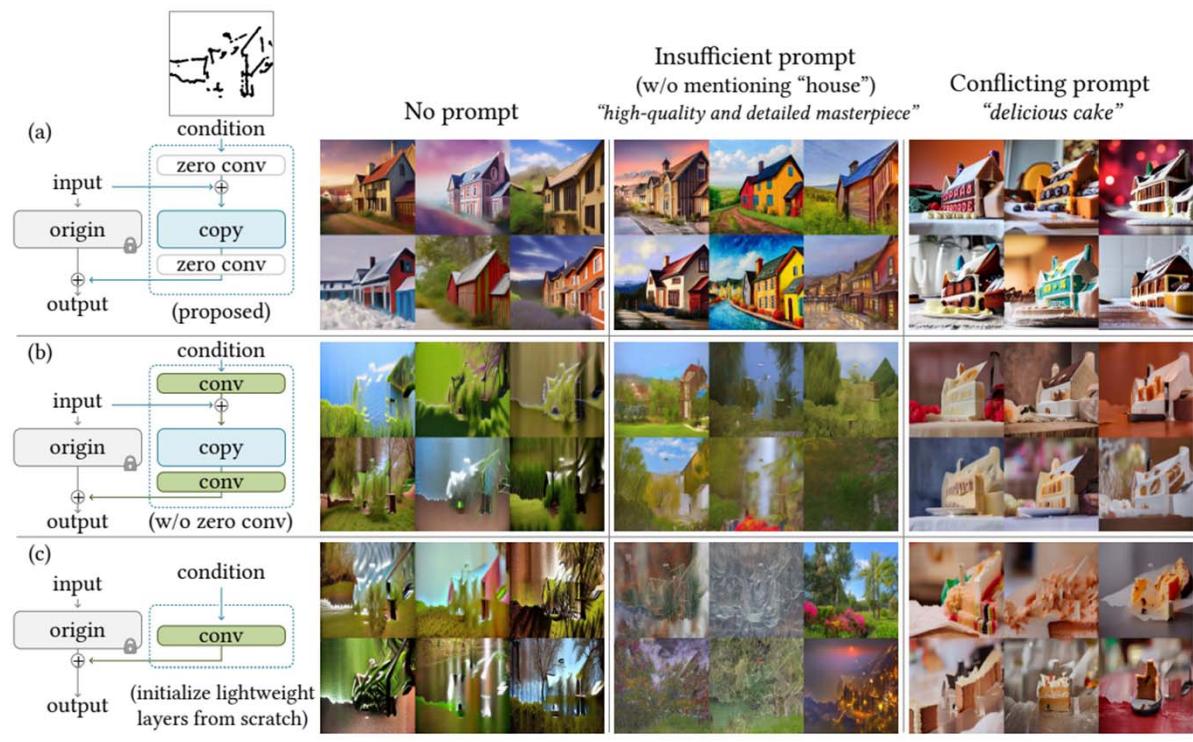


Figure 4: The sudden convergence phenomenon. Due to the zero convolutions, ControlNet always predicts high-quality images during the entire training. At a certain step in the training process (*e.g.*, the 6133 steps marked in bold), the model suddenly learns to follow the input condition.



# Diffusers

<https://github.com/huggingface/diffusers>

Task	Pipeline	🤗 Hub
Unconditional Image Generation	<a href="#">DDPM</a>	<a href="#">google/ddpm-ema-church-256</a>
Text-to-Image	<a href="#">Stable Diffusion Text-to-Image</a>	<a href="#">stable-diffusion-v1-5/stable-diffusion-v1-5</a>
Text-to-Image	<a href="#">unCLIP</a>	<a href="#">kakaobrain/karlo-v1-alpha</a>
Text-to-Image	<a href="#">DeepFloyd IF</a>	<a href="#">DeepFloyd/IF-I-XL-v1.0</a>
Text-to-Image	<a href="#">Kandinsky</a>	<a href="#">kandinsky-community/kandinsky-2-2-decoder</a>
Text-guided Image-to-Image	<a href="#">ControlNet</a>	<a href="#">Illyasviel/sd-controlnet-canny</a>
Text-guided Image-to-Image	<a href="#">InstructPix2Pix</a>	<a href="#">timbrooks/instruct-pix2pix</a>
Text-guided Image-to-Image	<a href="#">Stable Diffusion Image-to-Image</a>	<a href="#">stable-diffusion-v1-5/stable-diffusion-v1-5</a>
Text-guided Image Inpainting	<a href="#">Stable Diffusion Inpainting</a>	<a href="#">runwayml/stable-diffusion-inpainting</a>
Image Variation	<a href="#">Stable Diffusion Image Variation</a>	<a href="#">lambdalabs/sd-image-variations-diffusers</a>
Super Resolution	<a href="#">Stable Diffusion Upscale</a>	<a href="#">stabilityai/stable-diffusion-x4-upscaler</a>
Super Resolution	<a href="#">Stable Diffusion Latent Upscale</a>	<a href="#">stabilityai/sd-x2-latent-upscaler</a>

