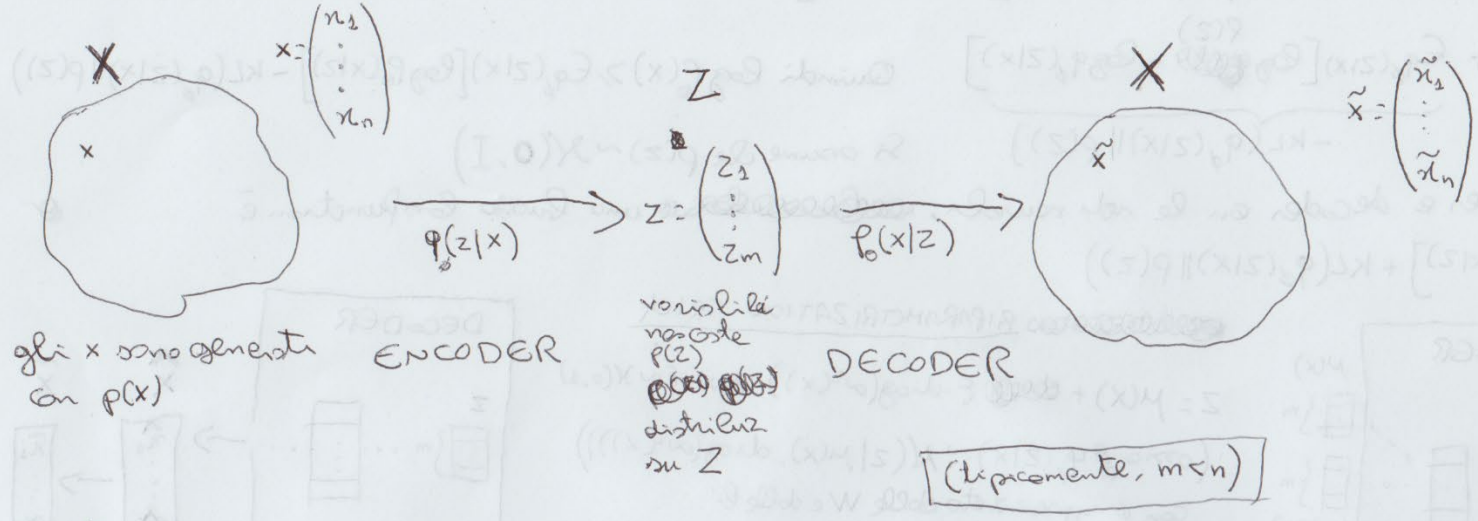


(tipicamente, $m < n$)

Sia $f : Z \times \Theta \to X$ una famiglia di funzioni (parametrizzata da un vettore θ su uno spazio Θ) | per semplicità $f_\theta(z) \equiv f(z, \theta)$

f_θ è deterministica, ma essendo z una variabile aleatoria, allora $f_\theta(z)$ è una variabile aleatoria nello spazio X

L'obiettivo è massimizzare la probabilità che venga generato ogni punto del training set di partenza (ossia la probabilità che $\tilde{x}$ sia il più vicino possibile ad x), al variare di θ. Cioè vorremo trovare $\max_\theta p_\theta(x)$

$p_\theta(x) = \int p_\theta(x|z) p(z) dz$ Si ipotizza che $p_\theta(x|z) \sim \mathcal{N}(x | f_\theta(z), \beta^2 I)$ (con β^2 parametro da determinare)

L'integrale è analiticamente intrattabile, pertanto si deve trovare un'altra soluzione.

Definiamo una nuova distribuzione di probabilità $q_\phi(z|x)$, cioè la probabilità di produrre z dato x come input, che si traduce nella probabilità sui valori di z che hanno probabilità di restituire x come output.

