Generative algorithms/models coming from learning from training data.

e.g. iid samples from p. Try to get some estimate $\hat{p}$.

Want to sample from $\hat{p}$ rather than evaluate $\hat{p}(x)$
(density estimation)

often have generative model where $\hat{p}$ is implicit and backing out $\hat{p}(x)$ is very difficult.

Suppose $\hat{p}$ is defined <u>implicitly</u> as the probability integral transformation of a density q by a generator function

$g : \mathcal{Z} \mapsto \mathcal{X}$.

So for some event A

$$\Pr[A] = \Pr\left[g^{-1}(A)\right] = \int_{g^{-1}(A)} q(z)\,dz = \int_A q\left(g^{-1}(x)\right) \left|\nabla_x g^{-1}(x)\right| dx$$

So $\hat{p}(x) = q\left(g^{-1}(x)\right) \left|\nabla_x g^{-1}(x)\right|$

when inverse and Jacobian of g are easy to compute.

Converting a generator to a density estimator is easy.

Is this possible? Yes with normalizing flows, which have nice generators.

In other cases, can still sample from $\hat{p}$ by sampling.

If $x \sim q$, then evaluate $g(z) \sim \hat{p}$

Normalizing Flows

want inverse and Jacobian to be simple (and also learnable).

want approach to how complicated $g(\cdot)$ be represented well.

→ use sequence of invertible transformations until desired level of complexity achieved.

Recall if we have invertible mapping $f: \mathbb{R}^d \mapsto \mathbb{R}^d$ with inverse $f^{-1} = g$, and we use this to transform r.v. X with distribution q.

Then $\tilde{X} = f(X)$ has distribution

$$q(\tilde{x}) = q(x) \left| \det \frac{\partial f^{-1}}{\partial \tilde{x}} \right| = q(x) \left| \det \frac{\partial f}{\partial x} \right|^{-1} \qquad (*)$$

where equality is by applying chain rule (inverse function theorem) and is property of Jacobian of invertible functions.

---

we can construct arbitrarily complex densities by composing several simple maps and successively applying $(*)$, the change of variable formula.

The density $q_K(x)$ obtained by successive transformation of r.v. X_0 with density q_0 through chain of K transformations f_K is

$$X_K = f_K \circ f_{K-1} \circ \cdots \circ f_2 \circ f_1(X_0)$$

$$\ln q_K(X_K) = \ln q_0(X_0) - \sum_{k=1}^{K} \ln \left| \det \frac{\partial f_k}{\partial X_{k-1}} \right|$$

The path traversed by the r.v. $X_k = f(X_{k-1})$ with initial distribution $q_0(X_0)$ is called the flow, path formed by successive distributions q_k is called normalizing flow.

A property of such transformations is LOTUS (law of the unconscious statistician)

$$X \sim p(x) \qquad \mathbb{E}[h(X)] = \sum_i h(u_i)\, p_X(u_i)$$

that expectations with respect to the transformed density q_K can be computed without explicitly knowing q_K.

Any expectation $\mathbb{E}_{q_K}[h(x)]$ can be written as an expectation under q_0:

$$\mathbb{E}_{q_K}[h(x)] = \mathbb{E}_{q_0}\left[h\left(f_K \circ f_{K-1} \circ \cdots \circ f_2 \circ f_1(x_0)\right)\right].$$

which does not require computation of the log det Jacobian terms when $h(x)$ does not depend on q_K.

infinitesimal flows: natural to consider many functions, limit as we get infinitesimal changes with infinite functions.

Describe how initial density $q_0(x)$ evolves over path using PDE $\frac{\partial}{\partial t} q_t(x) = \mathcal{T}_t[q_t(x)]$ where $\mathcal{T}_t$ describes the continuous-time dynamics.

Example: Langevin flow where r.v. X with initial density $q_0(x)$ has densities transformed by Fokker-Planck equation

Discrete Flows

Discrete flows:

determinant of Jacobian corrects for changes to volume in continuous spaces, but volume doesn't change in discrete spaces, so no need to account for it in settings with discrete distributions.

Let X be discrete r.v. and $Y = f(X)$

then pmf of Y is sum over preimage of f:

$$P_r[Y=y] = \sum_{x \in f^{-1}(y)} P_r[X=x]$$

for invertible f, $P_r[Y=y] = P_r[X = f^{-1}(y)]$

GLOW (how do we do learning)

Let X be high-dimensional random vector with unknown true distribution $X \sim p(x)$.

Collect some iid dataset $\mathcal{D}$ of size N and choose a model $P_\theta(x)$ with parameters θ.

Consider maximum likelihood objective:

$$\mathcal{J}(\mathcal{D}) = \frac{1}{N} \sum_{i=1}^{N} -\log P_\theta(x^{(i)}) \quad \text{for discrete data.}$$

or $$\mathcal{J}(\mathcal{D}) \approx \frac{1}{N} \sum_{i=1}^{N} -\log P_\theta(\tilde{x}^{(i)}) + c$$

where $\tilde{x}^{(i)} = x^{(i)} + u$ with $u \sim \mathcal{U}(0, a)$ and $c = -M \log a$

where a is determined by discretization levels of data and M is data dimension

Optimizing ML objective done through stochastic gradient descent

where parametric form of p_θ is often a neural network.

In flow-based generation:

$z \sim p_\theta(z)$ where z is latent variable and $p_\theta(z)$ is often simple like multivariate Gaussian (spherical)

$x = g_\theta(z)$.

where g_θ is invertible so $z = f_\theta(x) = g_\theta^{-1}(x)$.

since we think of $f = f_1 \circ f_2 \circ \dots \circ f_k$.

$$H_0 = X \xleftrightarrow{f_1} H_1 \xleftrightarrow{f_2} \dots \xleftrightarrow{f_k} Z = H_k.$$

$$\log p_\theta(x) = \log p_\theta(z) + \log \left| \det \frac{\partial z}{\partial x} \right|$$

$$= \log p_\theta(z) + \sum_{k=1}^{K} \log \left| \det \left(\frac{\partial H_i}{\partial H_{i-1}} \right) \right|$$

scalar quantity "log-det-Jacobian"

can be fairly easy to compute when Jacobian is lower triangular matrix

when Jacobian is triangular:

$$\log \left| \det \left(\frac{\partial H_i}{\partial H_{i-1}} \right) \right| = \text{sum} \left(\log \left| \text{diag} \left(\frac{\partial H_i}{\partial H_{i-1}} \right) \right| \right)$$

diagonal entries.

sum over all vector elements

element-wise log