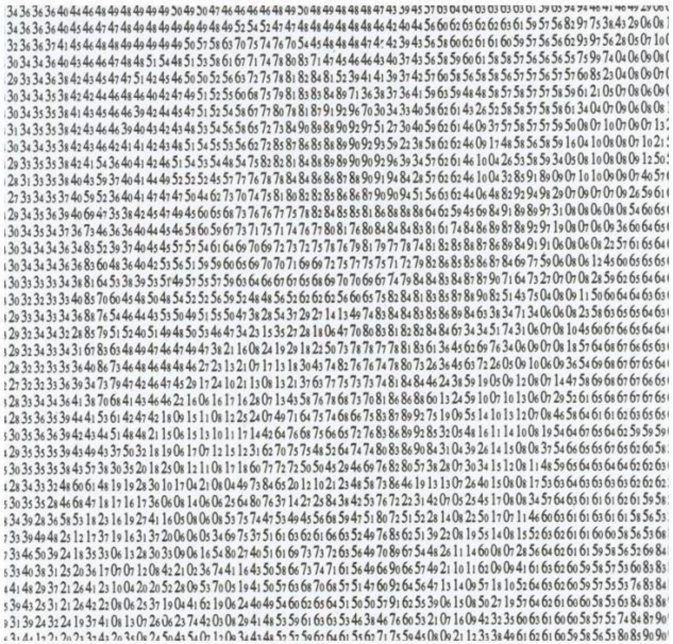
Energy-Based Generative Models

Probabilistic Models of Images



- An image is a collection of numbers indicating the intensity values of the pixels, and is a high dimensional object.

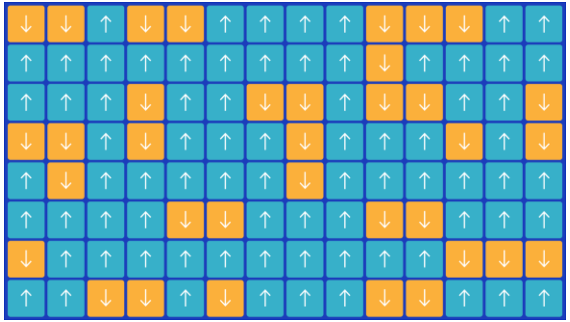
- A population of images (e.g., images of faces, cats) can be described by a probability distribution.

- A probabilistic model is a probability distribution parametrized by a set of parameters, which can be learned from the data.

Gibbs Distribution in Statistical Physics

$$p(x) = \frac{1}{Z} \exp\left(-\frac{E(x)}{T}\right)$$

$$Z = \int \exp\left(-\frac{E(x)}{T}\right) dx$$



Energy-based model originates from the Gibbs distribution in statistical physics:

- x is the state of a system (e.g., ferromagnetic substance, a cup of water, gas…).
- $E(x)$ is the energy of the system at state x.
- T is the temperature. As $T \to 0, p(x)$ focuses on the global minima of $E(x)$.
- Z is the normalizing constant, or partition function, to make $p(x)$ a probability density.
- The partition function is ubiquitous in statistics physics (also quantum physics).
- **States of low energies have high probabilities**

Energy-Based Model (EMB)

$$p_\theta(x) = \frac{1}{Z(\theta)} \exp(f_\theta(x)) \qquad Z(\theta) = \int \exp(f_\theta(x))dx$$

In this tutorial, we present energy-based model (EBM**):**

- x is an image (or video, text, etc.)
- $-E(x)/T$ will be parametrized by modern ConvNet $f_\theta(x)$, where θ denotes the parameters.
- $f_\theta(x)$ captures **regularities, rules, organizations and constraints** probabilistically.
- In conditional settings, $f_\theta(x)$ acts as **soft objective function, cost function, value function, or critic**.
- It actually is a **softmax probability**, recall in classification, for a category c, with logit score $f(c)$,

$$\Pr(c) = \frac{1}{Z}\exp(f(c)) = \frac{\exp(f(c))}{\sum_c \exp(f(c))}$$

- Here we assign score $f_\theta(x)$ to each x, and **softmax over all x** (as if each x is a category).

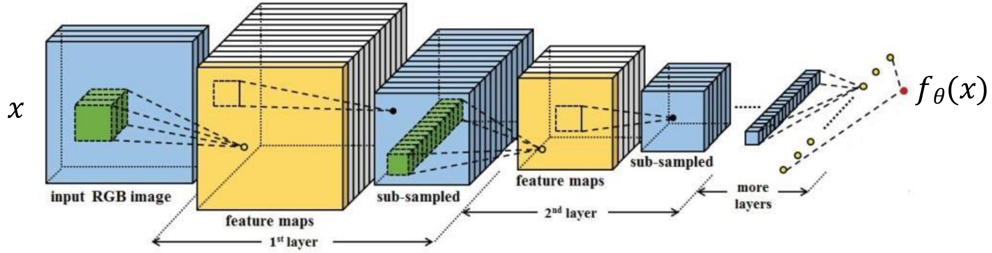
EBM Parameterized by Modern Neural Network

- Let x be an image defined on image domain D, the Generative ConvNet is a probability distribution defined on image domain

$$p(x) = \frac{1}{Z(\theta)} \exp(f_\theta(x))q(x)$$

where $q(x)$ is a reference distribution, e.g., uniform or Gaussian distribution $q(x) = \frac{1}{(2\pi\sigma^2)^{|D|/2}} \exp\left(-\frac{1}{2\sigma^2}\|x\|^2\right)$

- $Z(\theta)$ is the normalizing constant $Z(\theta) = \int_x \exp(f_\theta(x))q(x)dx$
- $f_\theta(x)$ is parameterized by a ConvNet structure that maps the input image to a scalar. θ contains all the parameters of the ConvNet.