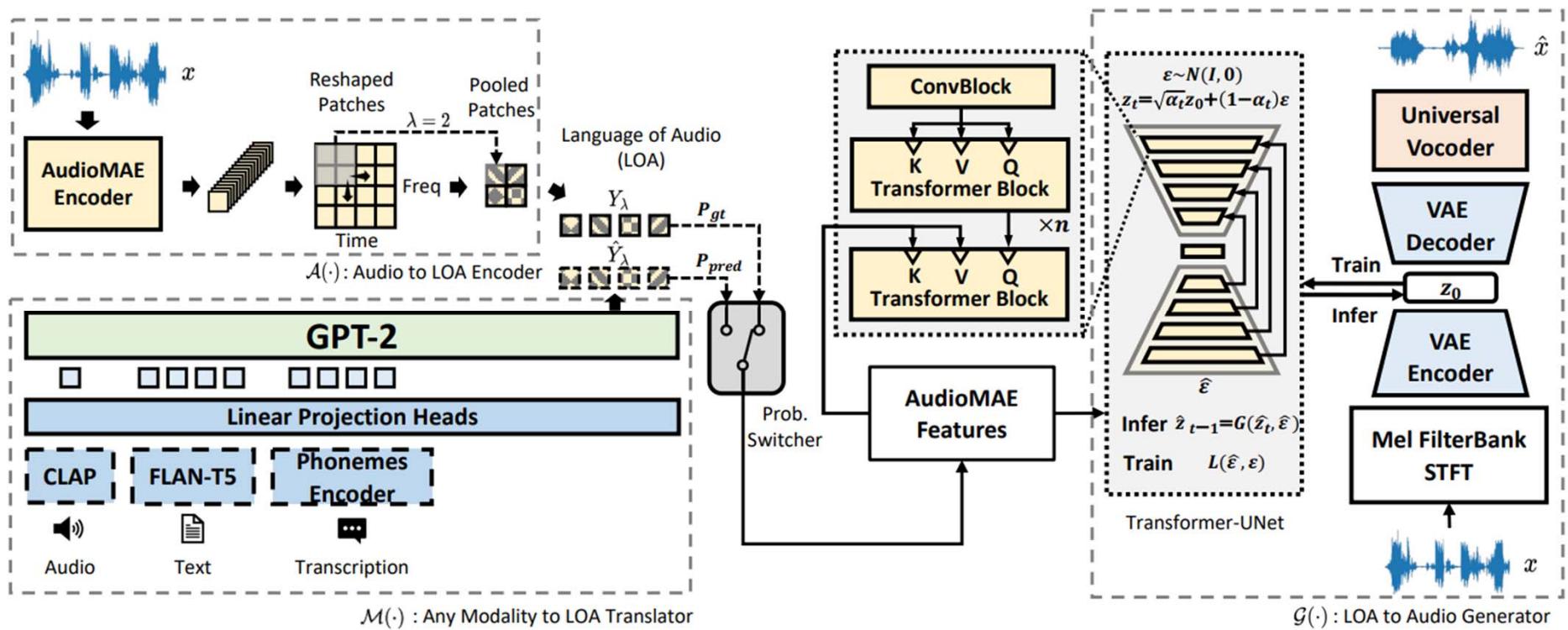
# Outline

- Diffusion models
- **Diffusion-based Text-to-music Generation**
- Music ControlNet & MuseControlLite

# AudioLDM2

<https://github.com/haoheliu/AudioLDM2>

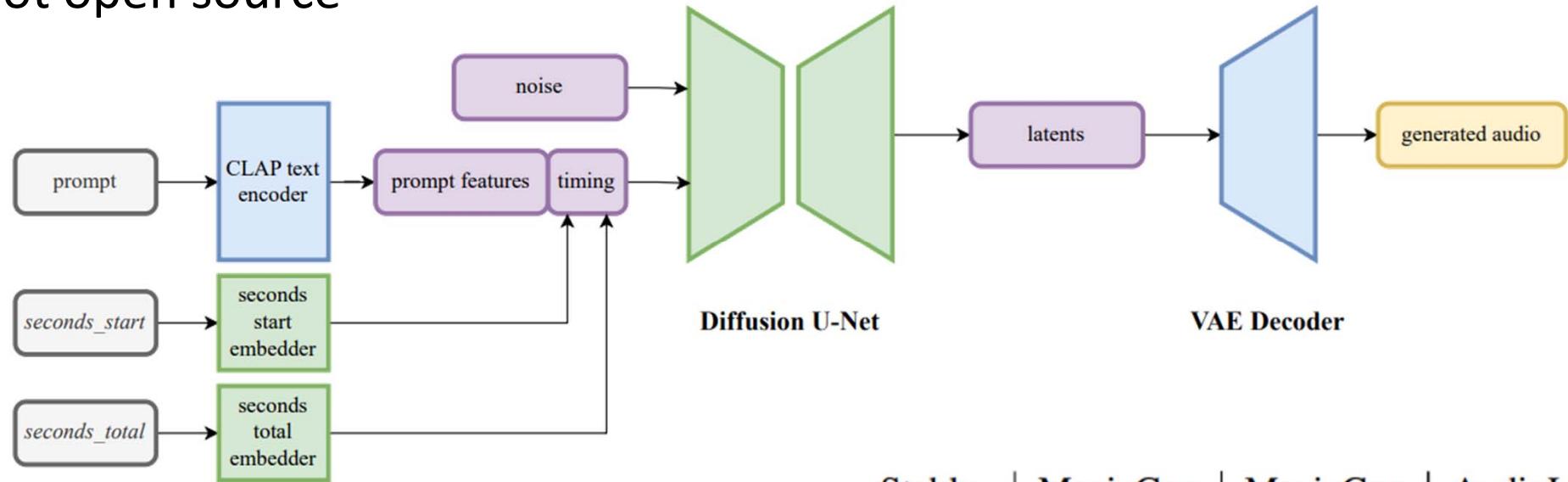
- Open source!
- Latent diffusion + Transformer U-Net



Ref: Liu et al, “AudioLDM 2: Learning holistic audio generation with self-supervised pretraining,” TASLP 2024

