Analisi di Immagini e Video (Computer Vision)

Giuseppe Manco

Outline

- Generative Modeling
  - Motivazioni
  - task
- Approcci
  - Variational Autoencoders
  - Generative Adversarial Networks

Crediti

- Slides adattate da altri corsi:
  - Ettore Ritacco (CS Unical)

Modelli Generativi

“What I cannot create, I do not understand.”

—Richard Feynman

Modelli Generativi

- Modelli probabilistici
  - Punto di partenza: x
    - Dato ad alta dimensionalità
  - Target: $P(x)$
    - Cattura la nozione d'incertezza sul dato
    - Densità
  - Obiettivo: generare (simulare) dati realistici
    - $x \sim P$
  - Strumento: parametrizzazione di P
    - $P(x) \approx P_\theta(x)$
    - $x \sim P_\theta$

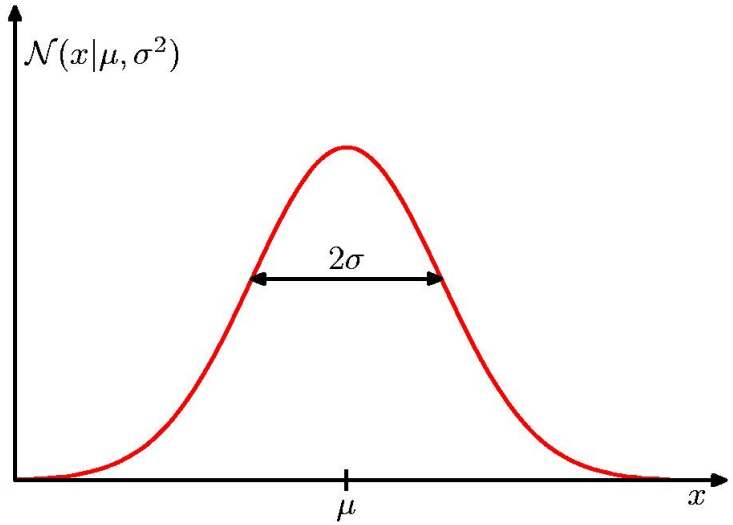
Modelli Generativi

- Tre componenti
  - Stima di parametri
    - Dati $\{x_1, \dots, x_n\}$ dove $x_i \sim P_{true}$
    - Trovare il θ ottimale
      - Il valore di θ per cui $P_\theta(x_i) \approx P_{true}(x_i)$
  - Inferenza
    - Calcolare $P(x|\theta)$
  - Generazione
    - Campionare $x \sim P(.|\theta)$

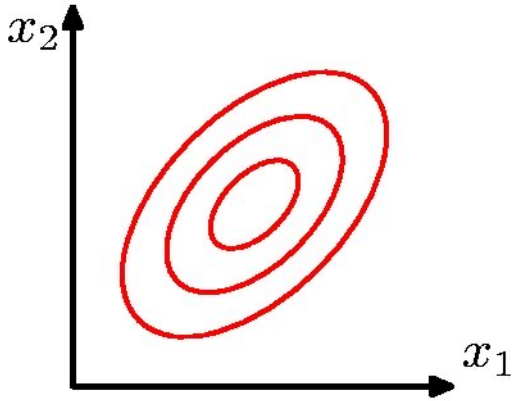
Esempio: Dati gaussiani

- $\theta = \{\mu, \Sigma\}$
- $x \sim \mathcal{N}(.|\mu, \Sigma)$

$$\mathcal{N}(x|\mu, \sigma^2) = \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left\{-\frac{1}{2\sigma^2}(x-\mu)^2\right\}$$



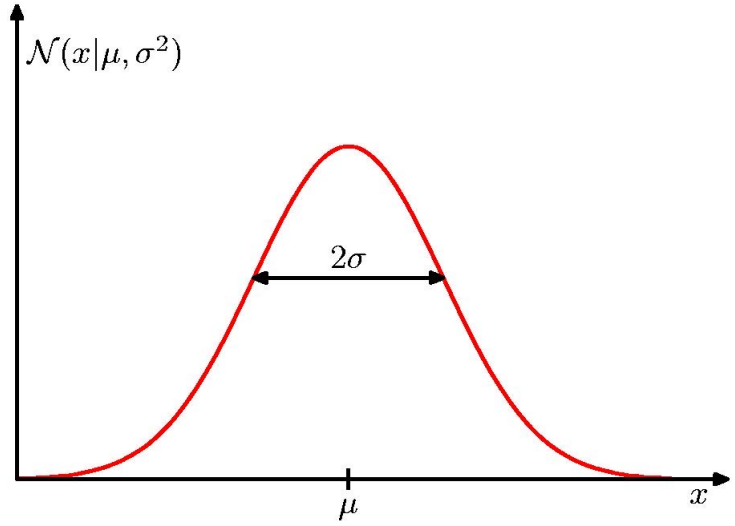
$$\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\boldsymbol{\Sigma}|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right\}$$



Esempio: Dati gaussiani

- Stima
  - Dati $\{x_1, \dots, x_n\}, \theta = \{\mu, \sigma\}$
    - $\mu = \frac{1}{n}\sum_i x_i$
    - $\sigma^2 = \frac{1}{n}\sum_i (x_i - \mu)^2$
- Inferenza
  - $p_\theta(x) = \mathcal{N}(x|\mu, \sigma)$
- Generazione
  - $x \sim p_\theta(.)$

$$\mathcal{N}(x|\mu, \sigma^2) = \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left\{-\frac{1}{2\sigma^2}(x-\mu)^2\right\}$$



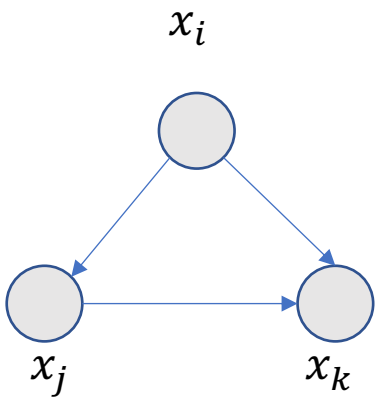
Esempio: immagini

Perché i modelli generativi?

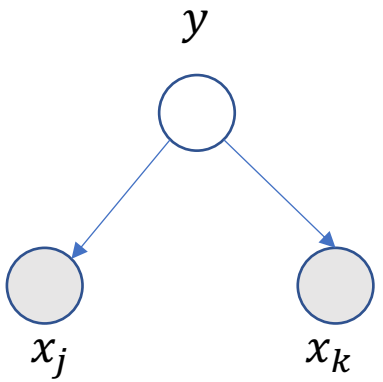
- Generazione di nuovi contenuti
  - Image completion
  - Data augmentation
- Image restoration
- Super-resolution
- Style transfer

Quali modelli generativi?

- Fully-observed models
  - Modellano le relazioni tra i dati
  - Variabili casuali

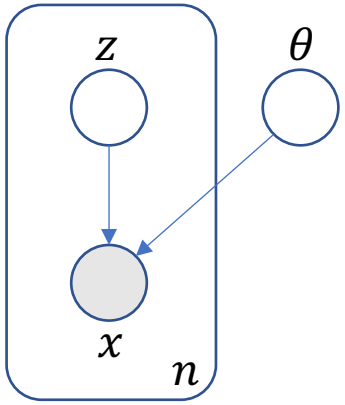


- Latent-Variable models
  - Variabili latenti associate ad ogni dato osservato
  - Esprimono causalità/correlazione nascosta con i dati osservati



Variational Autoencoders

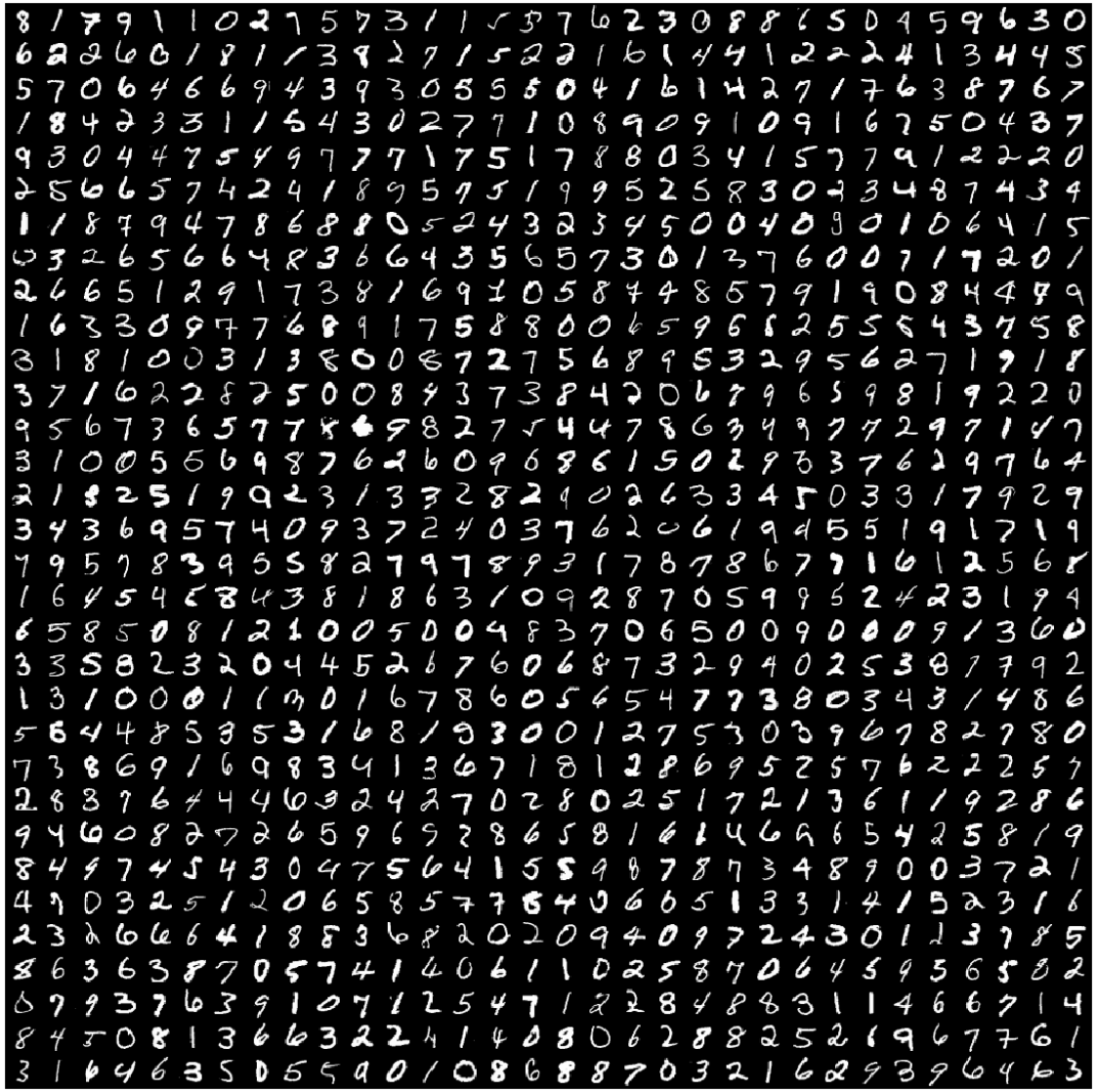
- Modello semplice
  - Assume che x sia generato condizionatamente ad una variabile latente
    - Influisce sul parametro θ



$$P(x) = \int P_\theta(x|z)P(z)\mathrm{d}z$$

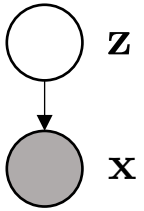
  - Generazione facile
  - Inferenza difficile…

