Introduction to Deep Generative Modeling

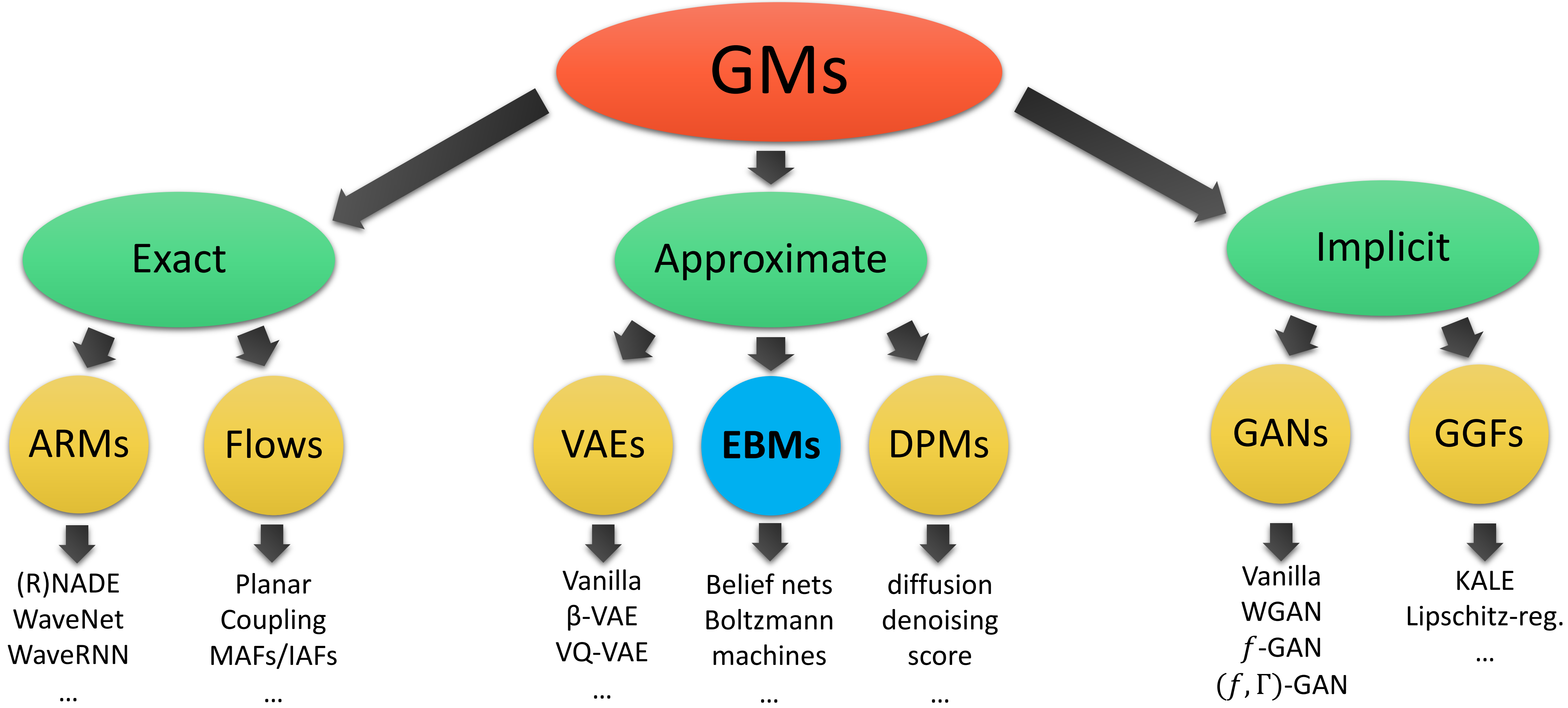
Lecture #12

HY-673 – Computer Science Dep., University of Crete

Professors: Yannis Pantazis, Yannis Stylianou

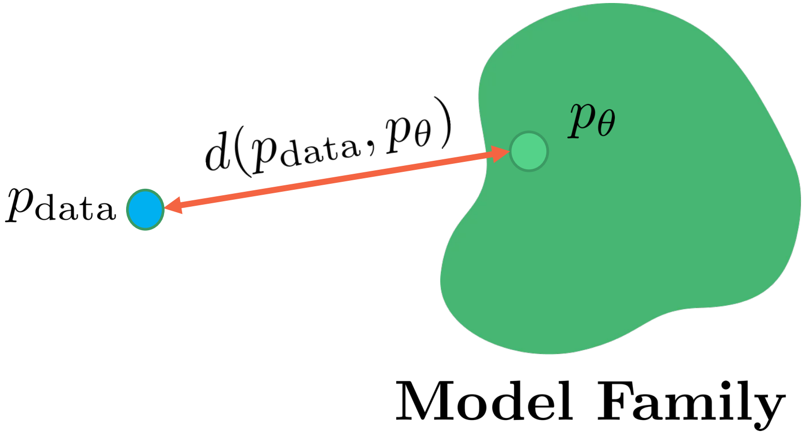
Teaching Assistant: Michail Raptakis

Taxonomy of GMs



Recap. of Last Lecture

$$x_i \sim p_{\text{data}}$$
$$i = 1, 2, \ldots, n$$



- <u>Energy-based models:</u>

$$p_\theta(x) = \frac{1}{Z_\theta} \exp\left(f_\theta(x)\right).$$

- Z_θ is intractable, so no access to likelihood.
- Comparing the probability of two points is easy: $p_\theta(x')/p_\theta(x) = \exp(f_\theta(x') - f_\theta(x))$.
- Maximum likelihood training: $\max_\theta \left\{ f_\theta(x_{\text{train}}) - \log Z_\theta \right\}$.
