

Generative Modeling

Vinay Setty
vinay.j.setty@uis.no



University
of Stavanger



Department of Electrical Engineering and Computer Science
University of Stavanger

January 6, 2026



Generators

Discriminators

Applications

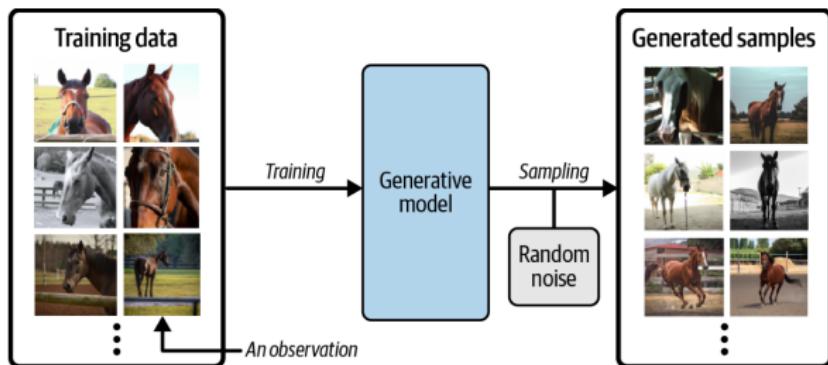
Representation Learning

Taxonomy

What are generative models?



- ▶ Given a training dataset, generate a data that similar to this.
- ▶ A generative model should learn from the features of the data
- ▶ E.g, Given a dataset of horses the features are pixels
- ▶ A generative model should be probabilistic and generate new features (pixels)





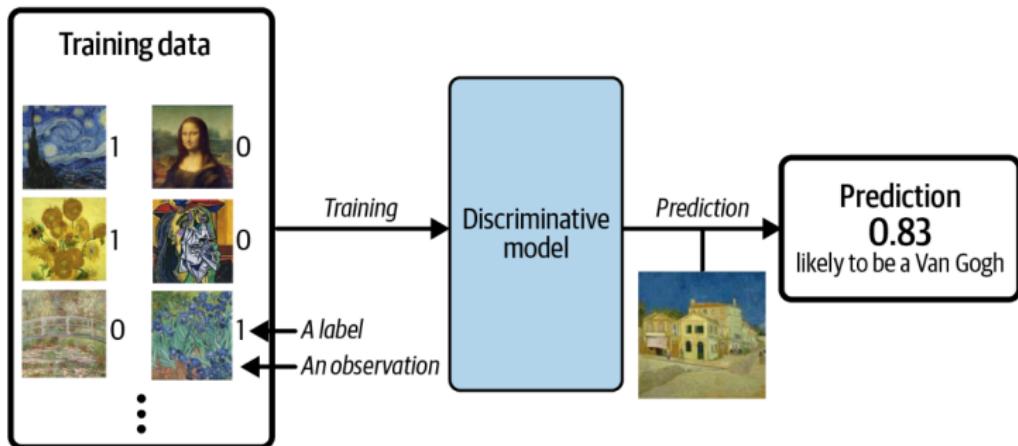
Large Language Models (LLMs)

- ▶ Are LLMs generative models?

What are discriminators?



- ▶ Traditional deep learning focuses on Discriminators.
- ▶ Given a set of features predict something, may be some subset of the features are enough to predict.



Discriminators



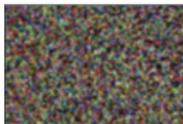
- ▶ Discriminative learning often focuses on modeling the conditional distribution:

$$p(y | x)$$

This can yield high accuracy, but may fail to capture semantic understanding and uncertainty.



+



noise

=



$p(y = \text{cat}|\mathbf{x}) = 0.05$
 $p(y = \text{dog}|\mathbf{x}) = 0.05$
 $p(y = \text{horse}|\mathbf{x}) = 0.90$

Small input changes can shift predicted label probabilities even when the input looks unchanged to humans.