Esempio: MNIST



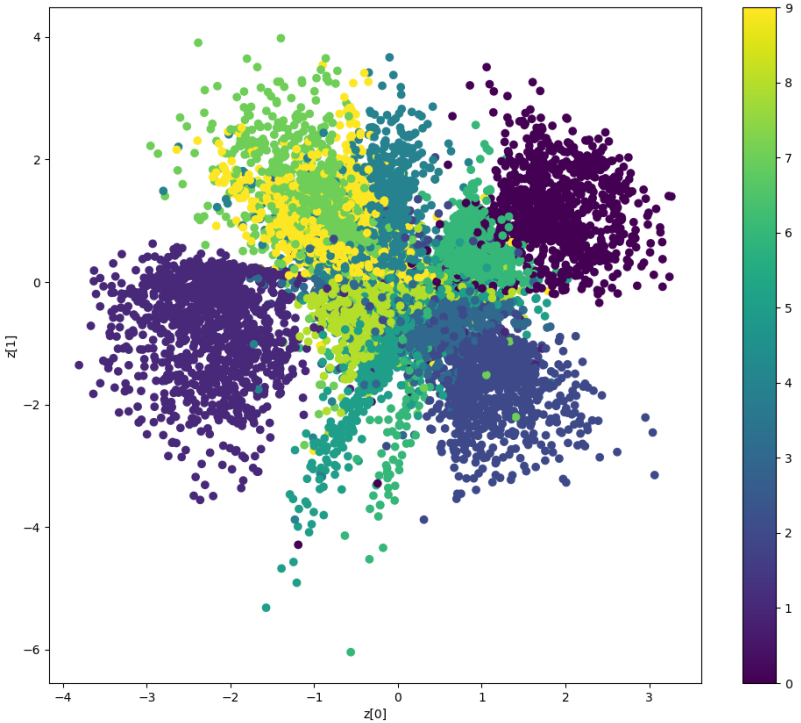
Generazione

- sorteggia $z \sim P_z$
- campiona $x \sim P_\theta(x|z)$

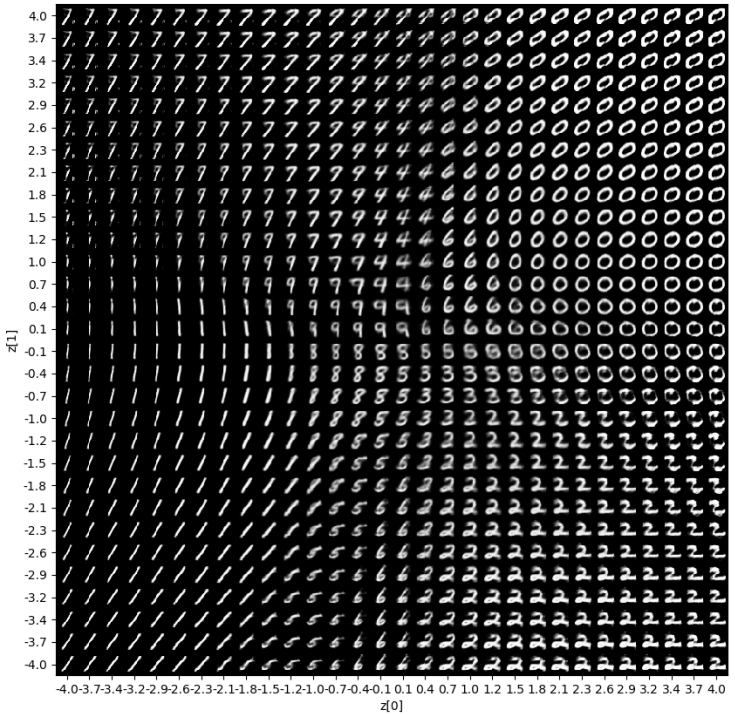


$$P(x) = \int P_\theta(x|z)P(z)\mathrm{d}z$$

$\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$



$\mathbf{x} \sim P_\theta(\mathbf{x}|\mathbf{z})$



Quali parametri per MNIST?

- Bernoullian model
  - $P_\theta(x|z) = \prod_{i,j} \theta_{ij}(z)^{x_{ij}} \left((1 - \theta_{ij}(z)\right)^{1-x_{ij}}$
    - $\theta_{ij}(z) \in [0,1]$ variabile bernoulliana che modella la probabilità bianco/nero
- Estensione: multinomial distribution
  - $P_\theta(x|z) = \prod_{i,j} \theta_{ij,k}(z)$
    - $\theta_{ij,k}(z)$ variabile multinomiale ($\sum_k \theta_{ijk}(z) = 1$) che modella la probabilità del colore k
- Gaussian model
  - $P_\theta(x|z) = \prod_{ij} \mathcal{N}(x, \mu(z), \sigma(z)I)$

Inferenza

- Problema
  - Dato x, bisogna calcolare

$$P(x) = \int P_\theta(x|z)P(z)\mathrm{d}z$$

  - Intrattabile!

Inferenza

- Approssimazione
- Metodi:
  - Monte Carlo (MC) Sampling
    - Approssimiamo l'integrale campionando dei valori di z e facendo averaging

$$P(x) = \int P_\theta(x|z)P(z)\mathrm{d}z \approx \sum_i P_\theta(x|z_i)P(z_i)$$

  - Variational Inference (VI)
    - Approssimiamo $P(x)$ in forma trattabile computazionalmente

Metodi Monte Carlo (MC)

- Una fimaglia di metodi di simulazione per inferire variabili target

- Algoritmo generale
  - Define a set of constraints or desiderata goals related to the target variables
  - Generate data from a suitable probability distribution related to the target variables
  - Check and evaluate how many successful trials you get out of all the experiments
  - Estimate your target variables

Metodi Monte Carlo (MC)

- Esempio: Stima di π
  - In uno spazio euclideo 2D, genera valori random $(x_i, y_i)_{i\in\{1,\dots,n\}}$ con una distribuzione uniforme su entrambi gli assi
  - Calcola m, il numero di punti che si trovano all'interno della circonferenza di raggio 1
    - Devono soddisfare $\sqrt{x^2 + y^2} \leq 1$
  - restituisci $4 \cdot \frac{m}{n}$

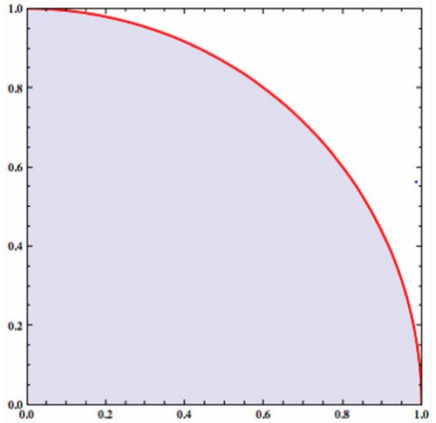
Metodi Monte Carlo (MC)

- Esempio: Stima di π
  - In uno spazio euclideo 2D, genera valori random $(x_i, y_i)_{i \in \{1,\dots,n\}}$ con una distribuzione uniforme su entrambi gli assi
  - Calcola m, il numero di punti che si trovano all'interno della circonferenza di raggio 1
    - Devono soddisfare $\sqrt{x^2 + y^2} \leq 1$
  - restituisci $4 \cdot \frac{m}{n}$

Che stiamo facendo?

Monte Carlo (MC)

- Esempio: Stima di π
  - In uno spazio euclideo 2D, genera valori random $(x_i, y_i)_{i\in\{1,\dots,n\}}$ con una distribuzione uniforme su entrambi gli assi
  - Calcola m, il numero di punti che si trovano all'interno della circonferenza di raggio 1
    - Devono soddisfare $\sqrt{x^2 + y^2} \leq 1$
  - restituisci $4 \cdot \frac{m}{n}$



Che stiamo facendo?

– stiamo approssimando quest'area

$$A = \frac{A_{sector}}{A_{square}} = A_{sector} = \frac{A_{circle}}{4} = \frac{\pi \cdot r^2}{4} = \frac{\pi}{4}$$

Metodi Monte Carlo (MC)

In [1]:
```
import numpy as np


def compute_pi(n_trials):
    n_successes = 0

    for _ in range(n_trials):
        x = np.random.rand()
        y = np.random.rand()

        if np.sqrt(x**2 + y**2) <= 1.:
            n_successes += 1

    return 4 * n_successes / n_trials


print(compute_pi(int(1e6)))
print(np.pi)
```

```
3.14368
3.141592653589793
```

Metodi Monte Carlo (MC)

- Limitazioni:
  - Il campionamento deve essere “semplice”
  - Servono molti dati
    - Non c’è convergenza asintotica

Idea

- Utilizziamo una proposal distribution $q_\lambda(z|x)$
- Utilizziamo i valori di z campionati da drawn from q_λ

Inferenza Variazionale

- Approssimiamo la posterior $P(z|x)$ con una famiglia di distribuzioni ”semplici” $q_\lambda(z|x)$

- Il parametro λ indicizza la famiglia di distribuzioni
  - Ad esempio, con q Gaussiana, $\lambda = \{\mu, \sigma\}$

$$P(z|x) = \frac{P(x|z) \cdot P(z)}{P(x)} \approx q_\lambda(z|x)$$

Inferenza Variazionale

- Due distribuzioni P e Q sono simili se la divergenza è bassa

- Divergenza di Kullback – Leibler:

$$KL(P||Q) = \int_{-\infty}^{+\infty} p(\omega) \cdot \ln \frac{p(\omega)}{q(\omega)} d\omega$$

$$= \int_{-\infty}^{+\infty} p(\omega) \cdot \ln p(\omega)\, d\omega - \int_{-\infty}^{+\infty} p(\omega) \cdot \ln q(\omega)\, d\omega$$

$$= \mathbb{E}_{x\sim p}[\ln p\,(x)] - \mathbb{E}_{x\sim p}[\ln q\,(x)]$$

Inferenza Variazionale

- Vogliamo una $q_\lambda(z|x)$ che sia una buona approssimazione di $p(z|x)$

$$KL\big(q_\lambda(z|x)||p(z|x)\big) = \mathbb{E}_{q_\lambda(z|x)}[\ln q_\lambda(z|x)] - \mathbb{E}_{q_\lambda(z|x)}[\ln p\,(z|x)]$$

$$= \mathbb{E}_{q_\lambda(z|x)}[\ln q_\lambda(z|x)] - \mathbb{E}_{q_\lambda(z|x)}\left[\ln \frac{p(z,x)}{p(x)}\right]$$

$$= \mathbb{E}_{q_\lambda(z|x)}[\ln q_\lambda(z|x)] - \mathbb{E}_{q_\lambda(z|x)}[\ln p\,(z,x)] + \mathbb{E}_{q_\lambda(z|x)}[\ln p(x)]$$

$$= \mathbb{E}_{q_\lambda(z|x)}[\ln q_\lambda(z|x)] - \mathbb{E}_{q_\lambda(z|x)}[\ln p\,(z,x)] + \ln p(x)$$

$$= -\big\{\mathbb{E}_{q_\lambda(z|x)}[\ln p\,(z,x)] - \mathbb{E}_{q_\lambda(z|x)}[\ln q_\lambda(z|x)]\big\} + \ln p(x)$$

