Deep Generative Models

10. Energy-Based Models

• 국가수리과학연구소 산업수학혁신센터 김민중

Recap

- Model Families
  - Autoregressive Models: $p_\theta(\boldsymbol{x}) = \prod_{i=1}^{d} p_\theta(x_i|\boldsymbol{x}_{<\boldsymbol{i}})$
  - Variational Autoencoders: $p_\theta(\boldsymbol{x}) = \int p_\theta(\boldsymbol{x}, \boldsymbol{z})d\boldsymbol{z}$
  - Normalizing Flow Models: $p_X(\boldsymbol{x};\theta) = p_Z\left(\boldsymbol{f}_\theta^{-1}(\boldsymbol{x})\right)\left|\det\left(\frac{\partial \boldsymbol{f}_\theta^{-1}(\boldsymbol{x})}{\partial \boldsymbol{x}}\right)\right|$
- All the above families are trained by minimizing KL divergence $D(p_{data} \parallel p_\theta)$ or equivalently maximizing likelihoods (or approximations)

Recap

- Generative Adversarial Networks (GANs)

  $$\min_{\theta} \max_{\phi} E_{\boldsymbol{x} \sim p_{data}}[\log D_{\phi}(\boldsymbol{x})] + E_{\boldsymbol{z} \sim p_{Z}}\left[\log\left(1 - D_{\phi}(G_{\theta}(\boldsymbol{z}))\right)\right]$$

  - Two sample tests
  - (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

Plan

- Energy-based models (EBMs).
  - Very flexible model architectures
  - Stable training
  - Relatively high sample quality
  - Flexible composition

Parameterizing probability distributions

- Probability distributions $p(\boldsymbol{x})$ are a key building block in generative modeling
  - **non-negative**: $p(\boldsymbol{x}) \geq 0$
  - **sum-to-one**: $\sum_x p(\boldsymbol{x})$ (or $\int p(\boldsymbol{x}) d\boldsymbol{x} = 1$ for continuous variables)
- Condition of non-negative function $p_\theta(\boldsymbol{x})$ is not difficult
- Given any function $f_\theta(\boldsymbol{x})$, we can choose
  - $g_\theta(\boldsymbol{x}) = f_\theta(\boldsymbol{x})^2$
  - $g_\theta(\boldsymbol{x}) = \exp f_\theta(\boldsymbol{x})$
  - $g_\theta(\boldsymbol{x}) = |f_\theta(\boldsymbol{x})|$
  - $g_\theta(\boldsymbol{x}) = \log(1 + \exp f_\theta(\boldsymbol{x}))$
  - etc.
  - In general, g_θ is not a normalized function of p_θ

Parameterizing probability distributions

- Probability distributions $p(\boldsymbol{x})$ are a key building block in generative modeling
  - **non-negative**: $p(\boldsymbol{x}) \geq 0$
  - **sum-to-one**: $\sum_x p(\boldsymbol{x})$ (or $\int p(\boldsymbol{x}) d\boldsymbol{x} = 1$ for continuous variables)
- Sum-to-one is key

Parameterizing probability distributions

- Probability distributions $p(\boldsymbol{x})$ are a key building block in generative modeling
  - **non-negative**: $p(\boldsymbol{x}) \geq 0$
  - **sum-to-one**: $\sum_x p(\boldsymbol{x})$ (or $\int p(\boldsymbol{x})d\boldsymbol{x} = 1$ for continuous variables)
- Total "**volume**" is fixed: increasing $p(\boldsymbol{x}_{train})$ guarantees that $\boldsymbol{x}_{train}$ becomes relatively more likely
- Problem:
  - $g_\theta(\boldsymbol{x})$ might not sum-to-one
  - $\sum_x g_\theta(\boldsymbol{x}) =: Z(\theta) \neq 1$ in general, so $g_\theta(\boldsymbol{x})$ is not a valid probability mass function or density

