DIFFUSION MODELS

Machine learning reading group, WG.
Omega Salvida

Used in generative AI.

So far we have seen SUPERVISED LEARNING.

- Data is of type (x = input, y = output), training set is of type $\{(x_i, y_i),\ i = 1, \dots, N\}$.
- The result of training is a function f s.t. for an unseen input x, the an appropriate output is $y = f(x)$.

- Examples: 1.) Digit recognition: input is an image [5], output is a single digit 5. (Pietro & Marc W7, Poppy W8.)

  2.) Property prices: input is a triplet (size, # of rooms, location), output is the value (£).

- All algorithms so far have been DETERMINISTIC: Training on two separate occasions using the same training set yields the same function f.
  Feeding the same input into f always gives the same output.

NOTE: Probability measures were used only for averaging, not for sampling: deterministic.

This week: UNSUPERVISED LEARNING.

- Data is not of type (x, y).
- The result of training is some sort of a generator, that does not necessarily take any input.
- Example: generating images of cats.
- The algorithm will be STOCHASTIC (non-determ.): ~~to training~~ the generator will ~~might~~ generate two distinct images if run twice, even if all other parameters are the same.

NOTE: the training procedure ~~is~~ will still be deterministic! training on the same training set always gives the same generator.

Notational differences with the source:

$I, \Theta \rightsquigarrow \Psi, x$

The general setup

- Assume images of cats are elements of $\mathbb{R}^d$ for some d. ~~For example,~~

  Example: resolution 800×600, each ~~pixel~~ pixel represented by a 32-bit number $\rightsquigarrow$ simplify to an arbitrary real number. $\Rightarrow d = 800 \cdot 600$.

- Assume $\exists$ a probability density Ψ on $\mathbb{R}^d$ s.t. for $u \in \mathbb{R}^d$:

  $$\Psi(u) \propto \mathbb{P}(u \text{ is an image of a cat}).$$

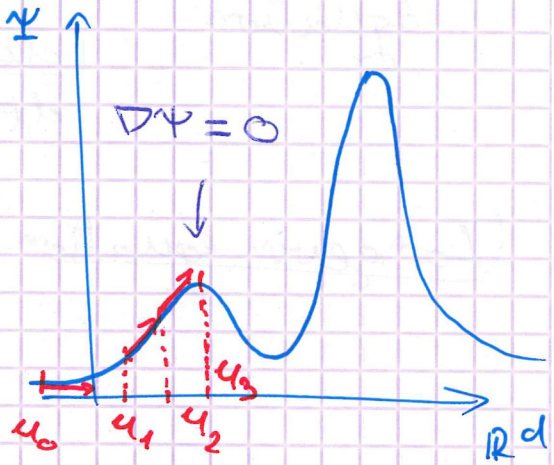
  We do not know Ψ; we will learn ~~Ψ~~ it from sample data.

- Assuming we know Ψ, we generate new images as follows (see figure below).

  1.) Take a random image $u_0 \in \mathbb{R}^d$.
  This will not be an image of a cat (most likely).

  2.) Update (transform) the image "in the direction" of $\nabla\Psi$ to get a new image $u_1 \in \mathbb{R}^d$.
  This image will ~~be~~ have a higher probability of looking like a cat.

  Repeat step 2.