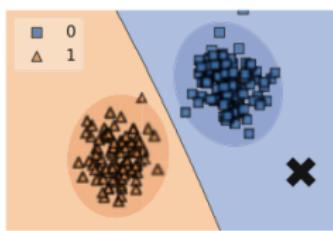
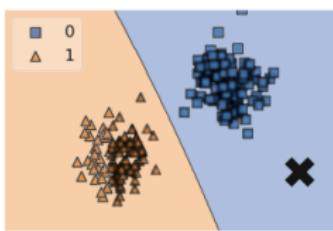
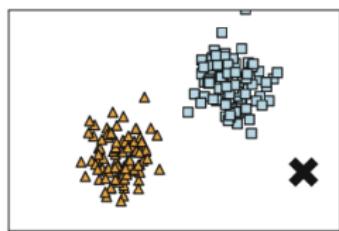
Two approaches to decision making



- ▶ Discriminative: learn $p(y | x)$ directly
- ▶ Generative: learn a joint model and factorize as

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

Generative modeling introduces an explicit notion of how typical an input x is under $p(x)$.



Data

$p(y|x)$

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

$p(\text{blue}|x)$ is high
= certain decision!

$p(\text{blue}|x)$ is high
and $p(x)$ is low
= uncertain decision!

Why $p(x)$ changes the story



If a point is far from the data manifold, $p(x)$ is low, so even if $p(y | x)$ is confident, the joint can be small:

$$p(x = \tilde{x}, y) = p(y | \tilde{x}) p(\tilde{x})$$

This enables uncertainty-aware decisions and novelty detection. [

Why estimate $p(x)$?



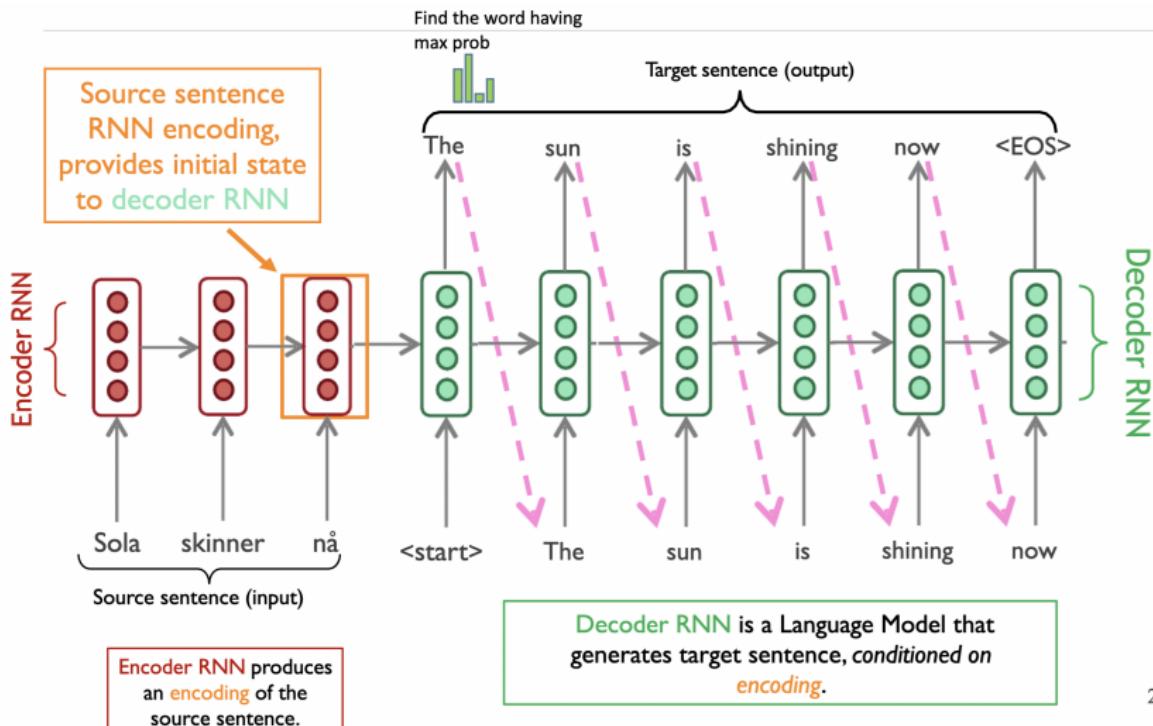
From the generative perspective, $p(x)$ is useful to:

- ▶ assess whether an object was observed before (out-of-distribution detection)
- ▶ weight decisions properly
- ▶ assess uncertainty about the environment
- ▶ support active learning (query low $p(x)$ points)
- ▶ generate (synthesize) new objects

Applications of deep generative modeling



Machine translation (Encoder - Decoder architecture)



Applications of deep generative modeling (cont.)

$$P(\text{Next word} | \text{Previous Words})$$

ChatGPT 5.2 >

To get an A in a Generative AI course the students have to

To get an A in a Generative AI course the students have to demonstrate a solid understanding of core models and algorithms, critically evaluate limitations and ethical implications, implement working systems with clear experimental methodology, and communicate results with rigor and originality.

🔗 ⓘ ...

Applications of deep generative modeling (cont.)



Code generation

ChatGPT 5.2 >

write code to compute fibonacci series in python

Here you go. The ancient ritual of Fibonacci, performed in Python, without unnecessary drama.

python

Copy

```
def fibonacci(n):
    if n <= 0:
        return []
    if n == 1:
        return [0]

    seq = [0, 1]
    for _ in range(2, n):
        seq.append(seq[-1] + seq[-2])
    return seq

# example
print(fibonacci(10))
```

This computes the first n Fibonacci numbers iteratively, which is faster and less self-destructive than recursion.

↪ ⏪ ...

Applications of deep generative modeling (cont.)



Text to image: $P(\text{Image}|\text{Caption})$

ChatGPT 5.2 >

Generate an image of students in University of Stavanger in a generative AI course

Image created

...

Applications of deep generative modeling (cont.)



Generation over years



- ▶ Generation is hard since they have to approximate all features.



2014



2015



2016



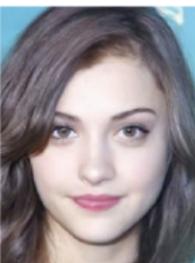
2017



2018



2019



2020



2021



2022



2023



- ▶ Deep generative models apply across diverse data modalities, including text, images, audio, graphs, reinforcement learning, active learning, and medical imaging.
- ▶ Beyond synthesis, generative models quantify uncertainty via $p(x)$, enabling tasks such as querying informative samples in active learning and predicting plausible future states in reinforcement learning.
- ▶ In high-stakes domains like medical imaging, generative modeling supports interpretability by expressing confidence over both observations and labels.
- ▶ These applications highlight generative modeling as a core capability underlying modern AI systems.

Generative Modeling and AI Applications (cont.)



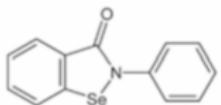
" i want to talk to you . "
"i want to be with you . "
"i do n't want to be with you . "
"i do n't want to be with you . "
she did n't want to be with him .

he was silent for a long moment .
he was silent for a moment .
it was quiet for a moment .
it was dark and cold .
there was a pause .
it was my turn .

