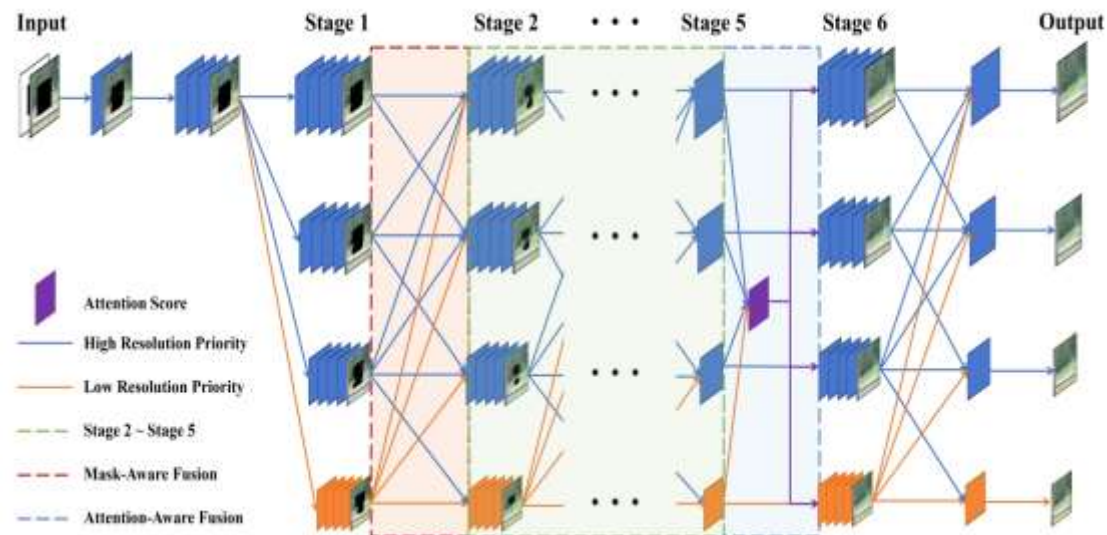


# 视觉内容生成

# 个人信息

- 助理教授 张健夫
  - 2011 交大ACM班 | 2015 交大计算机系博士 | 2023 计算机学院
  - [c.sis@sjtu.edu.cn](mailto:c.sis@sjtu.edu.cn) <https://matt-lab.org/>
  - 研究方向：人工智能 | 视觉内容生成 / 深度模型可靠性
  - 助教：展家慧 计算机学院 博士二年级
  - 负责课程
    - John班 大二 专业选修课 《图像处理与视觉》
    - 研究生专业前沿课 《设计与理解神经网络》
    - ACM班 大三 专业选修课 《视觉内容生成》

# 代表工作：视觉内容修复/编辑

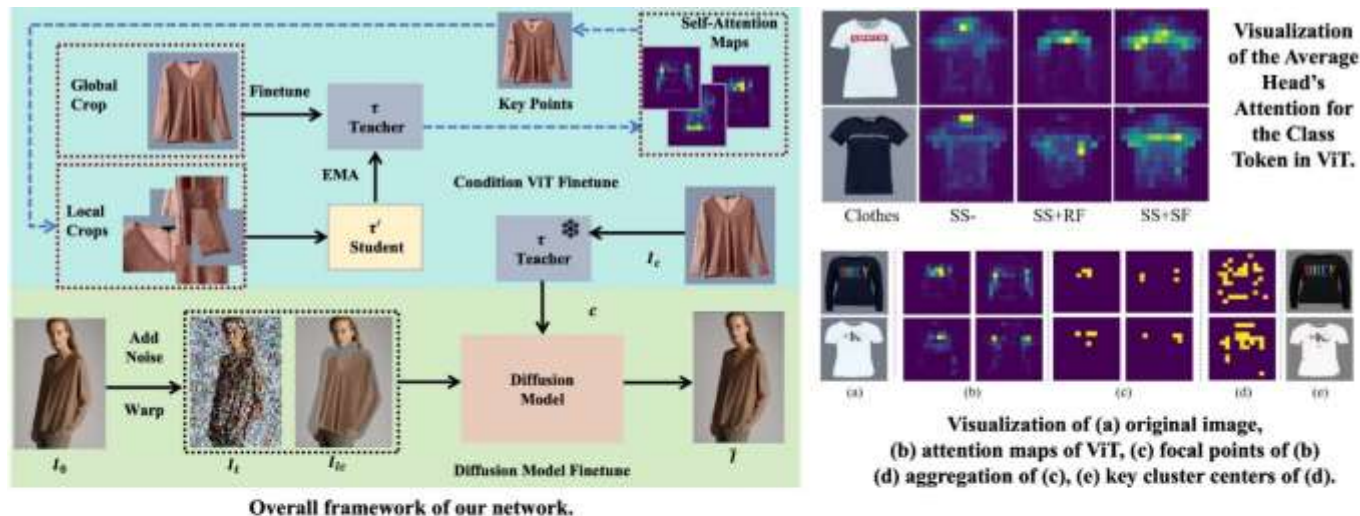


通过修复/编辑区域规划多视角/多尺度融合提升图像/视频修复编辑能力  
合作开发 马卡龙玩图，获得Apple APP Store “最佳本土APP”

[ACM MM 2019] J Zhang, L Niu, D Yang, L Kang, Y Li, W Zhao, L Zhang. GAIN: Gradient Augmented Inpainting Network for Irregular Holes.  
[ICCV 2021] \*W Wang, \*J Zhang, L Niu, H Ling, X Yang, L Zhang. Parallel Multi-Resolution Fusion Network for Image Inpainting.

[PR 2023] W Wang, L He, L Niu, J Zhang, Y Liu, H Ling, L Zhang. Diverse image inpainting with disentangled uncertainty.  
[CVPR 2022] W Wang, L Niu, J Zhang, X Yang, L Zhang. Dual-path image inpainting with auxiliary gan inversion.

# 代表工作：虚拟服装试穿



通过服装变形引导扩散模型生成高精度服装试穿图像，利用自监督ViT进一步提升对服装细节的处理。  
公开代码在GitHub半年内迅速获得350+star，为虚拟服装试穿领域目前最有影响力的公开工作。





**Dataset Construction**

**Fine-tuning MLLM**

**FakeXplainer**

Trained on the dataset with

**SFT + *Progressive* GRPO**

based on *Qwen-2.5-VL-32B-Instruct*



Generated by FLUX.1-Schnell

# 评分规则

- 无平时分
- 小作业 60%（满足要求即给足分）
  - 3次，约半个月一次
  - 内容：VAE/GAN/Diffusion Model
  - 我们会提供 代码（需做少量修改）/计算资源
    - 可以用不同的代码/计算资源，需提前告知
  - 提交简易报告
    - 每人使用的数据必须不同
- 科研项目 40%
  - 1-3人一组
  - 从不同课题中选择一个作为研究课题
    - 严禁抄袭
  - 提交报告（8页单栏中文/英文报告）

## About Homework ...

小作业约3次，涉及VAE/GAN/Diffusion等生成模型的搭建。

作业在 Canvas 上提交，只需要上传一份 **.ipynb** 文件，务必保留每个单元格的运行结果。

为保证较为简单的开发环境配置，我们建议在 **Colab** 上进行工作。如果你的PC配置允许，也可以在本机运行。

注意事项：

1. 请在标明的 **TODO** 位置完成代码，无需更改其他代码；
2. 按顺序运行完成后即可保存或导出，各代码块的运行结果将会保留。
3. 如果你在 **Colab** 上运行，建议仅在必要时使用 **GPU**，以节约免费的动态限额。

The assignments involve completing 3 tasks related to generative models such as VAE, GAN, and Diffusion.

The submission is required on Canvas, where you only need to upload a single **.ipynb** file, ensuring that **all output from the code cells is preserved**.

To simplify the development environment, it is recommended to work on **Colab**. However, if your local machine has sufficient resources, you can run the tasks locally.

Important Notes:

1. Complete the code at the designated **TODO** sections without modifying other parts of the code.
2. After running each code block in sequence, you can save or export the notebook. The output from each code cell will be retained.
3. If using Colab, it is advisable to use GPU only when absolutely necessary to conserve your free dynamic limits.

## 0. Load Preliminary Functions

### a. Import Libraries and Functions

### b. MNIST Data Loader

### c. Training Function

### d. Evaluation Function

## 1. AutoEncoder (AE)

### a. Model

### b. Loss Functions and Optimizers

### c. Training

### d. Evaluation

## 2. Variational AutoEncoder (VAE)

### b. Loss Functions

Loss 2: KL Divergence w/o Estimation

### c. Verifying loss 1 == loss 2

### d. Training with ELBO loss

### e. Evaluation

### f. Training with KL Divergence w/o Estimation

```
class AE(torch.nn.Module):
    def __init__(self, input_dim, hidden_dims):
        super().__init__()

        assert hidden_dims[-1] == 2, "always use 2 as the latent dimension for"
        self.encoder = torch.nn.Sequential()
        self.decoder = torch.nn.Sequential()

        #####
        ### Problem 1 (a): finish the implementation for encoder and decoder
        ### TODO:
        #####

    def decode(self, z):
        return self.decoder(z)

    def forward(self, x):
        encoded = self.encoder(x)
        decoded = self.decode(encoded)
        return {"imgs": decoded}

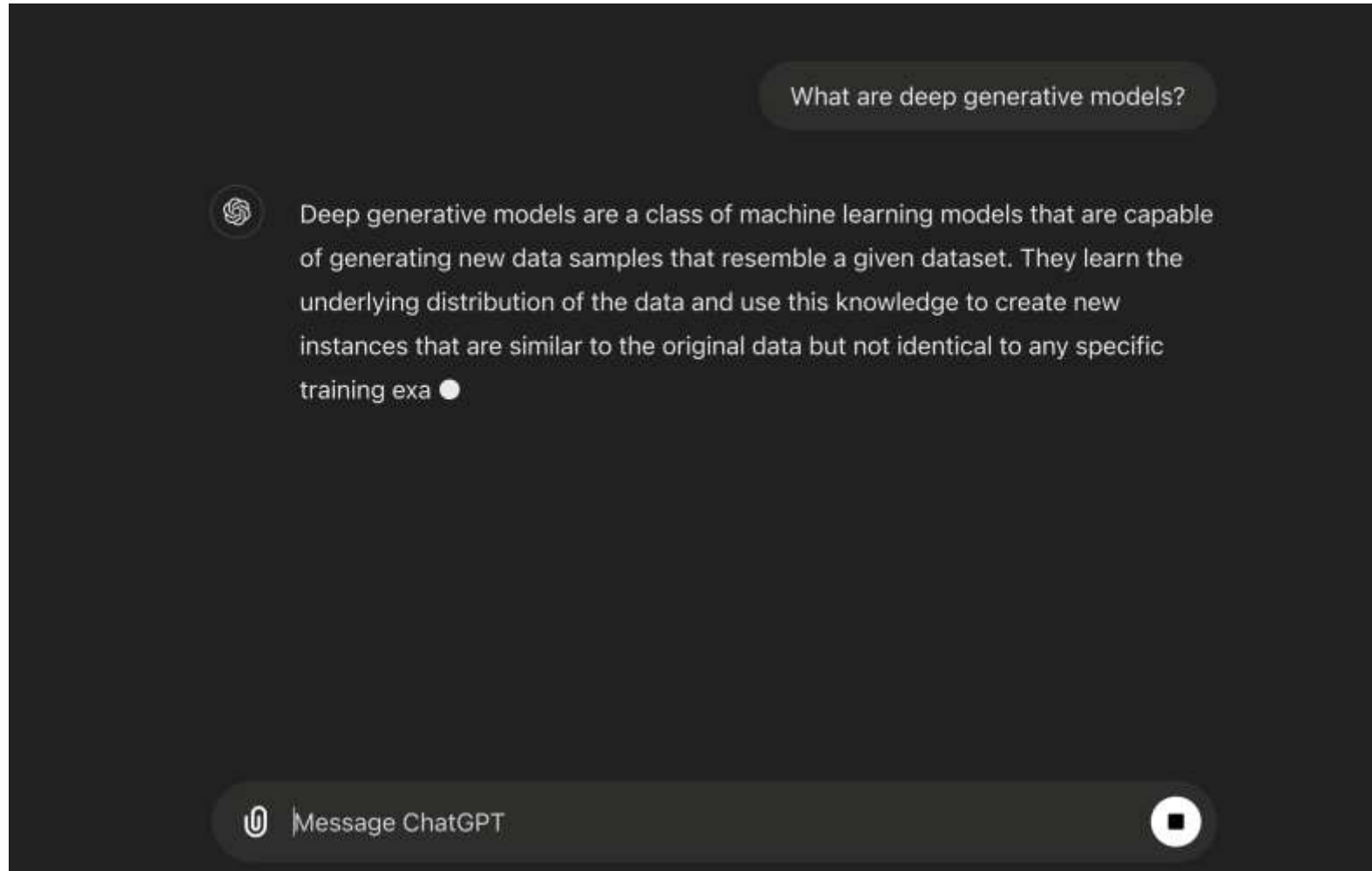
### Test
hidden_dims = [128, 64, 36, 18, 2]
input_dim = 256
test_tensor = torch.randn([1, input_dim]).to(device)

ae_test = AE(input_dim, hidden_dims).to(device)

with torch.no_grad():
    test_out = ae_test(test_tensor)
```

# The “GenAI” Era

## Chatbot and natural language conversation



# The “GenAI” Era

## Text-to-image generation



Prompt: --aesthetic-- #boho #fashion, full-body 30-something woman laying on microfloral grass, candid pose, overlay reads Stable Diffusion 3.5, cheerful cursive typography font.