# Stable Audio

- Not open source



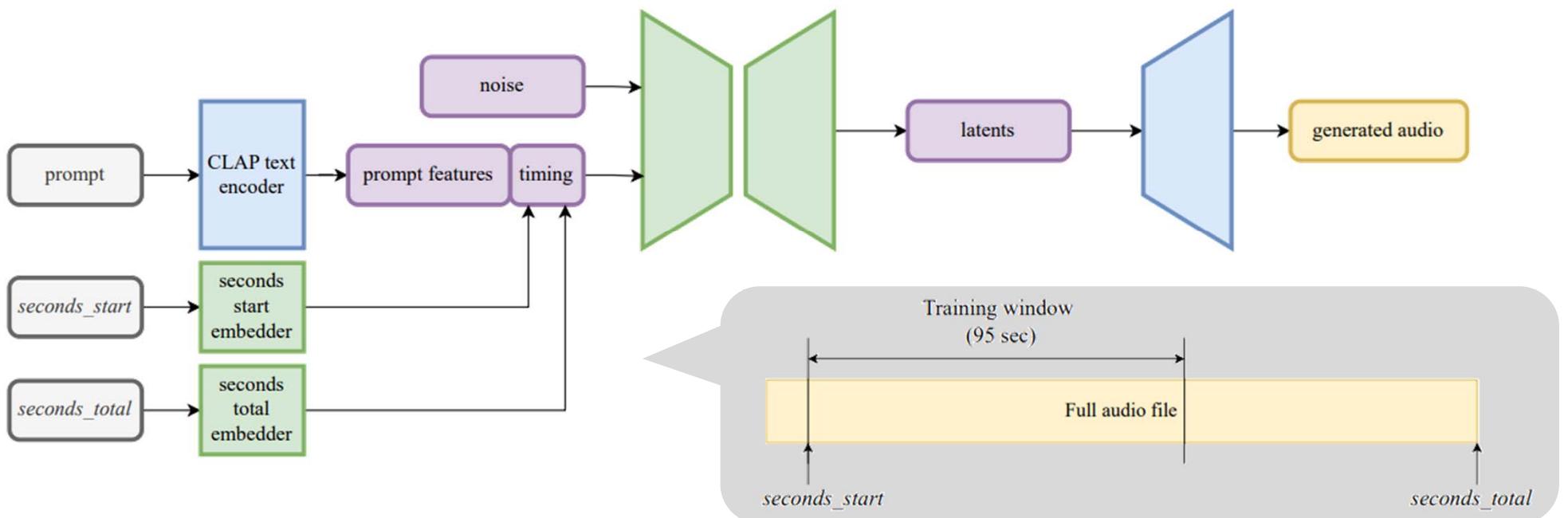
	Stable Audio	MusicGen large	MusicGen stereo	AudioLDM2 48kHz
Audio Quality	<b>3.0±0.7</b>	2.1±0.9	2.8±0.7	1.2±0.5
Text Alignment	<b>2.9±0.8</b>	2.4±0.9	2.4±0.9	1.3±0.6
Musicality	<b>2.7±0.9</b>	2.0±0.9	<b>2.7±0.9</b>	1.5±0.7

# Stable Audio

- Better latent representation
  - based on **DAC** but no VQ
    - “To allow for arbitrary-length audio encoding and decoding, we use a fully-convolutional architecture (133M parameters) that follows the Descript Audio Codec (Kumar et al., 2023) encoder and decoder (without the quantizer). We found that the **Snake activations** (Ziyin et al., 2020) in the Descript Audio Codec architecture improved audio reconstruction at high compression ratios compared to alternatives such as EnCodec (Défossez et al., 2022), at the expense of increased VRAM consumption.”
  - **sequence of latents**: maps a  $2 \times L$  input into  $64 \times (L/1024)$  latents
    - “The VAE is trained from scratch on our dataset and downsamples the input stereo audio sequence **by a factor of 1024**, with the resulting latent sequence having a **channel dimension of 64**.”
- Diffusion model
  - **U-Net** with 4 levels of down/upsampling, residual conv layers, self/cross-attention

# Stable Audio

- Conditioning
  - CLAP as the text encoder
  - **Timing embeddings:** `seconds\_start` & `seconds\_total` (for both training/inference)
    - represented as sinusoidal embeddings
    - this method allows the user generating **variable-length** music and sound effects