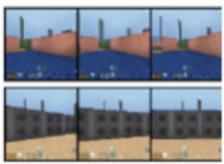
Text



Images



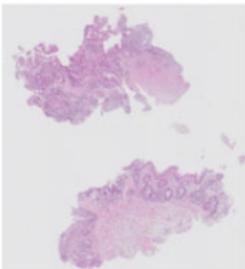
Graphs



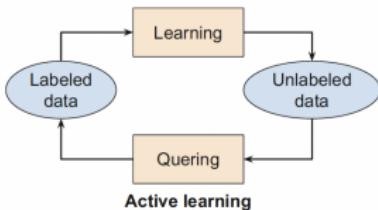
Reinforcement learning



Audio



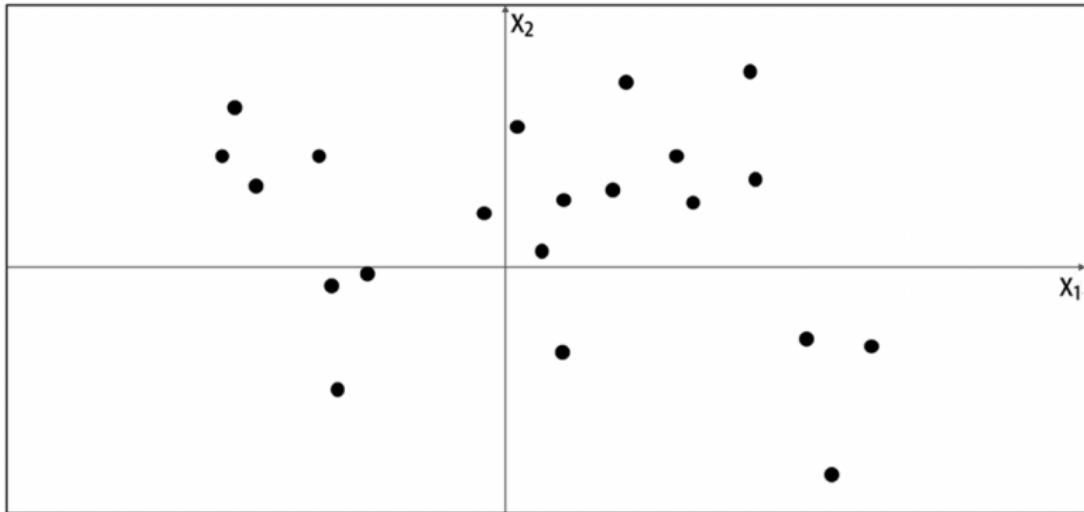
Medical imaging



Toy example



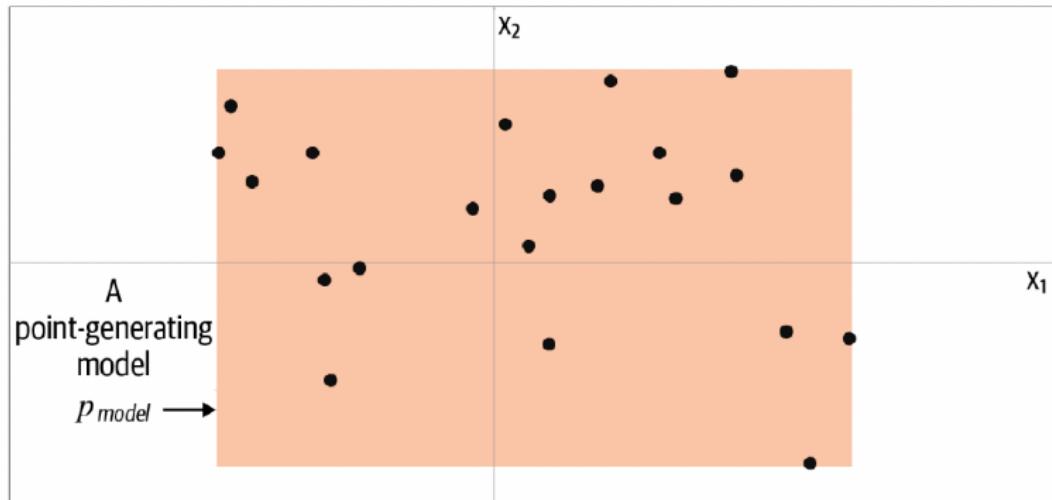
- ▶ Assume you are given this 2d-data.
- ▶ Propose a simple generator model.



Toy example (cont.)



