Modèles génératifs
Autoencodeurs variationnels
IA et Multimedia

G. Marie

2026

Modèles génératifs

Données d'entraînement



$p_{\text{données}}(x)$



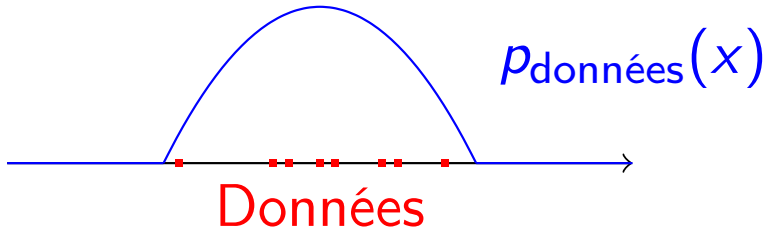
Données générées

$p_{\text{modèle}}(x)$

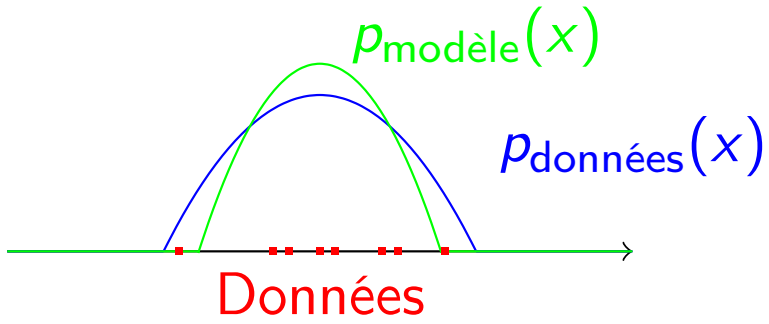
Étant donné un ensemble de données dites d'entraînement (échantillons), nous voulons être capable de générer des nouvelles données suivant la même distribution.

Estimation de densité

Il s'agit donc bien d'un problème d'apprentissage non supervisé, et plus spécifiquement d'estimation de densité.

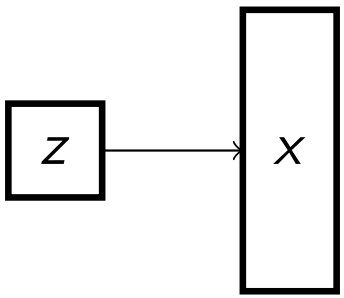


Un bon modèle génératif doit reproduire au plus près la densité de probabilité des données.



VAE - Modélisation du problème

On commence par supposer que nos données x sont issues de variables latentes z. Par exemple, des photos de voitures représentent des véhicules d'une certaine marque, d'un certain modèle, d'une certaine couleur, prises sous un certain angle, etc.



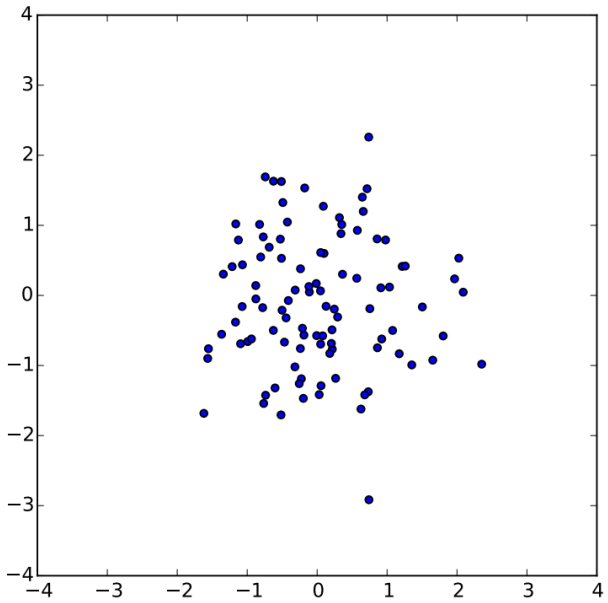
On cherche à apprendre une fonction qui génère les données x à partir de valeurs des variables latentes z.

