

Energy-Based Models

Stefano Ermon

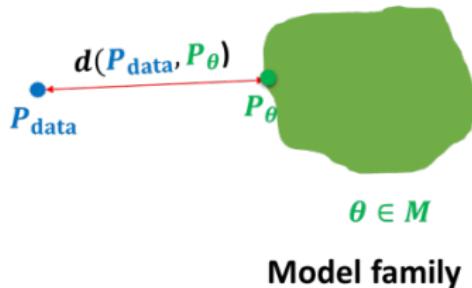
Stanford University

Lecture 12

Recap. of last lecture



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



- Energy-based models: $p_\theta(\mathbf{x}) = \frac{\exp\{f_\theta(\mathbf{x})\}}{Z(\theta)}$.
 - $Z(\theta)$ is intractable, so no access to likelihood.
 - Comparing the probability of two points is easy:
 $p_\theta(\mathbf{x}')/p_\theta(\mathbf{x}) = \exp(f_\theta(\mathbf{x}') - f_\theta(\mathbf{x}))$.
- Maximum likelihood training: $\max_\theta \{f_\theta(\mathbf{x}_{\text{train}}) - \log Z(\theta)\}$.
 - Contrastive divergence:

$$\nabla_\theta f_\theta(\mathbf{x}_{\text{train}}) - \nabla_\theta \log Z(\theta) \approx \nabla_\theta f_\theta(\mathbf{x}_{\text{train}}) - \nabla_\theta f_\theta(\mathbf{x}_{\text{sample}}),$$

where $\mathbf{x}_{\text{sample}} \sim p_\theta(\mathbf{x})$.

Sampling from EBMs: MH-MCMC

Metropolis-Hastings Markov chain Monte Carlo (MCMC).

- ① $\mathbf{x}^0 \sim \pi(\mathbf{x})$
- ② Repeat for $t = 0, 1, 2, \dots, T - 1$:
 - $\mathbf{x}' = \mathbf{x}^t + \text{noise}$
 - $\mathbf{x}^{t+1} = \mathbf{x}'$ if $f_\theta(\mathbf{x}') \geq f_\theta(\mathbf{x}^t)$
 - If $f_\theta(\mathbf{x}') < f_\theta(\mathbf{x}^t)$, set $\mathbf{x}^{t+1} = \mathbf{x}'$ with probability $\exp\{f_\theta(\mathbf{x}') - f_\theta(\mathbf{x}^t)\}$, otherwise set $\mathbf{x}^{t+1} = \mathbf{x}^t$.

Properties:

- In theory, \mathbf{x}^T converges to $p_\theta(\mathbf{x})$ when $T \rightarrow \infty$. Why?
 - Satisfies *detailed balance condition*: $p_\theta(\mathbf{x}) T_{\mathbf{x} \rightarrow \mathbf{x}'} = p_\theta(\mathbf{x}') T_{\mathbf{x}' \rightarrow \mathbf{x}}$ where $T_{\mathbf{x} \rightarrow \mathbf{x}'}$ is the probability of transitioning from \mathbf{x} to \mathbf{x}'
 - If \mathbf{x}^t is distributed as p_θ , then \mathbf{x}^{t+1} is distributed as p_θ .
- In practice, need a large number of iterations and convergence slows down exponentially in dimensionality.

Sampling from EBMs: unadjusted Langevin MCMC

Unadjusted Langevin MCMC:

- ① $\mathbf{x}^0 \sim \pi(\mathbf{x})$
- ② Repeat for $t = 0, 1, 2, \dots, T - 1$:
 - $\mathbf{z}^t \sim \mathcal{N}(0, I)$
 - $\mathbf{x}^{t+1} = \mathbf{x}^t + \epsilon \nabla_{\mathbf{x}} \log p_{\theta}(\mathbf{x})|_{\mathbf{x}=\mathbf{x}^t} + \sqrt{2\epsilon} \mathbf{z}^t$

Properties:

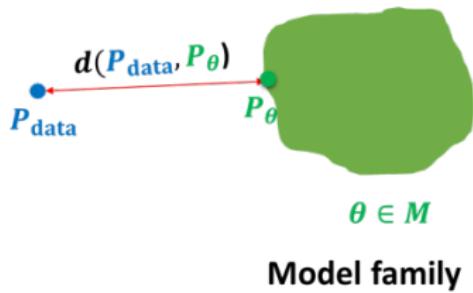
- \mathbf{x}^T converges to a sample from $p_{\theta}(\mathbf{x})$ when $T \rightarrow \infty$ and $\epsilon \rightarrow 0$.
- $\nabla_{\mathbf{x}} \log p_{\theta}(\mathbf{x}) = \nabla_{\mathbf{x}} f_{\theta}(\mathbf{x})$ for continuous energy-based models.
- Convergence slows down as dimensionality grows.

Sampling converges slowly in high dimensional spaces and is thus very expensive, yet we need sampling for **each training iteration** in contrastive divergence.

Today's lecture



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



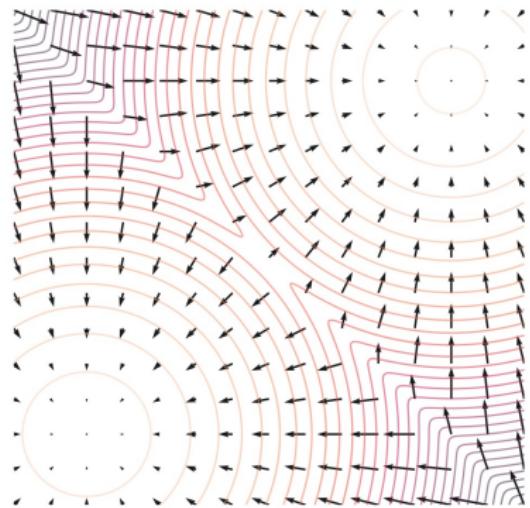
Goal: Training without sampling

- Score Matching
- Noise Contrastive Estimation
- Adversarial training

Score function

Energy-based model: $p_\theta(\mathbf{x}) = \frac{\exp\{f_\theta(\mathbf{x})\}}{Z(\theta)}$, $\log p_\theta(\mathbf{x}) = f_\theta(\mathbf{x}) - \log Z(\theta)$
(Stein) Score function:

$$s_\theta(\mathbf{x}) := \nabla_{\mathbf{x}} \log p_\theta(\mathbf{x}) = \nabla_{\mathbf{x}} f_\theta(\mathbf{x}) - \underbrace{\nabla_{\mathbf{x}} \log Z(\theta)}_{=0} = \nabla_{\mathbf{x}} f_\theta(\mathbf{x})$$



- Gaussian distribution
 $p_\theta(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$
 $\longrightarrow s_\theta(x) = -\frac{x-\mu}{\sigma^2}$
- Gamma distribution
 $p_\theta(x) = \frac{\beta^\alpha}{\Gamma(\alpha)} x^{\alpha-1} e^{-\beta x}$
 $\longrightarrow s_\theta(x) = \frac{\alpha-1}{x} - \beta$

Score matching

Observation

$s_\theta(\mathbf{x}) = \nabla_{\mathbf{x}} \log p_\theta(\mathbf{x})$ is independent of the partition function $Z(\theta)$.

Fisher divergence between $p(\mathbf{x})$ and $q(\mathbf{x})$:

$$D_F(p, q) := \frac{1}{2} E_{\mathbf{x} \sim p} [\|\nabla_{\mathbf{x}} \log p(\mathbf{x}) - \nabla_{\mathbf{x}} \log q(\mathbf{x})\|_2^2]$$