Energy-based model

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

- I.e., $g_\theta(\boldsymbol{x}) = \exp\left(f_\theta(\boldsymbol{x})\right)$
- The volume/normalization constant

$$\int \exp\left(f_\theta(\boldsymbol{x})\right) d\boldsymbol{x}$$

- is also called the partition function

Energy-based model

- **Why exponential** (and not e.g $f_\theta(\boldsymbol{x})^2$)?
  - Want to capture very large variations in probability. log-probability is the natural scale we want to work with Otherwise need highly non-smooth f_θ.
  - Exponential families. Many common distributions can be written in this form
  - These distributions arise under general assumptions in statistical physics (maximum entropy, second law of thermodynamics)
    - $-f_\theta$ is called the energy, hence the name
    - Intuitively, configurations $\boldsymbol{x}$ with low energy (high $f_\theta(\boldsymbol{x})$) are more likely

Idea

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

- I.e., $p_\theta(\boldsymbol{x}) \propto \exp\left(f_\theta(\boldsymbol{x})\right)$
- Given $\boldsymbol{x}$, $\boldsymbol{x}'$ evaluating $p_\theta(\boldsymbol{x})$ or $p_\theta(\boldsymbol{x}')$ requires $Z(\theta)$
- However, the ratio

$$\frac{p_\theta(\boldsymbol{x})}{p_\theta(\boldsymbol{x}')} = \exp\left(f_\theta(\boldsymbol{x}) - f_\theta(\boldsymbol{x}')\right)$$

- does not involve $Z(\theta)$

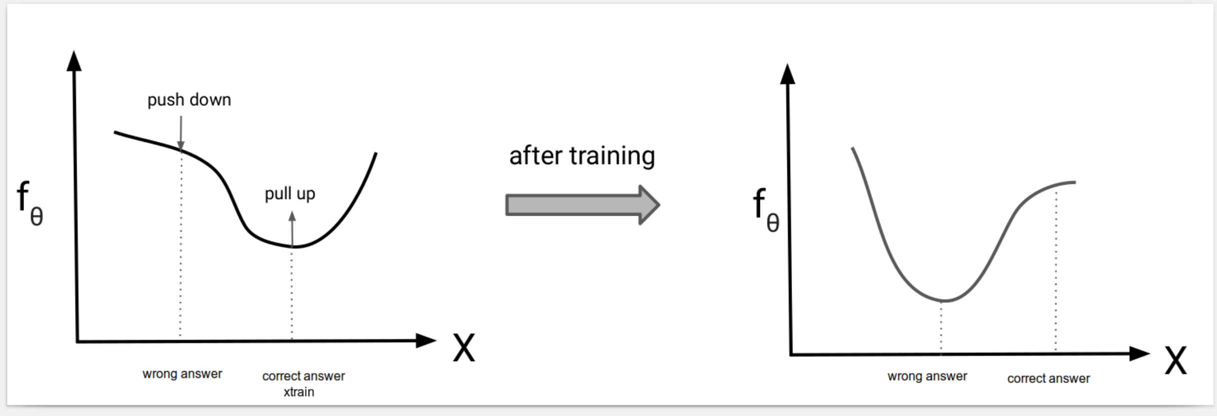
Energy-based model

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

- Pros:
  - Extreme flexibility: can use pretty much any function f_θ
- Cons:
  - Sampling from p_θ is difficult
  - Evaluating and optimizing likelihood p_θ is hard (learning is hard)
  - 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 $\boldsymbol{x}$
- Nevertheless, some tasks do not require knowing $Z(\theta)$