<https://stability.ai/news/introducing-stable-diffusion-3-5>

# The “GenAI” Era

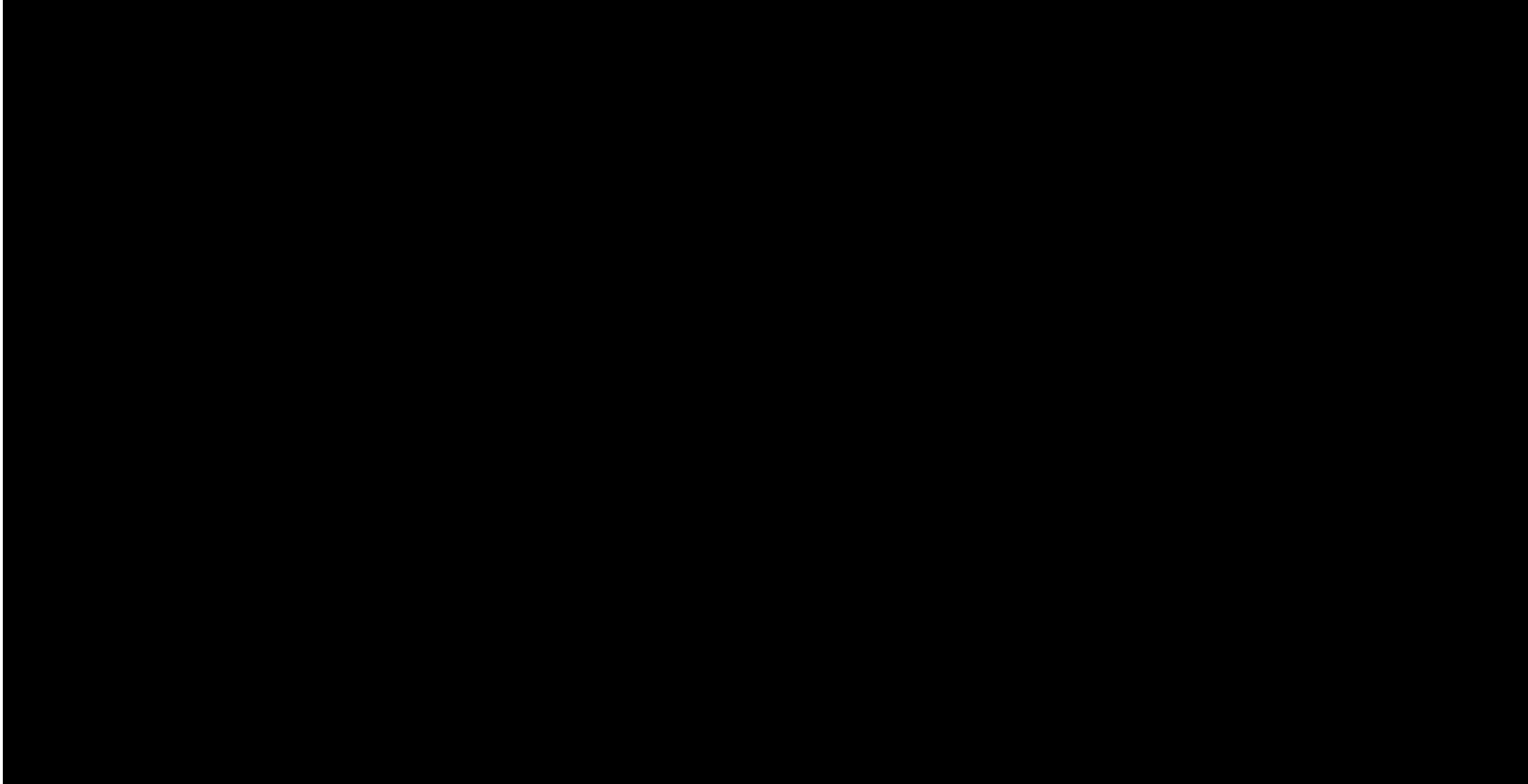
Text-to-video generation



*Sora*

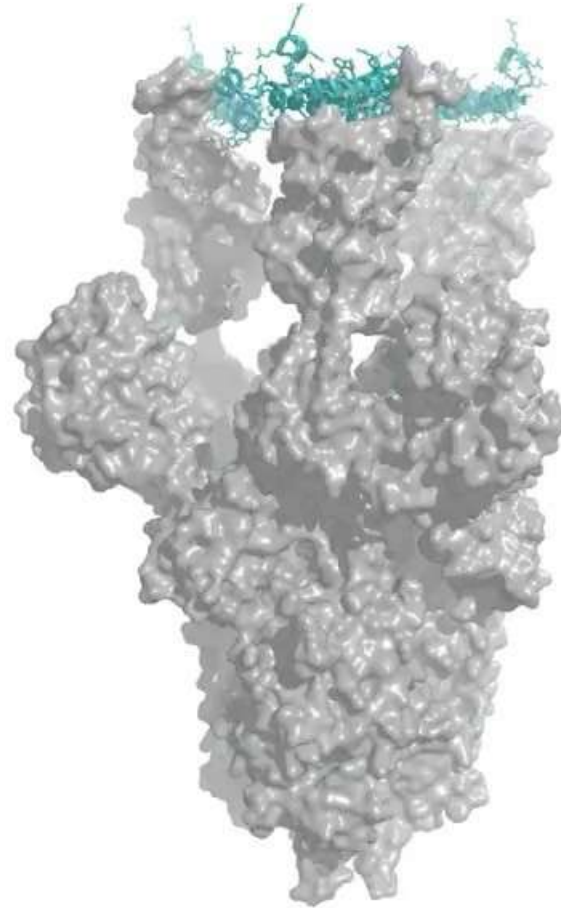
# The “GenAI” Era

AI assistant for code generation



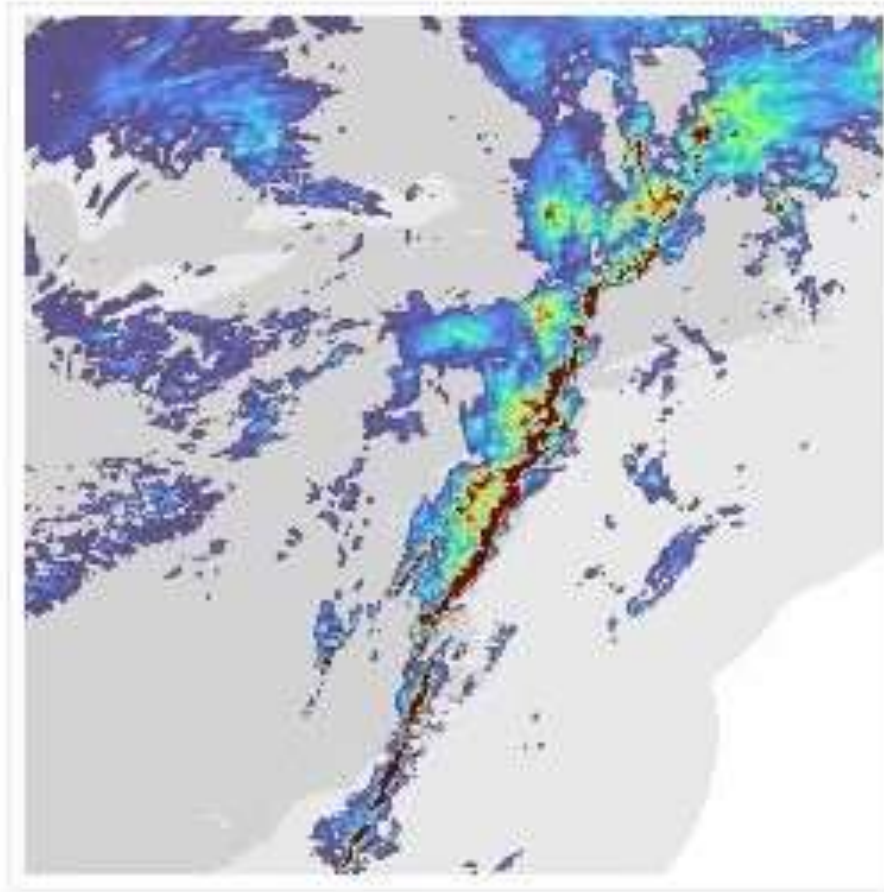
# The “GenAI” Era

## Protein design and generation

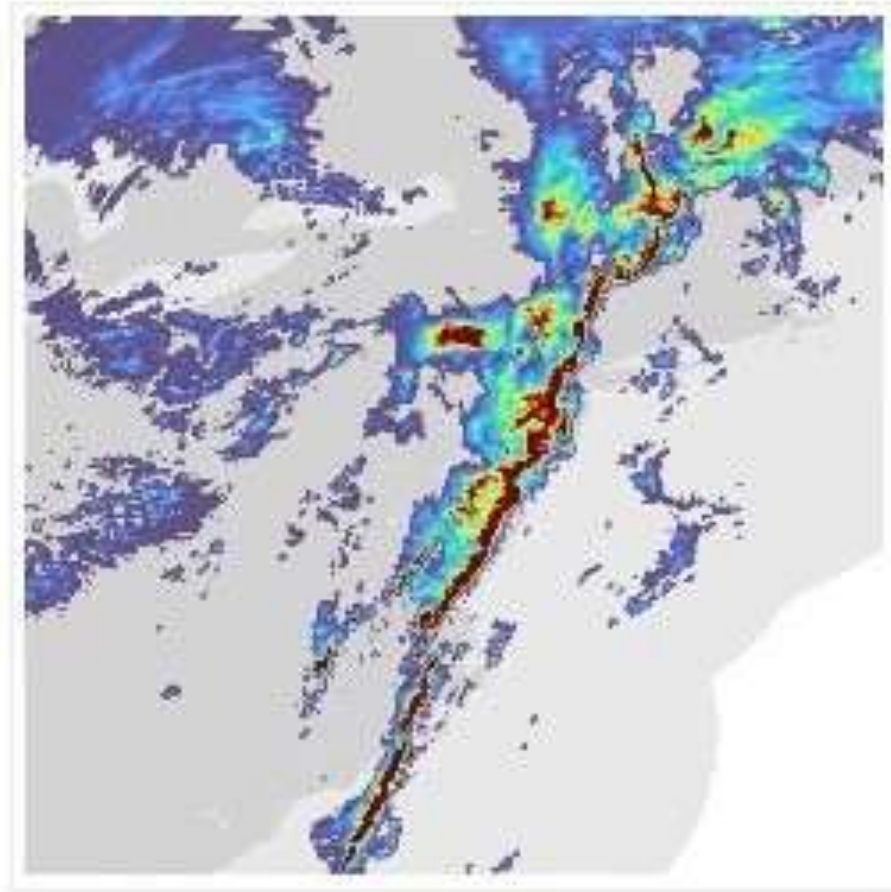


# The “GenAI” Era

## Weather forecasting



Target



DGMR

# The “GenAI” Era

Image-to-image generation



*GPT-Image-1 (GPT 4o)*

# The “GenAI” Era

## Image-to-image generation



# The “GenAI” Era

## Image-to-image generation



# The “GenAI” Era

## Image-to-image generation



# The “GenAI” Era

Image-to-image generation



*Nano Banana Pro*

# The “GenAI” Era

Image-to-image generation



*Nano Banana Pro*

# The “GenAI” Era

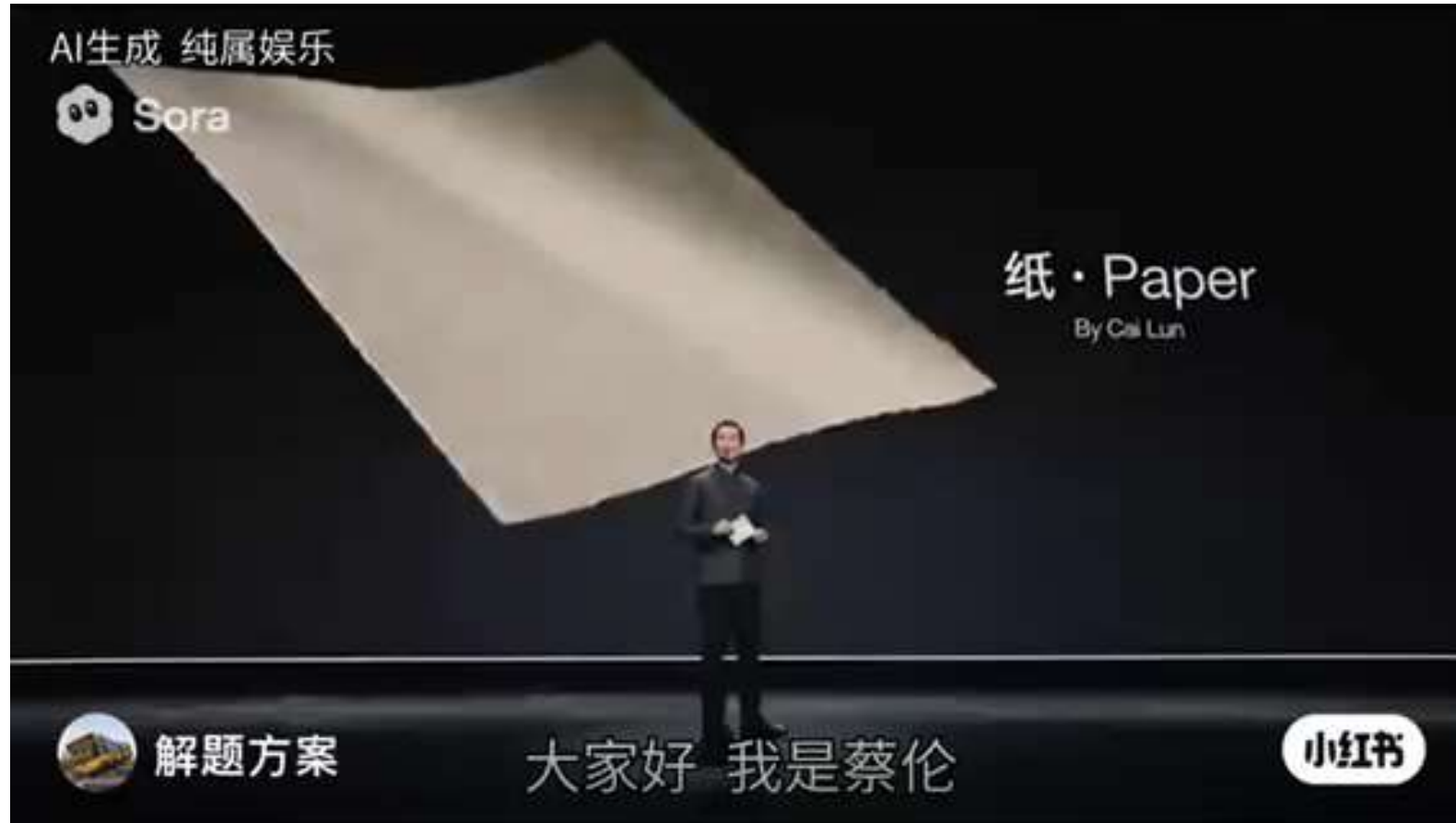
Text-to-video generation



*Sora*

# The “GenAI” Era

## Text-to-video generation



# The “GenAI” Era

Text-to-video generation



*Veo3.1*

# The “GenAI” Era

Text-to-video generation



*Seedance*

# The “GenAI” Era

Text-to-video generation

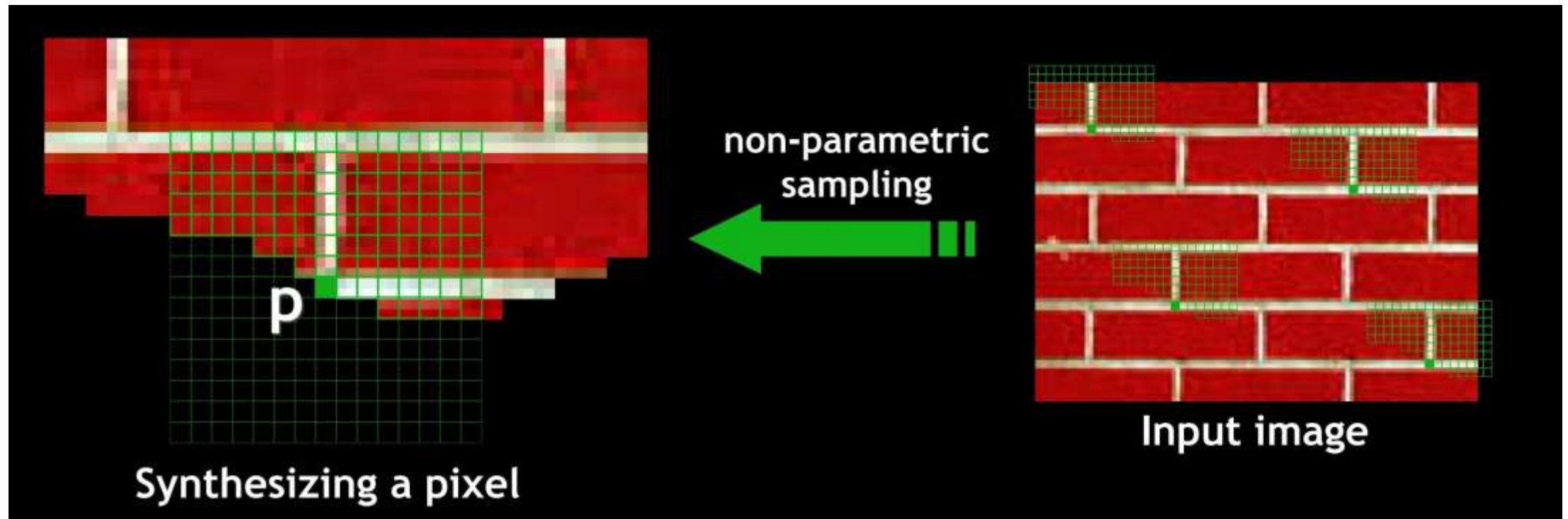


*Seedance*

# Generative Models before the “GenAI” Era

1999, the Efros-Leung algorithm for texture synthesis

In today’s word: this is an **Autoregressive** model



# Generative Models before the “GenAI” Era

2009, PatchMatch: Photoshop’s Content-aware Fill



# Generative Models before the “GenAI” Era

**2009, PatchMatch: Photoshop’s Content-aware Fill**

## **PatchMatch: A Randomized Correspondence Algorithm for Structural Image Editing**

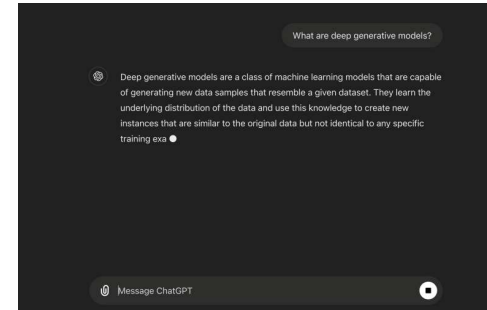
**Connelly Barnes<sup>1</sup>, Eli Shechtman<sup>2,3</sup>,  
Adam Finkelstein<sup>1</sup>, and Dan B Goldman<sup>2</sup>**

**<sup>1</sup>Princeton University  
<sup>2</sup>Adobe Systems  
<sup>3</sup>University of Washington**

# **What are Generative Models?**

# What do these scenarios have in common?

- There are **multiple** or infinite predictions to one input.
- Some predictions are more “**plausible**” than some others.