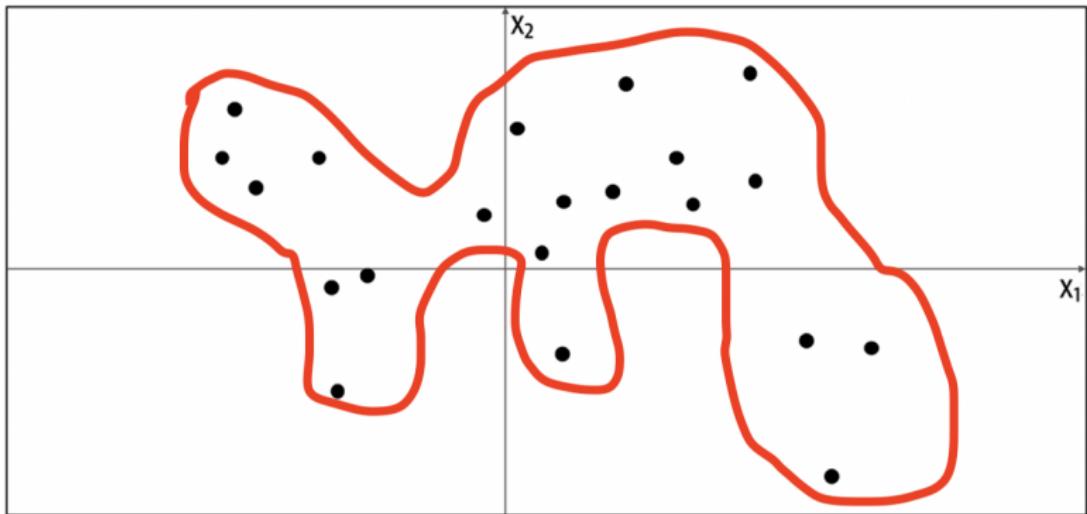
- ▶ The orange rectangle could be a reasonable approximation.



Toy example (cont.)



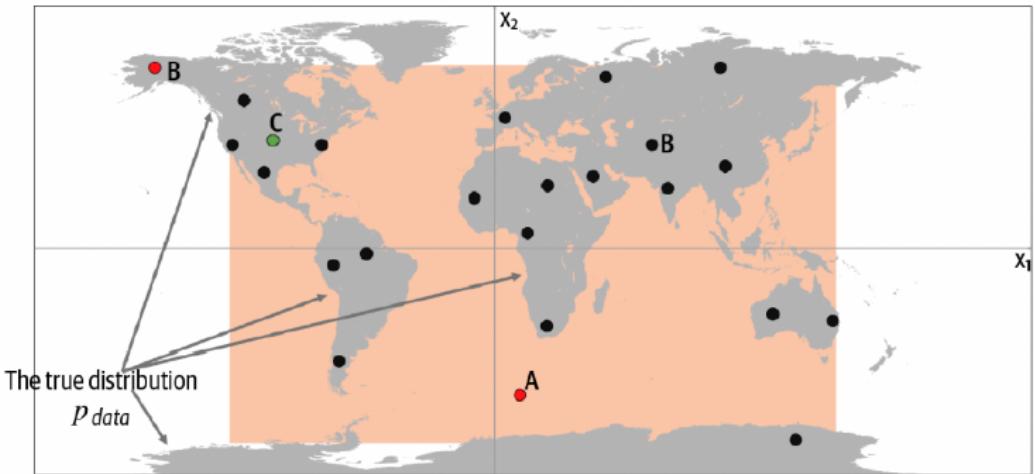
- ▶ What about this?



Toy example (cont.)



- ▶ Let's reveal the true distributions.
- ▶ A is out of distribution.
- ▶ B would never get generated.
- ▶ C is a correct generation.