[1] Jianwen Xie, Yang Lu, Song-Chun Zhu, Ying Nian Wu. A Theory of Generative ConvNet. ICML, 2016

Synthesis by Langevin dynamics

Kullback-Leibler Divergences in Two Directions

For two probability densities $p(x)$ and $q(x)$, the Kullback-Leibler Divergence (KL-divergence) is defined

$$\mathbb{D}_{\mathrm{KL}}(p\|q)=\mathbb{E}_p\left[\log\frac{p(x)}{q(x)}\right]=\int p(x)\log\frac{p(x)}{q(x)}dx$$

The KL-divergence appears in two scenarios:

(1) **Maximum likelihood estimation**: Suppose there are training examples $x_i \sim p_{\text{data}}(x)$ and we want to learn a model $p_\theta(x)$. The log-likelihood function is

$$L(\theta)=\frac{1}{n}\sum_{i=1}^{n}\log p_\theta\left(x_i\right)\rightarrow\mathbb{E}_{p_{\text{data}}}\left[\log p_\theta(x)\right]$$

Thus, for a large n, maximizing the log-likelihood is equivalent to minimizing the KL-divergence

$$\mathbb{D}_{\mathrm{KL}}\left(p_{\text{data}}\|p_\theta\right)=-\text{ entropy }\left(p_{\text{data}}\right)-\mathbb{E}_{p_{\text{data}}}\left[\log p_\theta(x)\right]\doteq-\text{ entropy }\left(p_{\text{data}}\right)-L(\theta)$$

Kullback-Leibler Divergences in Two Directions

(2) **Variational approximation**: Suppose there is a target distribution p_{target} and we know p_{target} up to a normalizing constant, e.g.,

$$p_{\text{target}}(x) = \frac{1}{Z} \exp(f(x))$$

where $f(x)$ is known but $Z = \int \exp\big(f(x)\big) dx$ is analytically intractable.

Suppose we want to approximate it by a distribution q_ϕ. We can find ϕ by minimizing

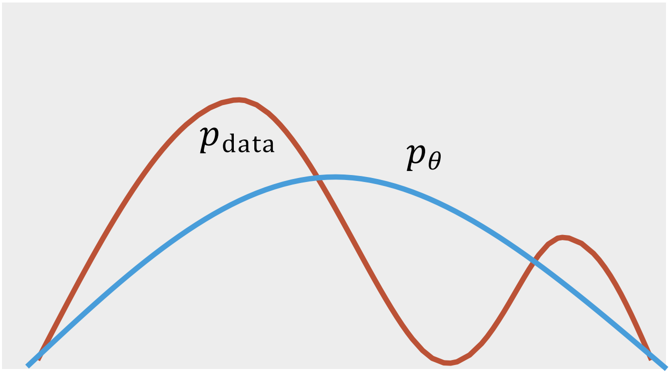
$$\mathbb{D}_{\mathrm{KL}}\left(q_\phi \| p_{\text{target}}\right) = \mathbb{E}_{q_\phi}\left[\log q_\phi(x)\right] - \mathbb{E}_{q_\phi}[f(x)] + \log Z$$

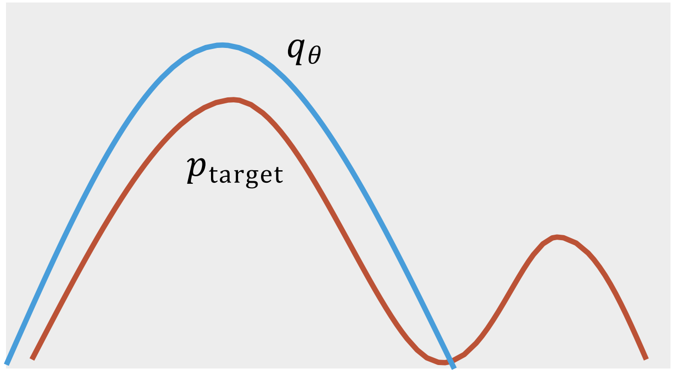
The above minimization does not require knowledge of $\log Z$.

Kullback-Leibler Divergences in Two Directions

The behaviors of $\mathbb{D}_{\mathrm{KL}}\left(p_{\mathrm{data}} \| p_\theta\right)$ in scenario (1) and $\mathbb{D}_{\mathrm{KL}}\left(q_\phi \| p_{\mathrm{target}}\right)$ in scenario (2) are different.

In (1), p_θ tends to cover all the modes of p_{data}, while in (2) q_ϕ tends to focus on some major modes of p_{target} while ignoring the minor modes.





$\mathbb{D}_{\mathrm{KL}}\left(q_\phi \| p_{\mathrm{target}}\right)$

Maximum Likelihood Estimation

- Observed data $\{x_1, ..., x_n\} \sim p_{\text{data}}(x)$

- Model: $$p_\theta(x) = \frac{1}{Z(\theta)} \exp(f_\theta(x))$$
$$Z(\theta) = \int \exp(f_\theta(x))dx$$

- Objective function of MLE learning is

$$L(\theta) = \frac{1}{n}\sum_{i=1}^{n} \log p_\theta(x_i)$$

- The gradient of the log-likelihood is

$$L'(\theta) = \frac{1}{n}\sum_{i=1}^{n} \nabla_\theta f_\theta(x_i) - \mathbb{E}_{p_\theta(x)}[\nabla_\theta f_\theta(x)]$$

Derivation of gradient of the log-likelihood:

$$\nabla_\theta \log p_\theta(x) = \nabla_\theta f_\theta(x) - \nabla_\theta \log Z(\theta)$$

where the term $\nabla_\theta \log Z(\theta)$ can be rewritten as

$$\begin{aligned}
\nabla_\theta \log Z(\theta) &= \frac{1}{Z(\theta)} \nabla_\theta Z(\theta) \\
&= \frac{1}{Z(\theta)} \nabla_\theta \int \exp(f_\theta(x))dx \\
&= \frac{1}{Z(\theta)} \int \exp(f_\theta(x)) \nabla_\theta f_\theta(x)dx \\
&= \int \frac{1}{Z(\theta)} \exp(f_\theta(x)) \nabla_\theta f_\theta(x)dx \\
&= \int p_\theta(x) \nabla_\theta f_\theta(x)dx \\
&= \mathbb{E}_{p_\theta(x)}[\nabla_\theta f_\theta(x)]
\end{aligned}$$

Maximum Likelihood Estimation

Given a set of observed images $\{x_1, ..., x_n\} \sim p_{\text{data}}(x)$

Gradient of MLE learning

$$\sum_{x} p_\theta(x) \nabla_\theta f_\theta(x)$$

$$L'(\theta) = \mathbb{E}_{p_{\text{data}}(x)}[\nabla_\theta f_\theta(x)] - \mathbb{E}_{p_\theta(x)}[\nabla_\theta f_\theta(x)]$$

$$\approx \frac{1}{n}\sum_{i=1}^{n} \nabla_\theta f_\theta(x_i) - \boxed{\frac{1}{\tilde{n}}\sum_{i=1}^{\tilde{n}} \nabla_\theta f_\theta(\tilde{x}_i)}$$

Approximated by MCMC $\{\tilde{x}_1, ..., \tilde{x}_{\tilde{n}}\} \sim p_\theta(x)$

e.g., x is a 100x100 grey-scale image

Each pixel ~ [0, 255].