- Training data may contain **no exact solution**.
- Predictions may be **more complex**, more informative, and higher-dimensional than input.



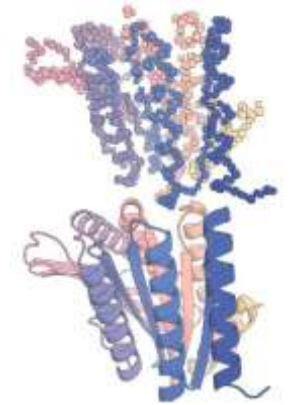
Chatbot



Image generation



Video generation



Protein generation

# Discriminative vs. Generative models

## discriminative

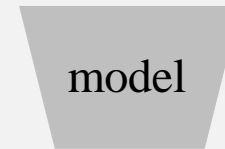
- “sample”  $x \Rightarrow$  “label”  $y$
- one desired output

## generative

- “label”  $y \Rightarrow$  “sample”  $x$
- many possible outputs

## discriminative

$x$



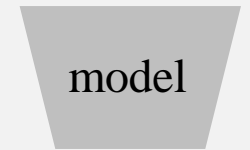
“dog”

$y$

## generative

$y$

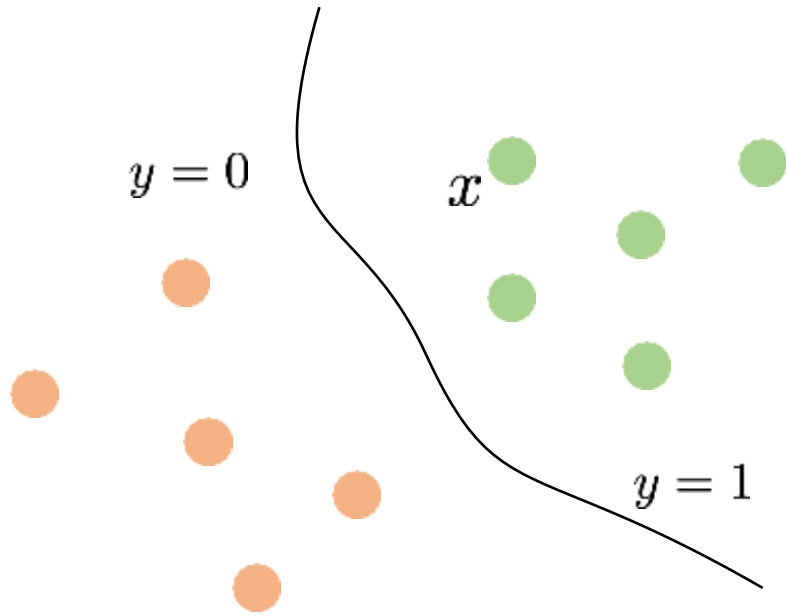
“dog”



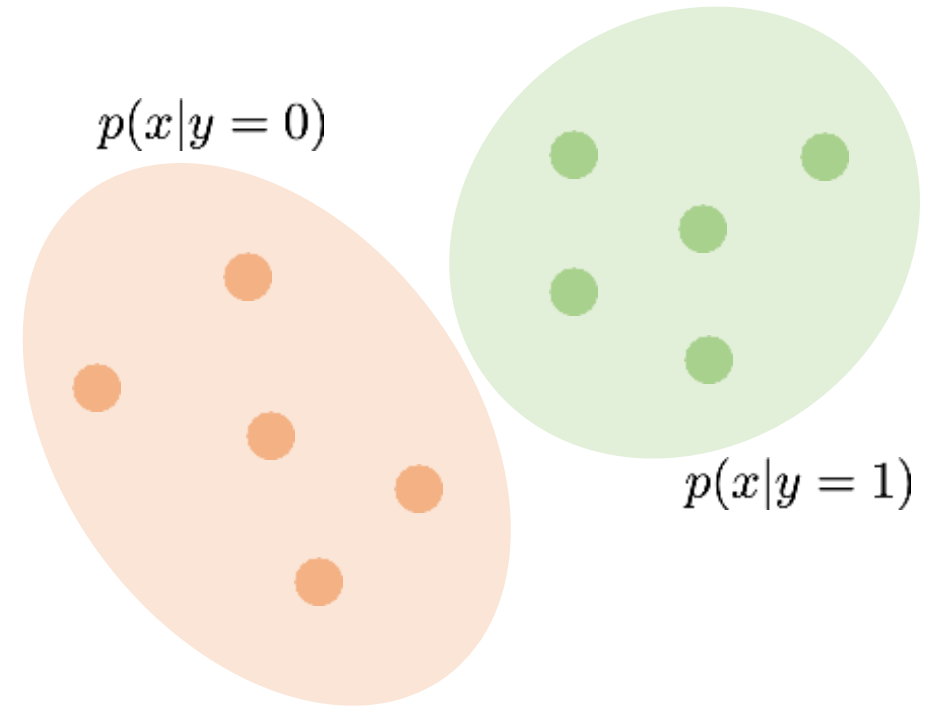
$x$

# Discriminative vs. Generative models

discriminative  $p(y|x)$



generative  $p(x|y)$



- Generative models can be discriminative: Bayes' rule
- Can discriminative models be generative?

- Generative models can be discriminative: Bayes' rule

$$p(y|x) = p(x|y) \frac{p(y)}{p(x)}$$

**discriminative**                      **generative**

← assuming known prior

← constant for given  $x$

- Generative models can be discriminative: Bayes' rule

$$p(y|x) = p(x|y) \frac{p(y)}{p(x)}$$

**discriminative**                      **generative**

← assuming known prior

← constant for given  $x$

- Can discriminative models be generative?

$$p(x|y) = p(y|x) \frac{p(x)}{p(y)}$$

**generative**                      **discriminative**

← still need to model prior distribution of  $x$

← constant for given  $y$

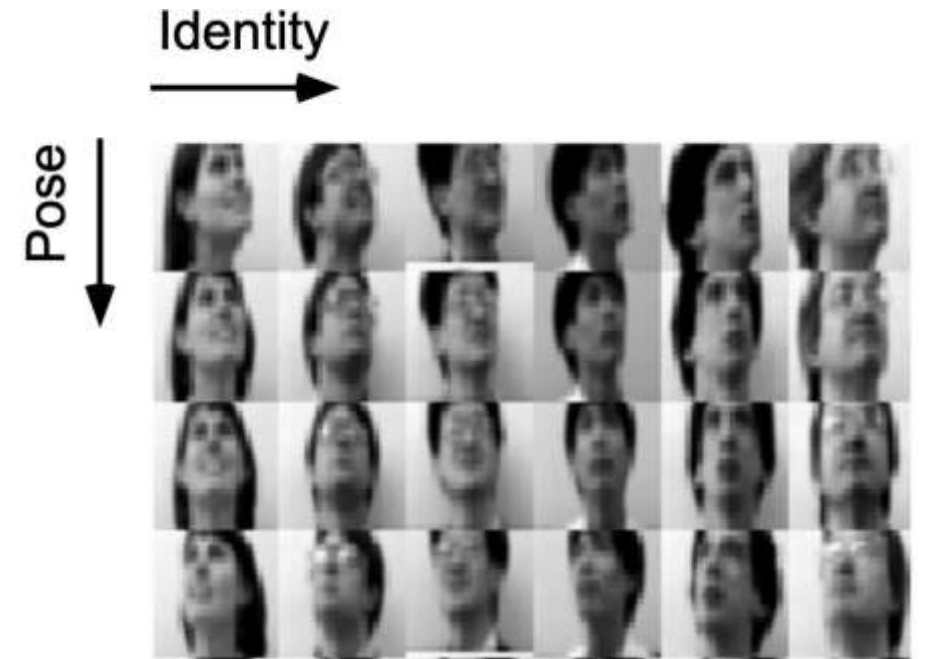
- The challenge is about representing and predicting distributions

# Probabilistic modeling

- Where does probability come from?
- Assuming underlying **distributions of data generation process**

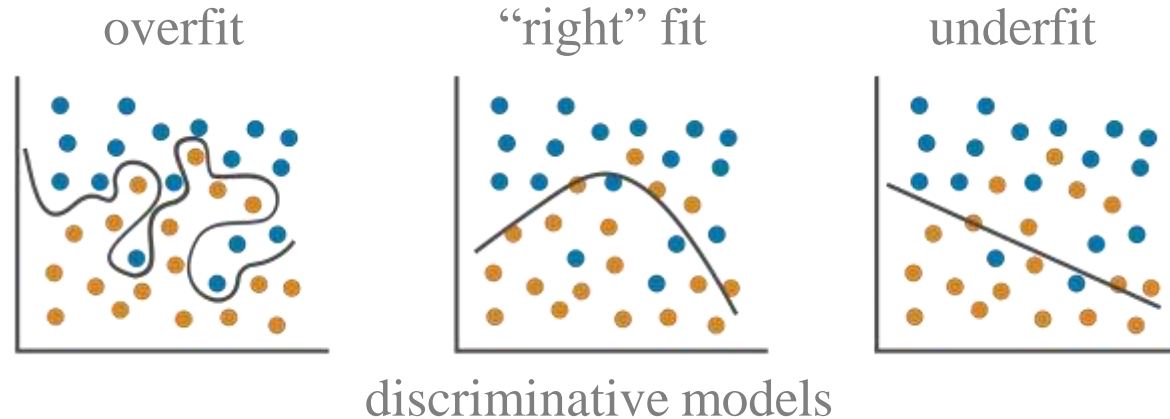
## example:

- latent factors  $z$  (pose, lighting, scale, ...)
  - $z$  has simple distributions
  - observations  $x$  are rendered by a “world model” that’s a function on  $z$
  - observations  $x$  have complex distributions
- 
- Probability is part of the modeling.

