Lecture 11 - Modeling High-Dimensional Data

Setup: data $x_1, \dots, x_n \sim p_D(x)$, $x_i \in \mathbb{R}^d$, large d

e.g. images, audio, text

Goal: Learn a good model for $p(x)$

→ Find parameters θ s.t. $p_\theta(x) \approx p_D(x)$

→ Leverage powerful function approximation (Neural Networks)

Why?

- obtain compact data description
- sample new data from similar distribution

Questions:

- What representation for $p_\theta(x)$?
  - autoregressive for sequences

$$p_\theta(x) = p_\theta(x_1)\, p_\theta(x_2|x_1) \cdots p_\theta(x_d|x_1, \dots, x_{d-1})$$

e.g. $p_\theta(x_s = k | x_{1:s-1}) \propto e^{f_\theta^k(x_{1:s-1})}$

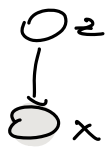
  - energy-based model

$$p_\theta(x) = \frac{1}{Z} e^{-f_\theta(x)}$$

f_θ: parameterized energy function

- mixture model

$$p_\theta(x) = \int p_\theta(x|z)\, p_z(z)\, dz$$



e.g. $p_\theta(x|z) = \mathcal{N}(x | g_\theta(z), \sigma^2 I)$

(used in Variational Auto-Encoder)

- transport model / implicit model

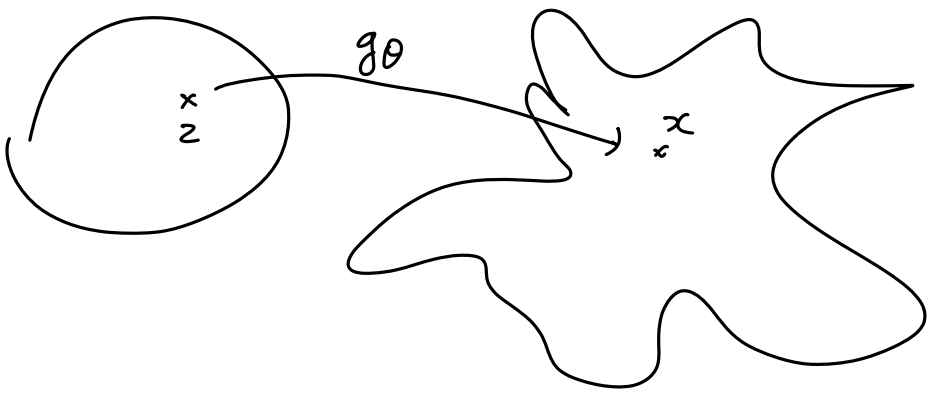
$p_\theta(x)$ s.t. $x = g_\theta(z)$ with

$z \sim p_z$

e.g. $z \sim \mathcal{N}(0, I_{d_z})$

p_θ defined implicitly by

$$\mathbb{E}_{x \sim p_\theta}[\varphi(x)] = \mathbb{E}_{z \sim p_z}[\varphi(g_\theta(z))] \quad \text{for all } \varphi$$



- What objective function?
  - maximum likelihood: $\max_\theta \frac{1}{n} \sum_{i=1}^{n} \log p(x_i)$
  - other metrics? $\min_\theta d(p_\theta, p_{data})$

(KL, φ-divergence, Ficher, TV, Wasserstein ...)

- What algorithm?
  → Back-propagation / (stochastic) gradient descent

1. Likelihood-based models

Objective: $\max_\theta \mathbb{E}_{x \sim p_{data}} [\log p_\theta(x)]$

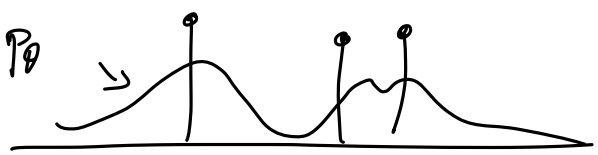
Remark: • This is equivalent to

$$\min_\theta d_{KL}(p_{data} \| p_\theta)$$

Indeed: $d_{KL}(p_{data} \| p_\theta) = \mathbb{E}_{p_{data}}\left[\log \frac{p_{data}(x)}{p_\theta(x)}\right]$

$$= -\mathbb{E}_{p_{data}}[\log p_\theta(x)] + C$$

- We need to have mass on all training data!
  ($p_\theta(x) > 0$ when $p_{data}(x) > 0$)



"coverage"

- Auto-regressive models

$x = (x_1, \dots, x_d)$, $p_\theta(x) = p_\theta(x_1) \cdots p_\theta(x_d | x_{1:d-1})$

→ Often, $p_\theta(x_s \mid x_{1:s-1})$ has a tractable form

e.g. softmax $p_\theta(x_s = k \mid x_{1:s-1}) \propto e^{f_{\theta,k}(x_{1:s-1})}$

$(k \in \{1, \dots, K\})$

$f_{\theta,k}(x_{1:s})$: "sequence" model (RNN, LSTM, Transformer, etc.)