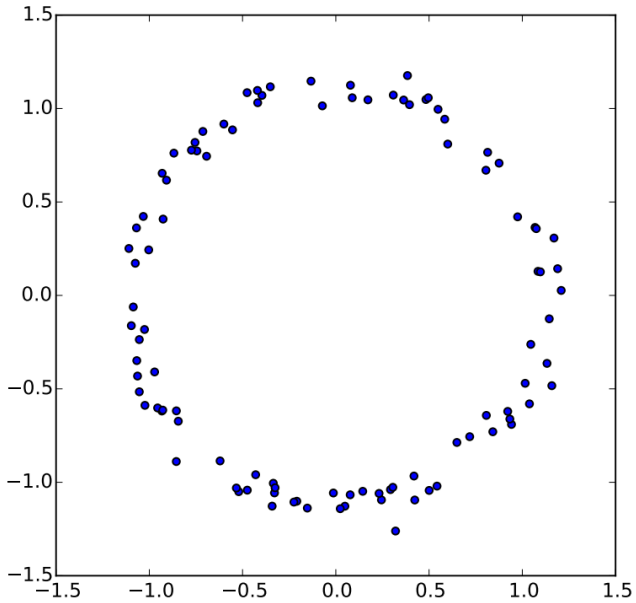
VAE - Hypothèses

On peut (hypothèse classique) supposer que z suit une loi Gaussienne :

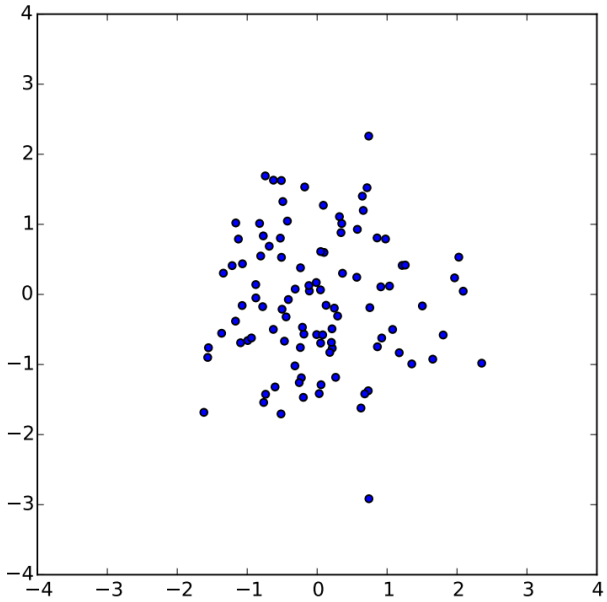
$$z \sim \mathcal{N}(0, I), \quad \text{i.e.} \quad p(z) = \mathcal{N}(z; 0, I)$$

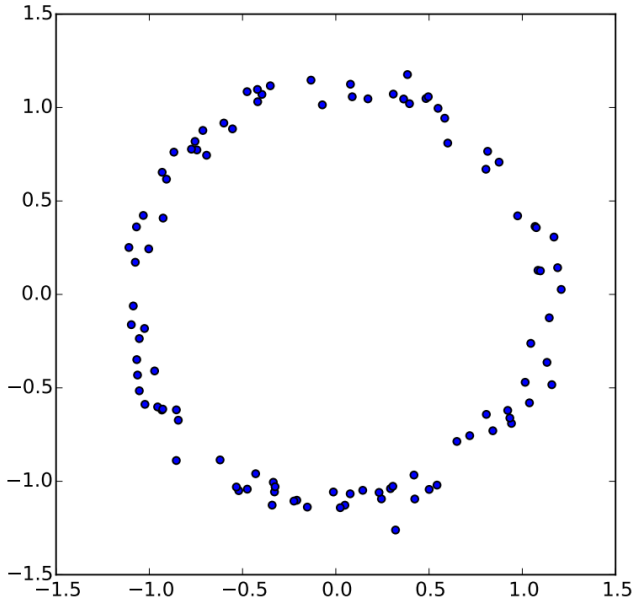
N.B. Choisir la matrice identité comme matrice de variance-covariance n'est pas anodin ; on suppose ici que nos variables latentes sont indépendantes.