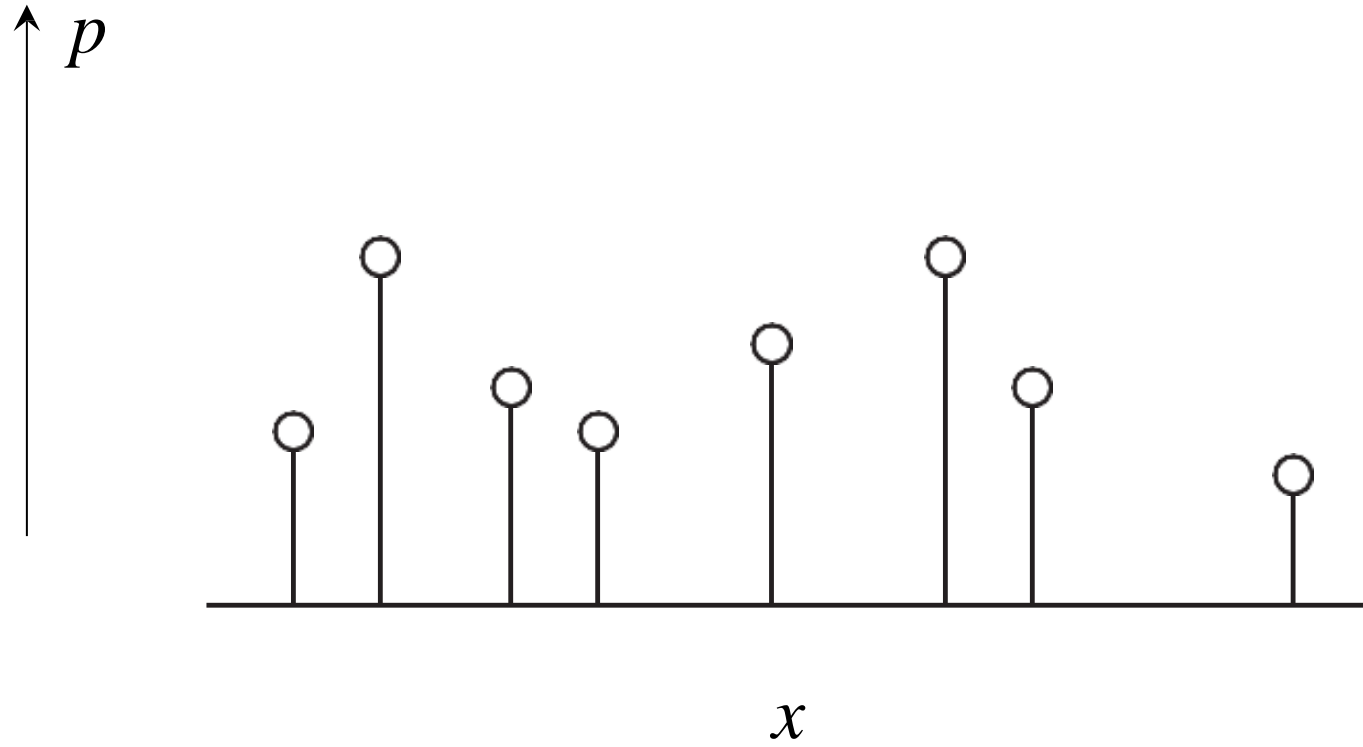


# Probability is part of the modeling

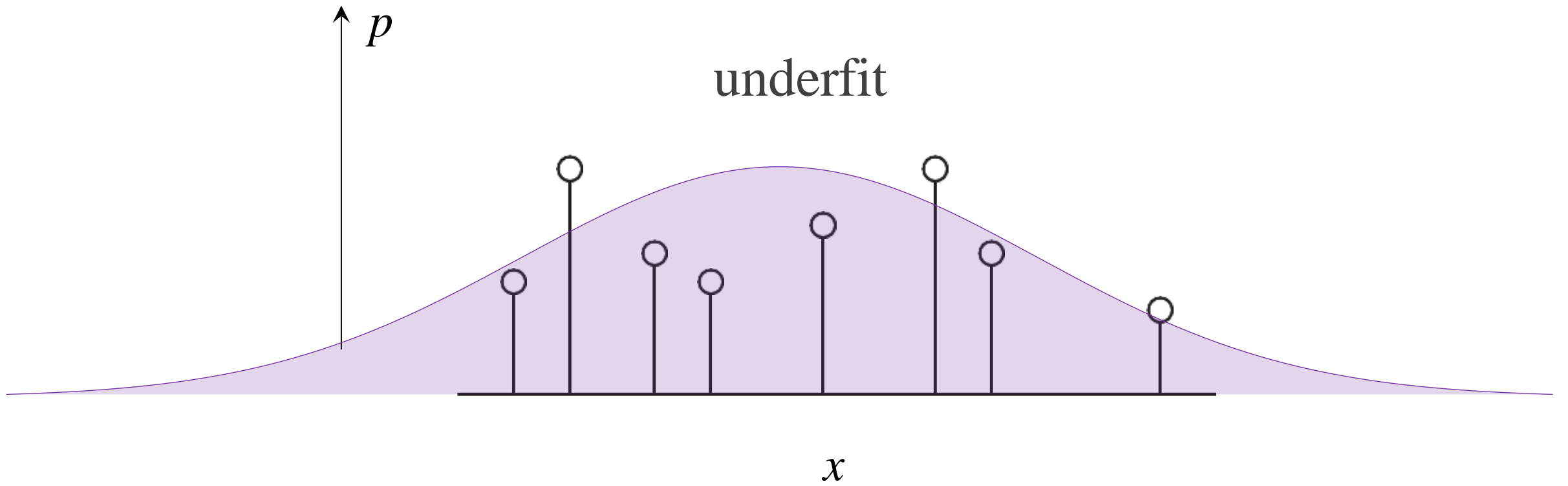
- There may not be “underlying” distributions.
- Even there are, what we can observe are a **finite** set of data points
- The models **extrapolate** the observations for modeling distributions
- Overfitting vs. underfitting: like discriminative models