Image space is $256^{10,000}$!

Intractable!!

The expectation is analytically intractable and has to be approximated by Markov chain Monte Carlo (MCMC), such as Langevin dynamics or Hamiltonian Monte Carlo (HMC).

[1] Jianwen Xie, Yang Lu, Song-Chun Zhu, Ying Nian Wu. A Theory of Generative ConvNet. ICML, 2016

Gradient-Based MCMC and Langevin Dynamics

For high dimensional data x, sampling from distribution $p_\theta(x) = \frac{1}{Z(\theta)} \exp(f_\theta(x))$ requires MCMC, such as Langevin dynamics

$$x_{t+\Delta t} = x_t + \underbrace{\frac{\Delta t}{2} \nabla_x f_\theta(x_t)}_{\text{Gradient ascent}} + \underbrace{\sqrt{\Delta t} e_t}_{\text{Brownian motion}} \qquad e_t \sim \mathcal{N}(0, I)$$

As $\Delta t \to 0$ and $t \to \infty$, the distribution of x_t converges to $p_\theta(x)$.
Δt corresponds to step size in implementation.

Different implementations of the synthesis step:

(i) **Persistent chain**: runs a finite-step MCMC from the synthesized examples generated from the previous epoch.

(ii) **Contrastive divergence**: runs a finite-step MCMC from the observed examples.

(iii) **Non-persistent short-run MCMC**: runs a finite-step MCMC from Gaussian white noise.

Analysis by Synthesis

Input: training images $\{x_1, \ldots, x_n\} \sim p_{\text{data}}(x)$

Output: model parameters θ

For *t* =1 to *N*

synthesis step: $\{\tilde{x}_1, \ldots, \tilde{x}_{\tilde{n}}\} \sim p_{\theta_t}(x)$

observed statistics

synthesized statistics

analysis step: $\theta_{t+1} = \theta_t + \eta_t \left[\boxed{\frac{1}{n} \sum_{i=1}^{n} \nabla_\theta f_\theta(x_i)} - \boxed{\frac{1}{\tilde{n}} \sum_{i=1}^{\tilde{n}} \nabla_\theta f_\theta(\tilde{x}_i)} \right]$

End

Adversarial Interpretation

- The update of θ is based on

$$L'(\theta) \approx \frac{1}{n}\sum_{i=1}^{n} \nabla_\theta f_\theta(x_i) - \frac{1}{\tilde{n}}\sum_{i=1}^{\tilde{n}} \nabla_\theta f_\theta(\tilde{x}_i)$$
$$= \nabla_\theta \left[\frac{1}{n}\sum_{i=1}^{n} f_\theta(x_i) - \frac{1}{\tilde{n}}\sum_{i=1}^{\tilde{n}} f_\theta(\tilde{x}_i)\right]$$

where $\{\tilde{x}_1, \ldots, \tilde{x}_{\tilde{n}}\}$ are the synthesized images generated by the Langevin dynamics

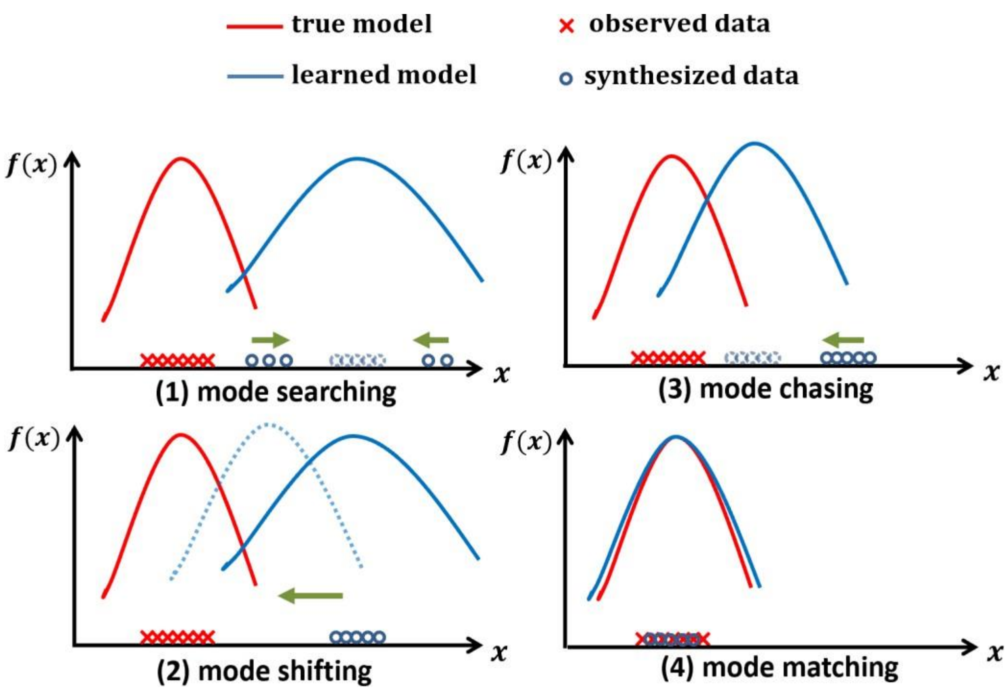
- Define a value function $V(\{\tilde{x}_i\}, \theta) = \frac{1}{n}\sum_{i=1}^{n} f_\theta(x_i) - \frac{1}{\tilde{n}}\sum_{i=1}^{\tilde{n}} f_\theta(\tilde{x}_i)$

