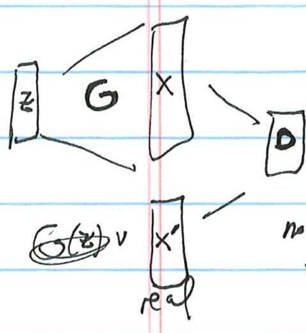
Lecture 4 Generative Modeling & Density Estimation

- Learn from data ~~to generate~~, sample from it $P(x)$ (possibly implicitly)

- Many applications in HEP, e.g. fast calorimeter simulation
  GEANT4 slow, ML fast
  "surrogate modeling"
  others: anomaly detection — LHC
  — stellar streams Gaia data
  learning phase space density $f(x, v) \to$ Boltzmann $\to \nabla\Phi$
  ρ
  $\to$ ρ

- 3 main paradigms
  - GANs
  - VAEs
  - Normalizing flows

GANs: Adversarial training (Goodfellow et al 2014)
2 NNs trying to defeat one another



G generator — noise $\to$ fake images
$z \qquad x$

D discriminator — fake vs real
$G(z) \qquad x$

$$L = -\left(\sum_{i \in data} \log D(x_i) + \sum_{i \in fake} \log\left[1 - D(G(z_i))\right]\right)$$

BCE b/w real & fake

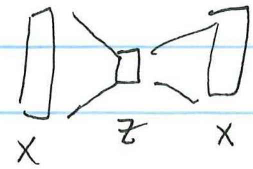
But objective is:

$$\boxed{\max_G \min_D L}$$

$\longrightarrow$ a min-max objective
saddle pt, non convex!

- GANs capable of generating very realistic ~~fake~~ images (e.g. celeb fa

— VAEs: idea — learn how to compress data to latent space and vice versa



"Autoencoder"

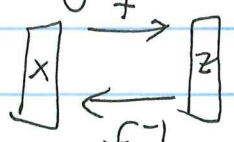
Sample from $z \sim$ noise

NOT REALLY STATE OF THE ART FOR ANYTHING

— Normalizing flows: learn invertible map from x to noise

★ state of the art!

will spend more time on this



can run in either direction in principle!

$x \longrightarrow z$: density estimation

$$p(x)\,dx = p(z)\,dz \qquad z = f(x)$$

simple e.g. unif or gaus

$$p(x) = p(z = f(x))\left|\frac{dz}{dx}\right| = p(z)\left|\frac{\partial f}{\partial x}\right|$$

need tractable Jacobian determinant!

~~Ref: Krause & DS 2105.05285 2110.11377 "CaloFlow"~~

$z \longrightarrow x$: generative modeling!

more in a minute

— Since NF provide access to $p(x)$, can train w/ MLE! → optimal → stable → convergent, much improved vs GANs

$$L = -\log p(x|\theta)$$

↳ parameters of f.

— How to get tractable Jacobian? det of general $d \times d$ mtx takes $O(d^3)$ operations

Basic idea: autoregressive transf

~~$z_1 \rightarrow x_1$~~
~~$z_1, z_2 \rightarrow x_2$~~
~~$(z_1, z_2, z_3) \rightarrow x_3$~~

~~$x_1 \rightarrow z_1$~~
~~$x_1, x_2 \rightarrow z_2$~~

$$\left.\begin{array}{l} z_1(x_1) \\ z_2(x_1, x_2) \\ z_3(x_1, x_2, x_3) \end{array}\right\} \longrightarrow \text{then } \frac{\partial z}{\partial x} = \begin{pmatrix} & \cdots \\ 0 & \ddots \end{pmatrix}$$

determinant is just product along diagonal! $O(d)$ not $O(d^3)$.

- Chain together multiple AFs w/ permutation of $(z_1, \dots z_d)$ in between to increase expressivity.

the "flow" in normalizing flow

- Still need family of functions for trans. Many choices here one possibility that works well: "affine"

$\alpha_1 = 1$

$\mu_1 = 0$

$$z_i = \alpha_i(x_1, \dots x_{i-1})\, x_i + \mu_i(x_1, \dots x_{i-1}) \rightarrow \text{"MAF"}$$

~~if $z_i \sim N(0,1)$, $x_i \sim N(\dots)$~~

w/ α_i, μ_i given by DNN.

inverse:

$x_1 = z_1$

$$x_2 = \frac{z_2 - \mu_2(x_1)}{\alpha_2(x_1)} = \frac{z_2 - \mu_2(z_1)}{\alpha_2(z_1)}$$

$$x_3 = \frac{z_3 - \mu_3(x_1, x_2)}{\alpha_3(x_1, x_2)} = \frac{z_3 - \mu_3\left(z_1, \frac{z_2 - \mu_2(z_1)}{\alpha_2(z_1)}\right)}{\alpha_3(z_1, \dots)}$$

$\vdots$

CAN SKIP

DE

So, $x \rightarrow z$: single eval of NN for $\vec{\alpha}, \vec{\mu}$ $\rightarrow$ fast

sampling $z \rightarrow x$: d successive evals of NN $\rightarrow$ slower by factor of d.

- Can also define "IAF" where α_i, μ_i fns. of z then sampling fast, DE slow.

- Other approaches exist where both directions fast