ELBO (evidence lower bound)

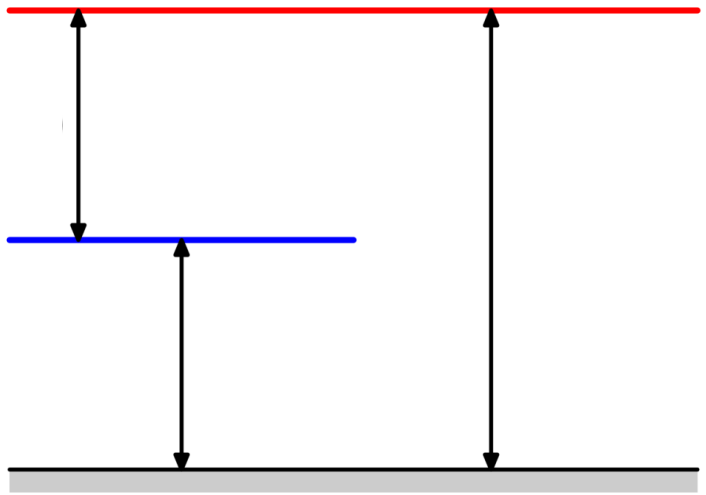
Log likelihood

$$
\begin{aligned}
\log P(x) &= \int q(z|x) \log P(x) \mathrm{d}z \\
&= \int q(z|x) \log P(x) \frac{P(z|x)}{P(z|x)} \mathrm{d}z \\
&= \int q(z|x) \log P(x) \frac{P(x|z)P(z)}{P(x)P(z|x)} \mathrm{d}z \\
&= \int q(z|x) \log P(x|z)P(z) \mathrm{d}z + \int q(z|x) \log \frac{q(z|x)}{q(z|x)P(z|x)} \mathrm{d}z \\
&= \int q(z|x) \log P(x|z) \mathrm{d}z + \int q(z|x) \log \frac{q(z|x)}{P(z|x)} \mathrm{d}z \\
&\qquad - \int q(z|x) \log \frac{q(z|x)}{P(z)} \mathrm{d}z \\
&= E_{z\sim q}[\log P(x|z)] + KL[q(z|x)||P(z|x)] \\
&\qquad -KL[q(z|x)||P(z)]
\end{aligned}
$$

Inferenza variazionale

$$KL[q_\lambda(z|x)||p(z|x)] = -ELBO(q_\lambda) + \ln p(x)$$

- Scelto q_λ, si stimano I parametri λ che minimizzano l'ELBO



Bishop – Pattern Recognition and Machine Learning

$$KL[q_\lambda(z|x)||p(z|x)] = E_{z\sim q_\lambda}[\log q_\lambda(z|x) - \log p(x|z)]$$

$$KL[q_\lambda(z|x)||p(z|x)] = E_{z\sim q_\lambda}[\log q_\lambda(z|x) - \log p(x|z)]$$
$$= KL[q_\lambda(z|x)||P(z)] - E_{z\sim q_\lambda}[p(x|z)] + \log P(x)$$

$$KL[q_\lambda(z|x)||p(z|x)] = E_{z\sim q_\lambda}[\log q_\lambda(z|x) - \log p(x|z)]$$
$$= KL[q_\lambda(z|x)||P(z)] - E_{z\sim q_\lambda}[p(x|z)] + \log P(x)$$

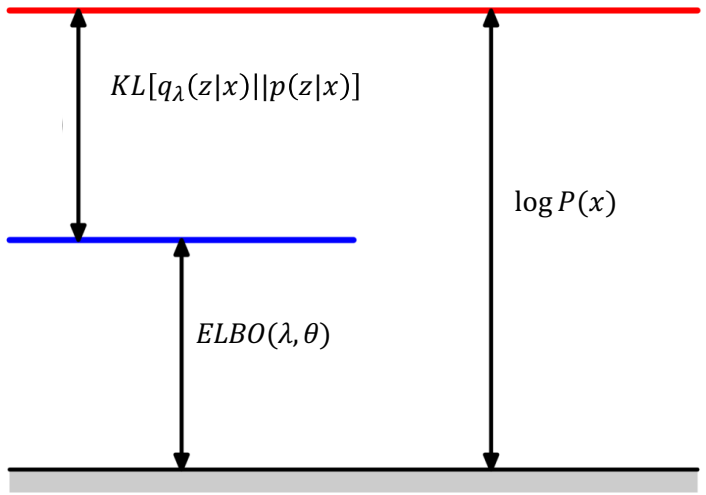
Evidence Lower Bound (ELBO)

Non dipende da z

$$KL[q_\lambda(z|x)||p(z|x)] = E_{z\sim q_\lambda}[\log q_\lambda(z|x) - \log p(x|z)]$$
$$= KL[q_\lambda(z|x)||P(z)] - E_{z\sim q_\lambda}[\log p(x|z)] + \log P(x)$$

Evidence Lower Bound (ELBO)

Non dipende da z

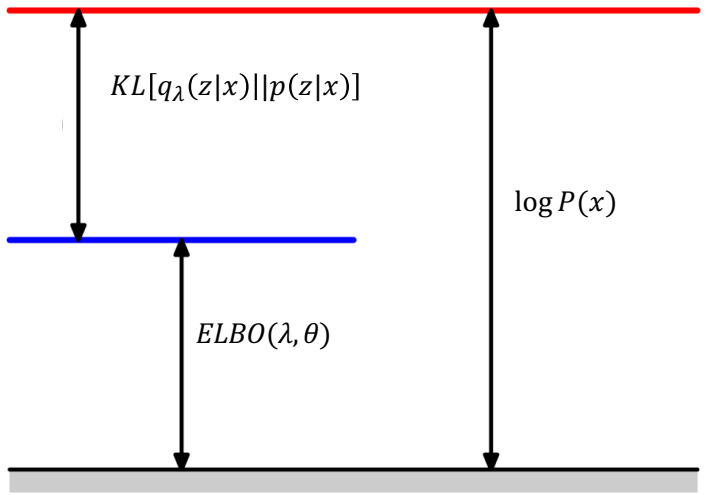


Bishop – Pattern Recognition and Machine Learning

Inferenza variazionale

$$KL[q_\lambda(z|x)||p(z|x)] = -ELBO(\lambda,\theta) + \ln p(x)$$

- Scelto q_λ, si stimano I parametri λ, θ che minimizzano l'ELBO



Bishop – Pattern Recognition and Machine Learning

Wrap up

- Si definisce una distribuzione q_λ
- Troviamo i parametri λ e θ che massimizzano

$$KL[q_\lambda(z|x)||P(z)] - E_{z\sim q_\lambda}[p(x|z)]$$

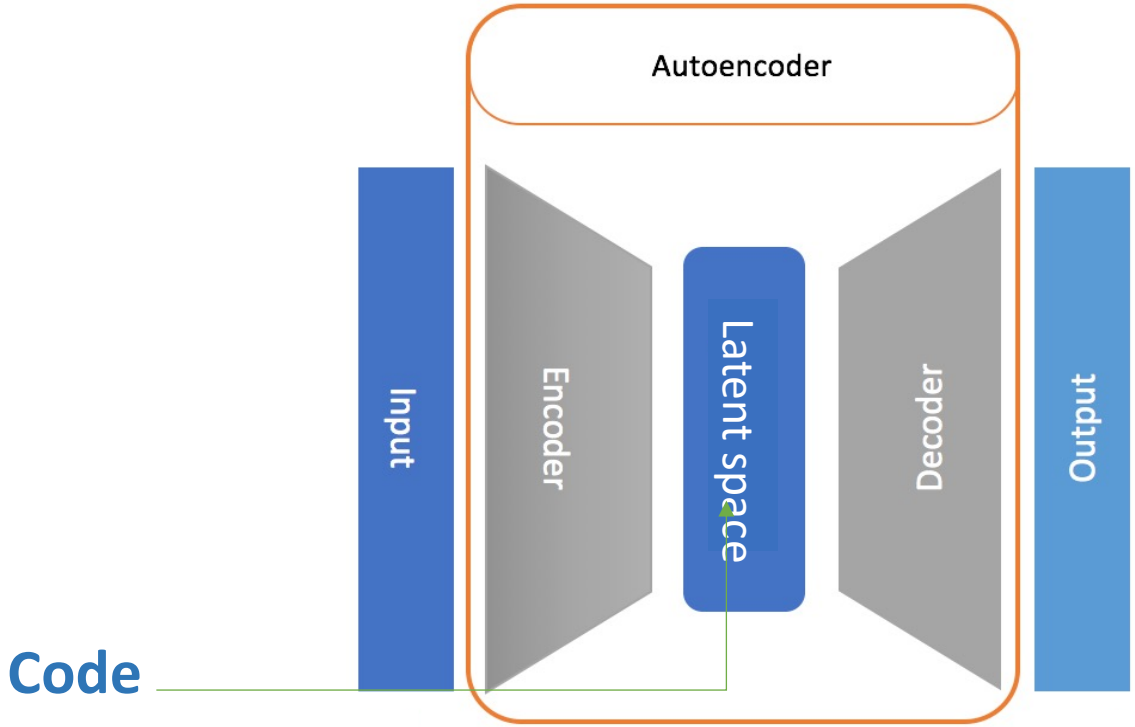
Wrap up

- Il framework dipende da due funzionali
  - $q_\lambda(z|x)$, che codifica x in una variabile latente z
  - $p(x|z)$, che decodifica z in x
- Autoencoder

Autoencoder

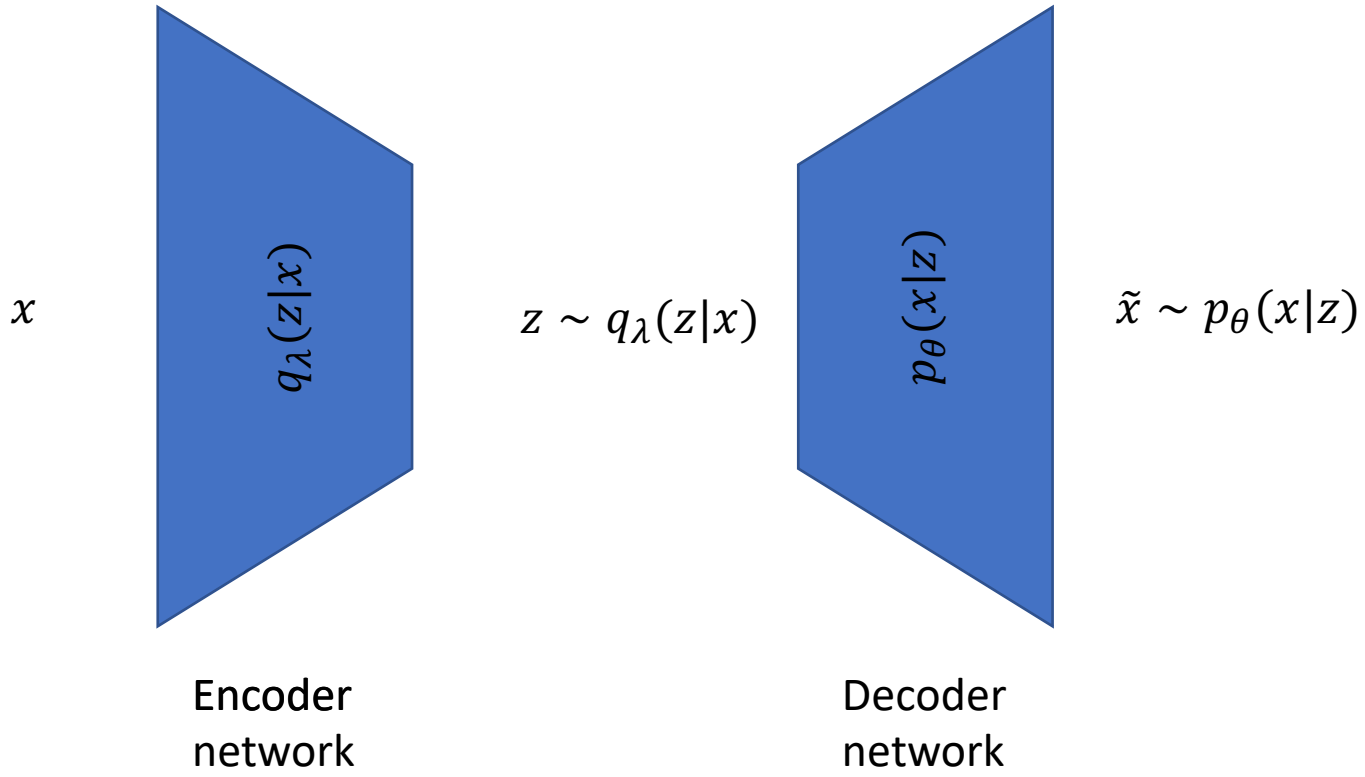
- un **autoencoder** è una rete neurale che produce un output (r) che è un **duplicato** dell'input (x).