- Contractive divergence: $\nabla_\theta f_\theta(x_{\text{train}}) - \nabla_\theta \log Z_\theta \approx \nabla_\theta f_\theta(x_{\text{sample}})$,
  where $x_{\text{sample}} \sim p_\theta(x)$.

Sampling from EBMs: MH-MCMC

- Metropolis-Hastings Markov chain Monte Carlo (MCMC).

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

Properties:

- In theory, x^t converges to $p_\theta(x)$ as $t \to \infty$.
- 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. $x^0 \sim \pi(x)$
2. Repeat for $t = 0, 1, 2, \cdots, T-1$:
   - $z^t \sim \mathcal{N}(0, I)$
   - $x^{t+1} = x^t + \epsilon \nabla_x \log p_\theta(x^t) + \sqrt{2\epsilon} z^t$

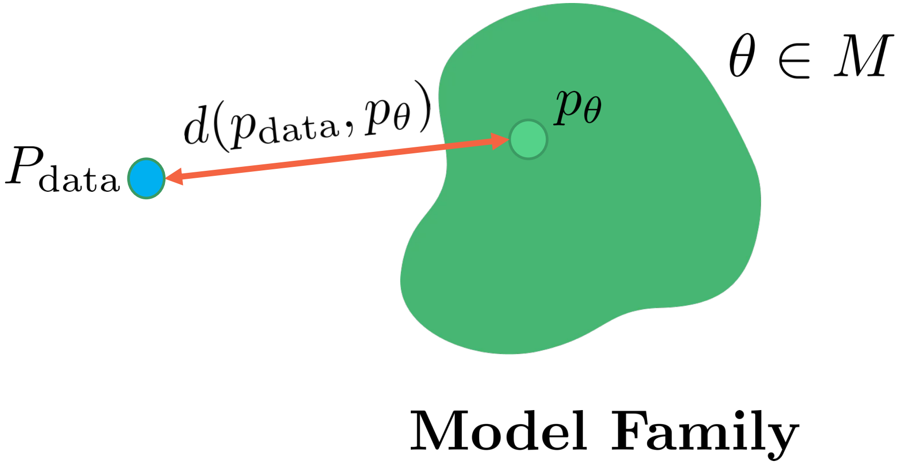
Properties:

- x^t converges to $p_\theta(x)$ as $t \to \infty$ and $\epsilon \to 0$.
- $\nabla_x \log p_\theta(x) = \nabla_x f_\theta(x)$ for continuous energy-based models.
- Convergence slows down as dimensonality 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

$$x_i \sim p_{\text{data}}$$
$$i = 1, 2, \ldots, n$$



Goal: Training without sampling

- Score matching
- Noise Contrastive Estimation

Score Function

Energy-Based model: $p_\theta(x) = \frac{1}{Z_\theta} \exp\{f_\theta(x)\}$

(Stein) Score function:

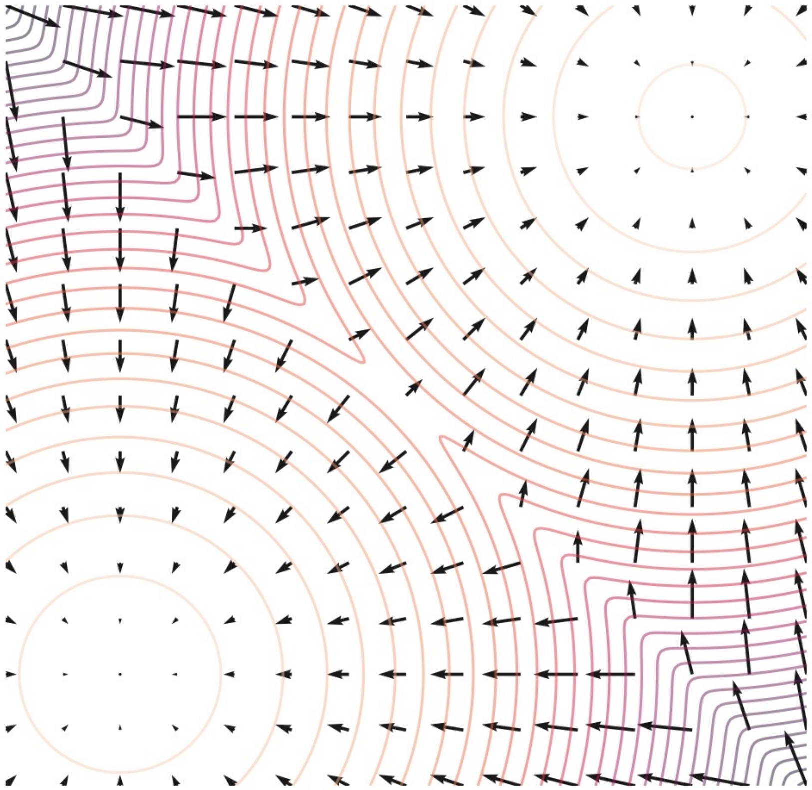
$$s_\theta(x) := \nabla_x \log p_\theta(x) = \nabla_x f_\theta(x) - \underbrace{\nabla_x \log Z_\theta}_{=0} = \nabla_x f_\theta(x)$$

- 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:

$$p_\theta = \frac{\beta^\alpha}{\Gamma(\alpha)} x^{\alpha-1} e^{-\beta x} \longrightarrow s_\theta(x) = \frac{\alpha-1}{x} - \beta$$



$p_\theta(x)$ vs. $s_\theta(x)$

Score Matching

- *Observation:*

$$s_\theta(x) = \nabla_x \log p_\theta(x) \text{ is independent of the partition function } Z_\theta.$$

- Fisher divergence between $p(x)$ and $q(x)$:

$$D_F(p||q) := \frac{1}{2}\mathbb{E}_{x\sim p}\left[\|\nabla_x \log p(x) - \nabla_x \log q(x)\|_2^2\right].$$

- **Score matching:** minimizing the Fisher divergence between $p_{\text{data}}(x)$ and the EBM $p_\theta(x)$:

$$\frac{1}{2}\mathbb{E}_{x\sim p_{\text{data}}}\left[\|\nabla_x \log p_{\text{data}}(x) - s_\theta(x)\|_2^2\right]$$

$$= \frac{1}{2}\mathbb{E}_{x\sim p_{\text{data}}}\left[\|\nabla_x \log p_{\text{data}}(x) - \nabla_x f_\theta(x)\|_2^2\right].$$