VAE - Hypothèses

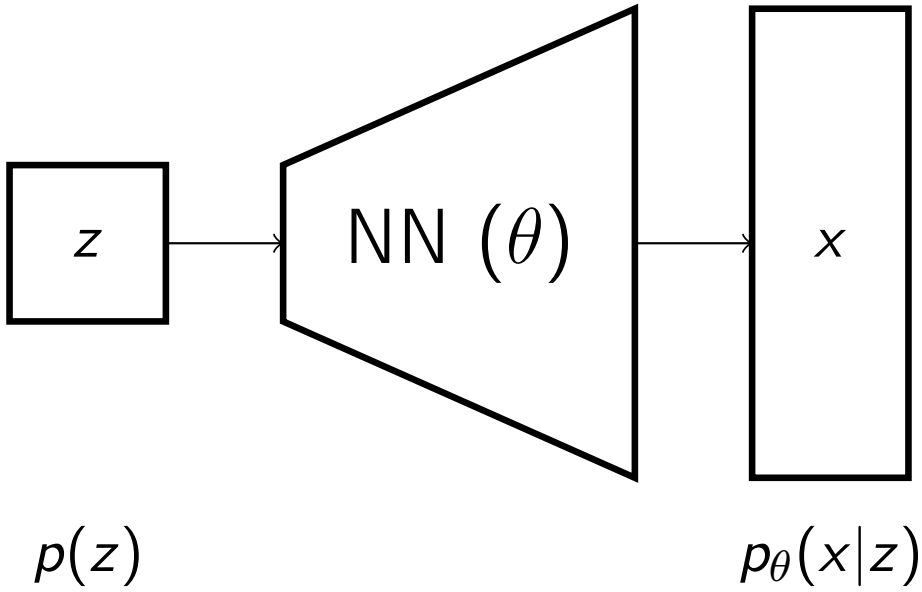




Il est possible de transférer une distribution gaussienne vers n'importe quelle distribution pour peu que l'on dispose d'un bon moyen d'approximer n'importe quelle fonction.

$\rightarrow$ On modélise cette fonction par un réseau de neurones !

VAE - Entraînement



Pour entraîner ce réseau, on cherche à établir les paramètres θ qui maximisent la vraisemblance $p_\theta(x)$ (ou, de manière équivalente, la log-vraisemblance) de générer nos données x.
Autrement dit, on veut résoudre :

$$\underset{\theta}{\operatorname{argmax}} \log p_\theta(x)$$

N.B. Dans le cours sur la diffusion, cette vraisemblance est notée simplement $p(x)$: elle dépend toujours de θ, à travers le décodeur $p_\theta(x|z)$.

VAE - Entraînement

Or, on peut écrire :

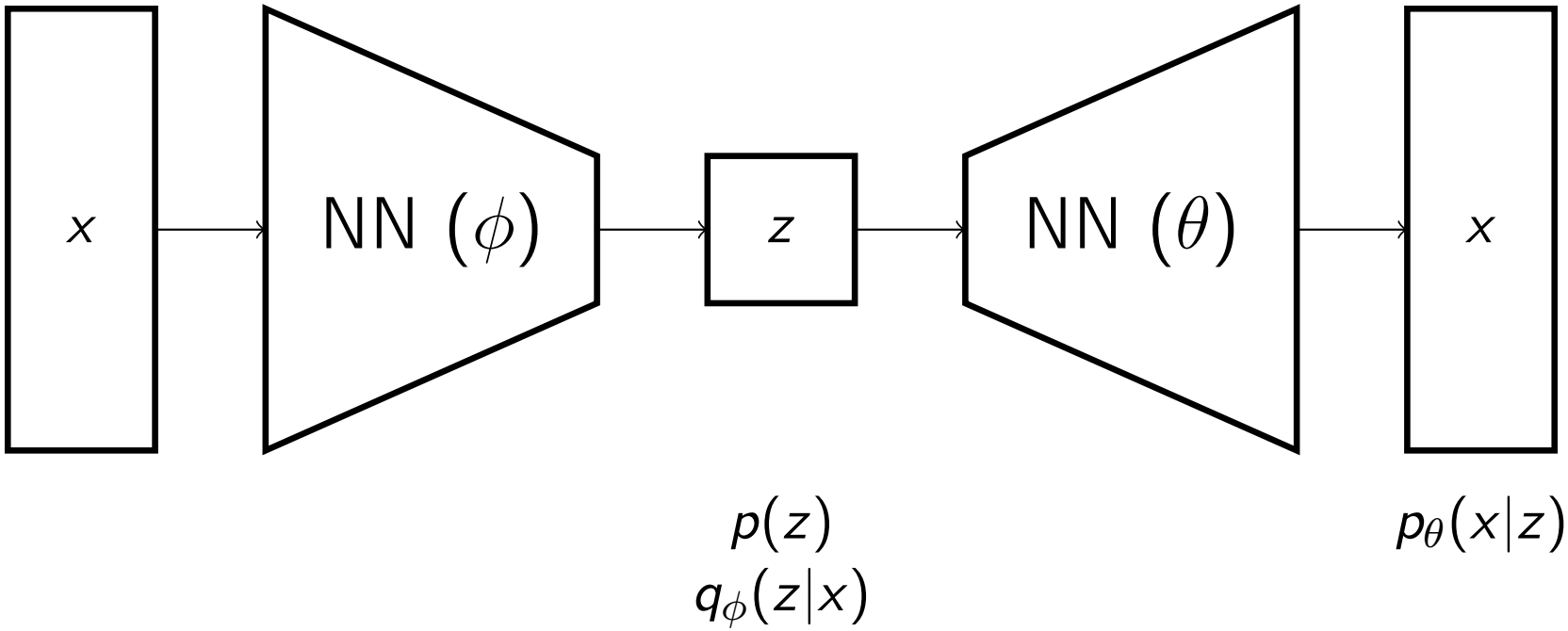
$$p_\theta(x) = \int p_\theta(x, z)\, dz = \int p(z)\, p_\theta(x|z)\, dz$$