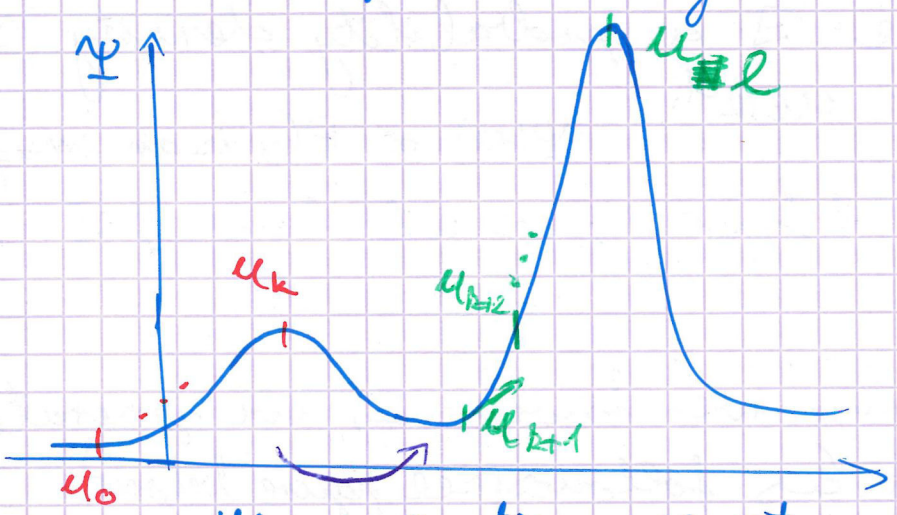
This can be put into an equation:

$$\frac{du}{dt} = \nabla\Psi(u) \qquad (u(0) = u_0)$$

(Discretising in time gives the steps 1 & 2.)

ISSUE: We get permanently stuck at the first stationary point we encounter.

SOLUTION: Add some noise; this will (eventually) perturb us out of stationary points:



After some time we get perturbed enough to start moving towards a more optimal peak.

Equation:

$$\frac{du}{dt} = \nabla\Psi(u) + \sqrt{\Sigma}\,\frac{dW}{dt}$$

(Langevin equation)

↑ Brownian/Wiener noise (standard).

ONE MORE ADJUSTMENT: Replace $\nabla\Psi$ with $\nabla \log \Psi$.

this is called a "score"

Benefits: 1.) In practice, Ψ is often a product of factors. Taking log produces sums, which are easier to work with.

See training set: a sequence of i.i.d. samples.

2.) Computationally more stable in regions when $\Psi \approx 0$.

Get

~~$\frac{dx}{dt} = \cdots$~~

$$\frac{dx}{dt} = \nabla \log \Psi(x) + \sqrt{2}\,\frac{dW}{dt}.$$

To run the generating dynamics, we need to learn $\nabla \log \Psi$.

GOAL: Find $s_\theta : \mathbb{R}^d \to \mathbb{R}^d$ from a family of functions $\Omega = \{ s_\theta : \mathbb{R}^d \to \mathbb{R}^d,\ \theta \in \mathbb{R}^n \text{ parametrization} \}$

s.t. $s_\theta \approx \nabla \log \Psi$ in an "appropriate" sense.

One option (not pursued here): Assume $\exists\, \Psi_\theta$ prob. density on $\mathbb{R}^d$ s.t. $s_\theta = \nabla \log \Psi_\theta$.

Find Ψ_θ that maximises the likelihood of the samples in the training set (see page 6)

this is MAXIMUM LIKELIHOOD ESTIMATION

INSTEAD, we will try to minimise (over θ)

$$J(\theta; \Psi) := \int_{\mathbb{R}^d} |s_\theta(x) - \nabla \log \Psi(x)|^2 \Psi(x)\, dx.$$

See note on page 6.

MINIMISATION

We are given $N \gg 1$ images of cats: $\{u_i\}_{i=1}^N \subset \mathbb{R}^d$.
We treat them as i.i.d. random variables distributed $\sim \Psi$.

↳ ~~ident~~ independent identically distributed

this cannot be used to estimate $\nabla \log \Psi$.
We do not know $\Psi(u)$, and we cannot do any series attempt of ~~take~~ differentiating.

SOLUTION: Def.

NOTE: $\operatorname{div} s_\theta = \operatorname{Tr} D s_\theta$, D ... Jacobian

$$K(\theta; \Psi) = \int_{\mathbb{R}^d} \left(|s_\theta(u)|^2 + 2 \operatorname{div} s_\theta(u) \right) \Psi(u)\, du.$$

PROP.: Minimising $J(\theta; \Psi)$ over θ is equivalent to minimising $K(\theta; \Psi)$ over θ.