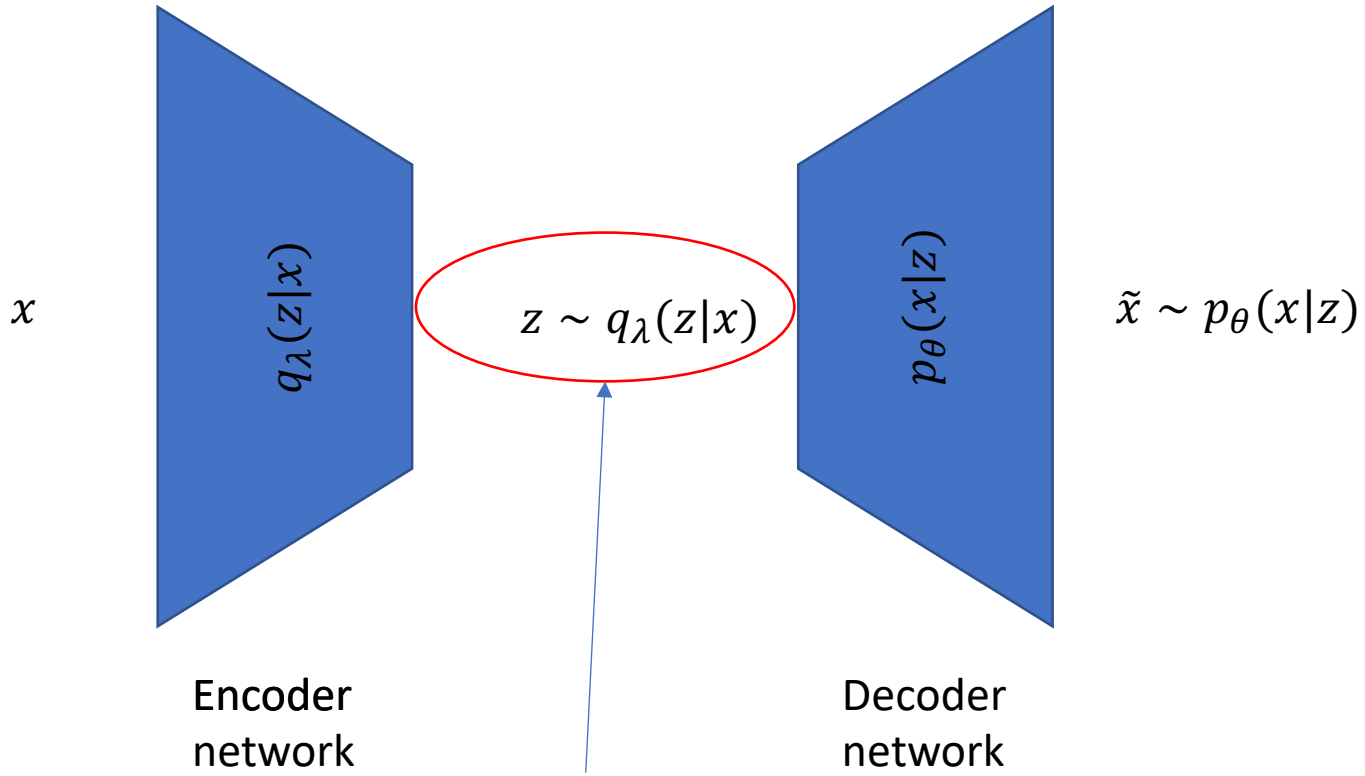
- Due componenti:
  - **encoder** function $z = f(x)$
  - z è il **Codice** (dell'input)
  - **decoder** che produce una ricostruzione $r = g(z)$
  - L'**obiettivo** è $x \approx r$
  - la **loss** espressa come $L\left(x, g\left(f(x)\right)\right)$



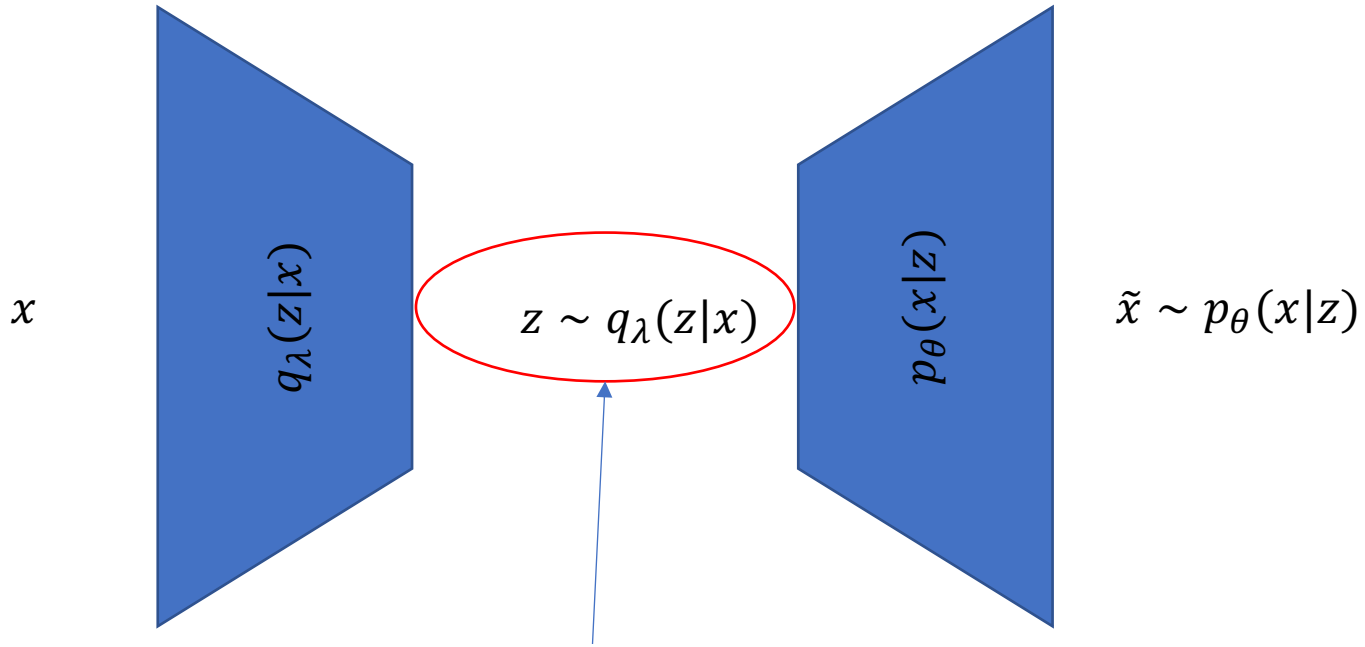
Proprietà

- Impara a sommarizzare le caratteristiche dell’input
- Rappresenta I dati originali in uno spazio a bassa dimensionalità

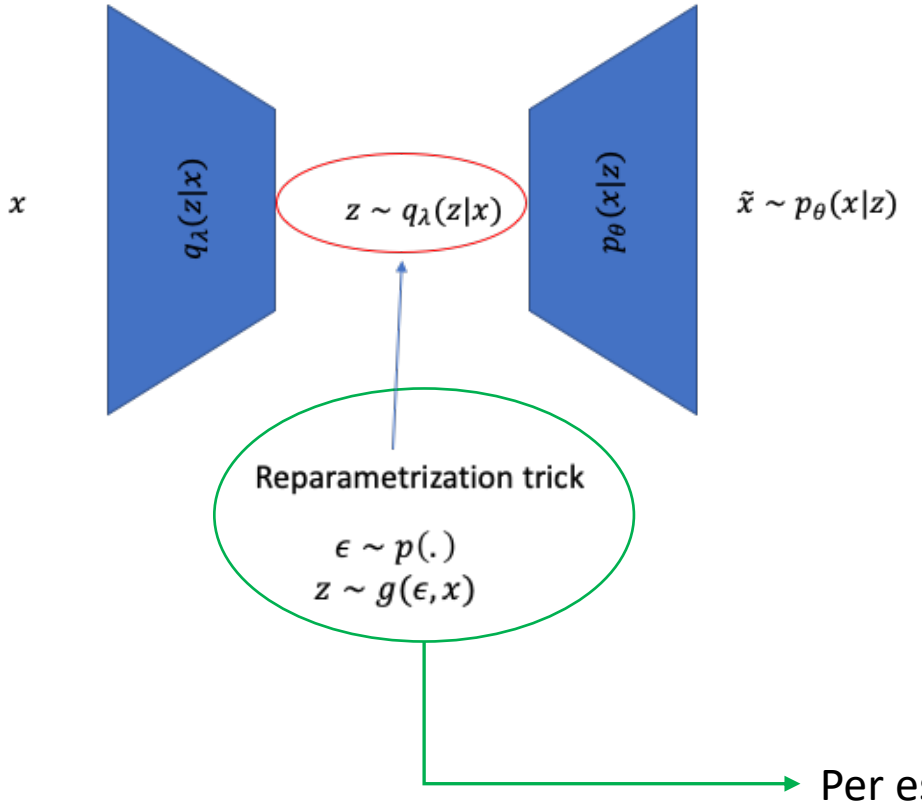

x
$q_\lambda(z|x)$
$z \sim q_\lambda(z|x)$
$p_\theta(x|z)$
$\tilde{x} \sim p_\theta(x|z)$
Encoder network
Decoder network



Come si ottimizza
Evitando di spezzare
La propagazione del gradiente?