Riscriviamo $p_\theta(x)$ in un'altra forma: $p_\theta(x) = \int p_{\theta,\phi}(x,z) dz$ Massimizzare $p_\theta(x)$ equivale a massimizzare $\log p_\theta(x)$

$\log p_\theta(x) = \log \int p_{\theta,\phi}(x,z) dz = \log \int q_\phi(z|x) \cdot \frac{p_{\theta,\phi}(x,z)}{q_\phi(z|x)} dz = \log E_{q_\phi(z|x)}\left[\frac{p_{\theta,\phi}(x,z)}{q_\phi(z|x)}\right] \geq E_{q_\phi(z|x)}\left[\log \frac{p_{\theta,\phi}(x,z)}{q_\phi(z|x)}\right]$ (disuguaglianza di Jensen perché log è concava)

$$\mathbb{E}_{q_\phi(z|x)}\left[\log\left(\frac{p_{\theta,\phi}(x,z)}{q_\phi(z|x)}\right)\right] = \mathbb{E}_{q_\phi(z|x)}\left[\log\left(\frac{p_\theta(x|z)\cdot p(z)}{q_\phi(z|x)}\right)\right] = \mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x|z) + \log p(z) - \log q_\phi(z|x)\right] =$$

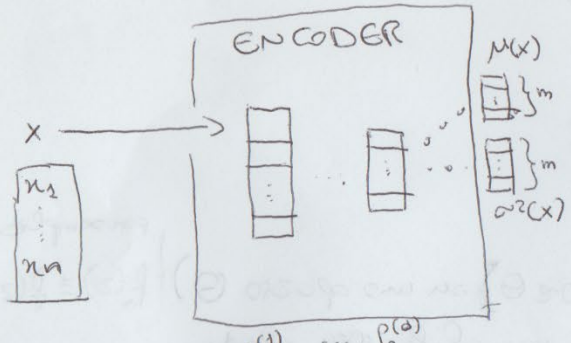
$$= \mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x|z)\right] + \underbrace{\mathbb{E}_{q_\phi(z|x)}\left[\log p(z) - \log q_\phi(z|x)\right]}_{-KL(q_\phi(z|x)\,\|\,p(z))}$$

Quindi $\log p_\theta(x) \geq \mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x|z)\right] - KL(q_\phi(z|x)\|p(z))$

Si assume che $p(z) \sim \mathcal{N}(0, I)$

Implementando encoder e decoder con le reti neurali, una buona loss function è

$-\mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x|z)\right] + KL(q_\phi(z|x)\|p(z))$



$h^{(1)} = ReLU(W^{(1)}x + b^{(1)})$

$h^{(d)} = ReLU(W^{(d)}h^{(d-1)} + b^{(d)})$

$\mu(x) = W_\mu h^{(d)} + b_\mu$

$\sigma^2(x) = W_\sigma h^{(d)} + b_{\sigma^2}$

(abbiamo usato ReLU ma può essere una qualsiasi funzione di attivazione, e la rete neurale può essere una CNN)

REPARAMETRIZATION TRICK

$z = \mu(x) + \varepsilon \cdot diag(\sigma^2(x))$ dove $\varepsilon \sim \mathcal{N}(0,1)$

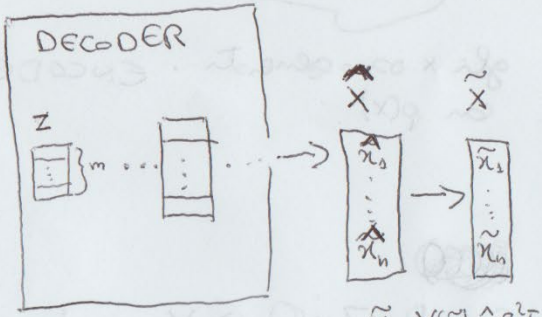
(ossia $q_\phi(z|x) \sim \mathcal{N}(z|\mu(x), diag(\sigma^2(x)))$)

con ϕ rappresentato dalle W e dalle b

Altro approccio:

$\lambda^{(x)} \equiv (\mu(x), \sigma(x))$

dati dai $\lambda^{(x)}$ sono parametri locali (dipendenti da x) da venellero appresi tramite la rete neurale



$\tilde{x} \sim \mathcal{N}(\tilde{x}|\hat{x}, \beta^2 I)$

Il decoder è un'altra rete neurale che implementa la funzione $f_\theta(z)$ (laddove θ è rappresentato dai pesi W di questa rete neurale)

$\hat{x} = f_\theta(z)$

e quindi

$\tilde{x} \sim \mathcal{N}(\tilde{x}|f_\theta(z), \beta^2 I)$

(β^2 può essere un iperparametro fisso, per esempio uguale ad 1, oppure può essere appreso)