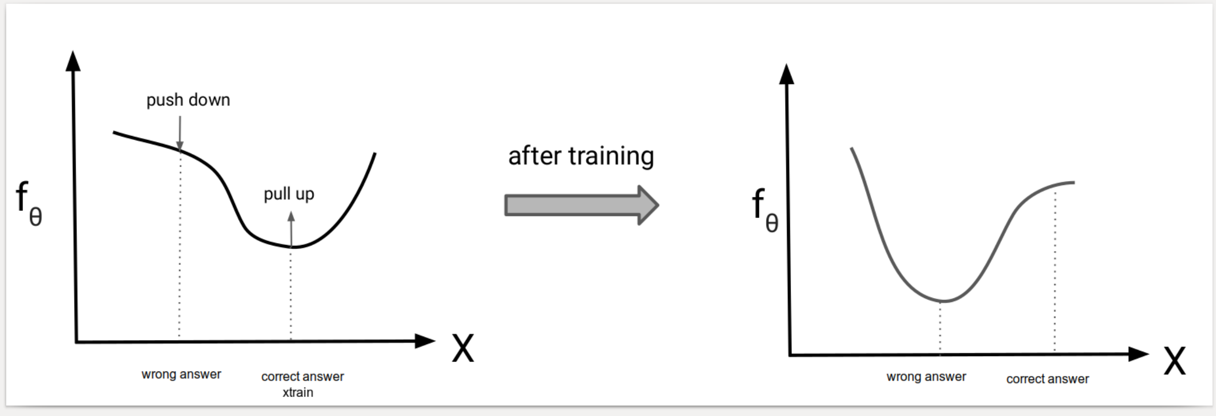
Training intuition

- **Goal**: maximize $\frac{1}{Z(\theta)}\exp\left(f_\theta(\boldsymbol{x}_{train})\right)$
- **Intuition**: because the model is not normalized, increasing the un-normalized log-probability $f_\theta(\boldsymbol{x}_{train})$ by changing θ does not guarantee that $\boldsymbol{x}_{train}$ becomes relatively more likely (compared to the rest)
- We also need to consider the effect on other "wrong points" and try to "push them down" to also make $Z(\theta)$ small



Contrastive Divergence

- Goal: maximize $\frac{1}{Z(\theta)}\exp\left(f_\theta(\boldsymbol{x}_{train})\right)$
- Instead of evaluating $Z(\theta)$ exactly, use a Monte Carlo estimate
- **Contrastive divergence algorithm**
  - Sample $\boldsymbol{x}_{sample} \sim p_\theta$ and maximize $f_\theta(\boldsymbol{x}_{train}) - f_\theta\left(\boldsymbol{x}_{sample}\right)$
  - Take step on $\nabla_\theta\left(f_\theta(\boldsymbol{x}_{train}) - f_\theta\left(\boldsymbol{x}_{sample}\right)\right)$
  - Make training data more likely than typical sample from the model



Contrastive Divergence

- Maximize log-likelihood: $f_\theta(\boldsymbol{x}_{train}) - \log Z(\theta)$
- Gradient of log-likelihood:

$$\nabla_\theta f_\theta(\boldsymbol{x}_{train}) - \nabla_\theta \log Z(\theta) = \nabla_\theta f_\theta(\boldsymbol{x}_{train}) - \frac{\nabla_\theta Z(\theta)}{Z(\theta)}$$

$$= \nabla_\theta f_\theta(\boldsymbol{x}_{train}) - \frac{1}{Z(\theta)} \int \nabla_\theta \exp\left(f_\theta(\boldsymbol{x})\right) d\boldsymbol{x}$$

$$= \nabla_\theta f_\theta(\boldsymbol{x}_{train}) - \frac{1}{Z(\theta)} \int \exp\left(f_\theta(\boldsymbol{x})\right) \nabla_\theta f_\theta(\boldsymbol{x}) d\boldsymbol{x}$$

$$= \nabla_\theta f_\theta(\boldsymbol{x}_{train}) - \int \frac{\exp\left(f_\theta(\boldsymbol{x})\right)}{Z(\theta)} \nabla_\theta f_\theta(\boldsymbol{x}) d\boldsymbol{x}$$

$$= \nabla_\theta f_\theta(\boldsymbol{x}_{train}) - E_{\boldsymbol{x}\sim p_\theta}[\nabla_\theta f_\theta(\boldsymbol{x})]$$