Score matching: minimizing the Fisher divergence between $p_{\text{data}}(\mathbf{x})$ and the EBM $p_\theta(\mathbf{x}) \propto \exp\{f_\theta(\mathbf{x})\}$

$$\begin{aligned} & \frac{1}{2} E_{\mathbf{x} \sim p_{\text{data}}} [\|\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x}) - s_\theta(\mathbf{x})\|_2^2] \\ &= \frac{1}{2} E_{\mathbf{x} \sim p_{\text{data}}} [\|\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x}) - \nabla_{\mathbf{x}} f_\theta(\mathbf{x})\|_2^2] \end{aligned}$$

Score matching

$$\frac{1}{2} E_{\mathbf{x} \sim p_{\text{data}}} [\|\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x}) - \nabla_{\mathbf{x}} \log p_{\theta}(\mathbf{x})\|_2^2]$$

How to deal with $\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x})$ given only samples? Integration by parts!

$$\begin{aligned} & \frac{1}{2} E_{\mathbf{x} \sim p_{\text{data}}} [(\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x}) - \nabla_{\mathbf{x}} \log p_{\theta}(\mathbf{x}))^2] \quad (\text{Univariate case}) \\ &= \frac{1}{2} \int p_{\text{data}}(\mathbf{x}) [(\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x}) - \nabla_{\mathbf{x}} \log p_{\theta}(\mathbf{x}))^2] d\mathbf{x} \\ &= \frac{1}{2} \int p_{\text{data}}(\mathbf{x}) (\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x}))^2 d\mathbf{x} + \frac{1}{2} \int p_{\text{data}}(\mathbf{x}) (\nabla_{\mathbf{x}} \log p_{\theta}(\mathbf{x}))^2 d\mathbf{x} \\ &\quad - \int p_{\text{data}}(\mathbf{x}) \nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x}) \nabla_{\mathbf{x}} \log p_{\theta}(\mathbf{x}) d\mathbf{x} \end{aligned}$$

Recall Integration by parts: $\int f'g = fg - \int g'f$.

$$\begin{aligned} & - \int p_{\text{data}}(\mathbf{x}) \nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x}) \nabla_{\mathbf{x}} \log p_{\theta}(\mathbf{x}) d\mathbf{x} \\ &= - \int p_{\text{data}}(\mathbf{x}) \frac{1}{p_{\text{data}}(\mathbf{x})} \nabla_{\mathbf{x}} p_{\text{data}}(\mathbf{x}) \nabla_{\mathbf{x}} \log p_{\theta}(\mathbf{x}) d\mathbf{x} \\ &= \underbrace{- p_{\text{data}}(\mathbf{x}) \nabla_{\mathbf{x}} \log p_{\theta}(\mathbf{x}) \big|_{x=-\infty}^{\infty}}_{=0} + \int p_{\text{data}}(\mathbf{x}) \nabla_{\mathbf{x}}^2 \log p_{\theta}(\mathbf{x}) d\mathbf{x} \\ &= \int p_{\text{data}}(\mathbf{x}) \nabla_{\mathbf{x}}^2 \log p_{\theta}(\mathbf{x}) d\mathbf{x} \end{aligned}$$

Note: we need to assume p_{data} decays sufficiently rapidly, $p_{\text{data}}(\mathbf{x}) \rightarrow 0$ when $\mathbf{x} \rightarrow \pm\infty$.

Score matching

Univariate score matching

$$\begin{aligned} & \frac{1}{2} E_{x \sim p_{\text{data}}} [(\nabla_x \log p_{\text{data}}(x) - \nabla_x \log p_{\theta}(x))^2] \\ &= \frac{1}{2} \int p_{\text{data}}(x) (\nabla_x \log p_{\text{data}}(x))^2 dx + \frac{1}{2} \int p_{\text{data}}(x) (\nabla_x \log p_{\theta}(x))^2 dx \\ & \quad - \int p_{\text{data}}(x) \nabla_x \log p_{\text{data}}(x) \nabla_x \log p_{\theta}(x) dx \\ &= \underbrace{\frac{1}{2} \int p_{\text{data}}(x) (\nabla_x \log p_{\text{data}}(x))^2 dx}_{\text{const. wrt } \theta} + \frac{1}{2} \int p_{\text{data}}(x) (\nabla_x \log p_{\theta}(x))^2 dx \\ & \quad + \int p_{\text{data}}(x) \nabla_x^2 \log p_{\theta}(x) dx \\ &= E_{x \sim p_{\text{data}}} \left[\frac{1}{2} (\nabla_x \log p_{\theta}(x))^2 + \nabla_x^2 \log p_{\theta}(x) \right] + \text{const.} \end{aligned}$$