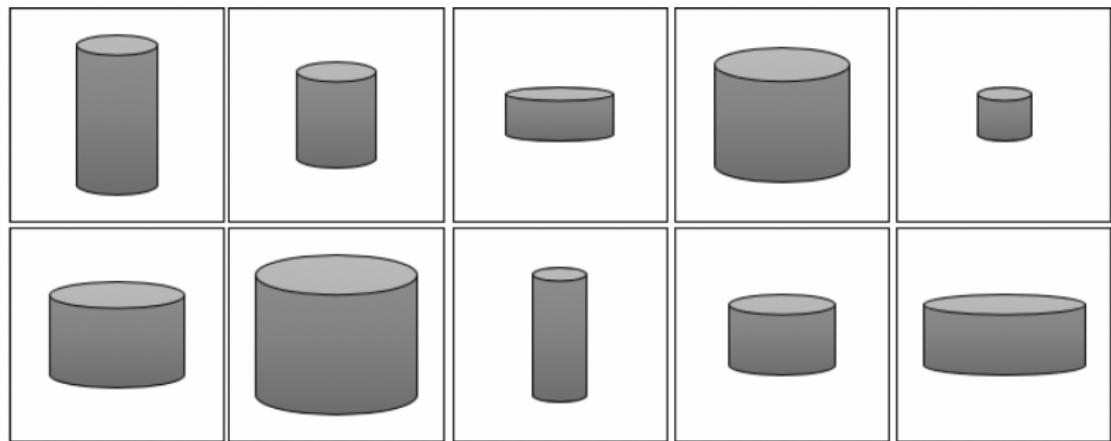


Representation Learning

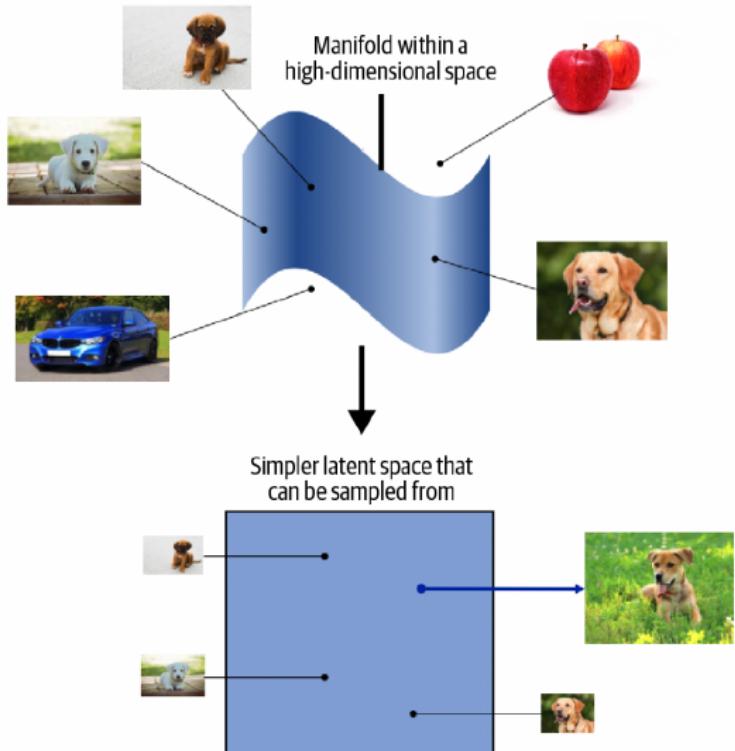


What does it mean by representation learning?