→ log-likelihood objective

$$\ell(\theta) = \mathbb{E}_{x \sim p_{data}}\left[\sum_{s=1}^{d} \underbrace{\log p_\theta(x_s \mid x_{1:s-1})}_{\text{"cross-entropy" loss}} \right]$$

can be maximized using (SGD).

- Mixture models : $p_\theta(x) = \int p_\theta(x \mid z)\, p(z)\, dz$

→ captures useful latent feature representations

→ but: leads to intractable likelihood !

need Variational Inference (V.A.E.)

$$\ell(\theta) = \mathbb{E}_{x \sim p_{data}}\left[\log p_\theta(x) \right]$$

$$\geq \mathbb{E}_{x \sim p_{data}}\left\{ \mathbb{E}_{z \sim q_\phi}\left[\log p_\theta(x, z) \right] - \mathbb{E}_{z \sim q_\phi}\left[\log q_\phi(z) \right] \right\} =: \tilde{\ell}(\theta, \phi)$$

(see Recitation 9)

Normalizing Flows

Transport model : $x = g_\theta(z)$, $z \sim p_z$

[similar to mixture w/ deterministic $p_\theta(x|z) = \delta_{g_\theta(z)}$]

Invertible : assume $\begin{cases} x, z \in \mathbb{R}^d \text{ (same dimension)} \\ g_\theta \text{ invertible} \end{cases}$

→ change-of-variables formula :

$$p_\theta(x) = p_z\left(g_\theta^{-1}(x)\right) \left| \det \partial g_\theta^{-1}(x) \right|$$

(∂g_θ : Jacobian matrix)

If $z = g_\theta^{-1}(x)$, we can write :

$$p_\theta(x) = p_z(z) \left| \det \partial g_\theta(z) \right|^{-1}$$

→ **Normalizing flows** : Use this formula to optimize the (exact) log-likelihood

→ Requires well-chosen models s.t. $\begin{cases} g_\theta \text{ is invertible} \\ g_\theta, g_\theta^{-1} \text{ have simple analytical forms} \end{cases}$

e.g. :
NICE, RealNVP (L. Dinh et al.)
IAF, MAF

Energy-based Models (EBMs)

$$p_\theta(x) = \frac{1}{Z_\theta} e^{-f_\theta(x)}$$

→ Gradient optimization requires sampling:

$$\nabla_\theta \log p_\theta(x) = \nabla_\theta \left(-f_\theta(x) - \log Z_\theta\right)$$
$$= -\nabla_\theta f_\theta(x) + \mathbb{E}_{x' \sim p_\theta}\left[\nabla_\theta f_\theta(x')\right]$$

$$\Rightarrow \quad \nabla_\theta \mathbb{E}_{x \sim p_{data}}\left[\log p_\theta(x)\right] = -\mathbb{E}_{x \sim p_{data}}\left[\nabla_\theta f_\theta(x)\right] + \mathbb{E}_{x \sim p_\theta}\left[\nabla_\theta f_\theta(x)\right]$$

$\mathbb{E}_{x \sim p_\theta}[\cdot]$ requires samples from p_θ!
E.g. with MCMC

Remarks:
- sampling from learned EBM also requires MCMC, e.g. Langevin using $\nabla_x f_\theta(x)$
- Can we avoid sampling during training?
  → Yes, if we give up MLE/KL, and use score matching / Fisher Divergence instead

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

[Hyvärinen, 2005]

- Score-based generative modeling [Song & Ermon]: directly learn the "score function" $g_\theta(x) = \nabla_x \log p(x)$

2. Alternatives to max. likelihood

⚠ Issues with maximum likelihood:

- Requires models with tractable likelihood
  - e.g. auto-regressive, invertible transport models, Gaussian conditionals in VAE
- Enforces coverage of training data (otherwise, $d_{KL}(p_{data}, p_\theta) = \infty$)

→ this could cause noisy/blurry samples

Q: Can we train implicit models $x = g_\theta(z)$ in a "black-box", "likelihood-free" fashion?

- Adversarial training (e.g. GANs)

Idea: train "discriminator/critic" f_ϕ jointly with g_θ.