Multivariate score matching (integration by parts, i.e. Gauss theorem)

$$\begin{aligned} & \frac{1}{2} E_{x \sim p_{\text{data}}} [\|\nabla_x \log p_{\text{data}}(x) - \nabla_x \log p_{\theta}(x)\|_2^2] \\ &= E_{x \sim p_{\text{data}}} \left[\frac{1}{2} \|\nabla_x \log p_{\theta}(x)\|_2^2 + \text{tr} \left(\underbrace{\nabla_x^2 \log p_{\theta}(x)}_{\text{Hessian of } \log p_{\theta}(x)} \right) \right] + \text{const.} \end{aligned}$$

Score matching

- ① Sample a mini-batch of datapoints $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\} \sim p_{\text{data}}(\mathbf{x})$.
- ② Estimate the score matching loss with the empirical mean

$$\begin{aligned} & \frac{1}{n} \sum_{i=1}^n \left[\frac{1}{2} \|\nabla_{\mathbf{x}} \log p_{\theta}(\mathbf{x}_i)\|_2^2 + \text{tr}(\nabla_{\mathbf{x}}^2 \log p_{\theta}(\mathbf{x}_i)) \right] \\ &= \frac{1}{n} \sum_{i=1}^n \left[\frac{1}{2} \|\nabla_{\mathbf{x}} f_{\theta}(\mathbf{x}_i)\|_2^2 + \text{trace}(\nabla_{\mathbf{x}}^2 f_{\theta}(\mathbf{x}_i)) \right] \end{aligned}$$

- ③ Stochastic gradient descent.
- ④ No need to sample from the EBM!

Caveat

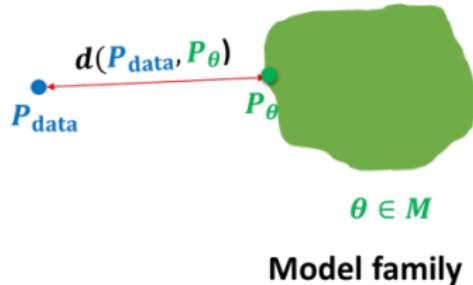
Computing the trace of Hessian $\text{tr}(\nabla_{\mathbf{x}}^2 \log p_{\theta}(\mathbf{x}))$ is in general very expensive for large models.

Denoising score matching (Vincent 2010) and sliced score matching (Song et al. 2019). More on this in the next lecture!

Recap.



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



Distances used for training energy-based models.

- KL divergence = maximum likelihood.

$$\nabla_{\theta} f_{\theta}(\mathbf{x}_{\text{data}}) - f_{\theta}(\mathbf{x}_{\text{sample}}) \quad (\text{contrastive divergence})$$

- Fisher divergence = score matching.

$$\frac{1}{2} E_{\mathbf{x} \sim p_{\text{data}}} [\|\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x}) - \nabla_{\mathbf{x}} f_{\theta}(\mathbf{x})\|_2^2]$$

Noise contrastive estimation

Learning an energy-based model by contrasting it with a noise distribution.