NOTE (forgot to write on p. 5):

$$J(\theta; \Psi) = \mathbb{E}_{u \sim \Psi} \left[|s_\theta(u) - \nabla \log \Psi(u)|^2 \right]$$

this gives probabilistic interpretation to J.

PROOF of Prop.:

$$|s_\theta - \nabla \log \Psi|^2 = \langle s_\theta - \nabla \log \Psi, s_\theta - \nabla \log \Psi \rangle$$
$$= |s_\theta|^2 - 2 \langle s_\theta, \nabla \log \Psi \rangle + |\nabla \log \Psi|^2$$

Observe: $\langle s_\theta, \nabla \log \Psi \rangle = \left\langle s_\theta, \frac{\nabla \Psi}{\Psi} \right\rangle$

$$\Rightarrow \int_{\mathbb{R}^d} \langle s_\theta, \nabla \log \Psi \rangle \Psi \, du = \int_{\mathbb{R}^d} \langle s_\theta, \nabla \Psi \rangle \, du$$

Detour to vector calculus: (Higher-dimensional integration by parts.)

Let $F: \mathbb{R}^d \to \mathbb{R}^d$ be a vector field,
$g: \mathbb{R}^d \to \mathbb{R}$ scalar field.

$$\operatorname{div}(g \cdot F) = \sum_{i=1}^{d} \frac{\partial}{\partial x_i}(g F_i)$$

$$= \sum_{i=1}^{d} \left(\frac{\partial}{\partial x_i} g\right) \cdot F_i + \sum_{i=1}^{d} g \cdot \frac{\partial}{\partial x_i} F_i$$

$$= \langle \nabla g, F \rangle + g \operatorname{div} F$$

$\Rightarrow$ ~~$\int_{\mathbb{R}} \langle \nabla g, F \rangle dx = \int_{\mathbb{R}}$~~ substitute $g = \Psi$, $F = s_\psi$.

$$\Rightarrow \int_{\mathbb{R}^d} \langle s_\psi, \nabla \Psi \rangle du = \int_{\mathbb{R}^d} \operatorname{div}(\Psi \cdot s_\psi) du - \int_{\mathbb{R}^d} \Psi \cdot \operatorname{div} s_\psi \, du$$

By Gauss / Divergence thm:

$$\int_{\mathbb{R}^d} \operatorname{div}(\Psi \cdot s_\psi) du = \underbrace{\int_{\partial \mathbb{R}^d} \Psi \langle s_\psi, \hat{n} \rangle dS}_{\text{"improper" integral}} \overset{(*)}{=} 0$$

surface

(*) $\lim_{|u| \to \infty} \Psi(u) = 0$ (assume), since s_ψ should be $\approx \nabla \log \Psi$, we can conclude $\Psi s_\psi(u) \to 0$ as $|u| \to \infty$.

$$\Rightarrow \int_{\mathbb{R}^d} \langle s_\psi, \nabla \Psi \rangle du = - \int_{\mathbb{R}^d} \Psi \cdot \operatorname{div} s_\psi \, du$$

"independent of ψ, so can be ignored when minimizing. Assume"

$$J(\psi, \Psi) = \int_{\mathbb{R}^d} \left(|s_\psi(u)|^2 + 2 \operatorname{div}(s_\psi) \right) \Psi(u) du + \int_{\mathbb{R}^d} |\nabla \log \Psi(u)|^2 \Psi(u) du$$

□

This is useful since K can be estimated from $\{u_i\}_{i=1}^N$:

$$K(x,y) \approx \frac{1}{N}\sum_{i=1}^{N} |s_x(u_i)|^2 + 2\,\mathrm{div}\left(s_x^{(u)}\right)\Big|_{u=u_i} \xrightarrow[N\to\infty]{\text{a.s.}} K(x,y)$$

a.s. means "almost surely", i.e. with probability 1. Follows from the strong law of large numbers under some assumptions.