# Stable Audio

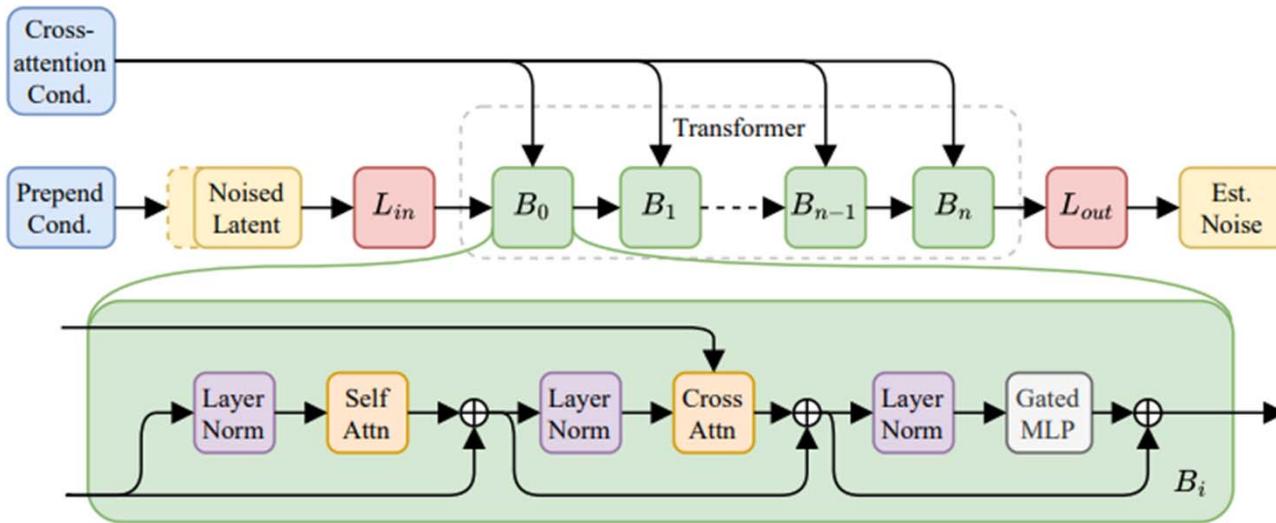
- Very fast

	channels/sr	output length	$FD_{openl3} \downarrow$	$KL_{passt} \downarrow$	$CLAP_{score} \uparrow$	inference time
Training data (upper bound)	2/44.1kHz	full songs	101.47	-	-	-
Autoencoded training data	2/44.1kHz	full songs	117.52	-	-	-
Stable Audio w/ CLAP <sub>ours</sub>	2/44.1kHz	23 sec	<u>118.09</u>	<u>0.97</u>	<u>0.44</u>	4 sec
Stable Audio w/ CLAP <sub>LAION</sub>	2/44.1kHz	23 sec	123.30	1.09	0.43	4 sec
Stable Audio w/ T5	2/44.1kHz	23 sec	126.93	1.06	0.41	4 sec
AudioLDM2-music	1/16kHz	95 sec	354.05	1.53	0.30	38 sec
AudioLDM2-large	1/16kHz	95 sec	339.25	1.46	0.30	37 sec
AudioLDM2-48kHz	<b>1/48kHz</b>	95 sec	299.47	2.77	0.22	242 sec
MusicGen-small	1/32kHz	95 sec	205.65	0.96	0.33	126 sec
MusicGen-large	1/32kHz	95 sec	197.12	0.85	0.36	242 sec
MusicGen-large-stereo	<b>2/32kHz</b>	95 sec	216.07	1.04	0.32	295 sec
Stable Audio	<b>2/44.1kHz</b>	95 sec	<b>108.69</b>	<b>0.80</b>	<b>0.46</b>	<b>8 sec</b>

# Stable Audio 2

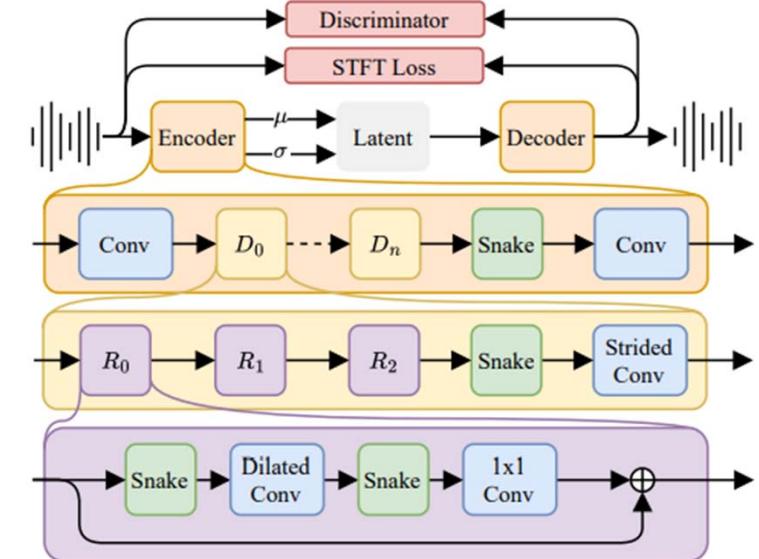
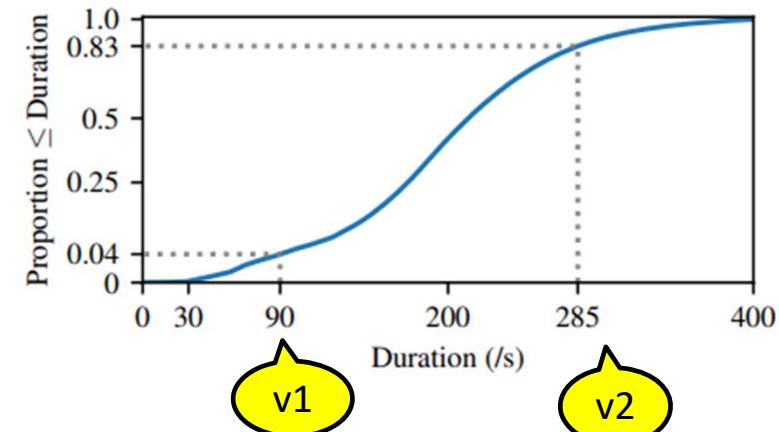
- Not open source

	DiT	AE	CLAP	Total
Parameters	1.1B	157M	125M	1.3B



**Figure 2:** Architecture of the diffusion-transformer (DiT).

Ref: Evans et al, "Long-form music generation with latent diffusion," ISMIR 2024



**Figure 3:** Architecture of the autoencoder.

# Stable Audio 1 vs Stable Audio 2

- U-Net → DiT
- 95 seconds → up to 4m 45s
- Reduced latent rate (43 Hz → 21.5 Hz)

	sampling rate	STFT distance ↓	MEL distance ↓	SI-SDR ↑	latent rate	latent (channels)
DAC [23]	44.1kHz	0.96	0.52	10.83	86 Hz	discrete
AudioGen [24]	48kHz	1.17	0.64	9.27	50 Hz	discrete
Encodec [8, 22]	32kHz	1.82	1.12	5.33	50 Hz	discrete
AudioGen [24]	48kHz	1.10	0.64	8.82	100 Hz	continuous (32)
Stable Audio [14]	44.1kHz	1.19	0.67	8.62	43 Hz	continuous (64)
Ours	44.1kHz	1.19	0.71	7.14	21.5 Hz	continuous (64)