Score Matching

- How to deal with $\nabla_x \log p_{\text{data}}(x)$?
  <u>*Answer:*</u> via Integration by Parts!
- For the univariate case:

$$\frac{1}{2}\mathbb{E}_{x\sim p_{\text{data}}}\left[((\log p_{\text{data}}(x))' - s_\theta(x))^2\right]$$

$$= \frac{1}{2}\int_{-\infty}^{\infty} p_{\text{data}}(x)((\log p_{\text{data}}(x))' - s_\theta(x))^2 dx$$

$$= \frac{1}{2}\int_{-\infty}^{\infty} p_{\text{data}}(x)((\log p_{data}(x))')^2 dx + \frac{1}{2}\int_{-\infty}^{\infty} p_{\text{data}}(x) s_\theta^2(x) dx$$
$$- \int_{-\infty}^{\infty} p_{\text{data}}(x)(\log p_{\text{data}}(x))' s_\theta(x) dx.$$

Score Matching

- The cross-correlation term is rewritten via interation-by-parts as:

$$-\int_{-\infty}^{\infty} p_{\text{data}}(x)(\log p_{\text{data}}(x))' s_\theta(x)dx$$

$$= -\int_{-\infty}^{\infty} p_{\text{data}}(x)\frac{1}{p_{\text{data}}(x)}p'_{\text{data}}(x)s_\theta(x)dx$$

$$= \underbrace{-p_{\text{data}}(x)s_\theta(x)\,|_{x=-\infty}^{\infty}}_{=0} + \int_{-\infty}^{\infty} p_{\text{data}}(x)s'_\theta(x)dx$$

$$= \int_{-\infty}^{\infty} p_{\text{data}}(x)s'_\theta(x)dx.$$

Score Matching

- Univariate score matching:

$$\frac{1}{2}\mathbb{E}_{x\sim p_{\text{data}}}\left[((\log p_{\text{data}}(x))' - s_\theta(x))^2\right]$$

$$= \frac{1}{2}\int_{-\infty}^{\infty} p_{\text{data}}(x)((\log p_{data}(x))')^2 dx + \frac{1}{2}\int_{-\infty}^{\infty} p_{\text{data}}(x) s_\theta^2(x) dx$$
$$- \int_{-\infty}^{\infty} p_{\text{data}}(x)(\log p_{\text{data}}(x))' s_\theta(x) dx$$

$$= \frac{1}{2}\int_{-\infty}^{\infty} p_{\text{data}}(x)((\log p_{data}(x))')^2 dx + \frac{1}{2}\int_{-\infty}^{\infty} p_{\text{data}}(x) s_\theta^2(x) dx$$
$$+ \int_{-\infty}^{\infty} p_{\text{data}}(x) s_\theta'(x) dx$$

$$= \text{const} + \mathbb{E}_{x\sim p_{\text{data}}}\left[\frac{1}{2} s_\theta^2(x) + s_\theta'(x)\right].$$

Score Matching

- Multivariate score matching:

$$s_\theta(x) = \nabla_x \log p_\theta(x).$$

$$\frac{1}{2}\mathbb{E}_{x\sim p_{\text{data}}}\left[\|\nabla_x \log p_{\text{data}}(x) - \nabla_x \log p_\theta(x)\|_2^2\right]$$

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

Trace operator
(sum of all diagonal elements of a matrix)

Hessian of $\log p_\theta(x)$

Score Matching – Training Algorithm

1. Sample a mini-batch of datapoints $\{x_1, x_2, \cdots, x_m\} \sim p_{\text{data}}(x)$.
2. Estimate the score matching loss with the empirical mean:

$$\frac{1}{m}\sum_{i=1}^{m}\left[\frac{1}{2}\left\|\nabla_x \log p_\theta(x_i)\right\|_2^2 + \operatorname{tr}(\nabla_x^2 \log p_\theta(x_i))\right]$$

$$\frac{1}{m}\sum_{i=1}^{m}\left[\frac{1}{2}\left\|\nabla_x f_\theta(x_i)\right\|_2^2 + \operatorname{tr}(\nabla_x^2 f_\theta(x_i))\right].$$