(formule des probabilités totales, puis définition de la probabilité conditionnelle)

Pour entraîner notre réseau de neurones génératif, il nous faut calculer cette intégrale sur z. **Impossible** !
(On pourrait l'estimer avec une méthode de Monte-Carlo, mais cela nécessiterait un trop grand nombre d'échantillons pour obtenir une estimation fiable à cause de la grande dimension de nos espaces).

VAE - Entraînement

Pour résoudre ce problème, on introduit une fonction $q_\phi(z|x)$ qui définit une distribution des z qui ont une forte probabilité de générer les données x (une approximation de la loi *a posteriori* $p_\theta(z|x)$, inaccessible).



L'ensemble forme ce que l'on appelle un **auto-encodeur variationnel**.

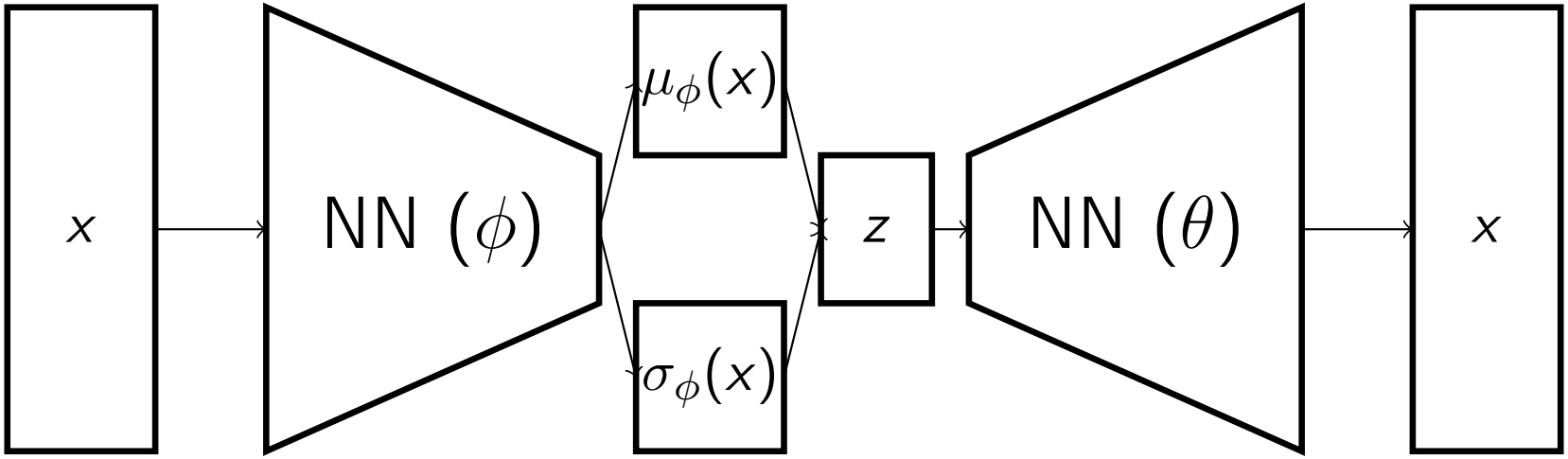
Auto-encodeurs variationnels

Une précision sémantique :

Les auto-encodeurs variationnels, en dépit de leur nom, sont assez différents des auto-encodeurs vus plus tôt dans le cours.
Ils partagent certes une architecture similaire et un vocabulaire commun (encodeur, décodeur, espace latent), mais les auto-encodeurs variationnels sont par nature **stochastiques**.