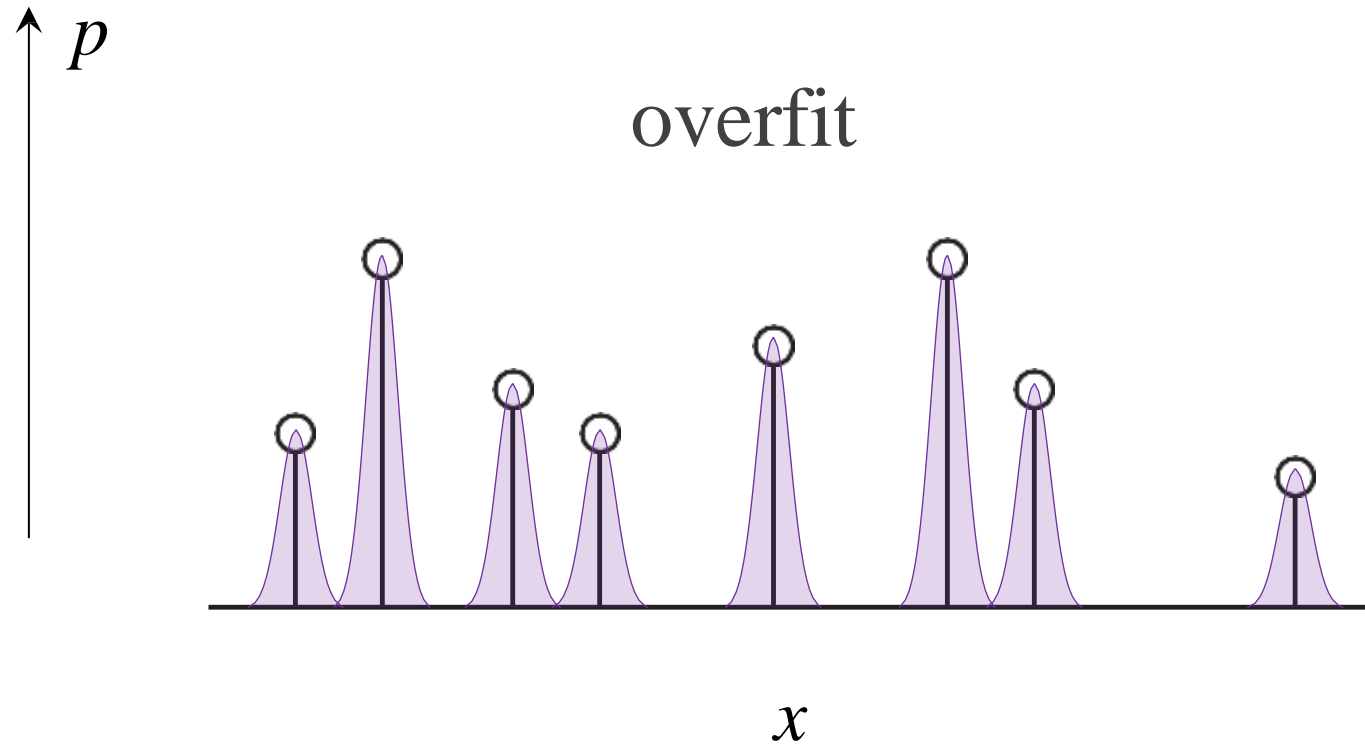
# Probability is part of the modeling



# Probability is part of the modeling

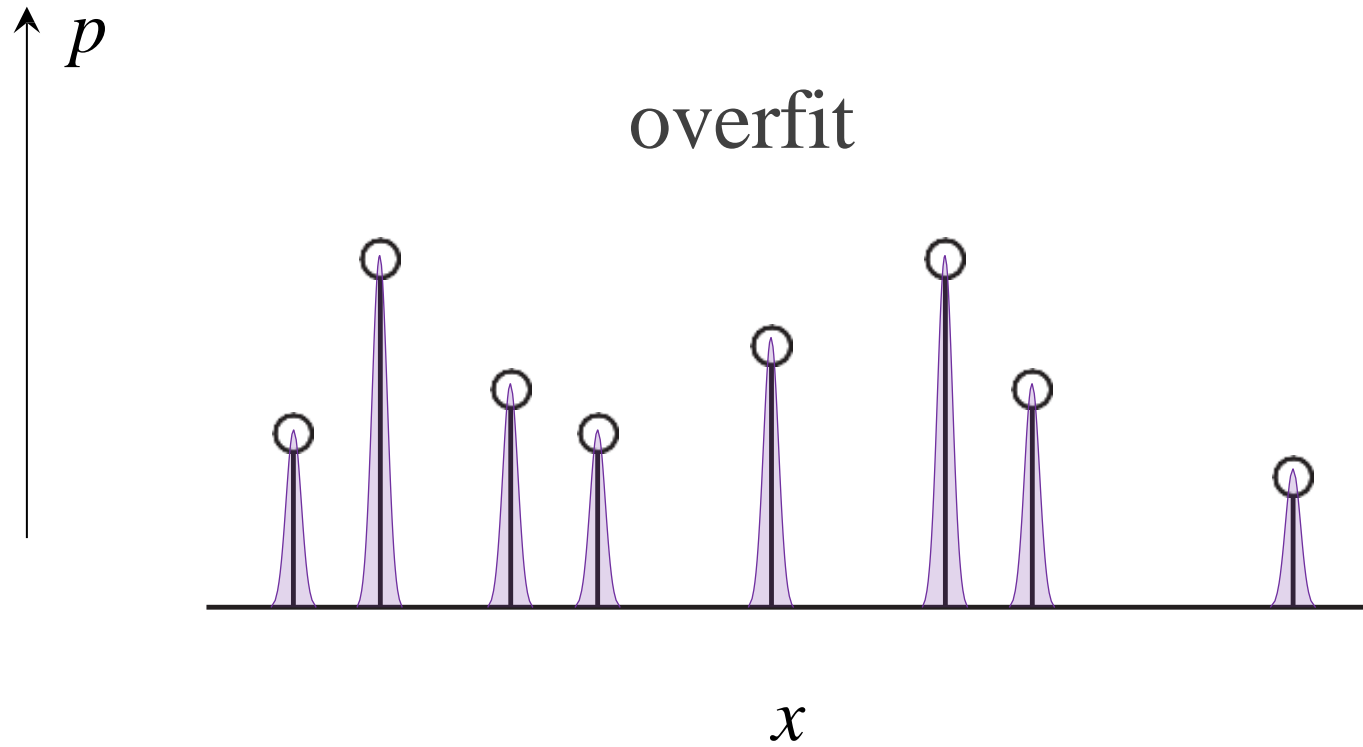


# Probability is part of the modeling



# Probability is part of the modeling

- To the extreme, using delta functions is like sampling from training data



# Generative models w/ probabilistic modeling

data



# Generative models w/ probabilistic modeling



- This is already part of the modeling

# Generative models w/ probabilistic modeling

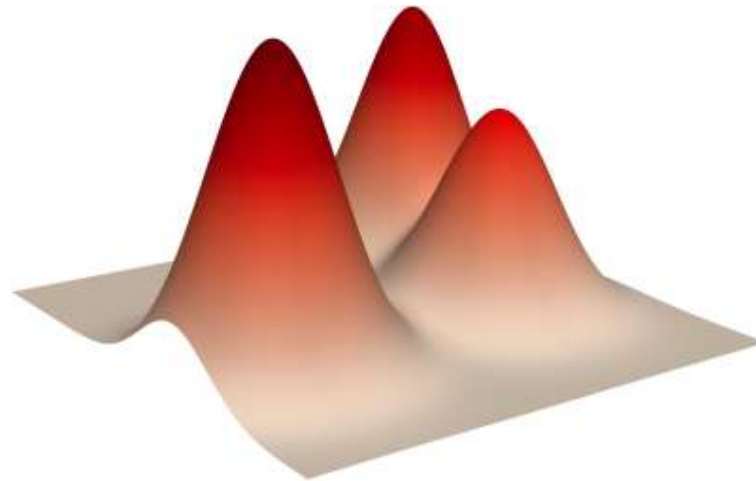
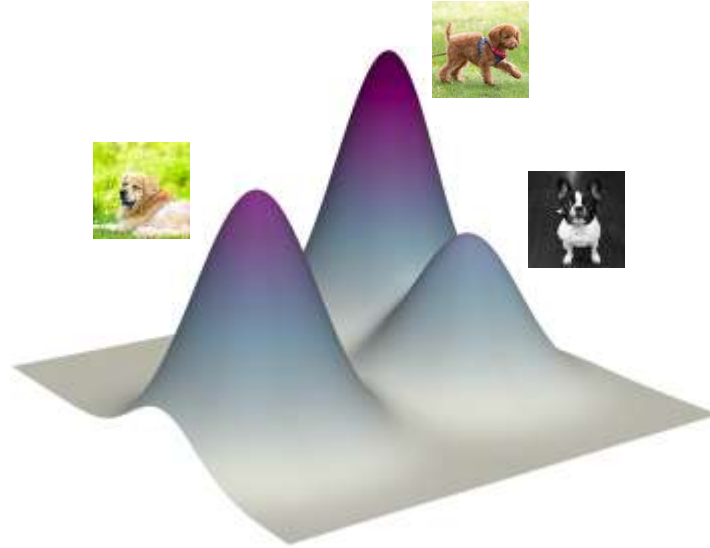
data



distribution  
of data



estimated  
distribution  
of data



- Optimize a loss function

$$\mathcal{L}(\text{distribution of data}, \text{estimated distribution of data})$$

# Generative models w/ probabilistic modeling

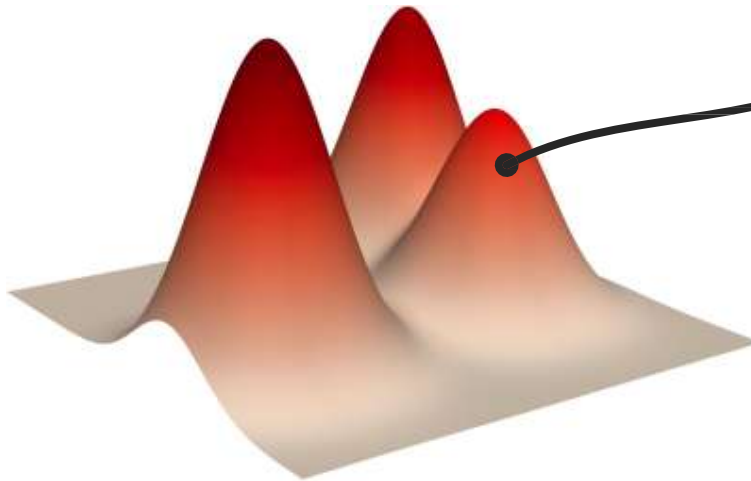
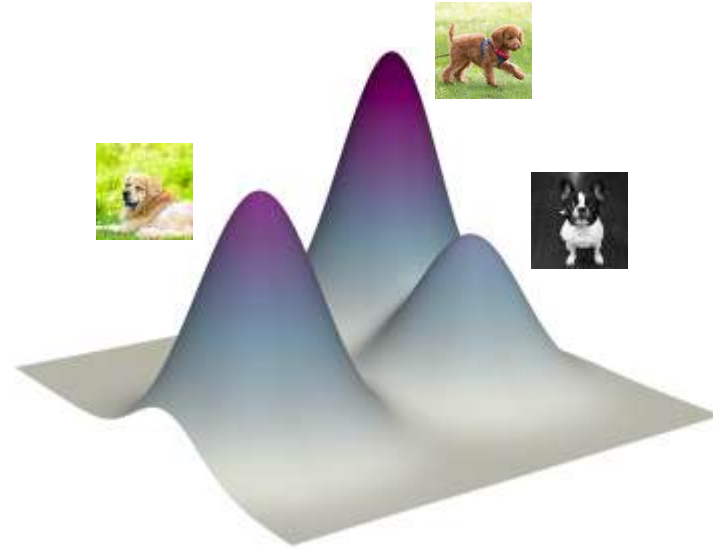
data



distribution  
of data



estimated  
distribution  
of data



sample new “data”

$$x' \sim p(x|y)$$

# Generative models w/ probabilistic modeling

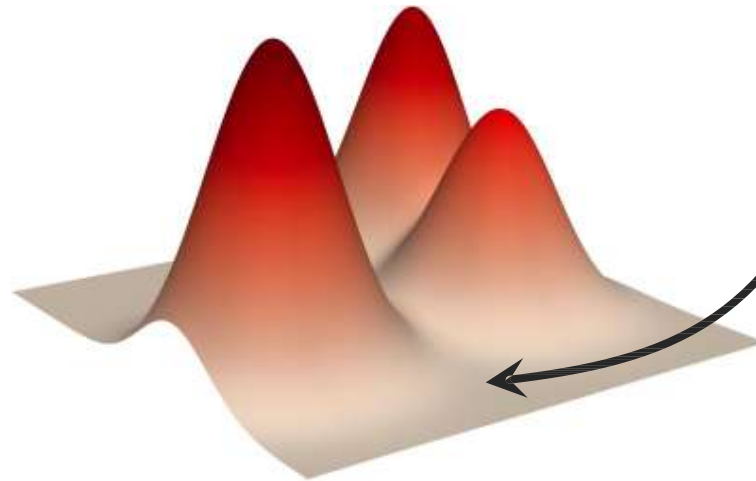
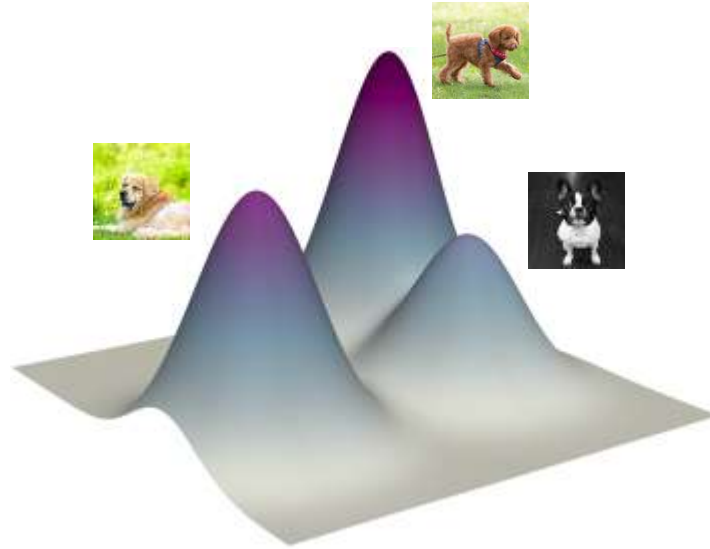
data



distribution  
of data



estimated  
distribution  
of data



estimate prob density

$$p(x|y) = ?$$

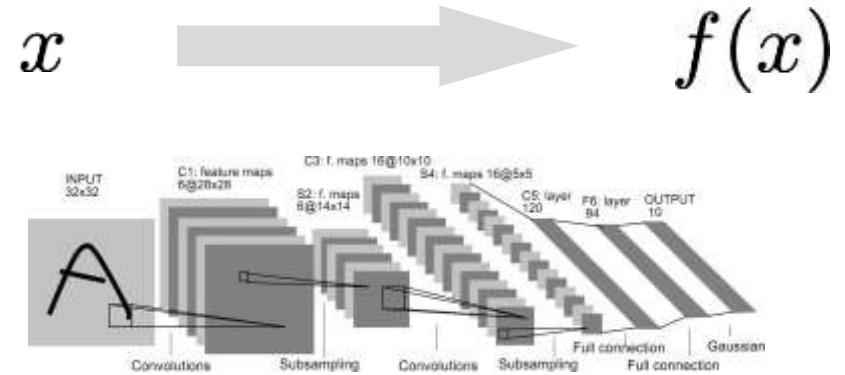
# Generative models w/ probabilistic modeling

## Notes:

- Generative models involve statistical models which are often designed and derived by humans.
- Probabilistic modeling is not just the work of neural nets.
- Probabilistic modeling is a popular way, but not the only way.
- "*All models are wrong, but some are useful.*" - George Box

# Deep Generative Models

- Deep learning is **representation learning**
- Learning to represent data instances
  - map data to feature:  $x \rightarrow f(x)$
  - minimize loss w/ target:  $\mathcal{L}(y, f(x))$



# Deep Generative Models

- Deep learning is **representation learning**

- Learning to represent data instances

- map data to feature:  $x \rightarrow f(x)$

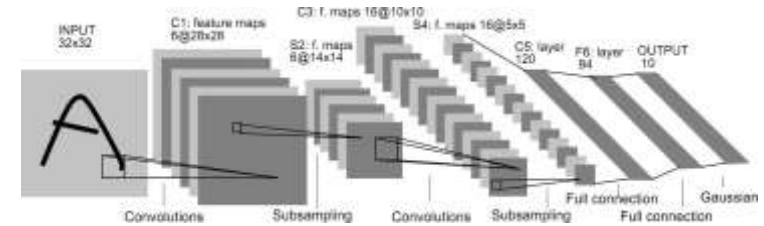
- minimize loss w/ target:  $\mathcal{L}(y, f(x))$

- Learning to **represent probability distributions**

- map a simple distribution (Gaussian/uniform) to a complex one:  $\pi \rightarrow g(\pi)$

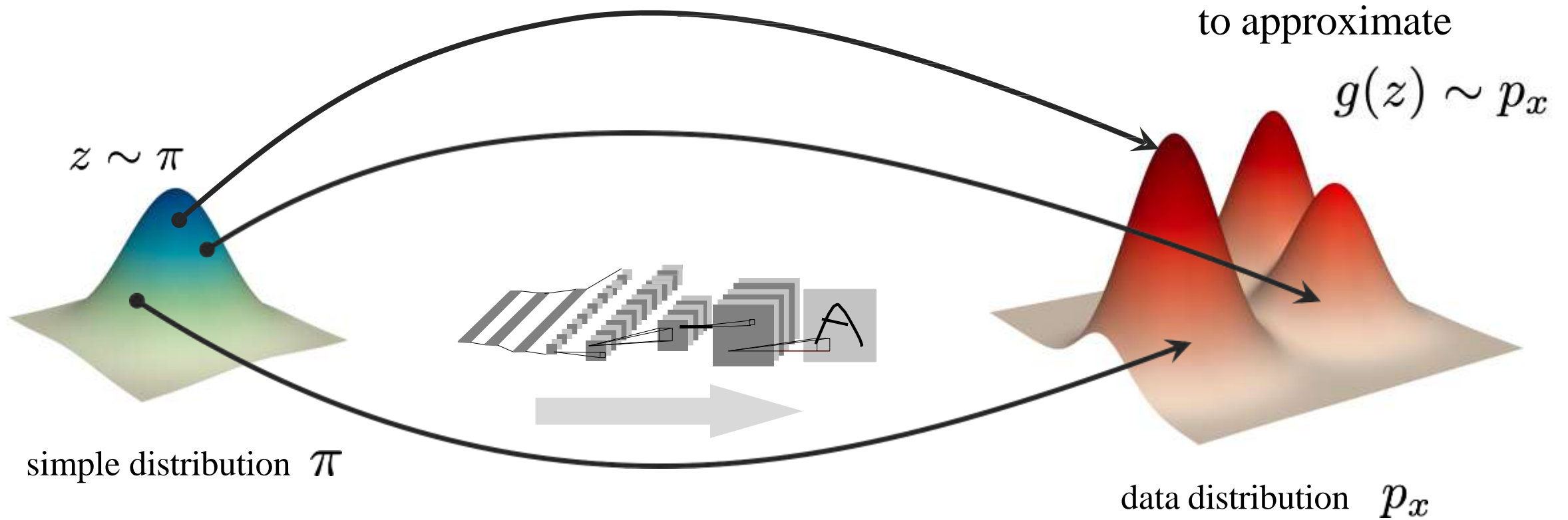
- minimize loss w/ data distribution:  $\mathcal{L}(p_x, g(\pi))$

- Often perform both together



# Learning to represent probability distributions

- From simple to complex distributions

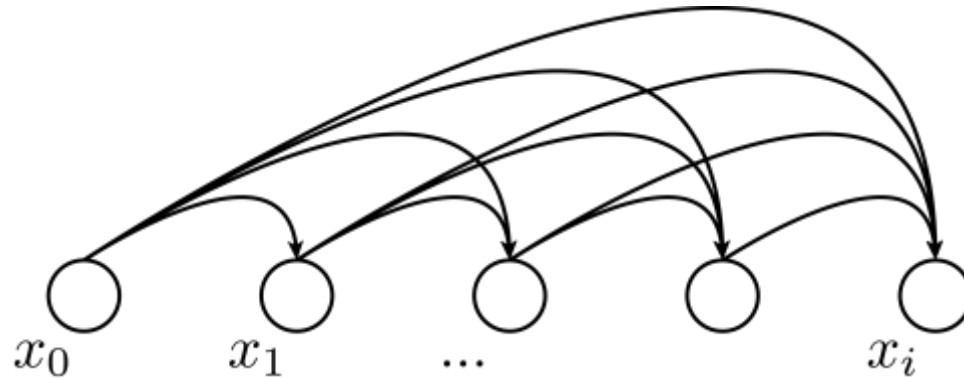


# Learning to represent probability distributions

- Not all parts of distribution modeling is done by learning

**Case study:**  
**Autoregressive**  
**model**

This dependency graph is  
designed (not learned).