- The learning and sampling steps play a minimax game: $\min_{\{\tilde{x}_i\}} \max_{\theta} V(\{\tilde{x}_i\}, \theta)$
- See Part 2 for adversarial contrastive divergence

[1] Jianwen Xie, Song-Chun Zhu, Ying Nian Wu. Synthesizing Dynamic Patterns by Spatial-Temporal Generative ConvNet. CVPR, 2017

Mode Seeking and Mode Shifting

Mode seeking and mode shifting



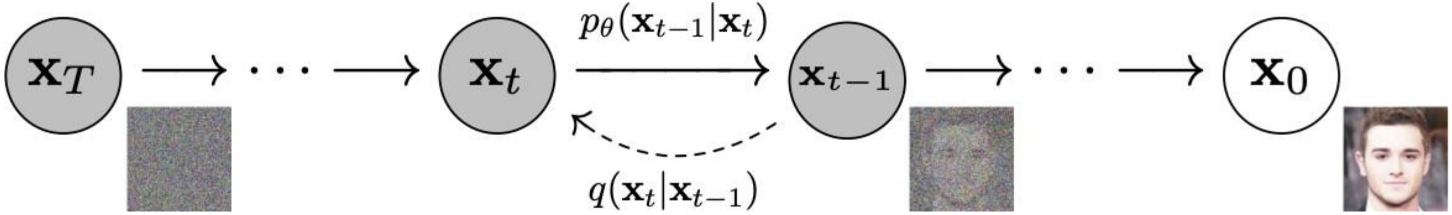
Multistage Coarse-to-Fine Expanding and Sampling

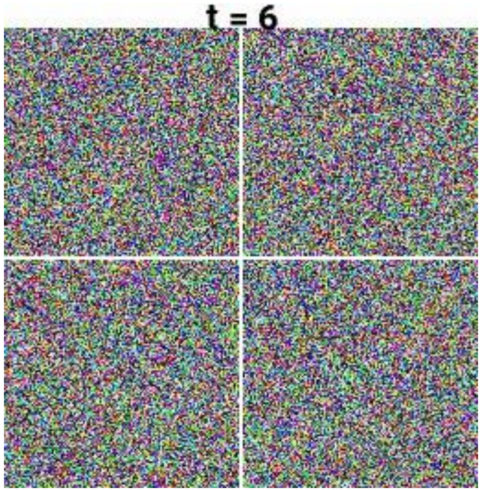
MCMC generative sequences on CelebA (50 Langevin steps)

Generated examples on CelebA-HQ at 512 × 512 resolution

[1] Yang Zhao, Jianwen Xie, Ping Li. Learning Energy-Based Generative Models via Coarse-to-Fine Expanding and Sampling. ICLR, 2021.

Diffusion-Based Modeling and Sampling





$$x_t = x_{t-1} + \sigma\epsilon_t \ \rightarrow q(x_t|x_{t-1})$$

$$p_\theta(x_t) = \frac{1}{Z(\theta, t)}\exp(f_\theta(x_t, t))$$

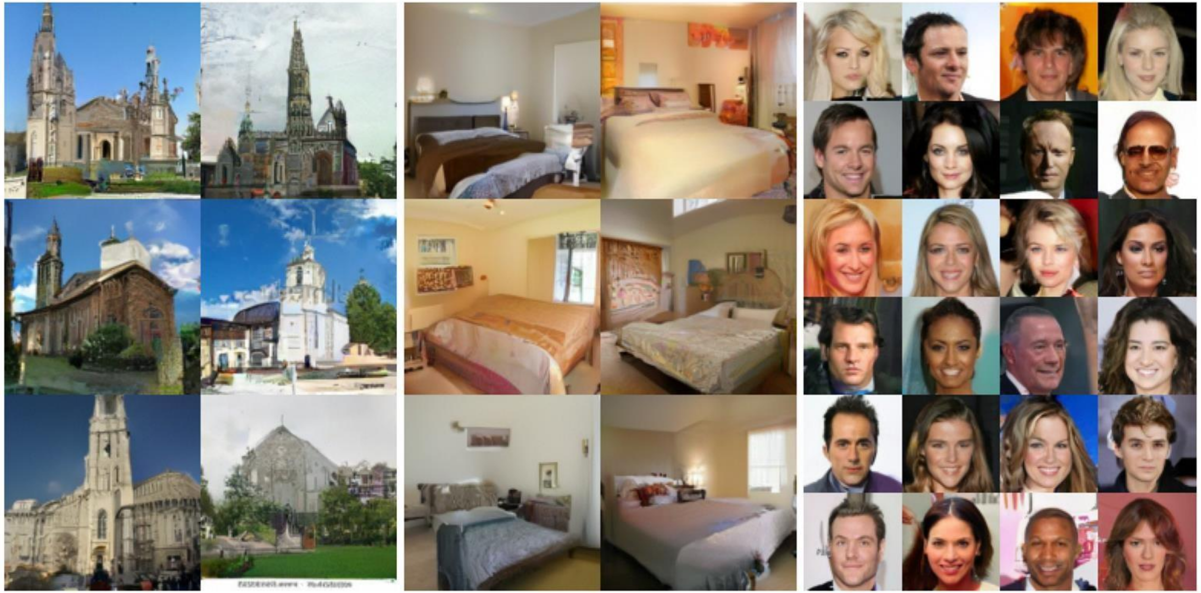
$$p_\theta(x_{t-1}|x_t) \propto \exp\left(f_\theta(x_{t-1}) - \frac{1}{2\sigma^2}\|x_t - x_{t-1}\|^2\right)$$

- Conditional distribution is easier to sample from than marginal
- Close to unimodal around x_t
- Denoising, recall x_{t-1} with hint x_t

[1] Ruiqi Gao, Yang Song, Ben Poole, Ying Nian Wu, and Diederik P. Kingma. Learning energy-based models by diffusion recovery likelihood. ICLR 2021