- Data distribution: $p_{\text{data}}(\mathbf{x})$.
- Noise distribution: $p_n(\mathbf{x})$. Should be analytically tractable and easy to sample from.
- Training a discriminator $D_\theta(\mathbf{x}) \in [0, 1]$ to distinguish between data samples and noise samples.

$$\max_{\theta} E_{\mathbf{x} \sim p_{\text{data}}} [\log D_\theta(\mathbf{x})] + E_{\mathbf{x} \sim p_n} [\log(1 - D_\theta(\mathbf{x}))]$$

- What is the Optimal discriminator $D_{\theta^*}(\mathbf{x})$?

$$D_{\theta^*}(\mathbf{x}) = \frac{p_{\text{data}}(\mathbf{x})}{p_{\text{data}}(\mathbf{x}) + p_n(\mathbf{x})}$$

Noise contrastive estimation

What if the discriminator is parameterized by

$$D_\theta(\mathbf{x}) = \frac{p_\theta(\mathbf{x})}{p_\theta(\mathbf{x}) + p_n(\mathbf{x})}$$

- The optimal discriminator $D_{\theta^*}(\mathbf{x})$ satisfies

$$D_{\theta^*}(\mathbf{x}) = \frac{p_{\theta^*}(\mathbf{x})}{p_{\theta^*}(\mathbf{x}) + p_n(\mathbf{x})} = \frac{p_{\text{data}}(\mathbf{x})}{p_{\text{data}}(\mathbf{x}) + p_n(\mathbf{x})}$$

- By training the discriminator, we are implicitly learning $p_{\theta^*}(\mathbf{x}) \approx p_{\text{data}}(\mathbf{x})$. Particularly suitable for cases where $p_\theta(\mathbf{x})$ is defined up to a normalization constant (EBMs)
- Equivalently,

$$p_{\theta^*}(\mathbf{x}) = \frac{p_n(\mathbf{x})D_{\theta^*}(\mathbf{x})}{1 - D_{\theta^*}(\mathbf{x})} = p_{\text{data}}(\mathbf{x})$$

Classifier is used to correct density estimates from p_n . Can be used to improve a base generative model (*Boosted Generative Models*, Grover et al., 2018)

Noise contrastive estimation for training EBMs

Energy-based model:

$$p_\theta(\mathbf{x}) = \frac{e^{f_\theta(\mathbf{x})}}{Z(\theta)}$$

The constraint $Z(\theta) = \int e^{f_\theta(\mathbf{x})} d\mathbf{x}$ is hard to satisfy.

Solution: Modeling $Z(\theta)$ with an additional trainable parameter Z that is not explicitly constrained to satisfy $Z = \int e^{f_\theta(\mathbf{x})} d\mathbf{x}$.

$$p_{\theta, Z}(\mathbf{x}) = \frac{e^{f_\theta(\mathbf{x})}}{Z}$$

With noise contrastive estimation, the optimal parameters θ^*, Z^* are

$$p_{\theta^*, Z^*}(\mathbf{x}) = \frac{e^{f_{\theta^*}(\mathbf{x})}}{Z^*} = p_{\text{data}}(\mathbf{x})$$

The optimal parameter Z^* is the correct partition function, because

$$\int \frac{e^{f_{\theta^*}(\mathbf{x})}}{Z^*} d\mathbf{x} = \int p_{\text{data}}(\mathbf{x}) d\mathbf{x} = 1 \implies Z^* = \int e^{f_{\theta^*}(\mathbf{x})} d\mathbf{x}$$

Noise contrastive estimation for training EBMs

The discriminator $D_{\theta, Z}(\mathbf{x})$ for probabilistic model $p_{\theta, Z}(\mathbf{x})$ is

$$D_{\theta, Z}(\mathbf{x}) = \frac{\frac{e^{f_{\theta}(\mathbf{x})}}{Z}}{\frac{e^{f_{\theta}(\mathbf{x})}}{Z} + p_n(\mathbf{x})} = \frac{e^{f_{\theta}(\mathbf{x})}}{e^{f_{\theta}(\mathbf{x})} + p_n(\mathbf{x})Z}$$

Noise contrastive estimation training

$$\max_{\theta, Z} E_{\mathbf{x} \sim p_{\text{data}}} [\log D_{\theta, Z}(\mathbf{x})] + E_{\mathbf{x} \sim p_n} [\log(1 - D_{\theta, Z}(\mathbf{x}))]$$