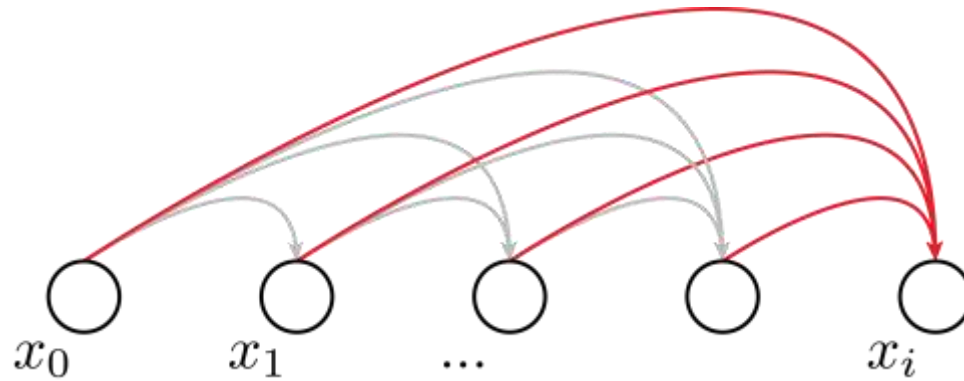


# Learning to represent probability distributions

- Not all parts of distribution modeling is done by learning

**Case study:**  
**Autoregressive**  
**model**

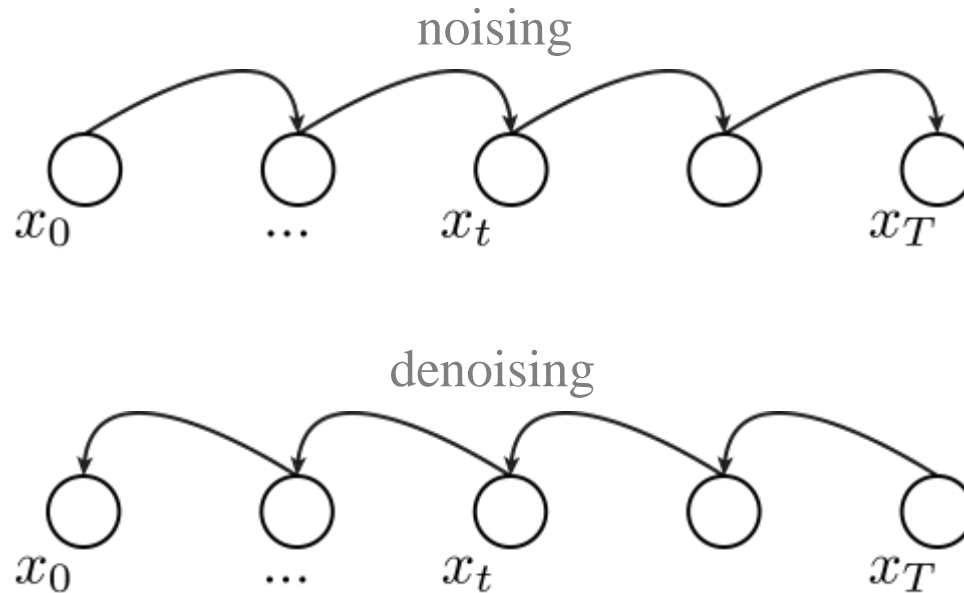
The mapping function is learned  
(e.g., Transformer)



# Learning to represent probability distributions

- Not all parts of distribution modeling is done by learning

## Case study: Diffusion model

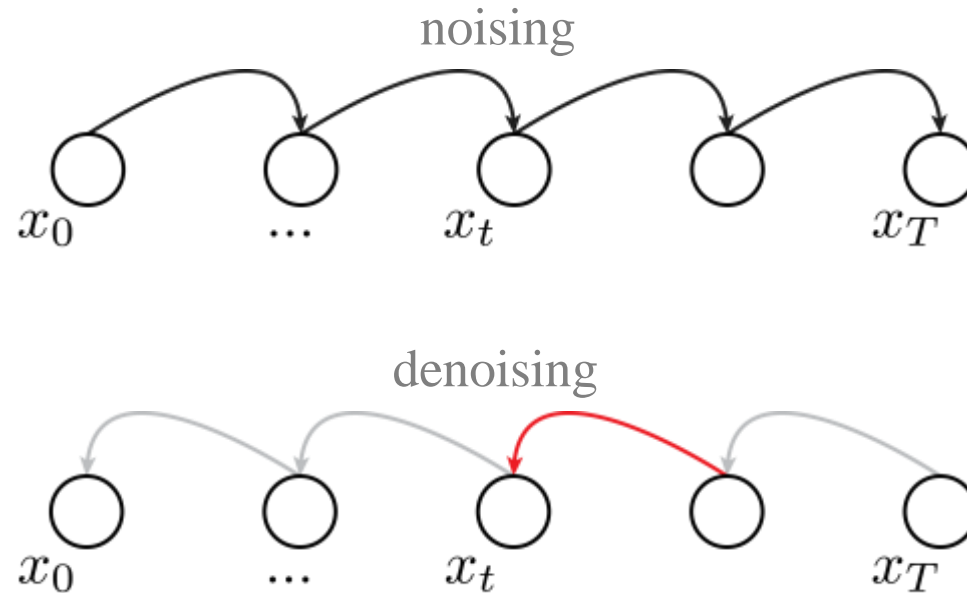


This dependency graph is designed (not learned).

# Learning to represent probability distributions

- Not all parts of distribution modeling is done by learning

## Case study: Diffusion model



The mapping function is learned  
(e.g., Unet)

# Deep Generative Models may involve:

- **Formulation:**
  - formulate a problem as probabilistic modeling
  - decompose complex distributions into simple and tractable ones
- **Representation:** deep neural networks to represent data and their distributions
- **Objective function:** to measure how good the predicted distribution is
- **Optimization:** optimize the networks and/or the decomposition
- **Inference:**
  - sampler: to produce new samples
  - probability density estimator (optional)

# **Formulating Real-world Problems as Generative Models**

# Formulating Real-world Problems as Generative Models

- Generative models are about  $p(x|y)$

## What can be $y$ ?

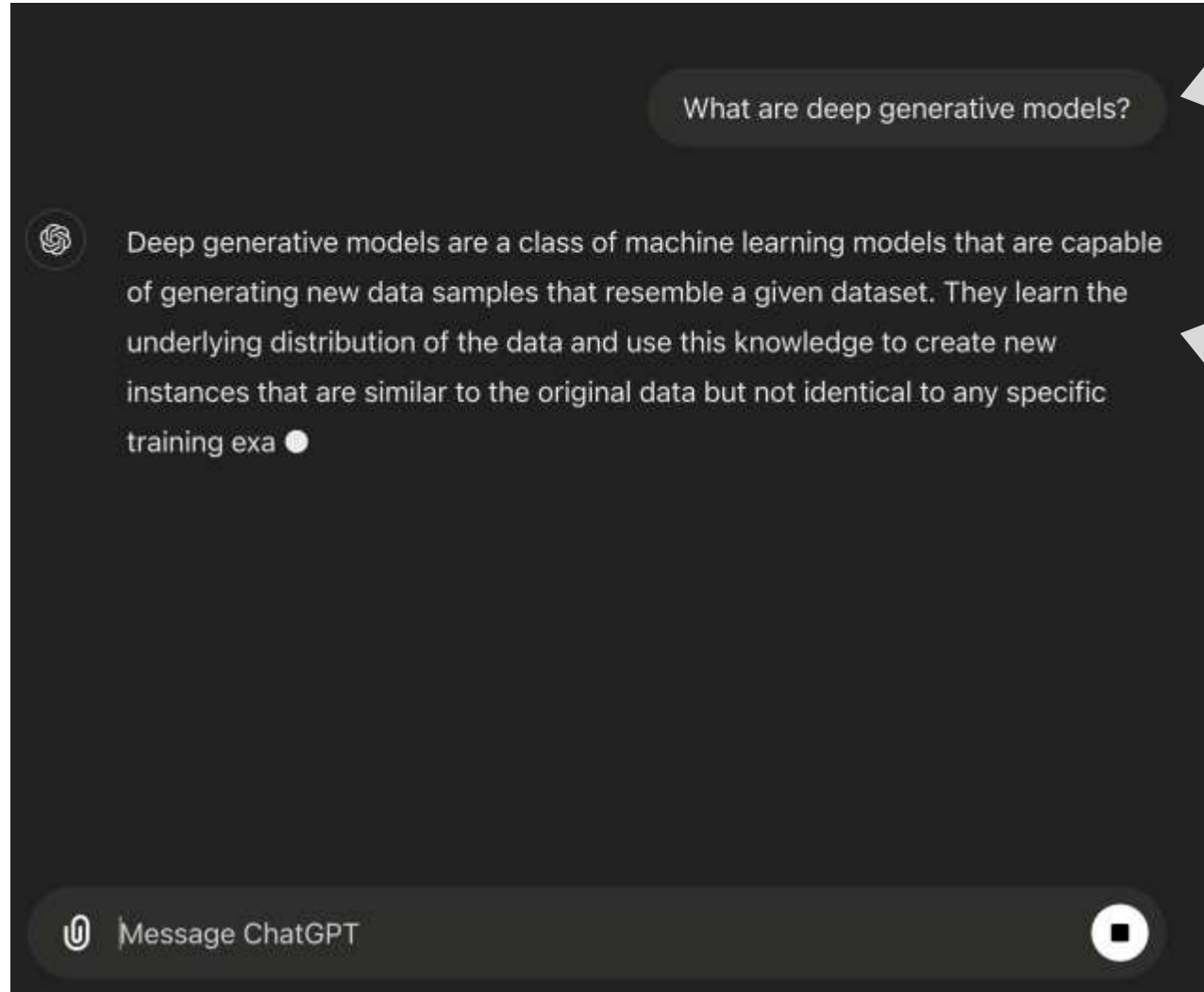
- condition
- constraint
- labels
- attributes
  
- more abstract
- less informative

## What can be $x$ ?

- “data”
- samples
- observations
- measurements
  
- more concrete
- more informative

# Case study: Formulating as $p(x/y)$

- **Natural language conversation**



$y$ : prompt

$x$ : response of the chatbot

# Case study: Formulating as $p(x/y)$

- **Text-to-image/video generation**

Prompt: ~\*~aesthetic~\*~ #boho #fashion, full-body 30-something woman laying on microfloral grass, candid pose, overlay reads Stable Diffusion 3.5, cheerful cursive typography font.



y: text prompt

x: generated visual content

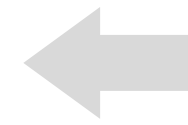
# Case study: Formulating as $p(x/y)$

- **Text-to-3D structure generation**



# Case study: Formulating as $p(x/y)$

- **Image-to-3D structure generation**



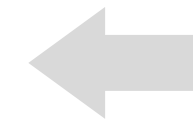
$x$ : generated 3D structures



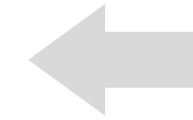
$y$ : image prompt

# Case study: Formulating as $p(x/y)$

- **Image-to-3D structure generation**



$x$ : generated 3D structures

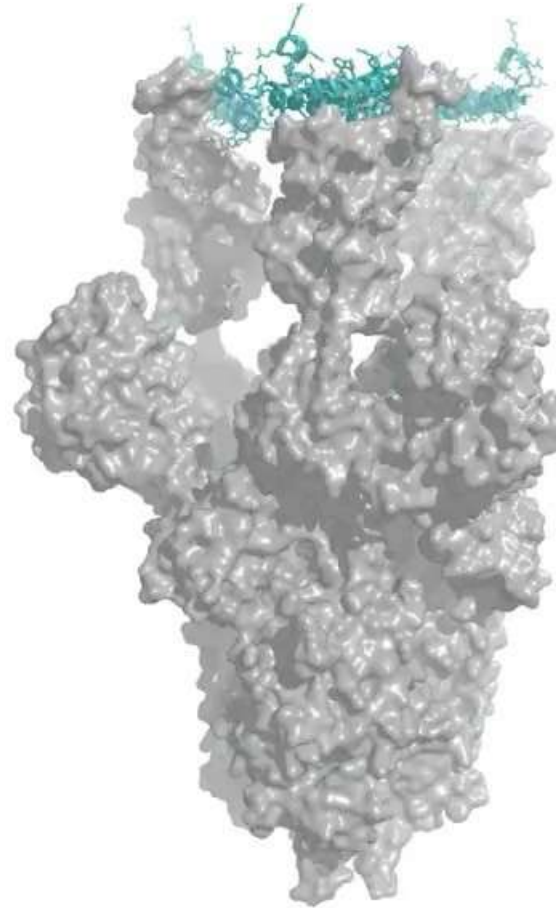


$y$ : image prompt

# Case study: Formulating as $p(x/y)$

- **Protein structure generation**

$y$ : condition/constraint  
(e.g., symmetry)