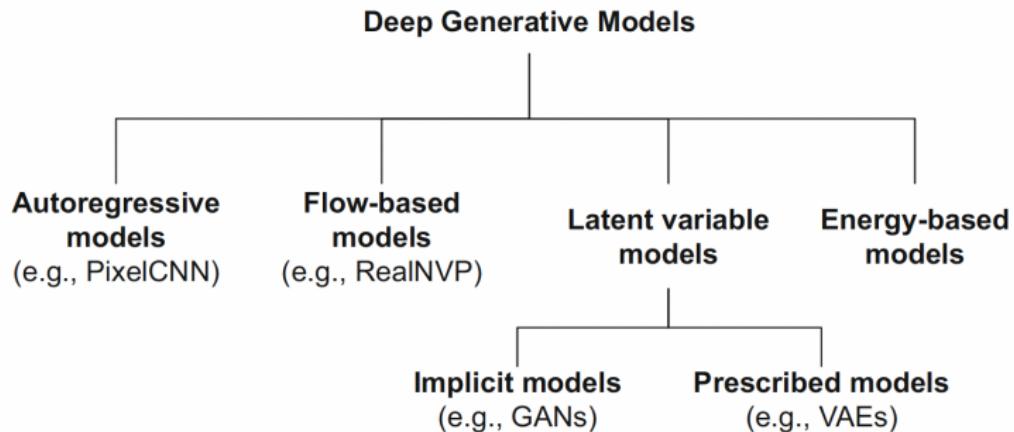
Suppose you want to describe a person how would you do it?



Representation Learning (cont.)



Generative Modeling Taxonomy





Autoregressive factorization. Let $x = (x_0, \dots, x_D)$ denote a multivariate random variable with a fixed ordering of its components. Autoregressive models represent the joint distribution as

$$p(x) = p(x_0) \prod_{i=1}^D p(x_i | x_{<i}), \quad x_{<i} := (x_0, \dots, x_{i-1}).$$



Computational challenge. Naively modeling each conditional distribution $p(x_i | x_{<i})$ is inefficient, as each component may depend on all previously ordered components.

Efficient implementation. Causal convolutions enforce the autoregressive constraint by restricting the model at index i to access only $x_{<i}$, while still allowing parallelized training. This approach has been successfully applied to:

- ▶ audio generation
- ▶ image generation with masked or causal convolutions

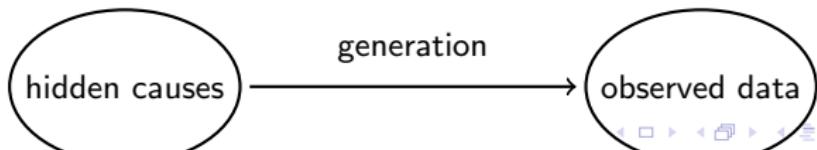


- ▶ Goal: model a complicated data distribution by **transforming** a simple one.
- ▶ Start with an easy distribution (e.g., Gaussian) and apply an **invertible function** f .
- ▶ The change-of-variables formula tells us how probabilities change under this transformation:

$$p(x) = p(z = f(x)) |\det J_f(x)|$$

- ▶ The determinant of the Jacobian measures how much the transformation stretches or squashes space.
- ▶ The function f is implemented with neural networks, but must be:
 - invertible
 - easy to compute the Jacobian determinant

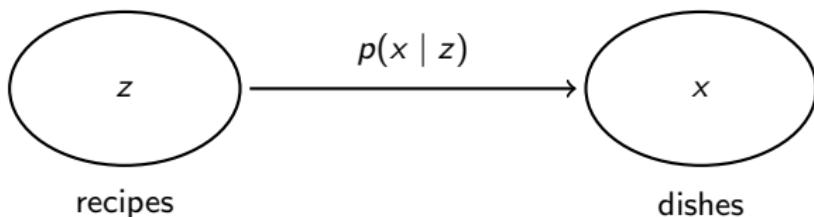
- ▶ Assume data is produced by **hidden causes** we cannot directly see.
- ▶ These hidden causes explain why data looks the way it does.
- ▶ The model works in two steps:
 - first choose hidden factors
 - then generate visible data from them
- ▶ Learning means:
 - figuring out what hidden factors are useful
 - learning how they produce realistic data
- ▶ Neural networks are used to model these relationships.
- ▶ Because the hidden causes are not observable, we can only **approximate** them.