- Trained via stochastic gradient descent. No need to sample from the EBM!
- <u>Caveat</u>: Computing the trace of Hessian $\operatorname{tr}(\nabla_x^2 \log p_\theta(x))$ is in general very expensive for large models.
- Denoising score matching (Vincent 2011) and sliced score matching (Song et al. 2019).

Score Matching for Learning Implicit VAEs

- **Model:** $p(\boldsymbol{z}), p_\theta(\boldsymbol{x} \mid \boldsymbol{z}), q_\phi(\boldsymbol{z} \mid \boldsymbol{x}) = \delta(\boldsymbol{z} = f_\phi(\boldsymbol{x}, \epsilon))$.

- **Goal:** maximize the evidence lower bound (ELBO):

$$\mathbb{E}_{\boldsymbol{z} \sim q_\phi(\boldsymbol{z}|\boldsymbol{x})} [\log p_\theta(\boldsymbol{x} \mid \boldsymbol{z}) p(\boldsymbol{z})] \underbrace{- \mathbb{E}_{\boldsymbol{z} \sim q_\phi(\boldsymbol{z}|\boldsymbol{x})} \log q_\phi(\boldsymbol{z} \mid \boldsymbol{x})}_{:= H(q_\phi(\boldsymbol{z} \mid \boldsymbol{x}))}.$$

- Estimate the gradient of the entropy term by training an energy-based model.

Score Matching for Learning Implicit VAEs

$$\nabla_\phi H(q_\phi(\boldsymbol{z} \mid \boldsymbol{x}))$$

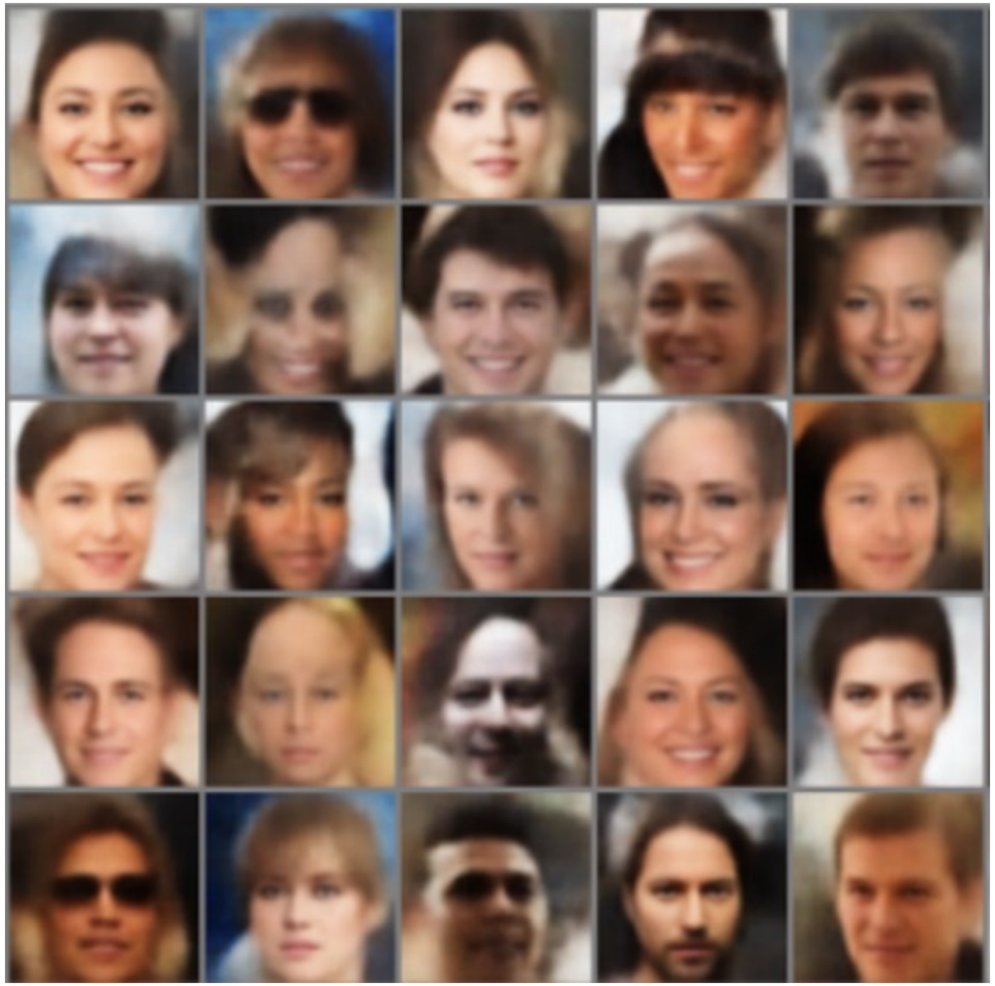
$$= -\nabla_\phi \mathbb{E}_{\boldsymbol{z} \sim q_\phi(\boldsymbol{z}|\boldsymbol{x})} [\log q_\phi(\boldsymbol{z} \mid \boldsymbol{x})]$$