# Stable Audio 2 vs MusicGen-large

	channels/sr	output length	$FD_{openl3} \downarrow$	$KL_{passt} \downarrow$	$CLAP_{score} \uparrow$	inference time
MusicGen-large-stereo [8]	2/32kHz	2m	204.03	0.49	0.28	6m 38s
Ours (fully-trained)	2/44.1kHz	2m <sup>†</sup>	79.09	0.35	0.40	13s
MusicGen-large-stereo [8]	2/32kHz	4m 45s	218.02	0.50	0.27	12m 53s
Ours (fully-trained)	2/44.1kHz	4m 45s	81.96	0.34	0.39	13s

Results with the fully-trained model:		2m long		4m 45s long		
		Stable Audio 2	MusicGen- large-stereo	ground truth	Stable Audio 2	MusicGen- large-stereo
Audio Quality		4.0±0.6	2.8±0.8	4.6±0.4	4.5±0.4	2.8±0.8
Text Alignment		4.3±0.7	3.1±0.8	4.6±0.5	4.6±0.4	2.9±1.0
Structure		3.5±1.3	2.4±0.7	4.3±0.8	4.0±1.0	2.1±0.7
Musicality		4.0±0.8	2.7±0.9	4.6±0.5	4.3±0.7	2.6±0.7
Stereo correctness		96%	61%	96%	100%	57%

# Stable Audio Open

<https://stability-ai.github.io/stable-audio-open-demo/>

- **Open source!**
  - trained on data from Freesound and FMA (about 500K recordings; or **7,300 h**)
  - all licensed under CC-0, CC-BY, or CC-Sampling+

	channels/sr	output length	$FD_{openT3} \downarrow$	$KL_{passt} \downarrow$	$CLAP_{score} \uparrow$
MusicGen-large-stereo [8]	2/32kHz	47	190.47	0.52	0.31
Stable Audio 1.0 [5]	2/44.1kHz	95 sec <sup>†</sup>	142.50	0.40	0.38
Stable Audio 2.0 [6]	2/44.1kHz	190 sec <sup>†</sup>	<b>71.25</b>	<b>0.37</b>	<b>0.42</b>
Stable Audio 2.0 [6]	2/44.1kHz	285 sec <sup>†</sup>	81.05	0.39	0.42
Stable Audio Open	2/44.1kHz	47	96.51	0.55	0.41

# Stable Audio Open

- Sequence length **47 sec**
  - It's actually a sequence of 1024 latents ( $47 \times 21.5$ )
- T5 as the text encoder
- DiT-based denoiser

	sampling rate	STFT distance ↓	MEL distance ↓	SI-SDR ↑	latent rate	latent (channels)
DAC [21]	44.1kHz	0.96	0.53	11.30	86Hz	discrete
AudioGen [20]	48kHz	1.16	0.66	9.64	50Hz	discrete
Encodec [22, 8]	32kHz	1.70	1.09	5.75	50Hz	discrete
AudioGen [20]	48kHz	1.10	0.65	9.21	100Hz	continuous (32)
Stable Audio 1.0 [5]	44.1kHz	1.20	0.67	9.12	43Hz	continuous (64)
Stable Audio 2.0 [6]	44.1kHz	<b>1.21</b>	<b>0.72</b>	<b>7.65</b>	21.5Hz	continuous (64)
Ours	44.1kHz	1.29	0.77	6.28	21.5Hz	continuous (64)

# Stable Audio Open: Training Data

- 486,492 recordings (7,300 h), where 472,618 (6,330 h) are from **Freesound** and 13,874 (970 h) from **FMA**
- Freesound
  - music/non-music classification by PANNs
  - the identified music was sent to a trusted content detection company to ensure the absence of copyrighted music
    - Most removed content was field recordings in which copyrighted music was in the background
  - most common tags: single-note, synthesizer, field-recording, drum, loop, ambient
- FMA
  - did a metadata search against a large database of copyrighted music and flagged any potential match to be reviewed by humans
  - Most common tags: instrumental, electronic, experimental, soundtrack, ambient

# Stable Audio Open: Training Data

- Freesound includes natural language descriptions as well as the title of the recording and tags
- FMA includes metadata like year, genres, album, title, and artist
- Generate text prompts from the metadata by concatenating a random subset of the metadata as a string
  - “For metadata-types with a list of values, like tags or genres, we shuffle the list”
  - “For half of the FMA prompts, we include the metadata-type (e.g., artist or album) and join them with a comma (e.g., “year: 2021, artist: dadabots, album: can’t play instruments, title: pizza hangover”).”
  - “For the other half, we do not include the metadata-type and join them with a comma (e.g., “dadabots, can’t play instruments, pizza hangover, 2021”).”