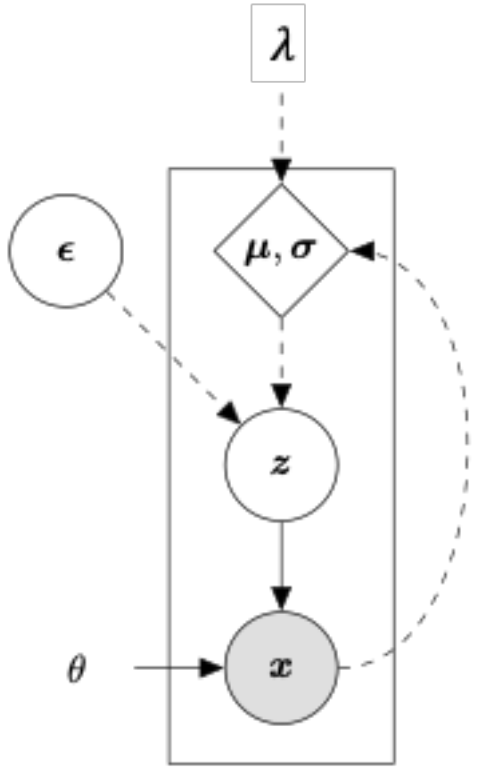


$$\epsilon \sim p(.)$$
$$z \sim g(\epsilon, x)$$



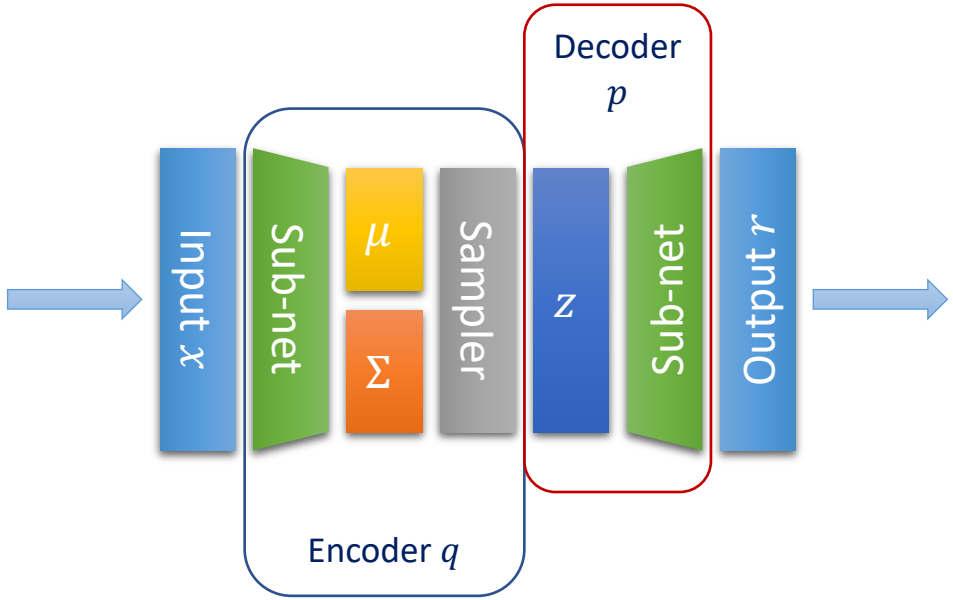
Per esempio,

$$\epsilon \sim \mathcal{N}(0, I)$$
$$\mu, \sigma = mlp(x)$$
$$z = \mu + \sigma \cdot \epsilon$$

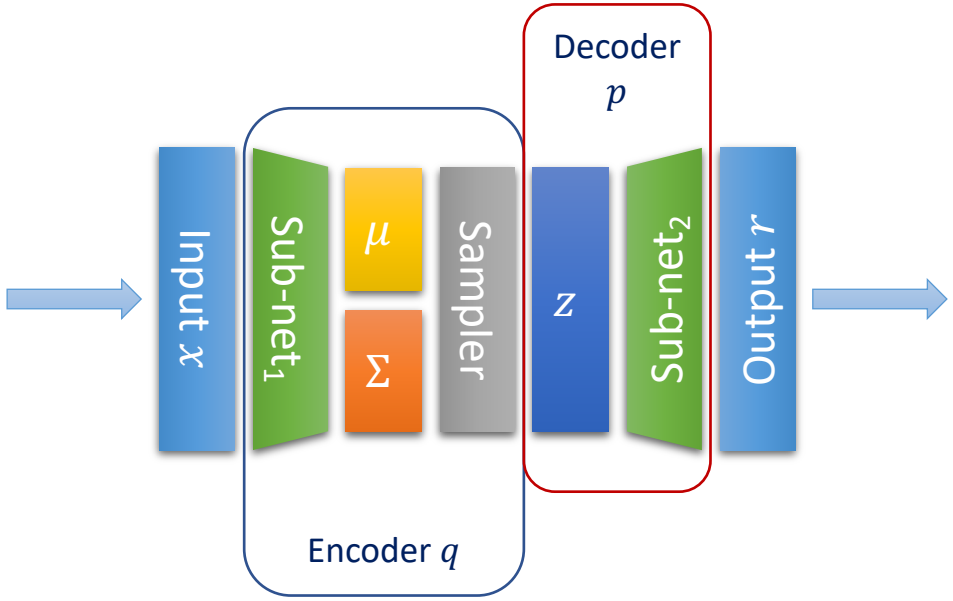


Variational Autoencoders

- Architettura generale



Variational Autoencoders



- La sottorete sub-net$_1$ calcola i parametri di una distribuzione gaussiana
- Il sampler sfrutta I parametri per campionare il codice
- La sub-net$_2$ genera l'output

Variational Autoencoders

- Cosa manca?

Variational Autoencoders

- Cosa manca?

$$ELBO(\lambda, \theta) = KL[q_\lambda(z|x)||P(z)] - E_{z\sim q_\lambda}[\log p(x|z)]$$

- Ammette una forma chiusa?

Variational Autoencoders

- Cosa manca?

$$ELBO(\lambda, \theta) = KL[q_\lambda(z|x)||P(z)] - E_{z\sim q_\lambda}[\log p(x|z)]$$

- Log-likelihood

Variational Autoencoders

- Cosa manca?

$$ELBO(\lambda, \theta) = KL[q_\lambda(z|x)||P(z)] - E_{z\sim q_\lambda}[\log p(x|z)]$$

- Se q_λ e P sono gaussiane ammette una forma chiusa

Variational Autoencoders

- Cosa manca?

$$ELBO(\lambda, \theta) = KL[q_\lambda(z|x)||P(z)] - E_{z\sim q_\lambda}[\log p(x|z)]$$

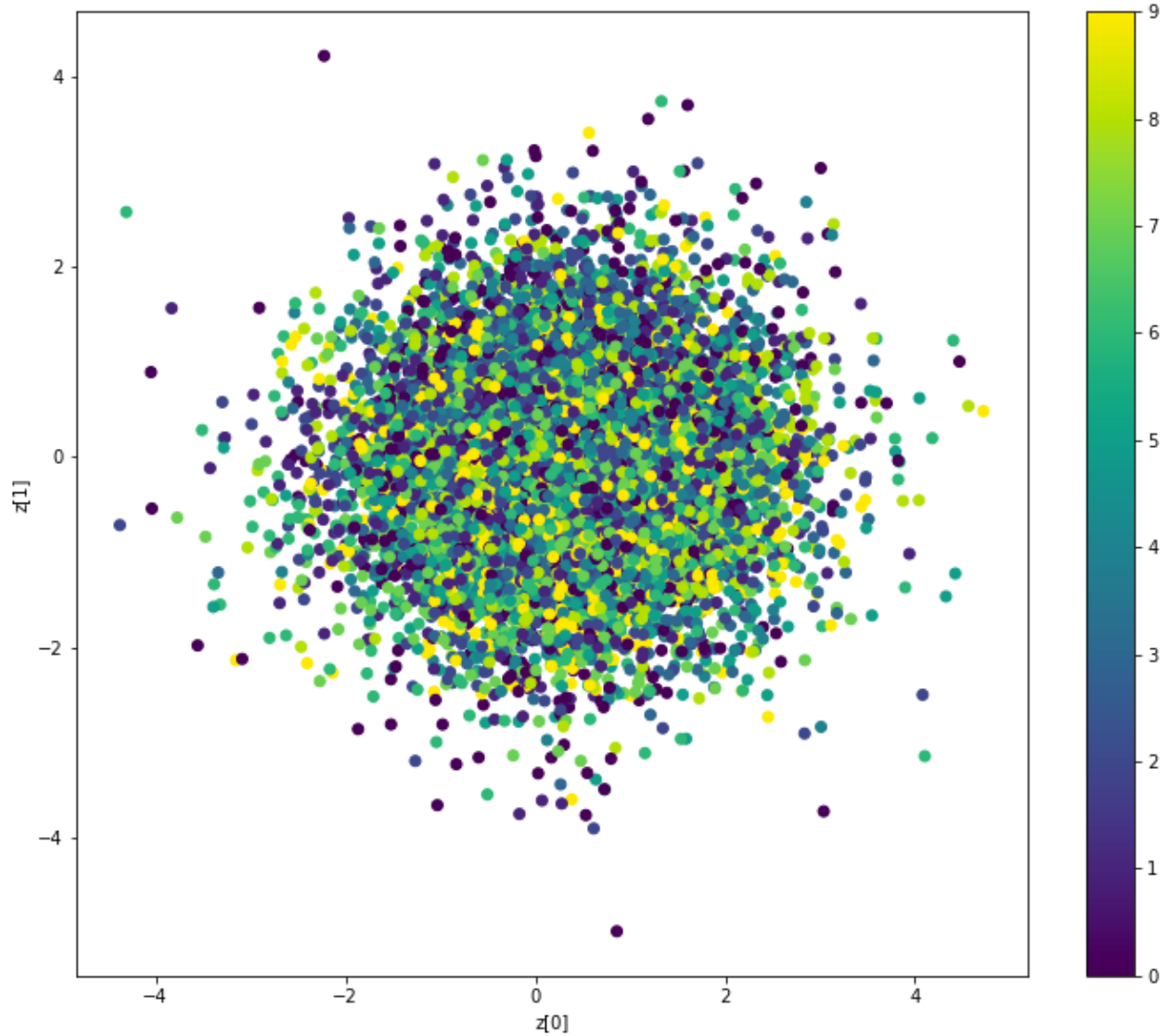
- Se q_λ e P sono gaussiane ammette una forma chiusa