$$= -\nabla_\phi \mathbb{E}_\epsilon [\log q_\phi(f_\phi(\boldsymbol{x}, \epsilon) \mid \boldsymbol{x})] = -\mathbb{E}_\epsilon [\nabla_\phi \log q_\phi(f_\phi(\boldsymbol{x}, \epsilon) \mid \boldsymbol{x})]$$

$$= -\mathbb{E}_\epsilon \left[\underbrace{\nabla_{\boldsymbol{z}} \log q_\phi(\boldsymbol{z} \mid \boldsymbol{x}) \mid_{\boldsymbol{z}=f_\phi(\boldsymbol{x},\epsilon)}} \nabla_\phi f_\phi(\boldsymbol{x}, \epsilon) \right].$$

Score function of $q_\phi(\boldsymbol{z} \mid \boldsymbol{x})$.

Score Matching for Learning Implicit VAEs



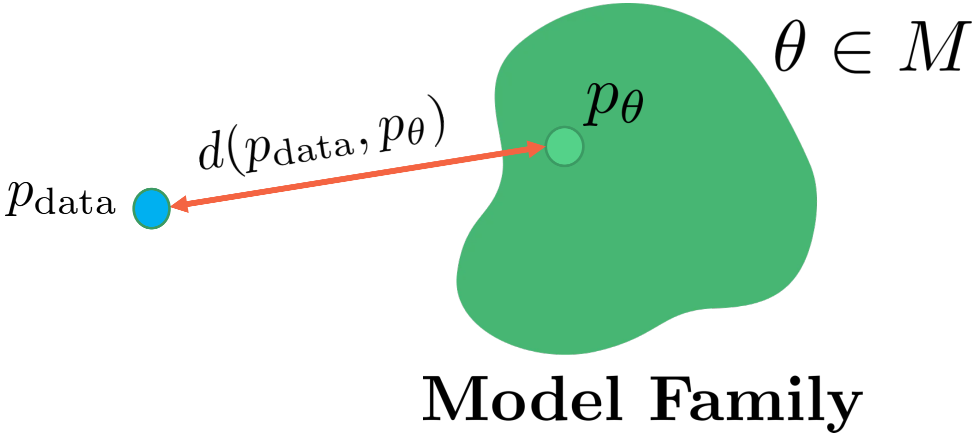
Samples on CelebA 64×64.

Image source: Song et al., 2019.

Recap

$x_i \sim p_{\text{data}}$

$i = 1, 2, \ldots, n$



Distances used for training energy-based models:

- KL divergence minimization $\Longleftrightarrow$ maximum likelihood maximization.