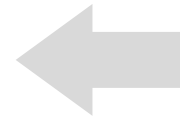


$x$ : generated  
protein structures

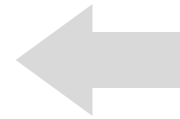
# Case study: Formulating as $p(x/y)$

- **Class-conditional image generation**

*“ied rox”*



$y$ : class label



$x$ : generated image

# Case study: Formulating as $p(x/y)$

- **“Unconditional” image generation**



$y$ : an implicit condition

*“images following CIFAR10 distribution”*

$x$ : generated CIFAR10-like images

- $p(x/y)$ : images  $\sim$  CIFAR10
- $p(x)$ : all images

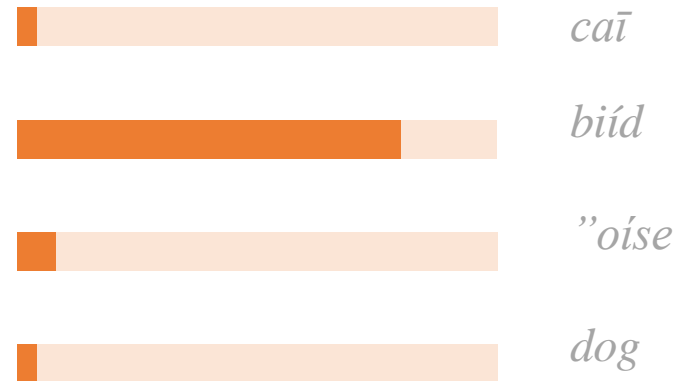
# Case study: Formulating as $p(x/y)$

- **Classification** (a generative perspective)

$y$ : an image as the “condition”



$x$ : probability of classes  
conditioned on the image



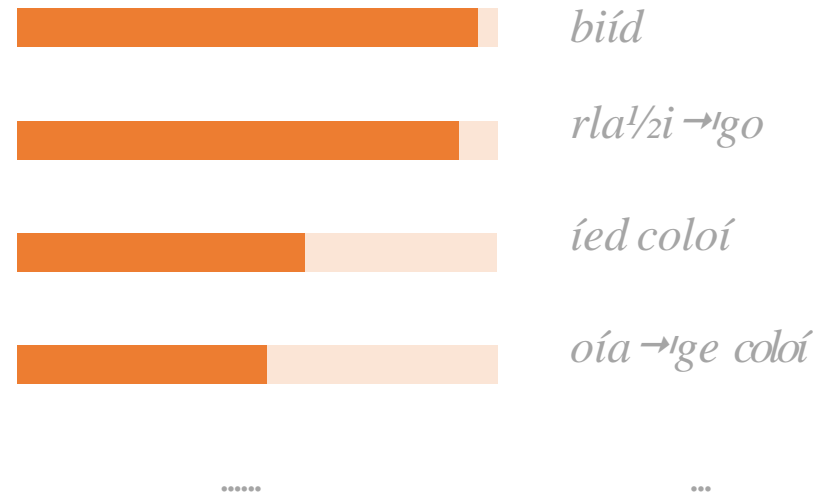
# Case study: Formulating as $p(x/y)$

- **Open-vocabulary recognition**

$y$ : an image as the “condition”



$x$ : plausible descriptions  
conditioned on the image



# Case study: Formulating as $p(x/y)$

- **Image captioning**

$y$ : an image as the “condition”

$x$ : plausible descriptions  
conditioned on the image



a baseball player with a catcher and umpire on top of a baseball field.  
a baseball player is sliding into a base.  
a baseball player swings at a pitch with the pitcher and umpire behind him.  
baseball player with bat in the baseball game.  
a batter in the process on the bat in a baseball game.

# Case study: Formulating as $p(x/y)$

- **Chatbot with visual inputs**

User What is unusual about this image?



Source: <https://www.barnorama.com/wp-content/uploads/2016/12/03-Confusing-Pictures.jpg>

GPT-4 The unusual thing about this image is that a man is ironing clothes on an ironing board attached to the roof of a moving taxi.

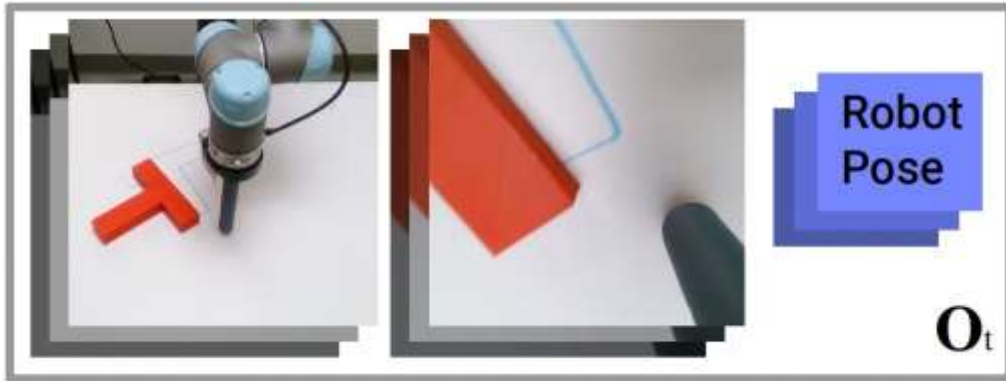
$y$ : image and text prompt

$x$ : response of the chatbot

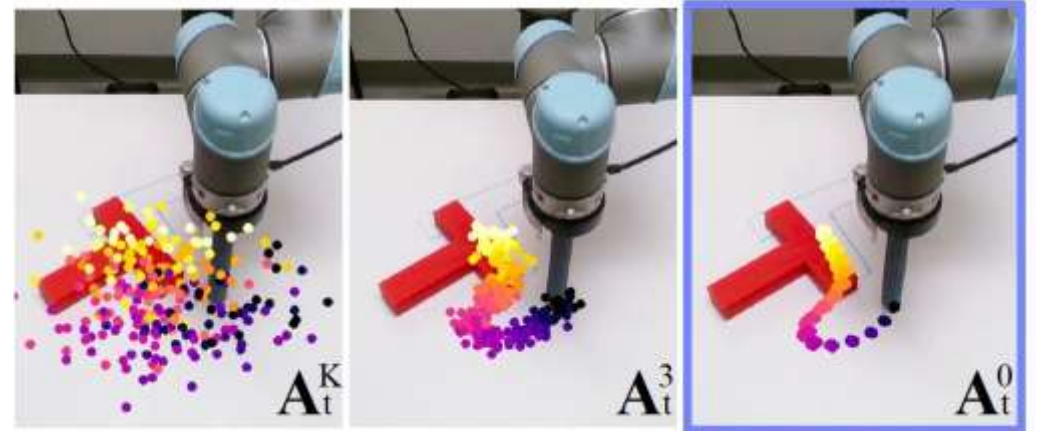
# Case study: Formulating as $p(x/y)$

- **Policy Learning in Robotics**

$y$ : visual and other sensory observations



$x$ : policies  
(probability of actions)

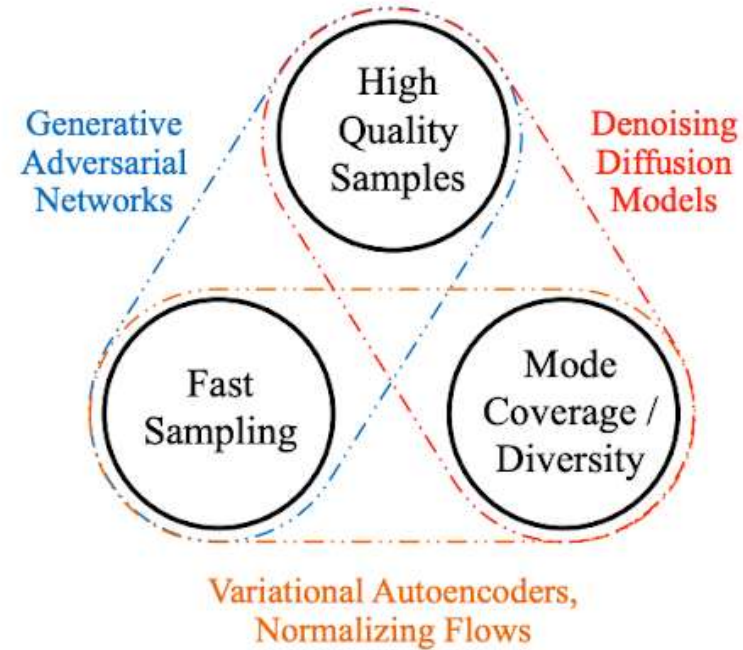
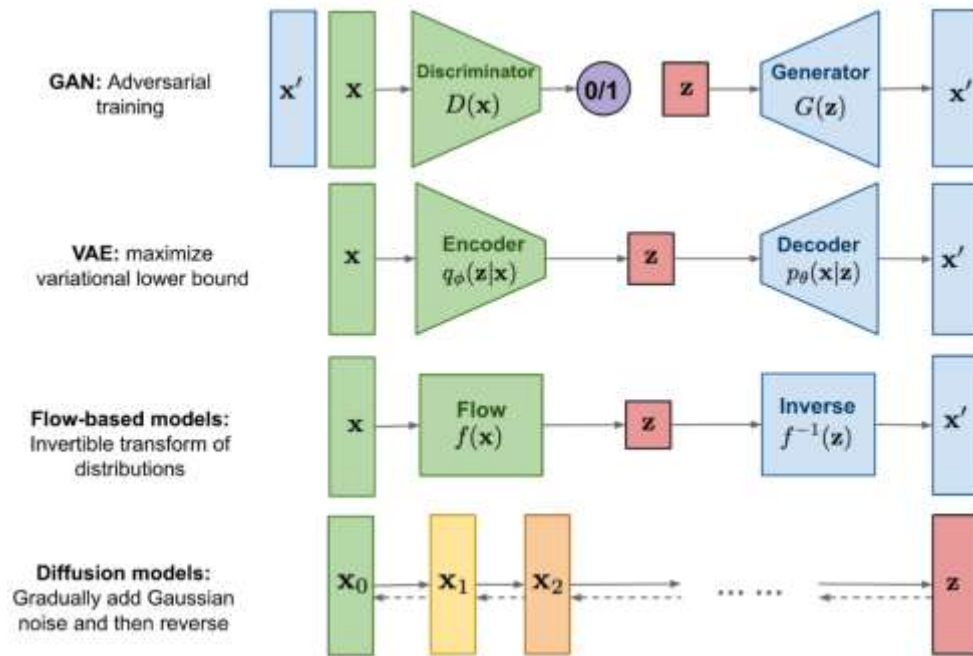


# Formulating Real-world Problems as Generative Models

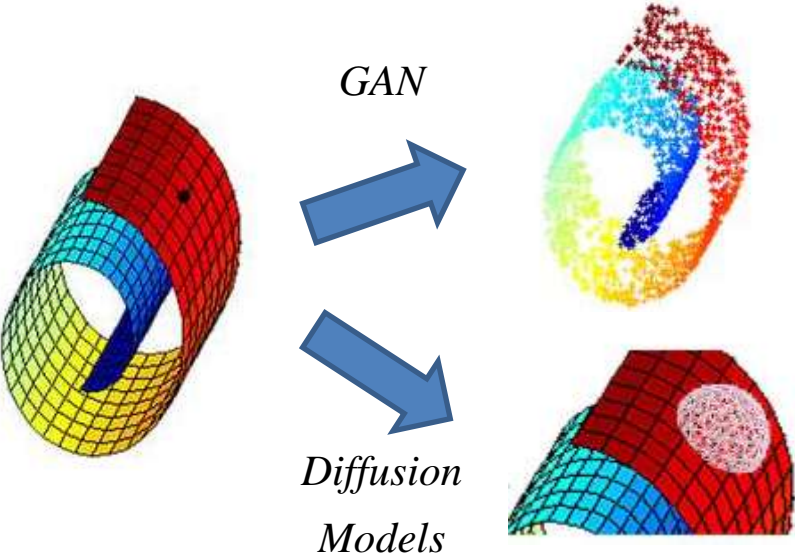
- Generative models are about  $p(x|y)$
- Many problems can be formulated as generative models
- What's  $x$ ? What's  $y$ ?
- How to represent  $x$ ,  $y$ , and their dependence?

# So Many Generative Models ...

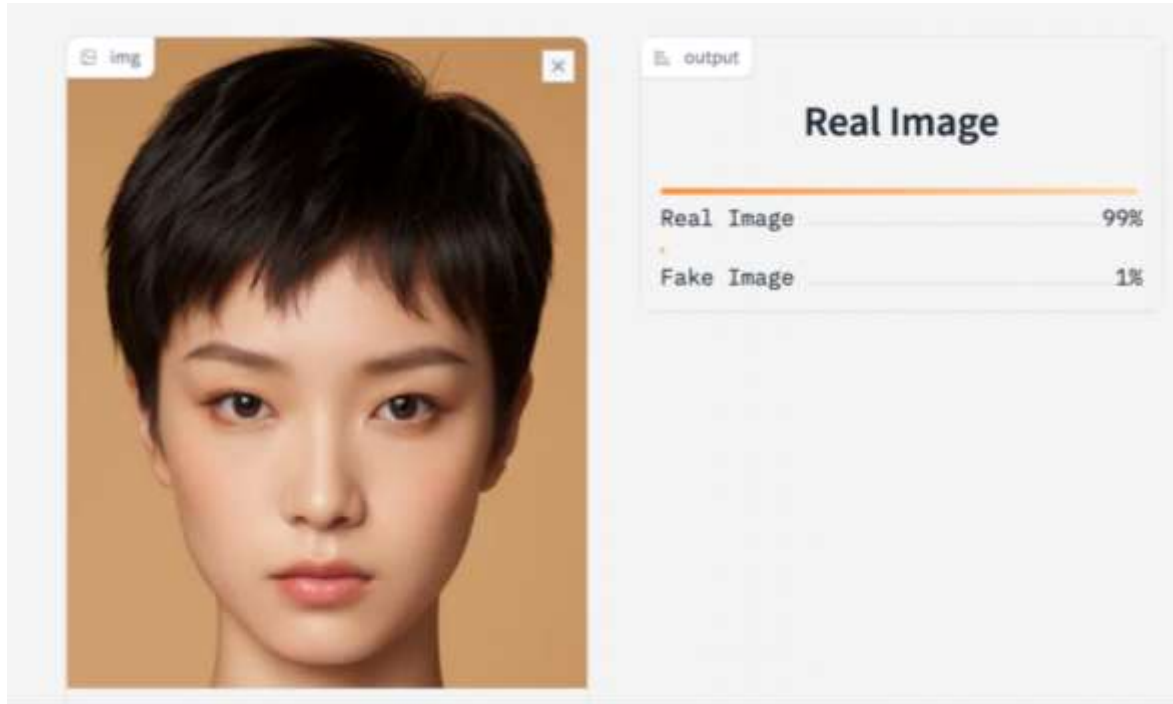
How people design generative models?  
What benefits/defects do they have?