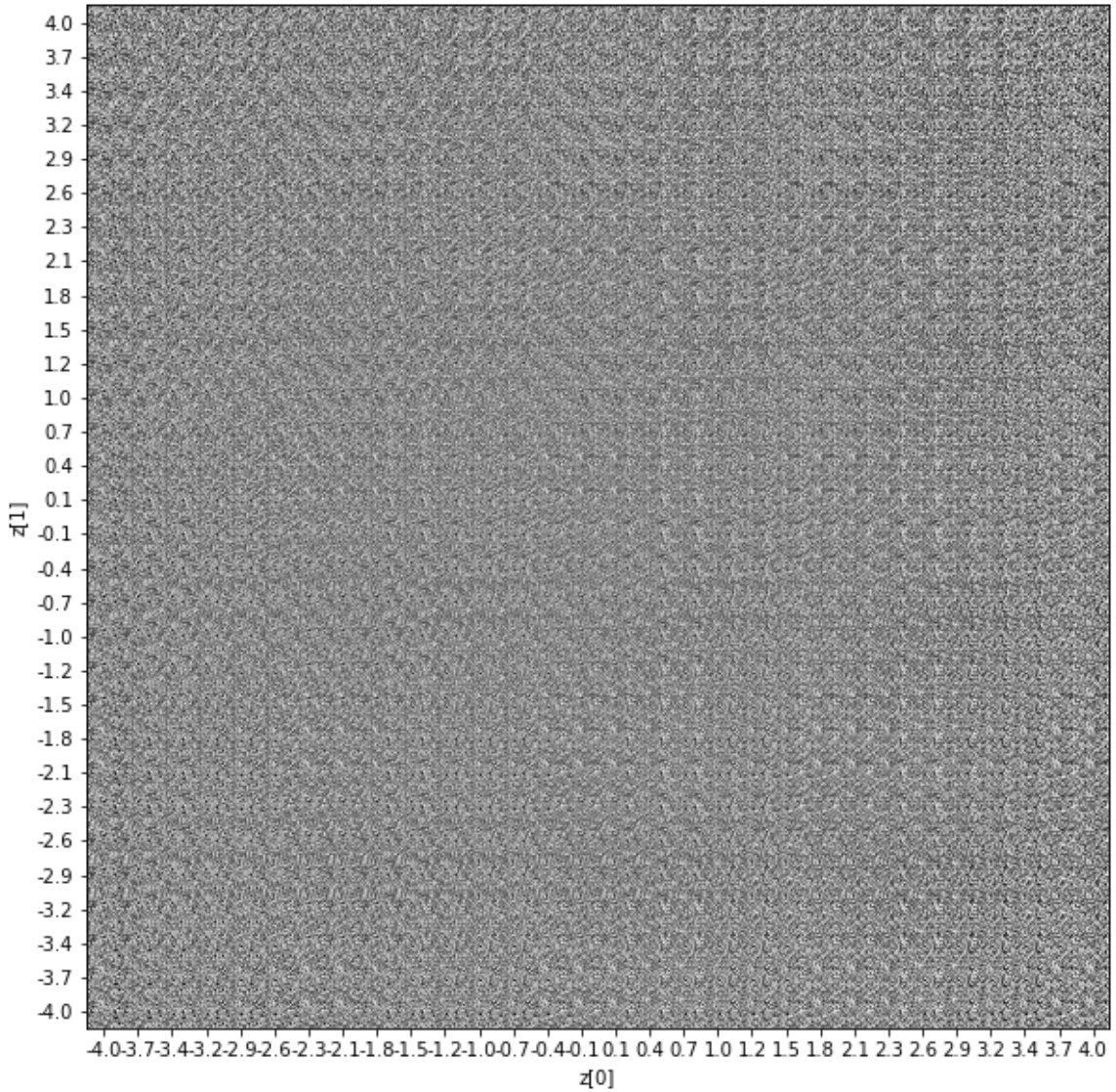
$$KL\left[\mathcal{N}(\boldsymbol{\mu}, \Sigma)||\mathcal{N}(\boldsymbol{m}, S)\right] =$$
$$\frac{1}{2}\left(log\frac{|S|}{|\Sigma|} - K + tr\left(S^{-1}\Sigma\right) + (\boldsymbol{m}-\boldsymbol{\mu})^T S^{-1}(\boldsymbol{m}-\boldsymbol{\mu})\right)$$

Recap: MC vs VI

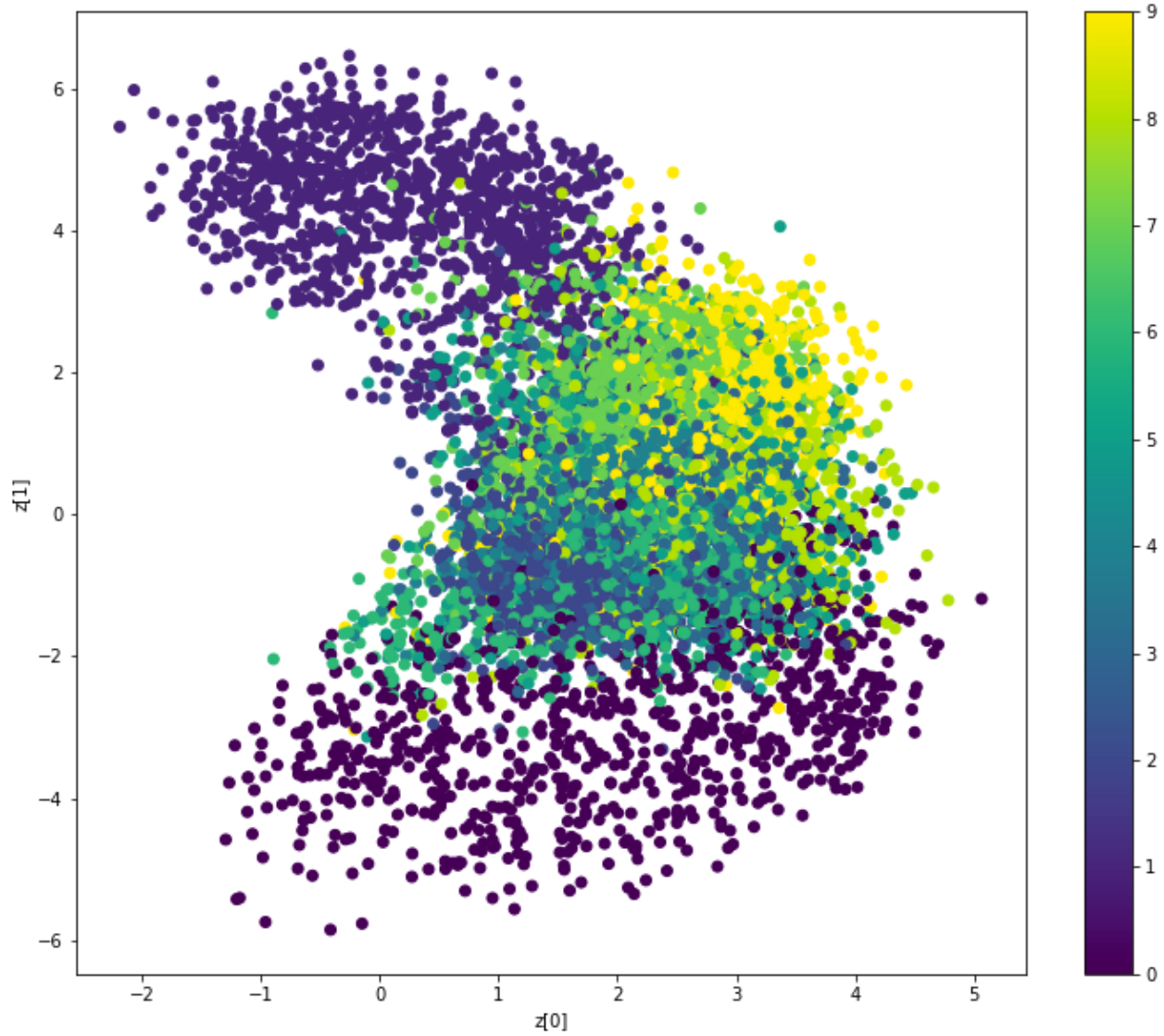
- Assunzioni sulla forma delle distribuzioni
  - MCMC: **no**
  - VI: **yes**
- Bias (dobbiamo forzare delle assunzioni?)
  - MCMC: **low**
  - VI: **high**
- Variance (quanto è grande lo spazio di ricerca?)
  - MCMC: **high**
  - VI: **low**
- Computational cost
  - MCMC: **high (lots of iterations)**
  - VI: **low**
- Accuracy
  - MCMC: **very good**
  - VI: **good**