# Stable Audio Open: Training Compute

- Autoencoder (156M)
  - “It trained for **183 h** on **x32 A100s** with a batch size of 4. At this point the encoder was frozen, and the decoder was trained for another **273 h** on **x32 A100s** with a batch size of 8.”
- DiT denoiser (1057M)
  - “The DiT is trained to predict a noise increment from noised groundtruth latents, following the v-objective approach.”
  - “During inference, we use the DPM-Solver++ [25] for 100 steps with classifierfree guidance (scale of 7.0).”
  - “The DiT is trained for **338 h** on **x64 A100s** with a batch size of 4.”
- T5 text encoder (109M)
  - use the publicly available T5-base checkpoint

# Stable Audio Tools

<https://github.com/Stability-AI/stable-audio-tools>

## Start training

To start a training run, run the `train.py` script in the repo root with:

```
$ python3 ./train.py --dataset-config /path/to/dataset/config --model-config /path/to/model/config --na
```

## Fine-tuning

Fine-tuning a model involves continuing a training run from a pre-trained checkpoint.

To continue a training run from a wrapped model checkpoint, you can pass in the checkpoint path to `train.py` with the `--ckpt-path` flag.

To start a fresh training run using a pre-trained unwrapped model, you can pass in the unwrapped checkpoint to `train.py` with the `--pretrained-ckpt-path` flag.

# Stable Audio Tools

<https://github.com/Stability-AI/stable-audio-tools>

## Start training

To start a training run, run the `train.py` script in the repo root with:

```
$ python3 ./train.py --dataset-config /path/to/dataset/config --model-config /path/to/model/config --na
```

## Fine-tuning

Fine-tuning a model involves continuing a training run from a pre-trained checkpoint.

To continue a training run from a wrapped model checkpoint, you can pass in the checkpoint path to `train.py` with the `--ckpt-path` flag.

To start a fresh training run using a pre-trained unwrapped model, you can pass in the unwrapped checkpoint to `train.py` with the `--pretrained-ckpt-path` flag.

# Stable Audio Open Small

<https://stability.ai/news/stability-ai-and-arm-release-stable-audio-open-small-enabling-real-world-deployment-for-on-device-audio-control>

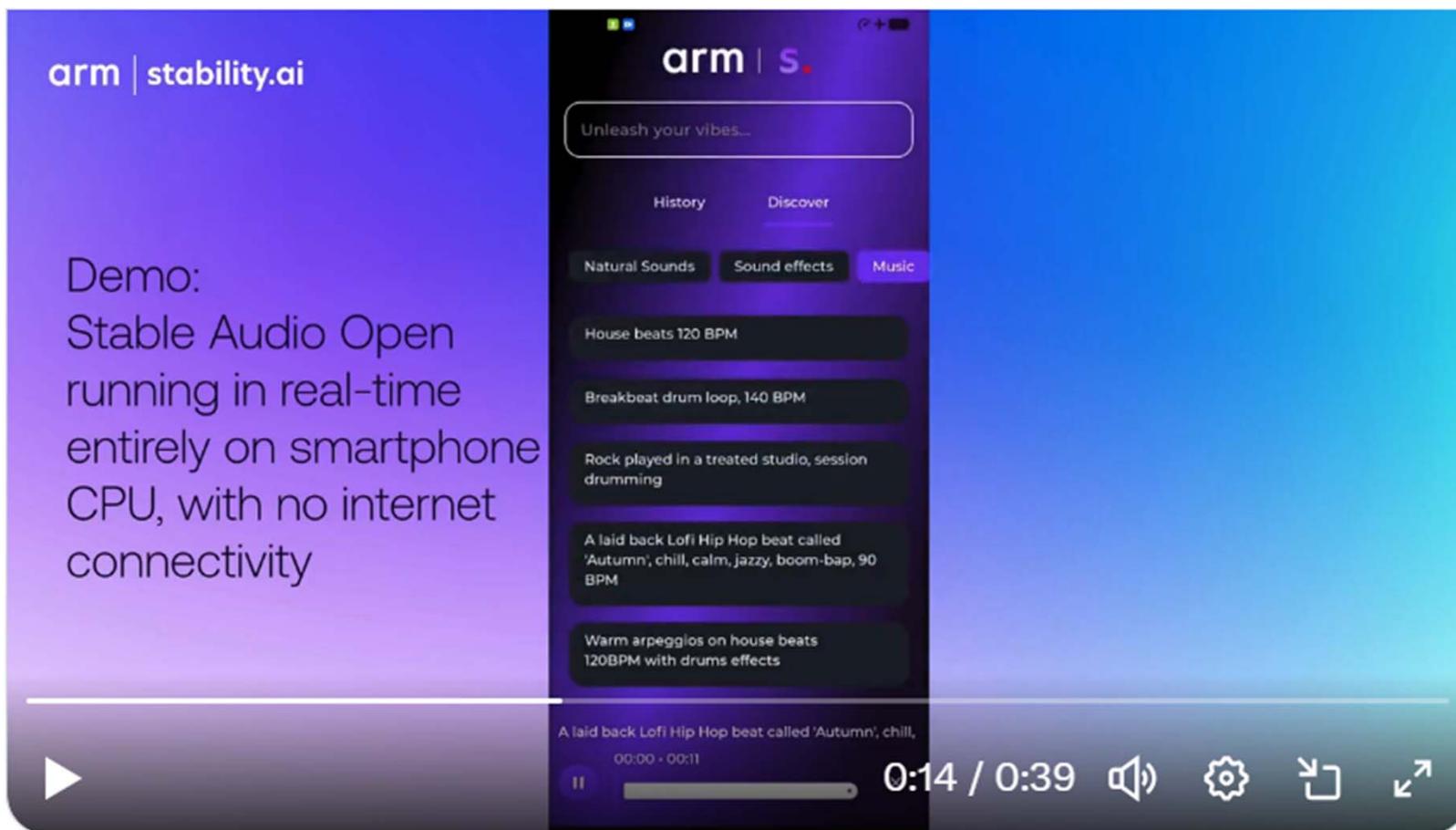
2025年5月14日

## Key Takeaways:

- We're open-sourcing Stable Audio Open Small, a 341 million parameter text-to-audio model optimized to run entirely on Arm CPUs. Designed for quickly generating short audio samples, it can produce up to 11 seconds of audio on a smartphone in less than 8 seconds.
- This release builds on our collaboration with Arm to bring generative audio creation to smartphones, following our [recent announcement](#) at Mobile World Congress.