Les applications des auto-encodeurs s'appuient principalement sur la partie **encodeur** du réseau, alors que dans les auto-encodeurs variationnels, nous sommes intéressés par la partie **décodeur** qui constitue notre modèle génératif.

Auto-encodeurs variationnels - entraînement



Étant donnés un *minibatch* de données x, on génère leur représentation latente que l'on utilise pour estimer les paramètres de la distribution $q_\phi(z|x) = \mathcal{N}(z; \mu_\phi(x), \sigma^2_\phi(x)I)$ (covariance diagonale) : $\mu_\phi(x)$ et $\sigma_\phi(x)$. On échantillonne ensuite un z qui suit cette distribution et on l'utilise pour obtenir une donnée x.

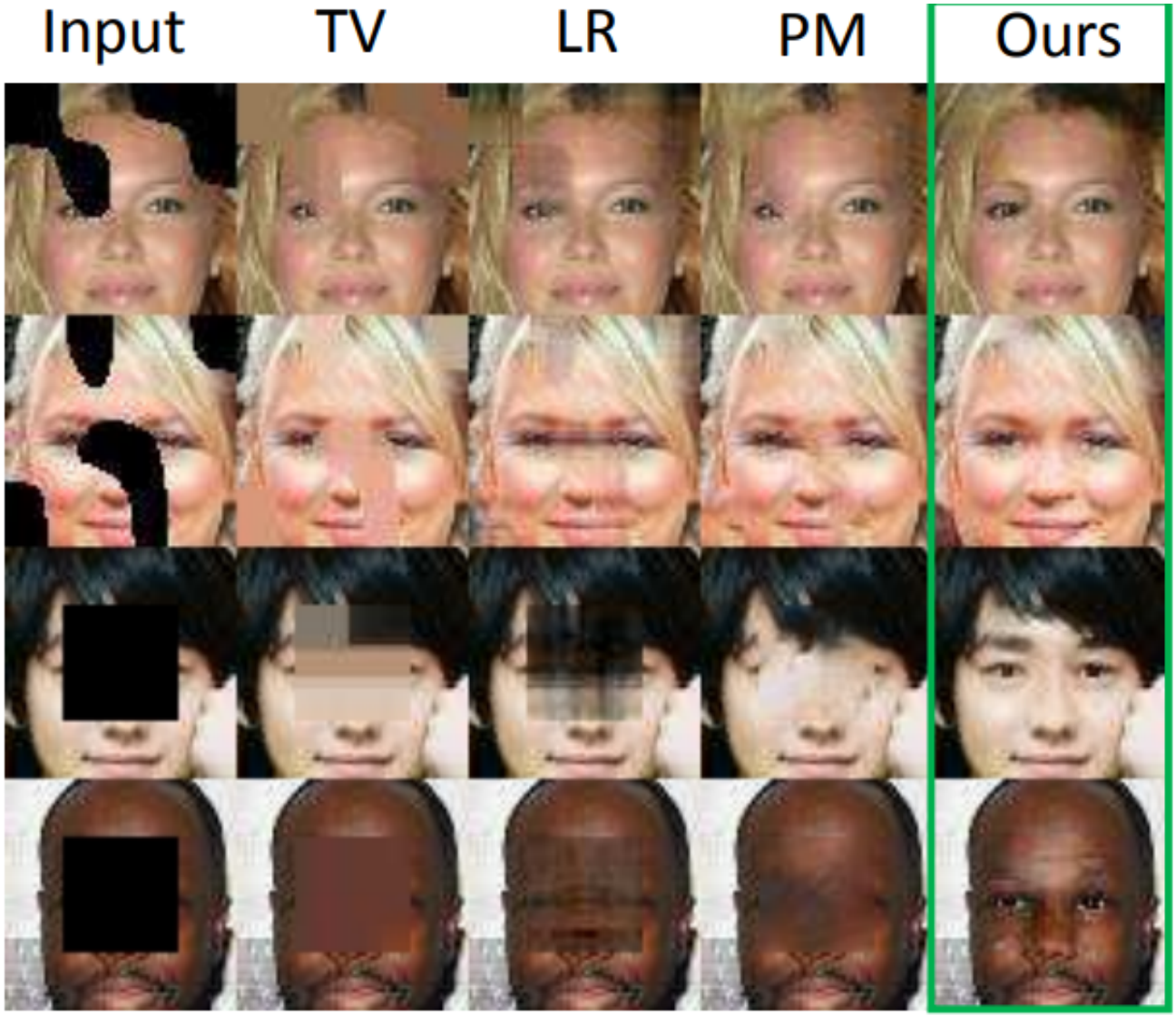
Auto-encodeurs variationnels

On maximise en fait, conjointement en θ et ϕ, l'**ELBO** (*Evidence Lower BOund*, une borne inférieure de $\log p_\theta(x)$), qui s'écrit dans ce contexte :