Contrastive Divergence

- Maximize log-likelihood: $\log p_\theta(\boldsymbol{x}_{train}) = f_\theta(\boldsymbol{x}_{train}) - \log Z(\theta)$
- Gradient of log-likelihood:

$$\nabla_\theta f_\theta(\boldsymbol{x}_{train}) - \nabla_\theta \log Z(\theta) = \nabla_\theta f_\theta(\boldsymbol{x}_{train}) - E_{\boldsymbol{x}\sim p_\theta}[\nabla_\theta f_\theta(\boldsymbol{x})]$$
$$\approx \nabla_\theta f_\theta(\boldsymbol{x}_{train}) - \nabla_\theta f_\theta\left(\boldsymbol{x}_{sample}\right)$$

- where $\boldsymbol{x}_{sample} \sim p_\theta(\boldsymbol{x}) = \frac{\exp\left(f_\theta(\boldsymbol{x})\right)}{Z(\theta)}$
- How to sample?

Sampling from Energy-based model

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

- No direct way to sample like in autoregressive or flow models
- **Main issue**: cannot easily compute how likely each possible sample is
- However, we can easily compare two samples $\boldsymbol{x}$, $\boldsymbol{x}'$
- Use an iterative approach called Markov Chain Monte Carlo:
  - Initialize $\boldsymbol{x}^{\boldsymbol{0}}$ randomly, $t = 0$
  - Let $\boldsymbol{x}' = \boldsymbol{x}^t + noise$
    - If $f_\theta(\boldsymbol{x}') \geq f_\theta(\boldsymbol{x}^t)$, let $\boldsymbol{x}^{t+1} = \boldsymbol{x}'$
    - Else let $\boldsymbol{x}^{t+1} = \boldsymbol{x}'$ with probability $\exp\big(f_\theta(\boldsymbol{x}') - f_\theta(\boldsymbol{x}^t)\big)$
- Works in theory, but can take a very long time to converge

Sampling from Energy-based model

- For any continuous distribution $p_\theta(\boldsymbol{x})$, suppose we can compute its gradient (the **score function**) $\nabla_{\boldsymbol{x}} \log p_\theta(\boldsymbol{x})$
- Let $\pi(\boldsymbol{x})$ be a prior distribution that is easy to sample
- **Langevin MCMC**
  - Initialize $\boldsymbol{x}^0 \sim \pi(\boldsymbol{x})$ from prior distribution
  - Repeat $\boldsymbol{x}^{t+1} \sim \boldsymbol{x}^t + \epsilon \nabla_{\boldsymbol{x}} \log p_\theta(\boldsymbol{x}^t) + \sqrt{2\epsilon}\boldsymbol{z}$ for $t = 0, \ldots, T-1$ where $\boldsymbol{z} \sim N(0, I)$
  - If $\epsilon \to 0$ and $T \to \infty$, then we have $\boldsymbol{x}^T \sim p_\theta$
- Note that for energy-based models, the score function is tractable

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

Training on Energy-based model

- Define the function $f_\theta(\boldsymbol{x})$ parametrized by θ
- Find $\boldsymbol{x}_{sample}$ that makes $f_\theta(\boldsymbol{x})$ relatively more likely using Langevin MCMC
  - $\boldsymbol{x}^{t+1} \sim \boldsymbol{x}^t + \epsilon \nabla_{\boldsymbol{x}} f_\theta(\boldsymbol{x}_t) + \sqrt{2\epsilon}\boldsymbol{z}$ for $t = 0, \dots, T-1$ where $\boldsymbol{z} \sim N(0, I)$ where ϵ is the step size
- Update the parameter θ

$$\theta^{t+1} = \theta^t + \eta \nabla_\theta \left(f_\theta(\boldsymbol{x}_{train}) - f_\theta\left(\boldsymbol{x}_{sample}\right) \right)$$

Modern Energy-based model

Figure 1: **Synthesis by short-run MCMC**: Generating synthesized examples by running 100 steps of Langevin dynamics initialized from uniform noise for CelebA (64×64).