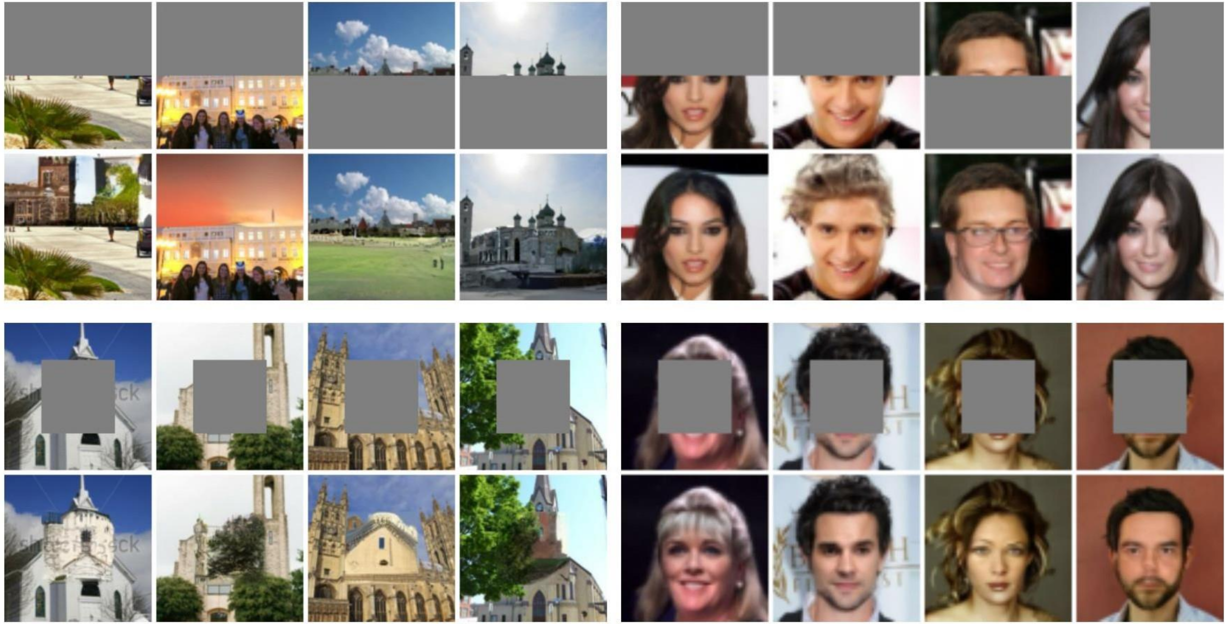
Diffusion-Based Modeling and Sampling

Diffusion recovery likelihood: SOTA synthesized results for pure EBMs.



[1] Ruiqi Gao, Yang Song, Ben Poole, Ying Nian Wu, and Diederik P. Kingma. Learning energy-based models by diffusion recovery likelihood. ICLR 2021

Diffusion-Based Modeling and Sampling

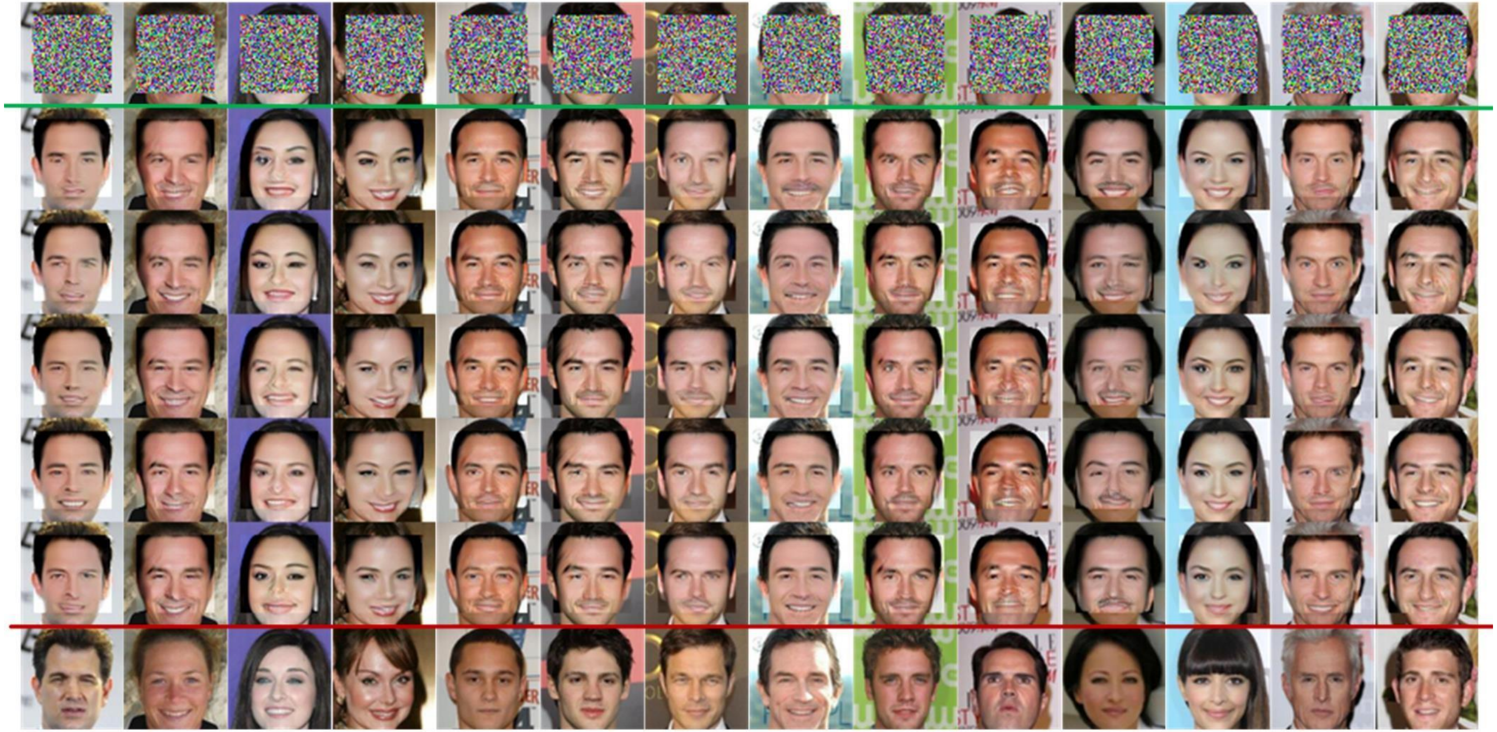


[1] Ruiqi Gao, Yang Song, Ben Poole, Ying Nian Wu, and Diederik P. Kingma. Learning energy-based models by diffusion recovery likelihood. ICLR 2021