$$\mathrm{ELBO} = \underbrace{\mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x|z)\right]}_{\text{reconstruction}} - \underbrace{D_{\mathrm{KL}}\left(q_\phi(z|x) \,\|\, p(z)\right)}_{\textit{prior matching}} \tag{1}$$

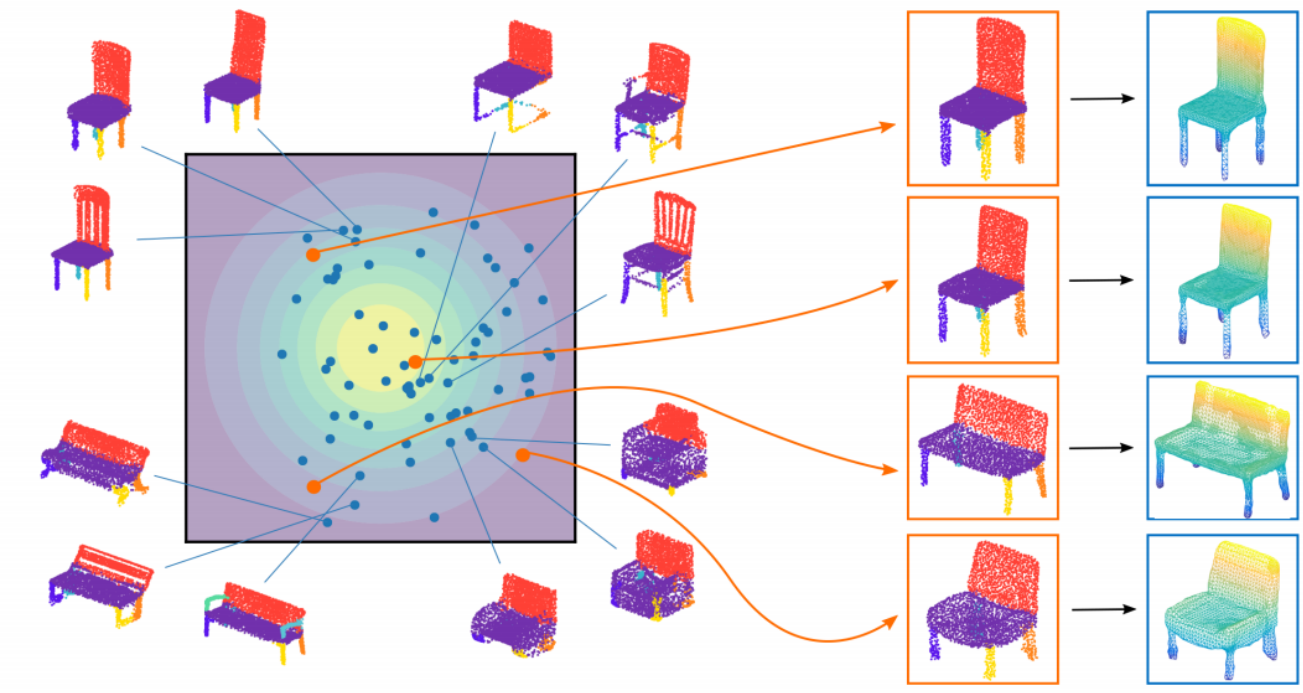
Le terme de gauche mesure la qualité de la reconstruction, et le terme de droite est la divergence de Kullback-Leibler (une mesure de dissimilarité, non symétrique, entre distributions) entre la distribution $q_\phi(z|x)$ estimée par l'encodeur et notre *a priori* $p(z)$ sur z.
Par bonheur, cette expression est différentiable et nous pouvons entraîner le réseau de neurones par descente de gradient (moyennant une "astuce de reparamétrisation" : $z = \mu_\phi(x) + \sigma_\phi(x) \odot \epsilon$, avec $\epsilon \sim \mathcal{N}(0, I)$).