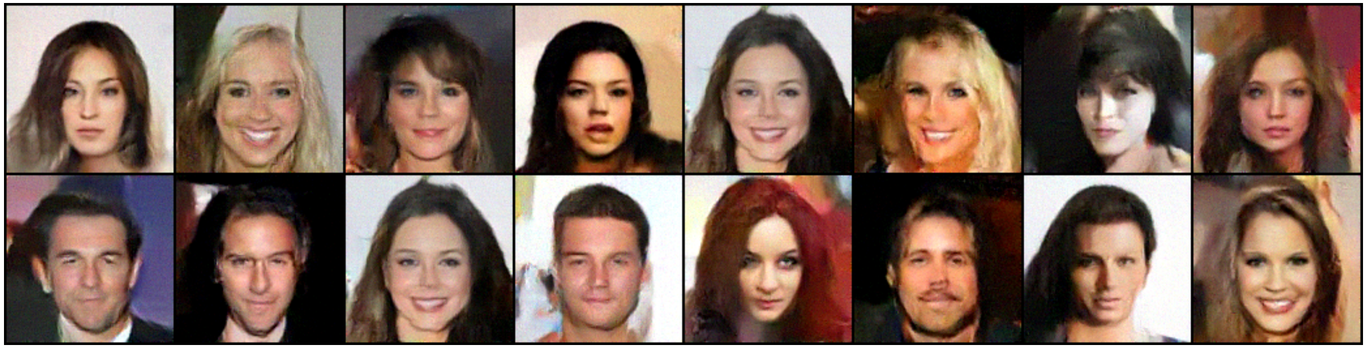


Figure 2: **Synthesis by short-run MCMC**: Generating synthesized examples by running 100 steps of Langevin dynamics initialized from uniform noise for CelebA (128×128).

Source: Nijkamp et al. 2019

Recap. of Energy-based model

- Energy-based models: $\frac{1}{Z(\theta)}\exp\left(f_\theta(\boldsymbol{x})\right)$
  - $Z(\theta)$ is intractable, so no access to likelihood
  - Comparing the probability of two points is easy

$$\frac{p_\theta(\boldsymbol{x})}{p_\theta(\boldsymbol{x}')} = \exp\left(f_\theta(\boldsymbol{x}) - f_\theta(\boldsymbol{x}')\right)$$

- Maximum likelihood training:

$$\max_\theta [f_\theta(\boldsymbol{x}_{train}) - \log Z(\theta)]$$

  - Contrastive divergence:

$$\nabla_\theta f_\theta(\boldsymbol{x}_{train}) - \nabla_\theta \log Z(\theta) \approx \nabla_\theta f_\theta(\boldsymbol{x}_{train}) - \nabla_\theta f_\theta\left(\boldsymbol{x}_{sample}\right)$$

  - where $\boldsymbol{x}_{sample} \sim p_\theta(\boldsymbol{x}) = \frac{\exp\left(f_\theta(\boldsymbol{x})\right)}{Z(\theta)}$

Sampling from Energy-based model

- Trained model f_θ is given
- Let $\pi(\boldsymbol{x})$ be a prior distribution that is easy to sample
- **Langevin MCMC**
  - Initialize $\boldsymbol{x}^0 \sim \pi(\boldsymbol{x})$ from prior distribution
  - Repeat $\boldsymbol{x}^{t+1} \sim \boldsymbol{x}^t + \epsilon \nabla_{\boldsymbol{x}} f_\theta(\boldsymbol{x}^t) + \sqrt{2\epsilon}\boldsymbol{z}$ for $t = 0, \ldots, T-1$ where $\boldsymbol{z} \sim N(0, I)$

Thanks