Image Synthesis

1 Jianwen Xie *, Yang Lu *, Song-Chun Zhu, Ying Nian Wu. A Theory of Generative ConvNet. ICML 2016
2 Yang Zhao, Jianwen Xie, Ping Li. Learning Energy-Based Generative Models via Coarse-to-Fine Expanding and Sampling. ICLR 2021
3 Ruiqi Gao, Yang Song, Ben Poole, Ying Nian Wu, and Diederik P. Kingma. Learning energy-based models by diffusion recovery likelihood. ICLR 2021

Image Inpainting

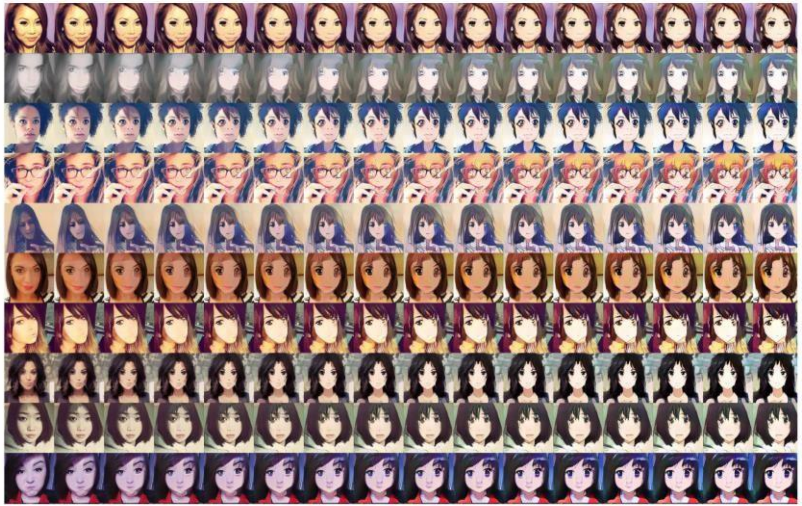


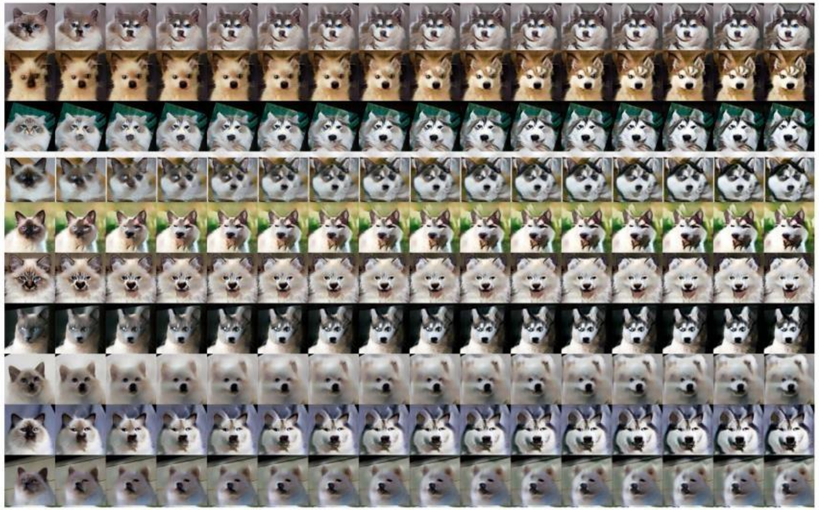
One-Sided Image-to-Image Translation

$$x \Rightarrow y$$

$$p(y) \propto \exp(f(y))$$

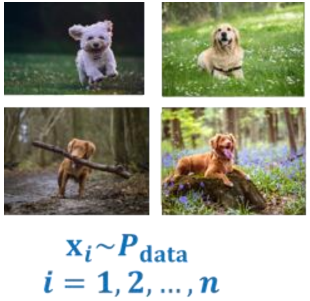
$$y_{t+\Delta t} = y_t + \frac{\Delta t}{2} \nabla_y f(y_t) + \sqrt{\Delta t} e_t \qquad y_0 = x \sim p_{\text{data}}(x)$$

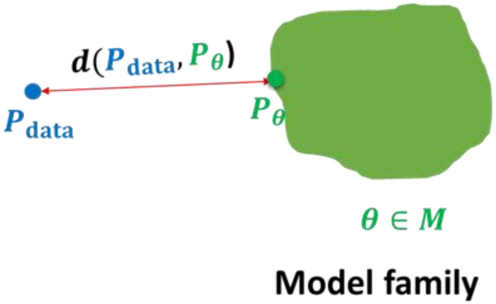




[1] Yang Zhao, Jianwen Xie, Ping Li. Learning Energy-Based Generative Models via Coarse-to-Fine Expanding and Sampling. ICLR 2021

Recap.



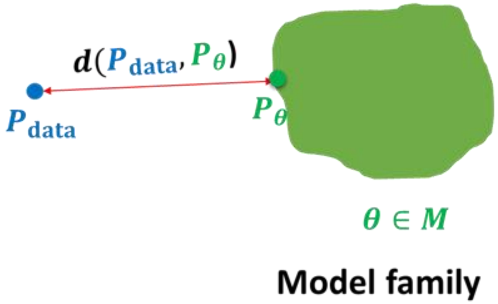


- Autoregressive models. $p_\theta(x_1, x_2, \cdots, x_n) = \prod_{i=1}^{n} p_\theta(x_i \mid x_{<i})$
- Normalizing flow models. $p_\theta(\mathbf{x}) = p(\mathbf{z})|\det J_{f_\theta}(\mathbf{x})|$, where $\mathbf{z} = f_\theta(\mathbf{x})$.
- Variational autoencoders: $p_\theta(\mathbf{x}) = \int p(\mathbf{z})p_\theta(\mathbf{x} \mid \mathbf{z})d\mathbf{z}$.