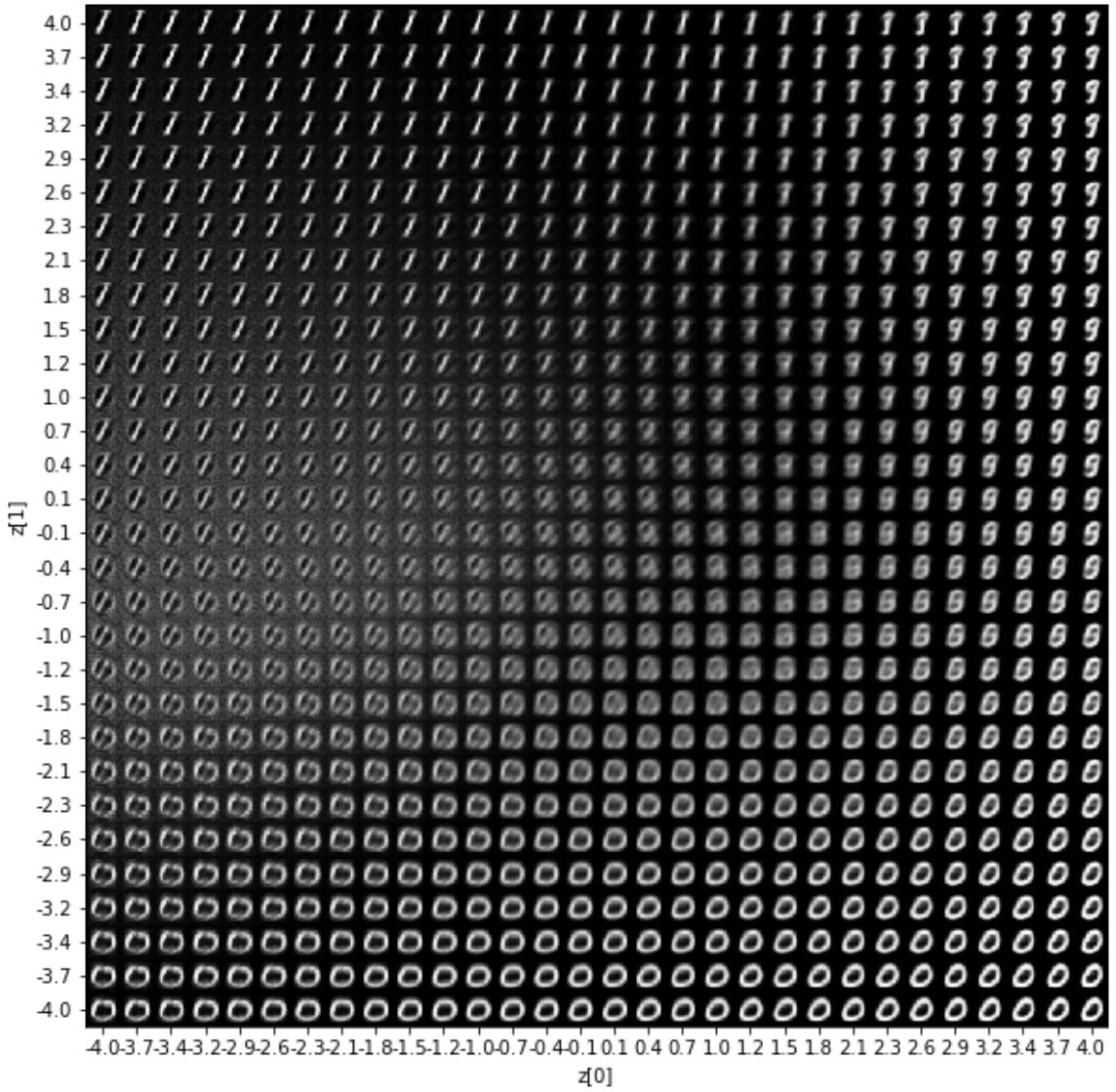
MNIST – epoca 0



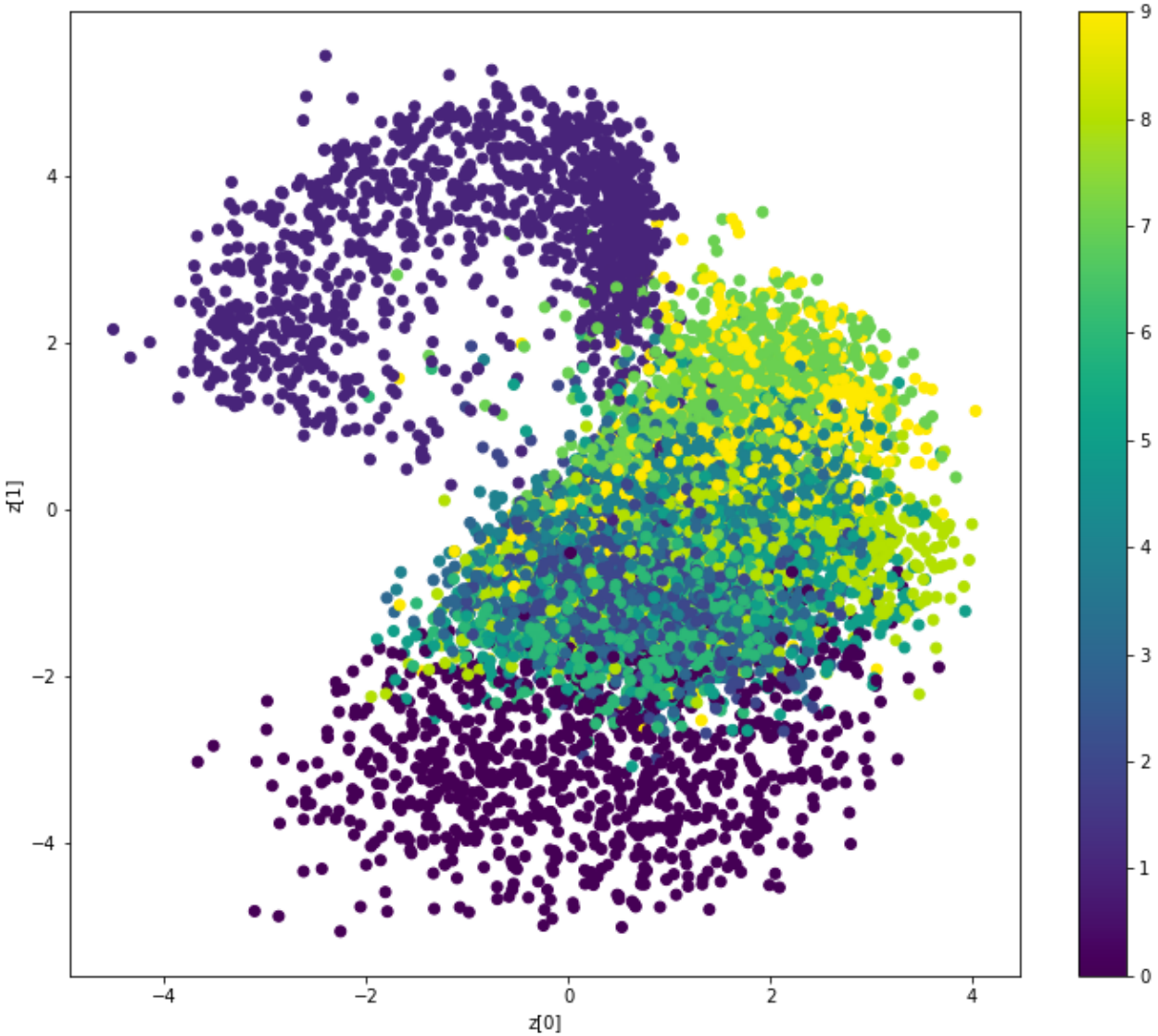


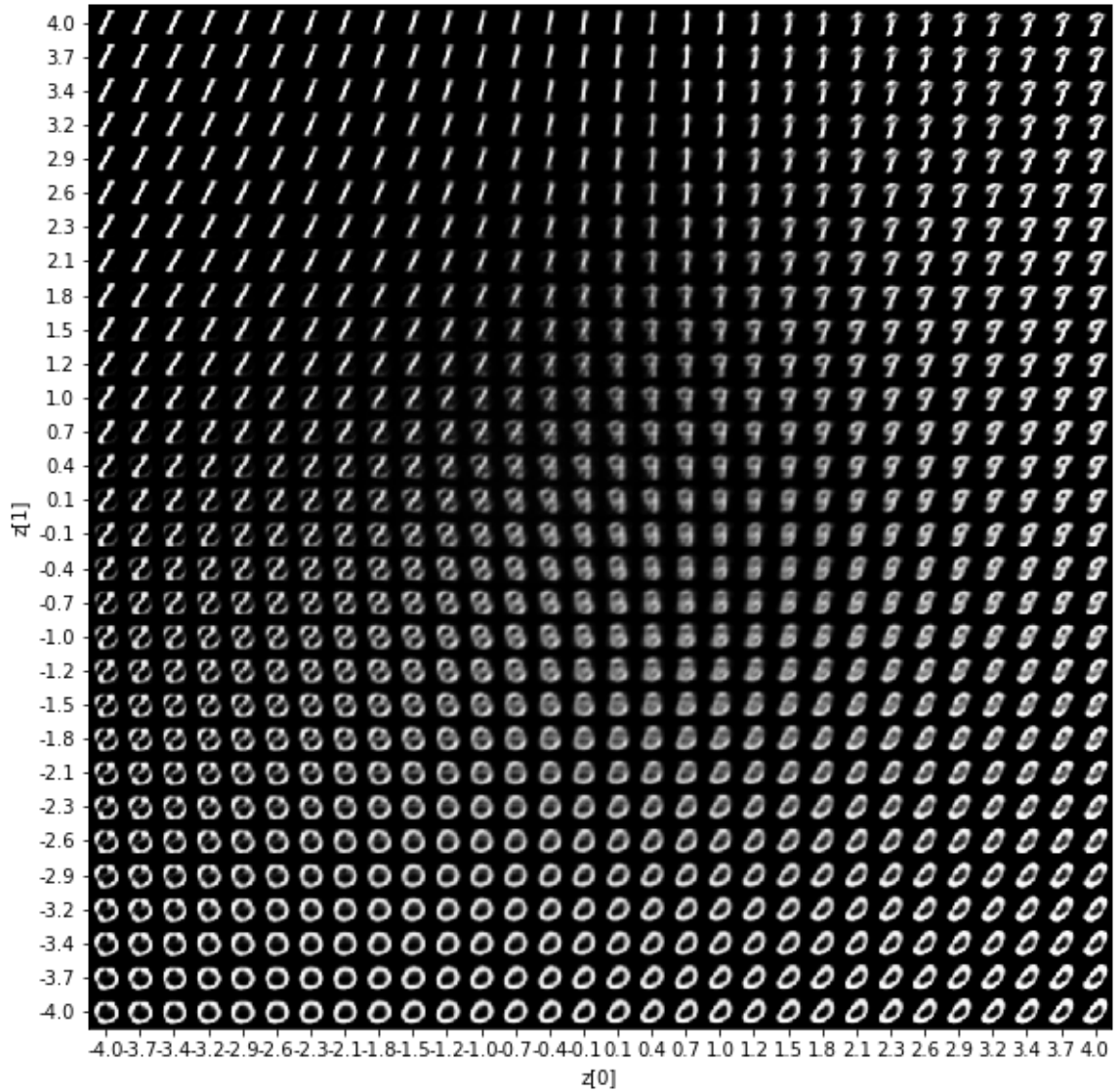
MNIST – epoca 10



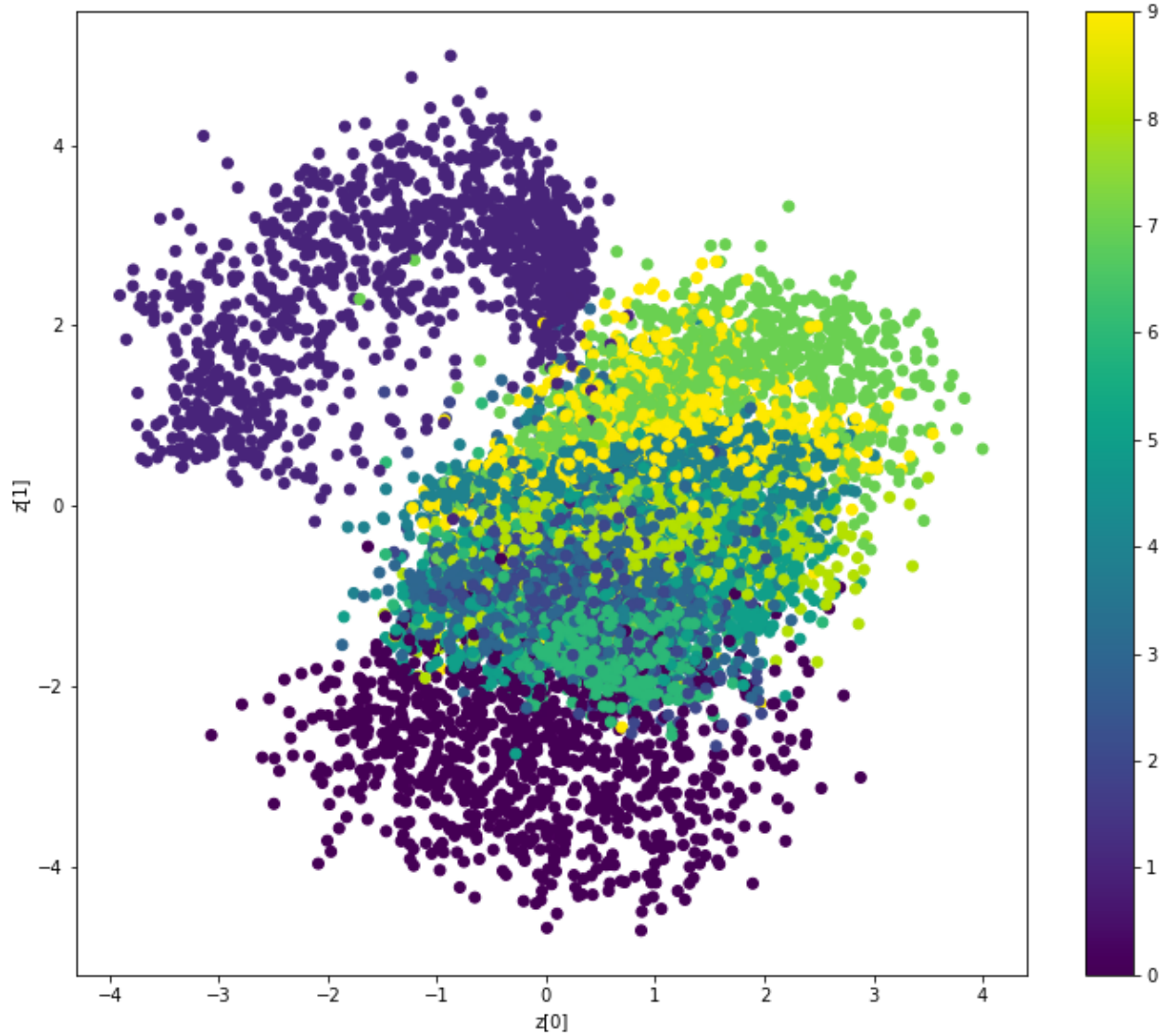