# Stable Audio Open Small: On-Device Generation

<https://x.com/zacknovack/status/1896602380244054168>



# Stable Audio 2.5

<https://stability.ai/news/stability-ai-introduces-stable-audio-25-the-first-audio-model-built-for-enterprise-sound-production-at-scale>

2025年9月10日

## Key Takeaways:

- We're launching Stable Audio 2.5, the first audio generation model designed specifically for enterprise-grade sound production.
- Stable Audio 2.5 is purpose-built for this challenge of creating customizable, high-quality audio at scale. That includes elevated musical composition, fast inference at less than two seconds on a GPU, and support for more control with audio inpainting.
- You can try Stable Audio 2.5 now at [StableAudio.com](#) or seamlessly deploy through the [Stability AI API](#); partner platforms such as [fal](#), [Replicate](#), and [ComfyUI](#); and on-premises with an [enterprise license](#).

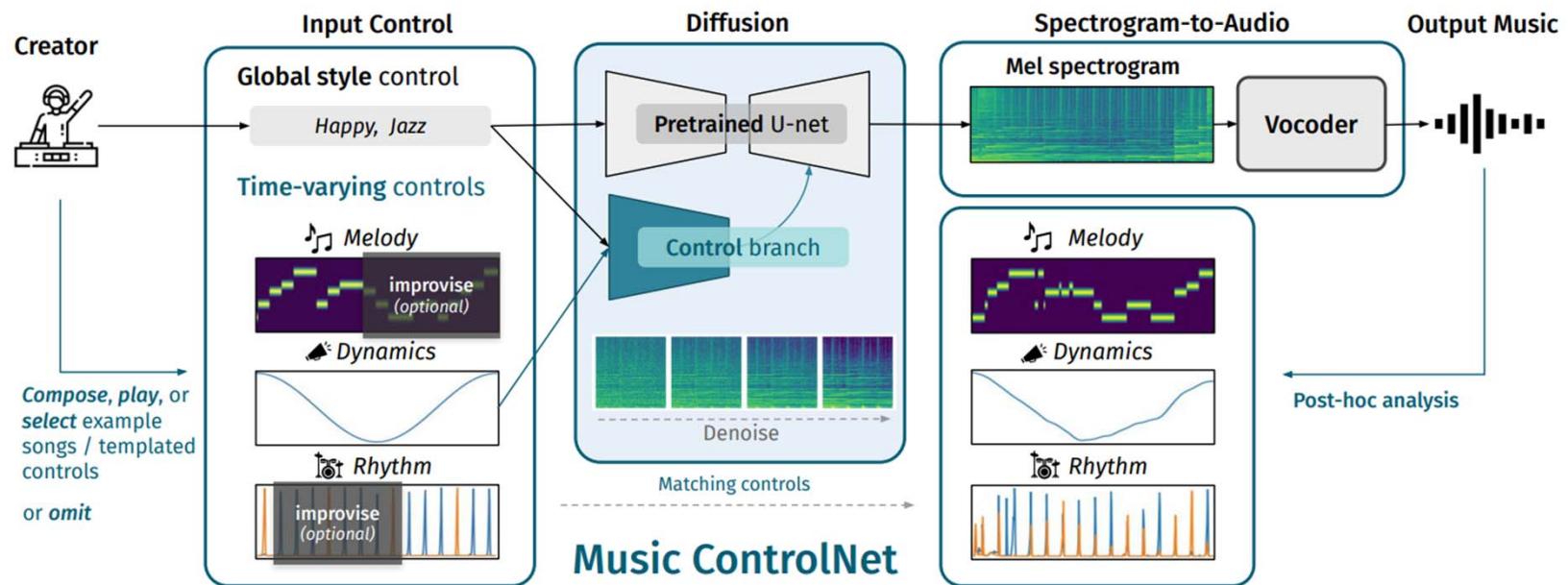
# Outline

- Diffusion models
- Diffusion-based Text-to-music Generation
- **Music ControlNet & MuseControlLite**