Equivalently,

$$\begin{aligned} \max_{\theta, Z} E_{\mathbf{x} \sim p_{\text{data}}} & [f_{\theta}(\mathbf{x}) - \log(e^{f_{\theta}(\mathbf{x})} + Zp_n(\mathbf{x}))] \\ & + E_{\mathbf{x} \sim p_n} [\log(Zp_n(\mathbf{x})) - \log(e^{f_{\theta}(\mathbf{x})} + Zp_n(\mathbf{x}))] \end{aligned}$$

Log-sum-exp trick for numerical stability:

$$\begin{aligned} \log(e^{f_{\theta}(\mathbf{x})} + Zp_n(\mathbf{x})) &= \log(e^{f_{\theta}(\mathbf{x})} + e^{\log Z + \log p_n(\mathbf{x})}) \\ &= \text{logsumexp}(f_{\theta}(\mathbf{x}), \log Z + \log p_n(\mathbf{x})) \end{aligned}$$

Noise contrastive estimation for training EBMs

- ① Sample a mini-batch of datapoints $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n \sim p_{\text{data}}(\mathbf{x})$.
- ② Sample a mini-batch of noise samples $\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_n \sim p_n(\mathbf{y})$.
- ③ Estimate the NCE loss.

$$\frac{1}{n} \sum_{i=1}^n [f_\theta(\mathbf{x}_i) - \text{logsumexp}(f_\theta(\mathbf{x}_i), \log Z + \log p_n(\mathbf{x}_i)) \\ + \log Z + p_n(\mathbf{y}_i) - \text{logsumexp}(f_\theta(\mathbf{y}_i), \log Z + \log p_n(\mathbf{y}_i))]$$

- ④ Stochastic gradient ascent with respect to θ and Z .
- ⑤ No need to sample from the EBM!

Comparing NCE and GAN

Similarities:

- Both involve training a discriminator to perform binary classification with a cross-entropy loss.
- Both are likelihood-free (recall likelihood not tractable in EBM).

Differences:

- GAN requires adversarial training or minimax optimization for training, while NCE does not.
- NCE requires the likelihood of the noise distribution for training, while GAN only requires efficient sampling from the prior.
- NCE trains an energy-based model, while GAN trains a deterministic sample generator.

Flow contrastive estimation (Gao et al. 2020)

Observations:

- We need to both evaluate the probability of $p_n(\mathbf{x})$, and sample from it efficiently.
- We hope to make the classification task as hard as possible, i.e., $p_n(\mathbf{x})$ should be close to $p_{\text{data}}(\mathbf{x})$ (but not exactly the same).

Flow contrastive estimation:

- Parameterize the noise distribution with a normalizing flow model $p_{n,\phi}(\mathbf{x})$.
- Parameterize the discriminator $D_{\theta,Z,\phi}(\mathbf{x})$ as

$$D_{\theta,Z,\phi}(\mathbf{x}) = \frac{\frac{e^{f_\theta(\mathbf{x})}}{Z}}{\frac{e^{f_\theta(\mathbf{x})}}{Z} + p_{n,\phi}(\mathbf{x})} = \frac{e^{f_\theta(\mathbf{x})}}{e^{f_\theta(\mathbf{x})} + p_{n,\phi}(\mathbf{x})Z}$$

- Train the flow model to minimize $D_{JS}(p_{\text{data}}, p_{n,\phi})$:

$$\min_{\phi} \max_{\theta, Z} E_{\mathbf{x} \sim p_{\text{data}}} [\log D_{\theta,Z,\phi}(\mathbf{x})] + E_{\mathbf{x} \sim p_{n,\phi}} [\log(1 - D_{\theta,Z,\phi}(\mathbf{x}))]$$

Flow contrastive estimation (Gao et al. 2020)



Samples from SVHN, CIFAR-10, and CelebA datasets.

Image source: Gao et al. 2020.