Cons: Model architectures are restricted.

Recap.

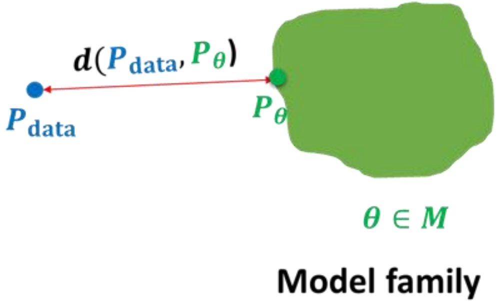
$\mathbf{x}_i \sim P_{\text{data}}$
$i = 1, 2, \ldots, n$



- Generative Adversarial Networks (GANs).
  - $\min_\theta \max_\phi E_{\mathbf{x}\sim p_{\text{data}}}[\log D_\phi(\mathbf{x})] + E_{\mathbf{z}\sim p(\mathbf{z})}[\log(1 - D_\phi(G_\theta(\mathbf{z})))]$.
  - Two sample tests. Can (approximately) optimize f-divergences and the Wasserstein distance.
  - Very flexible model architectures. But likelihood is intractable, training is unstable, hard to evaluate, and has mode collapse issues.

Today's lecture

$\mathbf{x}_i \sim P_{\text{data}}$
$i = 1, 2, \ldots, n$



Energy-based models (EBMs).

- Very flexible model architectures.
- Stable training.
- Relatively high sample quality.
- Flexible composition.

Parameterizing probability distributions

Probability distributions $p(x)$ are a key building block in generative modeling.

1. non-negative: $p(x) \geq 0$
2. sum-to-one: $\sum_x p(x) = 1$ (or $\int p(x)dx = 1$ for continuous variables)

Coming up with a non-negative function $p_\theta(\mathbf{x})$ is not hard.
Given any function $f_\theta(\mathbf{x})$, we can choose

- $g_\theta(\mathbf{x}) = f_\theta(\mathbf{x})^2$
- $g_\theta(\mathbf{x}) = \exp(f_\theta(\mathbf{x}))$
- $g_\theta(\mathbf{x}) = |f_\theta(\mathbf{x})|$
- $g_\theta(\mathbf{x}) = \log(1 + \exp(f_\theta(\mathbf{x})))$
- etc.

Parameterizing probability distributions

Probability distributions $p(\mathbf{x})$ are a key building block in generative modeling.

1. non-negative: $p(\mathbf{x}) \geq 0$
2. sum-to-one: $\sum_{\mathbf{x}} p(\mathbf{x}) = 1$ (or $\int p(\mathbf{x})d\mathbf{x} = 1$ for continuous variables)

Sum-to-one is key:

Total "volume" is fixed: increasing $p(x_{train})$ guarantees that x_{train} becomes relatively more likely (compared to the rest).

Energy-based model

$$p_\theta(\mathbf{x}) = \frac{1}{\int \exp(f_\theta(\mathbf{x}))d\mathbf{x}} \exp(f_\theta(\mathbf{x})) = \frac{1}{Z(\theta)} \exp(f_\theta(\mathbf{x}))$$

Pros:

1. extreme flexibility: can use pretty much any function $f_\theta(\mathbf{x})$ you want

Cons:

1. Sampling from $p_\theta(\mathbf{x})$ is hard
2. Evaluating and optimizing likelihood $p_\theta(\mathbf{x})$ is hard (learning is hard)
3. No feature learning (but can add latent variables)

Curse of dimensionality: The fundamental issue is that computing $Z(\theta)$ numerically (when no analytic solution is available) scales exponentially in the number of dimensions of **x.**

Nevertheless, **some tasks do not require knowing** $Z(\theta)$

Energy-based models: learning and inference

$$p_\theta(\mathbf{x}) = \frac{1}{\int \exp(f_\theta(\mathbf{x}))} \exp(f_\theta(\mathbf{x})) = \frac{1}{Z(\theta)} \exp(f_\theta(\mathbf{x}))$$

Pros:

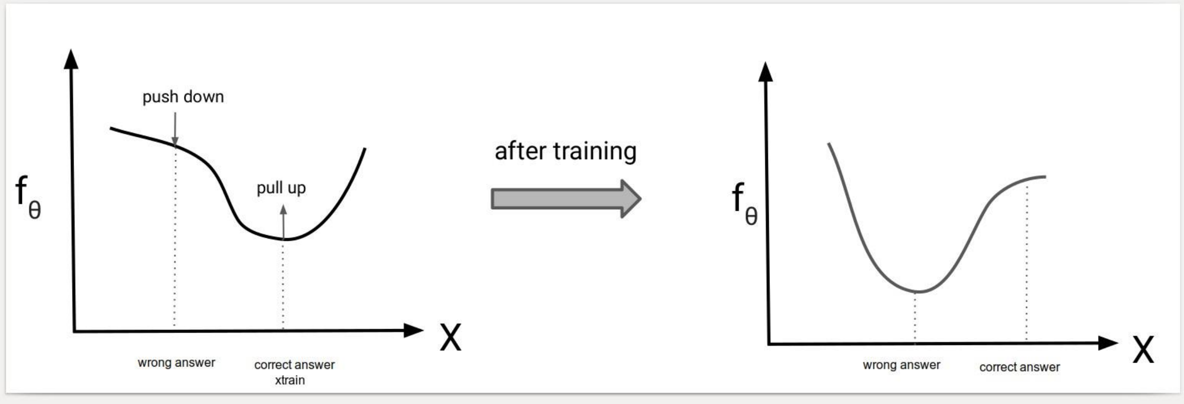
1. can plug in pretty much any function $f_\theta(\mathbf{x})$ you want

Cons (lots of them):