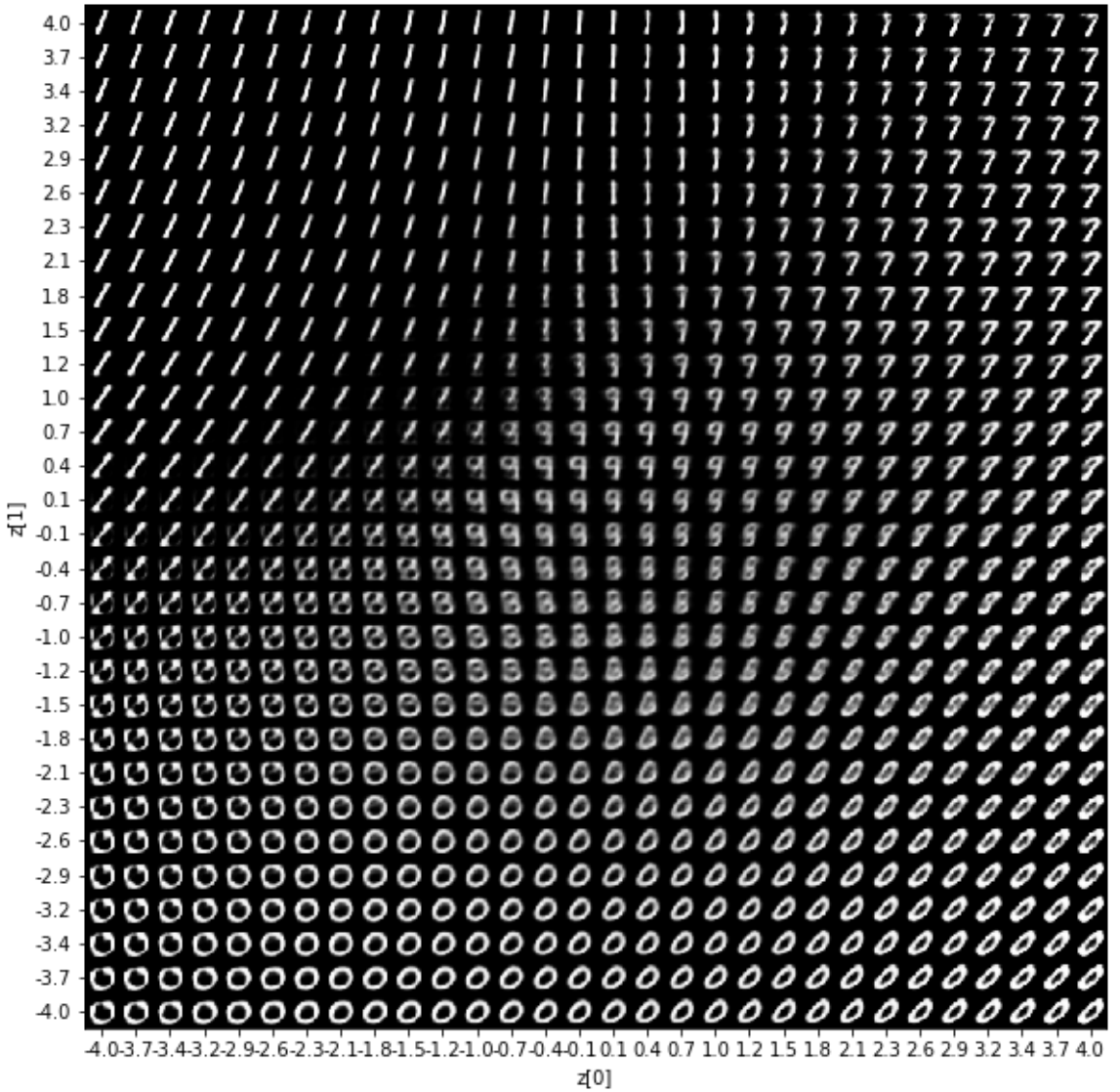
MNIST – epoca 20



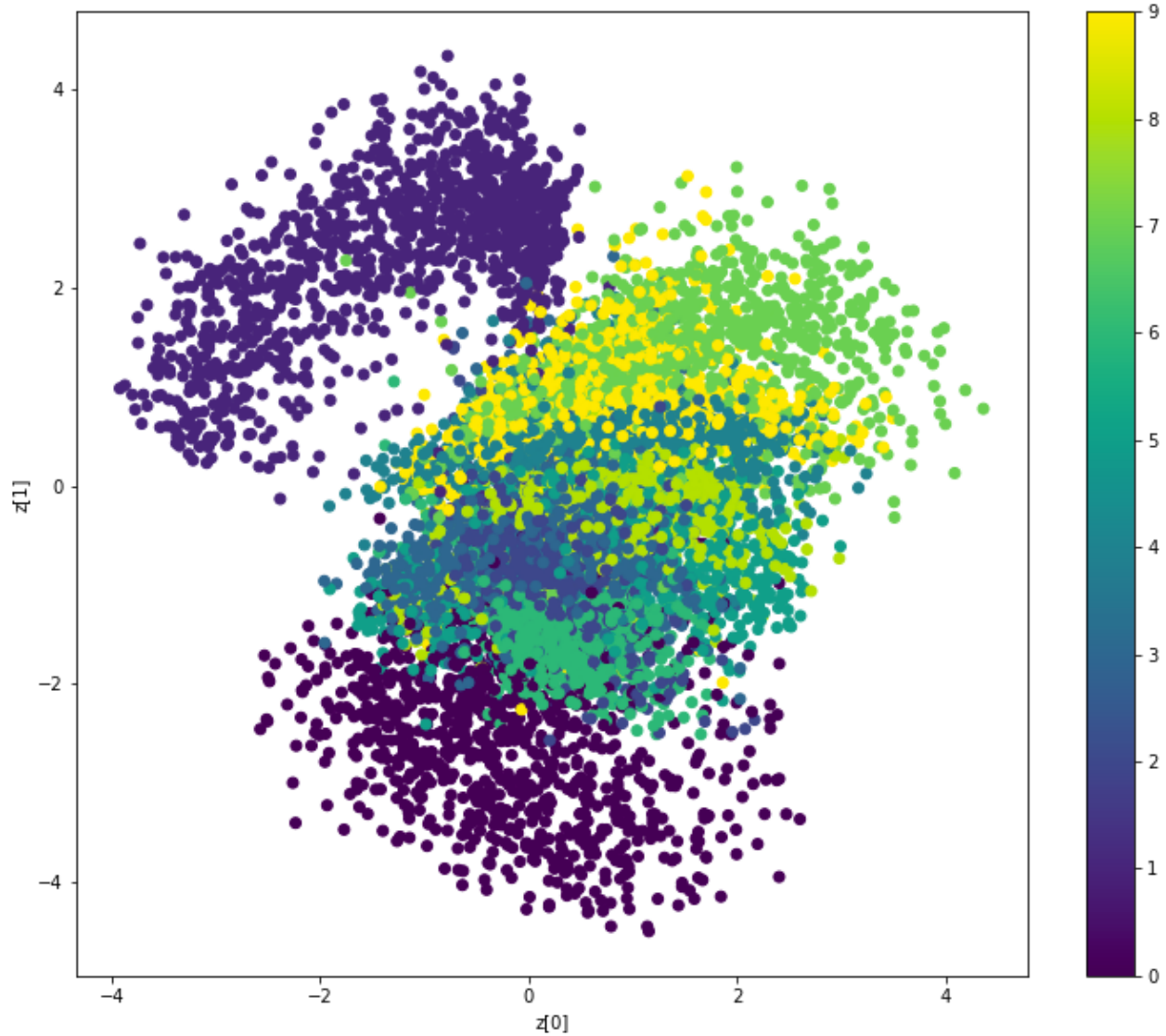


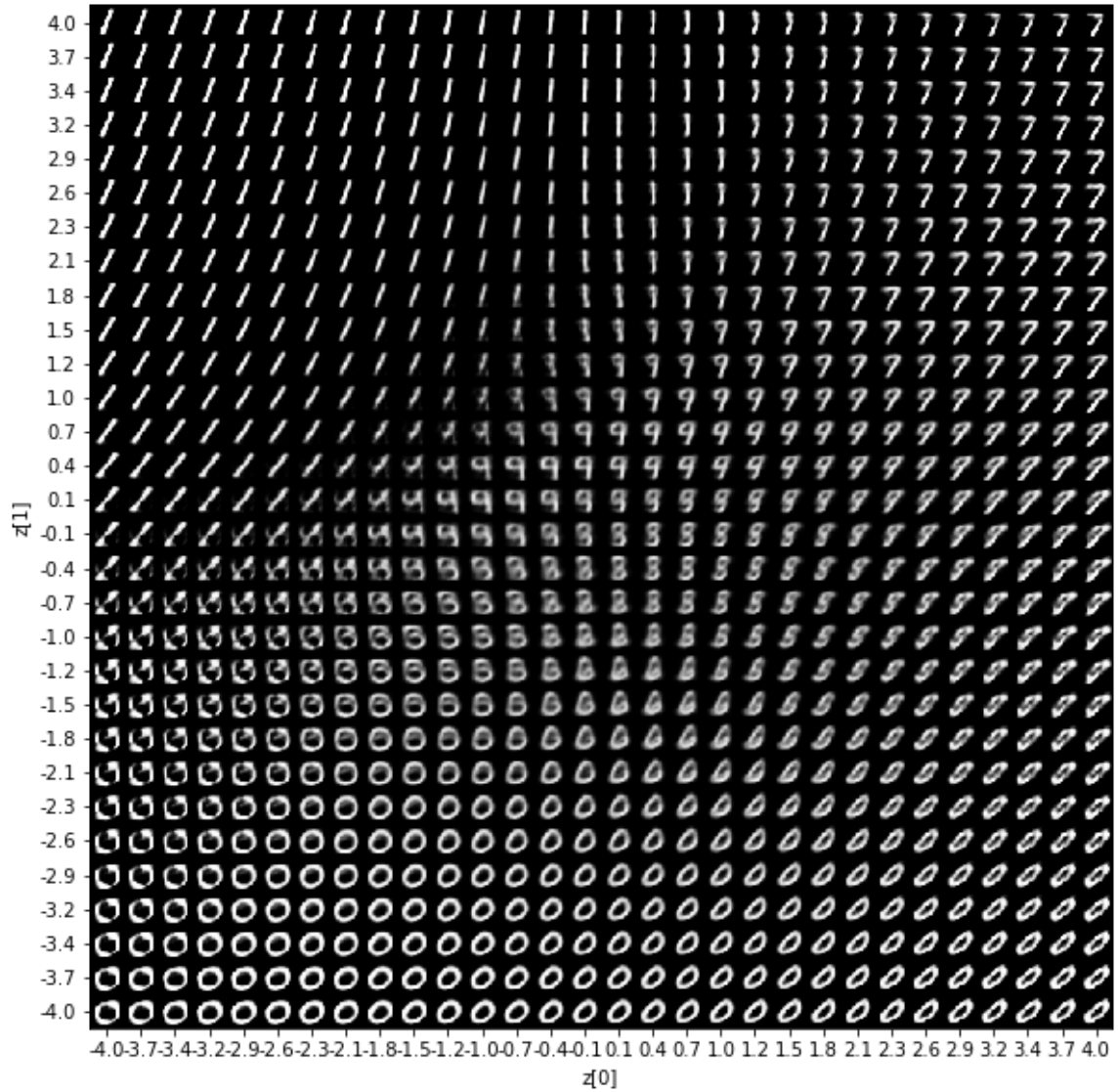
MNIST – epoca 50





MNIST – epoca 100





MNIST – epoca 200