# Music ControlNet (for Attribute Control)

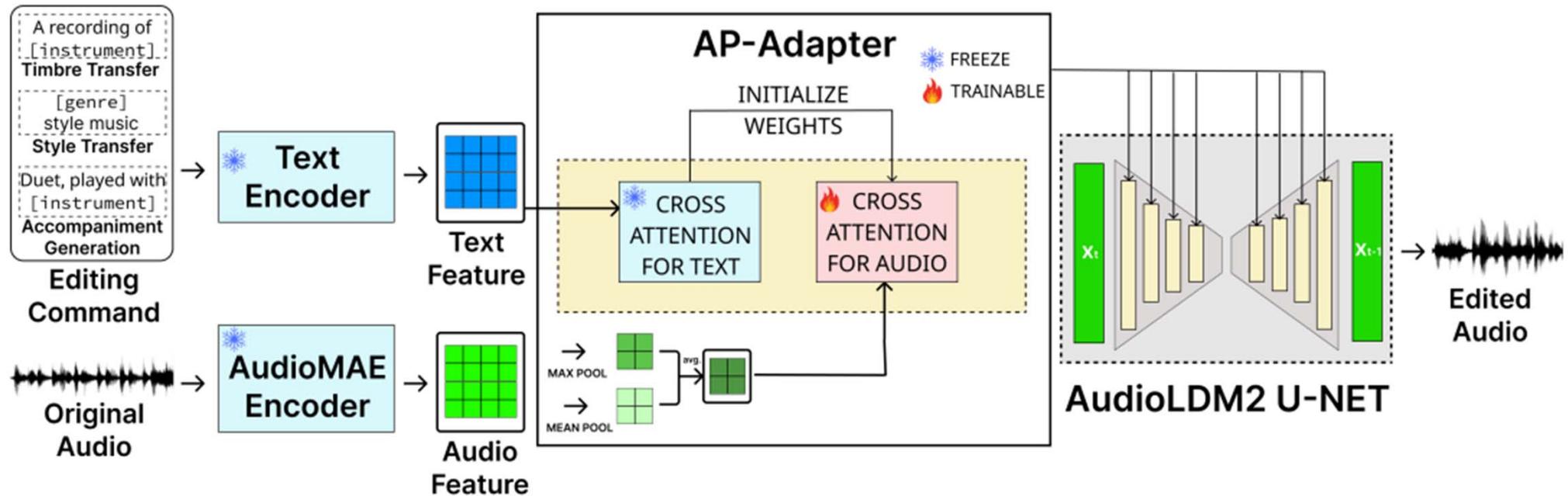
<https://musiccontrolnet.github.io/web/>

- Work in the “pixel” domain
- Not open source



# AP-Adapter (for Text+Audio Control)

<https://young-almond-689.notion.site/Audio-Prompt-Adapter-Unleashing-Music-Editing-Abilities-For-Text-To-Music-with-Lightweight-Finetuni-fbbfeb0608664f61a6bf894d56e85820>



Ref: Tsai et al, "Audio Prompt Adapter: Unleashing music editing abilities for text-to-music with lightweight finetuning," ISMIR 2024

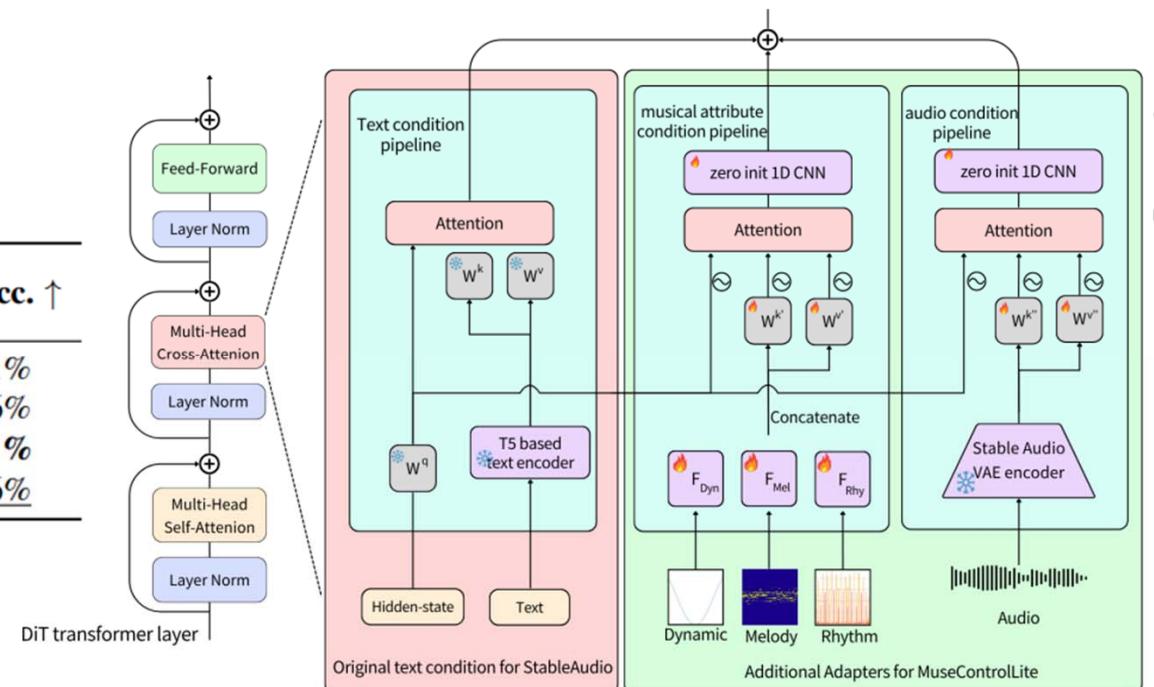
# MusicControlLite (for Attribute & Audio Control)

<https://musecontrollite.github.io/web/>

[https://colab.research.google.com/drive/1rR-Ncng\\_gSeb6hX0LY20SA4O9RCF-ZF3](https://colab.research.google.com/drive/1rR-Ncng_gSeb6hX0LY20SA4O9RCF-ZF3)

- Latent diffusion
- Open source

Model	FD ↓	KL ↓	CLAP ↑	Mel acc. ↑
MusicGen-Stereo-Large-Melody	193.66	0.436	0.354	43.1%
Stable Audio Open ControlNet	97.73	<b>0.265</b>	<b>0.396</b>	56.6%
Ours (MuseControlLite-Melody)	<b>76.42</b>	0.289	0.372	<b>61.1%</b>
Ours (MuseControlLite-Attr)	<u>80.79</u>	<u>0.271</u>	<u>0.373</u>	<u>60.6%</u>



Ref: Tsai et al, "MuseControlLite: Multifunctional music generation with lightweight conditioners," ICML 2025

# MusicControlLite (for Singing Accompaniment Generation)

<https://musecontrollite.github.io/web/>