Latent Variable Models: An Analogy



- ▶ Think of each data point as a **finished dish** served at a restaurant.
- ▶ The latent variable z is the **recipe card**:
 - hidden ingredients
 - proportions and style
- ▶ The observation x is the **dish you see on the table**.
- ▶ Generative process:
 - randomly choose a recipe
 - cook a dish from it



Different recipes can produce similar dishes

Comparing Deep Generative Models



	VAEs [1]	Flows [2]	GANs [3]
Latent variables	Yes	Yes	Yes
Invertible mapping	No	Yes	No
Exact likelihood	Approx.	Exact	No
Training objective	ELBO (KL-based)	Log-likelihood	Adversarial
Sampling	Easy	Easy	Easy
Density evaluation	Easy	Easy	Hard / impossible

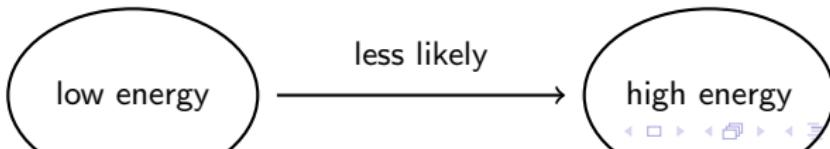
- ▶ **VAEs**: prescribed probabilistic models with approximate inference.
- ▶ **Flows**: exact likelihood via invertible transformations.
- ▶ **GANs**: implicit models trained via a discriminator, no explicit density.

Choice reflects a trade-off between likelihood tractability, flexibility, and training stability.

Energy-Based Models [4]



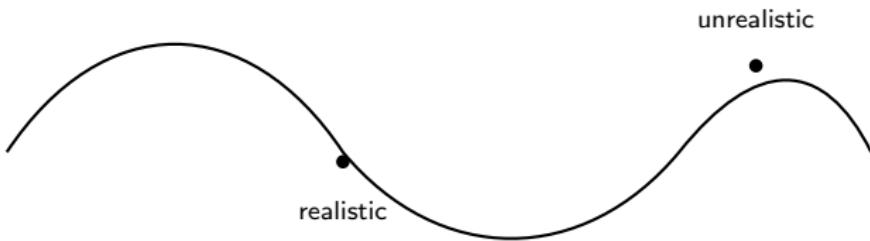
- ▶ Rather than modeling probabilities, we assign an **energy score** to each data point.
- ▶ Low energy means the model thinks the data looks realistic.
- ▶ High energy means the data looks unlikely or wrong.
- ▶ Probabilities are obtained by:
 - preferring low-energy configurations
 - normalizing over all possible inputs
- ▶ Learning means shaping the energy landscape so that:
 - real data has low energy
 - unrealistic data has high energy
- ▶ Neural networks can be used to represent the energy function.



Energy-Based Models: An Analogy



- ▶ Imagine a **landscape of hills and valleys**.
- ▶ Each possible data point is a location in this landscape.
- ▶ The **height of the land** is the energy:
 - valleys correspond to realistic data
 - hills correspond to unrealistic data
- ▶ The model believes data points found in valleys are much more likely.
- ▶ Learning reshapes the terrain so real data falls into deep valleys.



Learning digs valleys where real data lives

Bibliography



- [1] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *International Conference on Learning Representations*, Origin of the VAE framework, 2014.
- [2] D. J. Rezende and S. Mohamed, "Variational inference with normalizing flows," in *Proceedings of the 32nd International Conference on Machine Learning*, Normalizing flows for flexible posterior approximation, PMLR, 2015, pp. 1530–1538.
- [3] I. Goodfellow et al., "Generative adversarial nets," *Advances in Neural Information Processing Systems*, vol. 27, 2014, Foundational work on GANs.
- [4] Y. W. Teh, M. Welling, S. Osindero, and G. E. Hinton, "Energy-based models for sparse overcomplete representations," *Journal of Machine Learning Research*, vol. 4, pp. 1235–1260, 2003, Early work on general energy-based models.