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

- Fisher divergence minimization $\Longleftrightarrow$ score matching.

$$\frac{1}{2}\mathbb{E}_{x \sim p_{\text{data}}}\left[\|\nabla_x \log p_{\text{data}}(x) - \nabla_x f_\theta(x)\|_2^2\right].$$

Noise Contrastive Estimation (NCE)

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

- *Data distribution:* $p_{\mathrm{data}}(x)$.
- *Noise distribution:* $p_n(x)$.
  It should be analytically tractable and easy to sample from.
- Train a discriminator (binary classifier) $D(x) \in [0, 1]$ to distinguish between data sample and noise samples via MLE:

$$\max_D \; \mathbb{E}_{x \sim p_{\mathrm{data}}} [\log D(x)] + \mathbb{E}_{x \sim p_n} [\log(1 - D(x))] \, .$$

- Given enough capacity, the *optimal discriminator* is given by:

$$D^*(x) = \frac{p_{\mathrm{data}}(x)}{p_{\mathrm{data}}(x) + p_n(x)}.$$

Noise Contrastive Estimation (NCE)

- What if the discriminator is parameterized by:

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

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

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

- Equivalently,

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

NCE for Training EBMs

- Energy-based models: $p_\theta(x) = \frac{1}{Z_\theta} \exp\left(f_\theta(x)\right)$.

  The normalization constraint $Z_\theta = \int e^{f_\theta(x)} dx$ is hard to satisfy.

- **Solution:** Modeling Z_θ with an additional trainable parameter Z that disregards the normalization constraint:

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

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

$$p_{\theta^*, Z^*}(x) = \frac{1}{Z^*} e^{f_{\theta^*}(x)} = p_{\text{data}}(x).$$

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

$$\int \frac{1}{Z^*} e^{f_{\theta^*}(x)} dx = \int p_{\text{data}}(x) dx = 1 \Longrightarrow Z^* = \int e^{f_{\theta^*}(x)} dx.$$

NCE for Training EBMs

- The discriminator $D_{\theta,Z}(x)$ for $p_{\theta,Z}(x)$ is given by:

$$D_{\theta,Z}(x) = \frac{\frac{1}{Z}e^{f_\theta(x)}}{\frac{1}{Z}e^{f_\theta(x)} + p_n(x)} = \frac{e^{f_\theta(x)}}{e^{f_\theta(x)} + p_n(x)Z}.$$

- Noise contrastive estimation training:

$$\max_{\theta,Z} \; \mathbb{E}_{x\sim p_{\text{data}}}\left[\log D_{\theta,Z}(x)\right] + \mathbb{E}_{x\sim p_n}\left[\log(1 - D_{\theta,Z}(x))\right].$$

- Equivalently,

$$\max_{\theta,Z} \; \mathbb{E}_{x\sim p_{\text{data}}}\left[f_\theta(x) - \log(e^{f_\theta(x)} + Zp_n(x))\right] + \mathbb{E}_{x\sim p_n}\left[\log(Zp_n(x)) - \log(e^{f_\theta(x)} + Zp_n(x))\right].$$

- Use LogSumExp (LSE) function for numerical stability:

$$\log\left(e^{f_\theta(x)} + Zp_n(x)\right) = \log\left(e^{f_\theta(x)} + e^{\log Z + \log p_n(x)}\right) = \text{LSE}(f_\theta(x), \log Z + \log p_n(x)).$$

NCE for Training EBMs

1. Sample a mini-batch of datapoints $x_1, x_2, \cdots, x_n \sim p_{\text{data}}(x)$.
2. Sample a mini-batch of noise samples $y_1, y_2, \cdots, y_n \sim p_n(y)$.
3. Estimate the NCE loss.

$$\frac{1}{n}\sum_{i=1}^{n}\Big[f_\theta(x_i) - \text{LSE}(f_\theta(x_i), \log Z + \log p_n(x_i))$$
$$+ \log Z + \log p_n(y_i) - \text{LSE}(f_\theta(y_i), \log Z + \log p_n(y_i))\Big]$$

4. Compute the gradient and then update θ & Z (Stochastic gradient ascent).

No need to sample from the EBM!
However, the noise distribution needs to be "close" to the data distribution.