1. Sampling is hard
2. Evaluating likelihood (learning) is hard
3. No feature learning

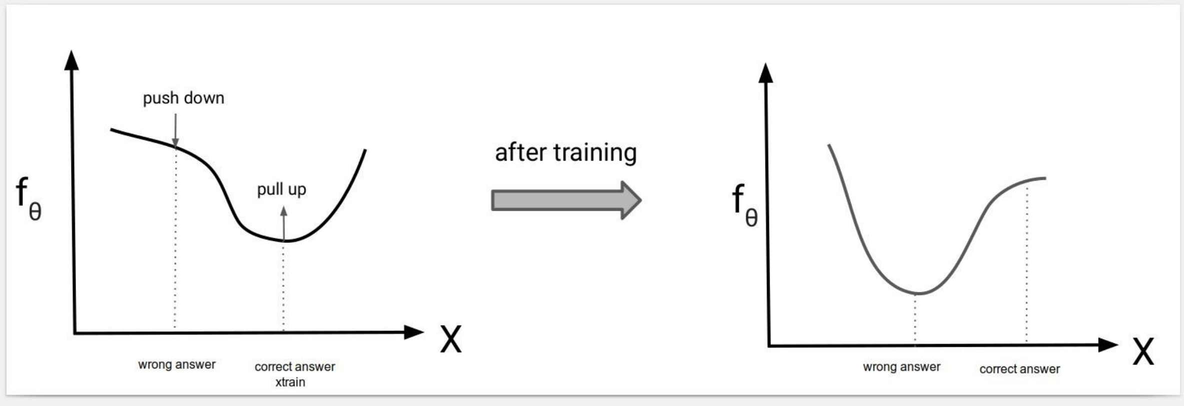
Curse of dimensionality: The fundamental issue is that computing $Z(\theta)$ numerically (when no analytic solution is available) scales exponentially in the number of dimensions of **x**.

Training intuition



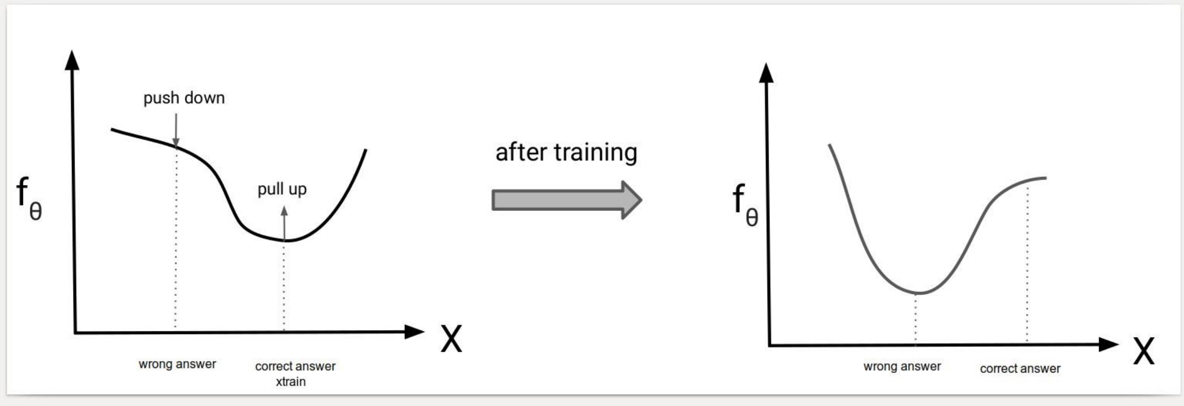
- Goal: maximize $\frac{\exp\{f_\theta(\mathbf{x}_{train})\}}{Z(\theta)}$. Increase numerator, decrease denominator.
- **Intuition**: because the model is not normalized, increasing the un-normalized log-probability $f_\theta(\mathbf{x}_{train})$ by changing θ does **not** guarantee that $\mathbf{x}_{train}$ becomes relatively more likely (compared to the rest).
- We also need to take into account the effect on other "wrong points" and try to "push them down" to *also* make $Z(\theta)$ small.

Contrastive Divergence



- Goal: maximize $\frac{\exp\{f_\theta(x_{train})\}}{Z(\theta)}$
- **Idea**: Instead of evaluating $Z(\theta)$ exactly, use a Monte Carlo estimate.
- **Contrastive divergence algorithm**: sample $x_{sample} \sim p_\theta$, take step on $\nabla_\theta \left(f_\theta(x_{train}) - f_\theta(x_{sample})\right)$. Make training data more likely than typical sample from the model.

Contrastive Divergence



$$
\begin{aligned}
\nabla_\theta \mathcal{L}_{\text{MLE}}(\theta; p) &= -\mathbb{E}_{p(\mathbf{x})}\left[\nabla_\theta \log q_\theta(\mathbf{x})\right] \\
&= \mathbb{E}_{p(\mathbf{x})}\left[\nabla_\theta E_\theta(\mathbf{x})\right] - \mathbb{E}_{q_\theta(\mathbf{x})}\left[\nabla_\theta E_\theta(\mathbf{x})\right]
\end{aligned}
$$

Sampling from Energy-Based Models

Algorithm 1 Sampling from an energy-based model

Sample $\tilde{\boldsymbol{x}}^0$ from a Gaussian or uniform distribution;

for sample step $k = 1$ to K **do** ▷ Generate sample via Langevin dynamics

$\quad \tilde{\boldsymbol{x}}^k \leftarrow \tilde{\boldsymbol{x}}^{k-1} - \eta \nabla_{\boldsymbol{x}} E_\theta(\tilde{\boldsymbol{x}}^{k-1}) + \omega, \quad \text{where } \omega \sim \mathcal{N}(0, \sigma)$

end for

$\boldsymbol{x}_{\text{sample}} \leftarrow \tilde{\boldsymbol{x}}^K$

Refs

Tutorial 8: Deep Energy-Based Generative Models — UvA DL Notebooks v1.2 documentation (uvadlc-notebooks.readthedocs.io)

cs236_lecture11.pdf (deepgenerativemodels.github.io)

CVPR 2021 Tutorial on Theory and Application of Energy-Based Generative Models (energy-based-models.github.io)