We now consider another approach.

Def. (for $\sigma>0$):

$$\rho_\sigma(u) := \frac{1}{(2\pi\sigma)^{d/2}} e^{-\frac{1}{2\sigma^2}|u|^2}$$

Multivariate (d-dim) Gaussian prob. density.

$$\Psi_\sigma(u) := (\rho_\sigma * \Psi)(u) = \int_{\mathbb{R}^d} \frac{1}{(2\pi\sigma)^{d/2}} \Psi(w)\,dw$$

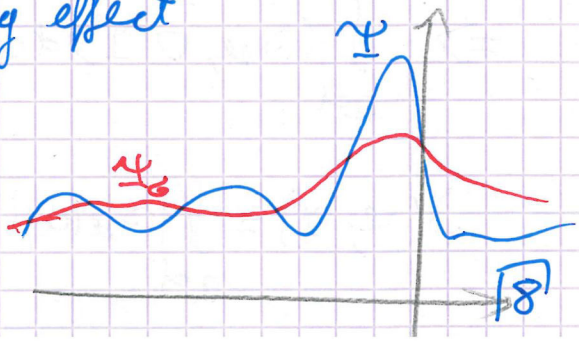
$$= \int_{\mathbb{R}^d} \rho_\sigma(u-w)\,\Psi(w)\,dw.$$

Interpretation of Ψ_σ: the value of $\Psi_\sigma(u)$ is a "weighted average" of Ψ, where the weight is given by $\rho_\sigma(u)$. The averaging effect increases with σ.

$\Psi_\sigma \xrightarrow{\sigma\to\infty}$ "mostly constant function"

(TAKE WITH 3 PINCHES OF SALT)

$\Psi_\sigma \xrightarrow{\sigma\to 0} \Psi$



We will minimise $J(\mathcal{R}; \Psi_\sigma)$ instead of $J(\mathcal{R}; \Psi)$.
In practice, this is done for a range of σ's.

MOTIVATION:

1.) ~~As mentioned early~~ It is challenging to properly learn $\nabla \log \Psi$ in the regions where $\Psi \approx 0$. This is because all our samples (training set) are from areas where $\Psi \gg 1$.
This poses a problem: when starting the generation algorithm from a random point (where $\Psi \approx 0$), we do not know well how to update the image. Our knowledge of $\nabla \log \Psi$ is ~~bad~~ poor in this region of $\mathbb{R}^d$. We thus learn $\nabla \log \Psi_\sigma$ for σ large first, and progressively decrease σ.

2.) Minimising $J(\mathcal{R}; \Psi_\sigma)$ is easier than minimising $K(\mathcal{R}; \Psi)$. Problem with $K(\mathcal{R}; \Psi)$: $\operatorname{div}(s_\mathcal{R})$ is costly to compute.
We use the following proposition.

PROP.: Minimising $J(\mathcal{R}; \Psi_\sigma)$ over $\mathcal{R}$ is $\Leftrightarrow$ to minimising $L(\mathcal{R}; \Psi_\sigma)$ over $\mathcal{R}$, where

$$L(\mathcal{R}; \Psi_\sigma) := \int_{\mathbb{R}^d} \int_{\mathbb{R}^d} \left| s_\mathcal{R}(u) - \nabla_u \log p_\sigma(u+w) \right|^2 p_\sigma(w-u) \Psi(w)^2 \, dw \, du$$

PROOF: see the book (Prop. 13.13, p. 181 in ~~the~~ ~~my~~ Github/drive version).

Again, $L(\mathcal{R}; \Psi_\sigma)$ is easily estimated from the training set:

$$L(\mathcal{R}; \Psi_\sigma) \approx \frac{1}{N} \sum_{i=1}^{N} \int_{\mathbb{R}^d} \left| s_\mathcal{R}(u_i) - \nabla_u \log p_\sigma(w-u) \Big|_{u=u_i} \right|^2 p_\sigma(w-u_i) \, dw.$$