Comparing NCE and GAN

Similarities:

- Both involve training a discriminator to perform binary classification with cross-entropy loss.
- Both are likelihood-free.

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

Flow contrastive estimation:

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

$$D_{\theta,Z,\phi}(x) = \frac{\frac{1}{Z}e^{f_\theta(x)}}{\frac{1}{Z}e^{f_\theta(x)} + p_{n,\phi}(x)} = \frac{e^{f_\theta(x)}}{e^{f_\theta(x)} + p_{n,\phi}(x)Z}$$

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

$$\min_{\phi} \max_{\theta,Z} \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}}\left[\log D_{\theta,Z,\phi}(\boldsymbol{x})\right] + \mathbb{E}_{\boldsymbol{x}\sim p_{n,\phi}}\left[\log(1 - D_{\theta,Z,\phi}(\boldsymbol{x}))\right]$$

Flow Contrastive Estimation (Gao et al. 2020)

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

Image source: Gao et al. 2020.

Adversarial training for EBMs

Energy-based model: $p_\theta(\boldsymbol{x}) = \frac{e^{f_\theta(\boldsymbol{x})}}{Z(\theta)}$.

Upper bounding its log-likelihood with a variational distribution $q_\phi(\boldsymbol{x})$:

$$\begin{aligned}
\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}}[\log p_\theta(\boldsymbol{x})] &= \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}}[f_\theta] - \log Z(\theta) \\
&= \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}}[f_\theta(\boldsymbol{x})] - \log \int e^{f_\theta(\boldsymbol{x})} d\boldsymbol{x} \\
&= \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}}[f_\theta(\boldsymbol{x})] - \log \int q_\phi(\boldsymbol{x}) \tfrac{e^{f_\theta(\boldsymbol{x})}}{q_\phi(\boldsymbol{x})} d\boldsymbol{x} \\
&\leq \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}}[f_\theta(\boldsymbol{x})] - \int q_\phi(\boldsymbol{x}) \log \tfrac{e^{f_\theta(\boldsymbol{x})}}{q_\phi(\boldsymbol{x})} d\boldsymbol{x} \\
&= \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}}[f_\theta(\boldsymbol{x})] - \mathbb{E}_{\boldsymbol{x}\sim q_\phi}[f_\theta(\boldsymbol{x})] + H(q_\phi(\boldsymbol{x})).
\end{aligned}$$

What do we require for the model $q_\phi(\boldsymbol{x})$?

Adversarial training: $\max_\theta \min_\phi \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}}[f_\theta(\boldsymbol{x})] - \mathbb{E}_{\boldsymbol{x}\sim q_\phi}[f_\theta(\boldsymbol{x})] + H(q_\phi(\boldsymbol{x}))$.

Conclusions

- Energy-based models are very flexible probabilistic models with intractable partition functions.
- Sampling is hard and typically requires iterative MCMC approaches.
- Training is hard because computing likelihood is hard.
- Comparing the likelihood/probability of two different points is tractable.
- Maximum likelihood training by contrastive divergence.
  However, it requires sampling for each training iteration.
- Sampling-free training: score matching and its extensions.
- Sampling-free training: noise contrastive estimation.
  Additionally, it provides an estimate of the partition function.

References

1. Probabilistic Machine Learning: Advanced Topics (*Chapter 23*)
   Kevin P Murphy, The MIT Press (2023)

2. How to Train Your Energy-Based Models
   https://arxiv.org/pdf/2101.03288.pdf

3. https://github.com/yataobian/awesome-ebm

Introduction to Deep Generative Modeling

Lecture #12

HY-673 – Computer Science Dep., University of Crete

Professors: Yannis Pantazis, Yannis Stylianou

Teaching Assistant: Michail Raptakis