Auto-encodeurs variationnels - applications



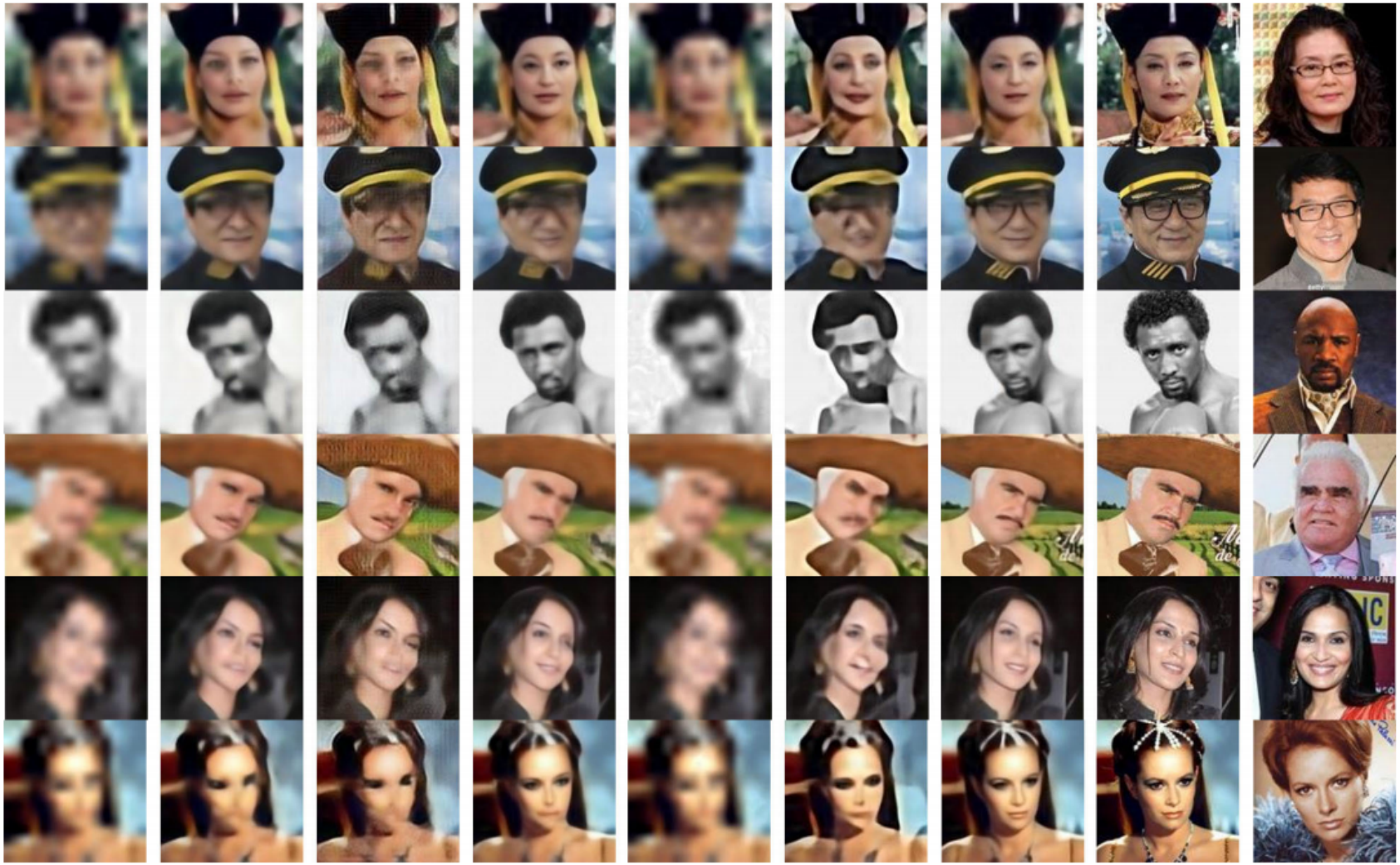
[Yeh et al.] Semantic Image Inpainting with Deep Generative Models

Auto-encodeurs variationnels - applications



[Nash et al.] The shape variational autoencoder : A deep generative model of part-segmented 3D objects

Auto-encodeurs variationnels - applications



[Liu et al.] Reference Based Face Super-Resolution