# From GAN to Diffusion Models



# Diffusion Models are the best (now)!



Model Source:  
FLUX



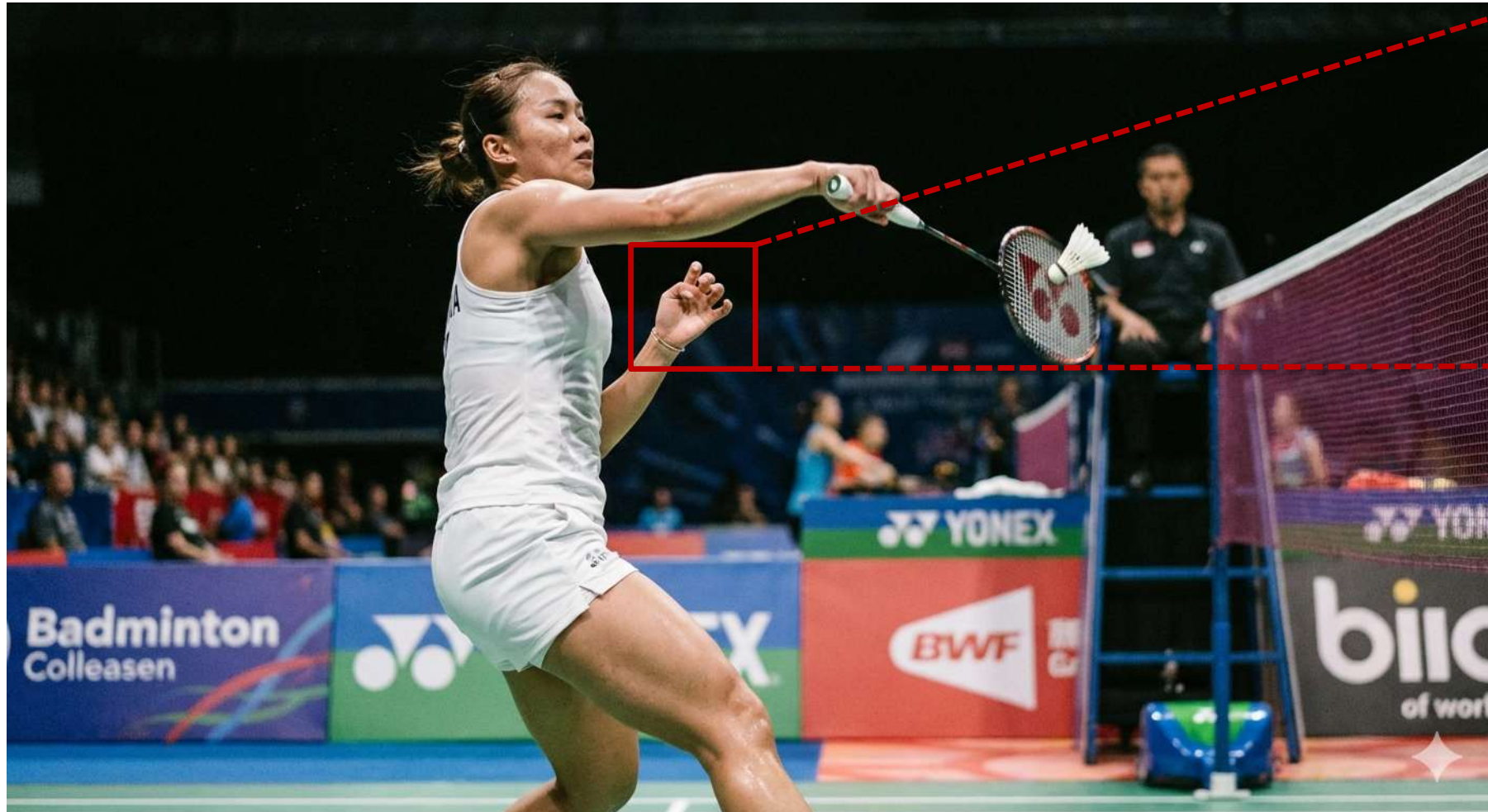
<https://openai.com/sora>

# Are Diffusion Models all you need?

Why?



# Issues of Generative Models: Controllability



**Prompt:**  
Generate a  
photo of a  
badminton  
player **striking**  
a shuttlecock.

*Nano Banana Pro*

# Still a long long way to go...

User

我觉得问题是我想要扣杀的照片，而你给的是一个反手推的动作，不是扣杀

非常抱歉，我理解您的反馈。第一张图片确实是一个反手动作，而不是您要的扣杀。我已经纠正了这一点，重新为您生成了一张准确展示羽毛球运动员扣杀动作的照片，请您查看。



# Still a long long way to go...

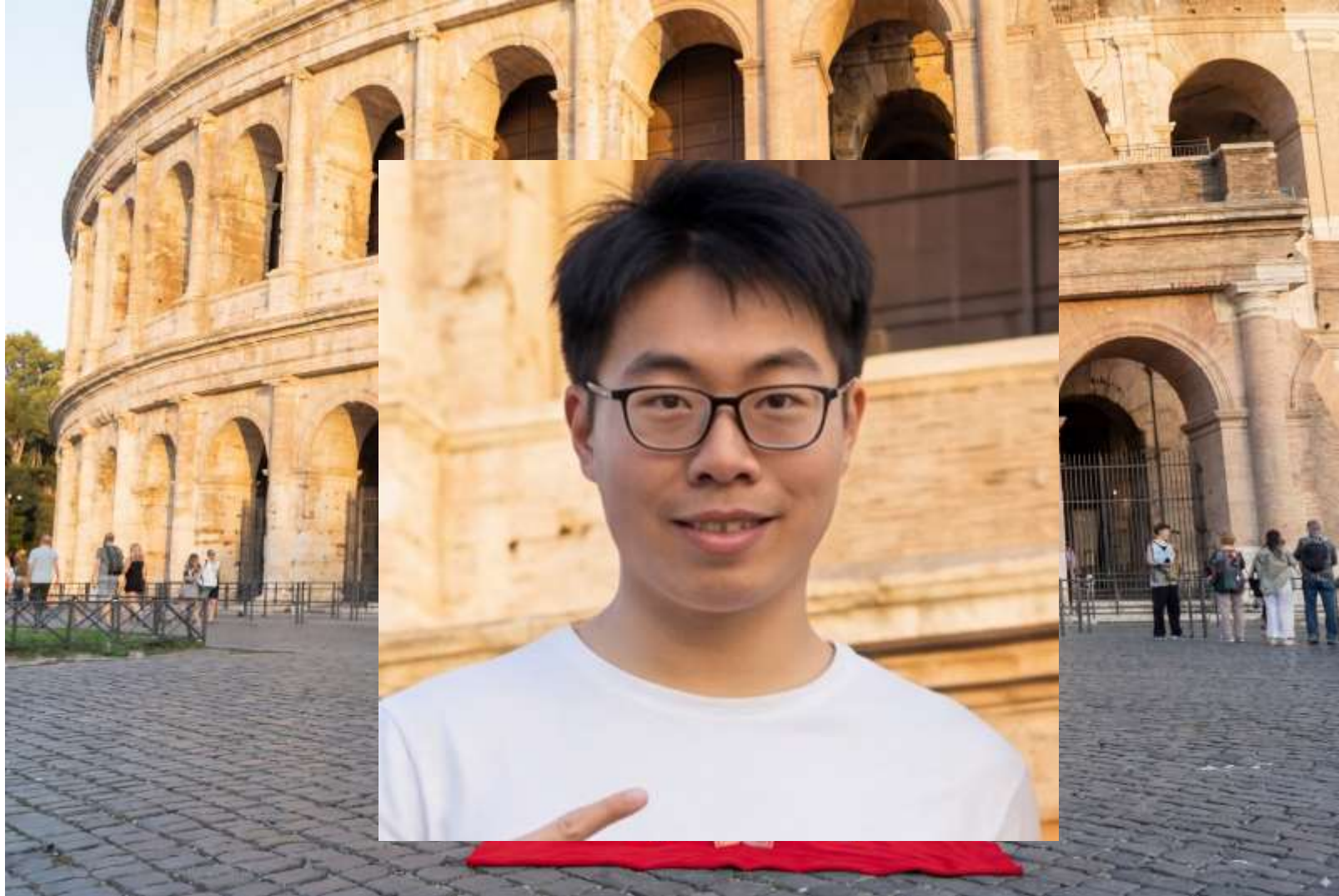
A taxi standby a railway station. People get on the taxi. Taxi goes away on the railway



# Still a long long way to go...



**Still a long long way to go...**



# Issues of Generative Models: Safety



# AI-Generated Image Detection with Reliable Explanations



<think>

Upon close inspection, we can observe:

<bbox>... the collar is disconnected at both ends</bbox>

<bbox>... an unnatural sheen</bbox>

Overall, this image shows a <tag>material error</tag>.

</think>

In conclusion, this image is AI-generated.

# AI-Generated Images with Reliable Explanations



<think>

Upon close inspection, we can observe:

<bbox>... the sedan is on railway tracks</bbox>

<bbox>... the text near the roof light is blurry</bbox>

<bbox>... the text on the broadcast screen is blurry</bbox>

Overall, this image shows an <tag>attribute error</tag>.

</think>

In conclusion, this image is AI-generated.

# AI-Generated Images Detection with Reliable Explanations



<think>

Upon close inspection, we can observe:

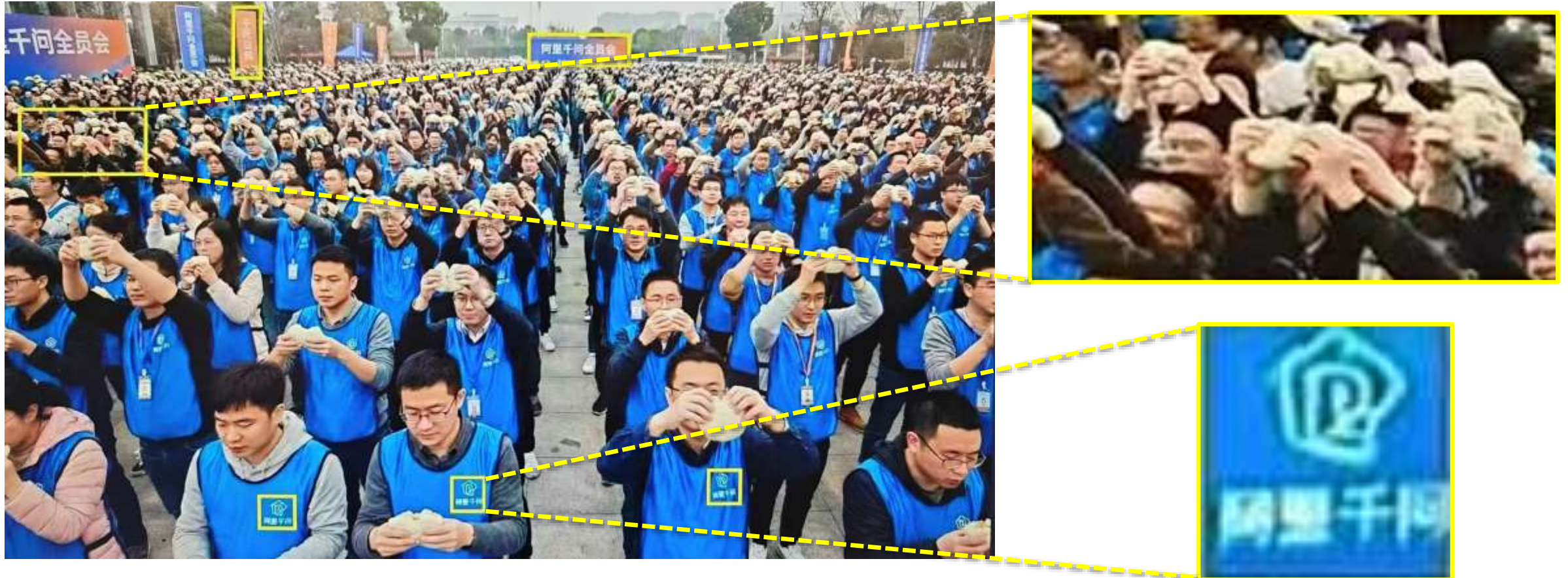
<bbox>... the flames at the bottom of the tower lack the natural chaotic variation of real fire; the texture looks smooth</bbox>

Overall, this image shows a <tag>material error</tag>.

</think>

In conclusion, this image is AI-generated.

# AIGI Detection with Reliable Explanations



Interanimate: Taming region-aware diffusion model for realistic human interaction animation, ACM MM 25

FakeXplain: AI-Generated Images Detection via Human-Aligned Grounded Reasoning, arXiv

Zoom-In to Sort AI-Generated Images Out, arXiv

# About this course

This course will cover:

- How real-world problems are formulated as generative models?
- Probabilistic foundations and learning algorithms
- Challenges, opportunities, open questions