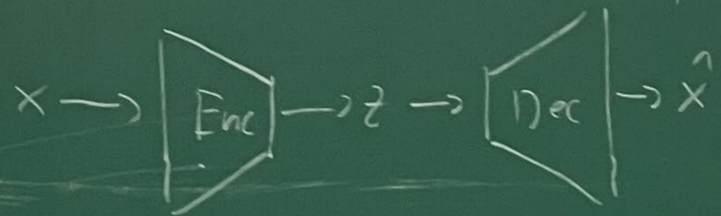
Autoencoder:

- "learned compression". $f(x)$ and $g(z)$ are neural networks
- lossy compression because usually $d = \dim(z) \ll \dim(x) = D$
  "bottleneck", d is a hyperparameter



train by reconstruction error

$$\hat{f}, \hat{g} = \arg\min_{f,g} \mathbb{E}_{x \sim p^*(x)} \left[\| x - g(f(x)) \|^2 \right]$$

- decoder manifold $\mathcal{M} = \{ x : x = g(z) \text{ for some } z \}$
  ↑ 1-dimensional
  if trained well: centerline of moons
- fibers: equivalence classes of data mapped to same code

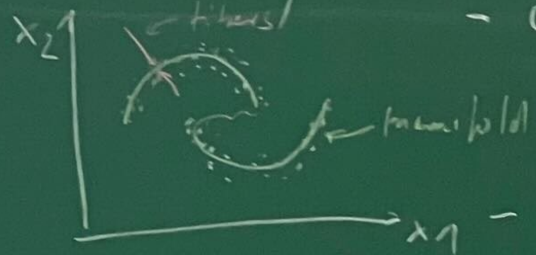
$$\mathcal{F}(z) = \{ x : f(x) = z \}$$

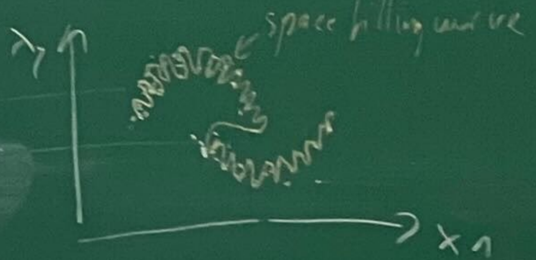
if trained well: orthogonal to centerline, & centerline is average of $\mathcal{F}(z)$

- example (homework)

two moons

$D = 2$
$d = 1$





- distance fcts. for the loss: – L_2 is most common
  – L_1: if applied to images, bit less blurry $\hat{x}$
  – multi-resolution L_p: compute image pyramid

- denoising autoencoder: add more noise to the data to make clear to the networks what noise is, i.e. what can be dropped

$$\tilde{x} = x + \varepsilon \qquad \varepsilon \sim \mathcal{N}(\varepsilon \,|\, 0, \sigma^2 \mathbb{I})$$

$$\hat{f}, \hat{g} = \underset{f,g}{\arg\min}\ \mathbb{E}_{x \sim p^*(x),\, \varepsilon \sim \mathcal{N}(\varepsilon | 0, \sigma^2 \mathbb{I})} \left[\| x - g(f(x + \varepsilon)) \|^2 \right]$$

often better, deep mathematical properties (later?)

- autoencoder is <u>not</u> a generative model (according to our definition)
  – no inference, i.e. no formula/algorithm to calculate $p(x)$ (later)
  – no generative capability, because we do not know $p_Z(z)$

to generate data with autoencoder:

① ex-post learning of $p_E(z)$ by a second generative model

often simpler than learning $p^*(x)$ directly, e.g. $d < D$

ex stable diffusion

② joint optimization – predefine $q(z)$ desired code dimension

– measure $MMD(p_E(z), q(z))$ ⟹ add new loss term

$$\hat{f}, \hat{g} = \arg\min_{f,g} \mathbb{E}_{x \sim p^*(x)} \left[\| x - g(f(x)) \|^2 \right] + \lambda \cdot MMD[f_\# p^*, q]$$

variant of InfoVAE [2019]