- f_ϕ learns to distinguish p_{data} from p_θ
- g_θ "fools" discriminator

→ leads to min-max optimization problem

(GAN) $\min_{\theta} \max_{\varphi} \mathbb{E}_{x\sim p_{data}}\left[\log p_\varphi(y=1|x)\right] + \mathbb{E}_{x\sim p_\theta}\left[\log p_\varphi(y=0|x)\right]$

(f_φ tries to classify $\begin{cases} p_{data} \text{ as } +1 \\ p_\theta \text{ as } 0 \end{cases}$

- **Integral probability metrics** (IPM)

(GAN) resembles the optimization problem

$$\min_{\theta} d_{\mathcal{F}}(p_{data}, p_\theta)$$

where $d_{\mathcal{F}}(p,q) := \sup_{f\in\mathcal{F}} \mathbb{E}_{x\sim p}[f(x)] - \mathbb{E}_{x\sim q}[f(x)]$

↳ IPM

Ex: . Total Variation distance (TV):

$$\mathcal{F} = \{ f : \|f\|_\infty \le 1 \}$$

. Wasserstein-1 distance

$$\mathcal{F} = \{ f : \|f\|_{Lip} \le 1 \}$$

↳ $\|f\|_{Lip} = \sup_{x,y} \frac{|f(x)-f(y)|}{\|x-y\|}$

(this is the Kantorovich dual of

$$W_1(p,q) = \inf_{(x,y)} \{ \mathbb{E}[\|x-y\|] : x\sim p, y\sim q \}$$)

(see Recitation yesterday)

- **Maximum Mean Discrepancy**

$$\mathcal{F} = \{ f : \|f\|_{\mathcal{H}} \le 1 \}$$

$\mathcal{H}$ is a Hilbert space of functions, e.g. Sobolev space or RKHS.

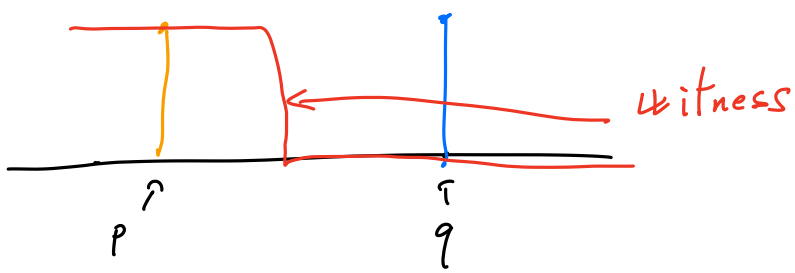
- "**Neural net divergence**":

$$\mathcal{F} = \{ f_\varphi : \text{a set of neural networks parameterized by } \varphi \}$$

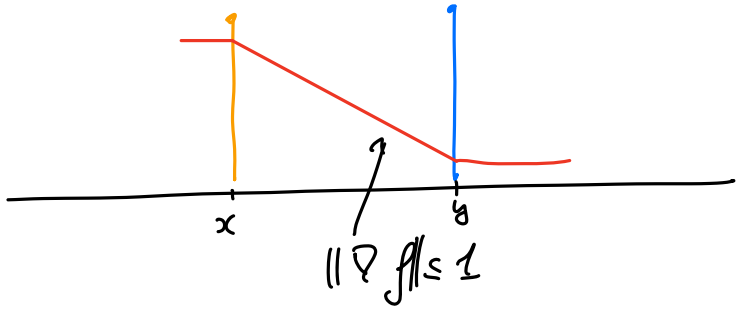
Remarks:

- optimal $f \in \mathcal{F}$ in $\sup_{f \in \mathcal{F}} \hat{E}_p[f] - \hat{E}_q[f]$ is sometimes called "witness function"

- In TV, the data geometry ($\|x-y\|$) plays no role!



- In W_1, the data metric is important (via Lipschitz condition)



$\rightarrow W_1(\delta_x, \delta_y)$ varies with $\|x-y\|$

- Yet, W_1 suffers from the "curse of dimensionality"

$$\left| W_1(\hat{p}, \hat{q}) - W_1(p, q) \right| \leq n^{-\frac{1}{d}}$$

$$\hat{p} = \frac{1}{n} \sum_{i=1}^{n} \delta_{x_i} \quad \text{with } x_i \sim p$$

$$\hat{q} = \frac{1}{n} \sum_{i=1}^{n} \delta_{y_i} \quad y_i \sim q$$

$\Rightarrow$ very slow convergence to true distance when using empirical distributions !

- MMD and "Neural Net" distance can improve this :

$$\left| d_{\mathcal{F}}(\hat{p}, \hat{q}) - d_{\mathcal{F}}(p, q) \right| \leq \frac{1}{\sqrt{n}}$$

But : they are "weaker" metrics / less discriminative

- "Information" divergences (KL, reverse KL, Hellinger, ...) can also be written in similar "variational" form

(f-GAN paper [Nowozin et al. '16])