

VARIATIONAL AUTOENCODERS (VAEs)

Jérémie Cabessa

Laboratoire DAVID, UVSQ

INTRODUCTION

- ▶ **But d'un autoencodeur (rappel):** projeter les data dans un espace de dimension plus petite – l'**espace latent (latent space)** – et être capable de les reconstruire ces data.
 - ▶ On avait vu que l'**encodeur** pouvait être utilisé comme un **data compressor**...
 - ▶ et le **décodeur** comme un **data generator**.
 - ▶ Pour cela, il suffisait de sampler des points dans l'**espace latent** et de les décoder.

INTRODUCTION

- ▶ **But d'un autoencodeur (rappel):** projeter les data dans un espace de dimension plus petite – l'**espace latent (latent space)** – et être capable de les reconstruire ces data.
 - ▶ On avait vu que l'**encodeur** pouvait être utilisé comme un **data compressor**...
 - ▶ et le **décodeur** comme un **data generator**.
 - ▶ Pour cela, il suffisait de sampler des points dans l'espace latent et de les décoder.

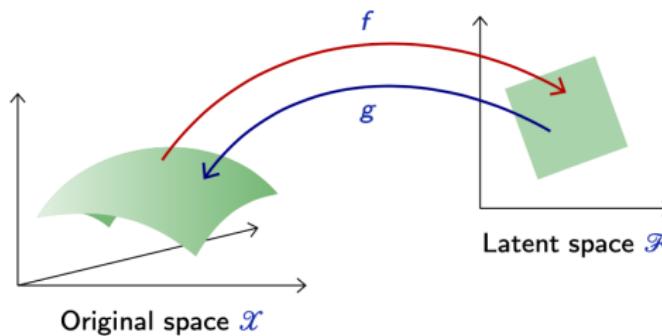
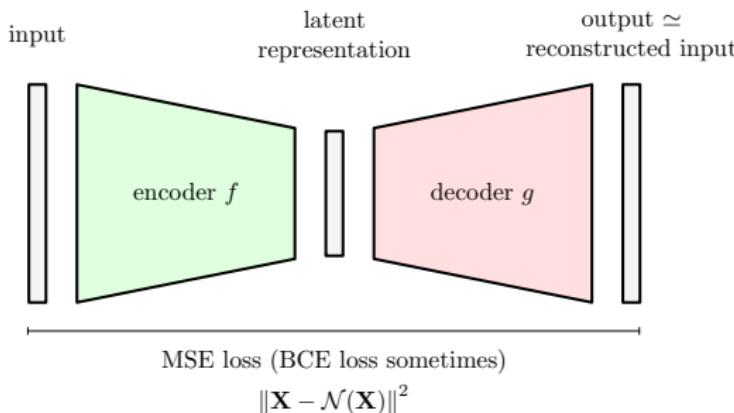
INTRODUCTION

- ▶ **But d'un autoencodeur (rappel):** projeter les data dans un espace de dimension plus petite – l'**espace latent (latent space)** – et être capable de les reconstruire ces data.
 - ▶ On avait vu que l'**encodeur** pouvait être utilisé comme un **data compressor**...
 - ▶ et le **décodeur** comme un **data generator**.
 - ▶ Pour cela, il suffisait de sampler des points dans l'espace latent et de les décoder.

INTRODUCTION

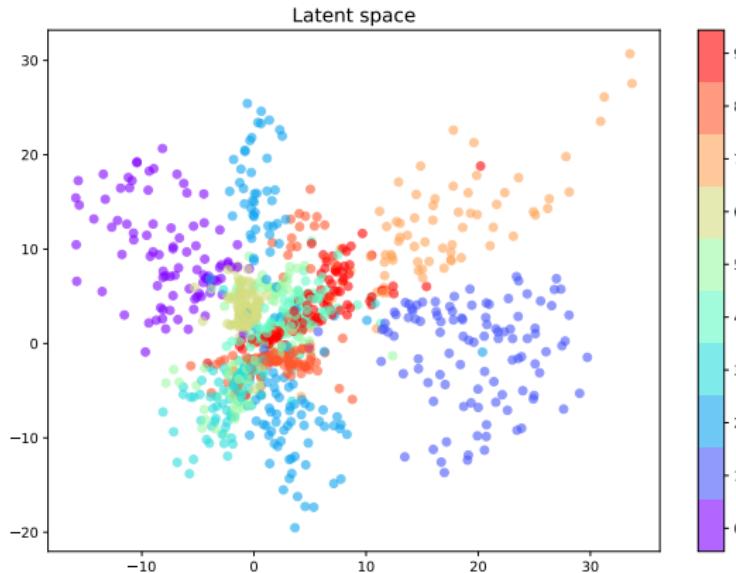
- ▶ **But d'un autoencodeur (rappel):** projeter les data dans un espace de dimension plus petite – l'**espace latent (latent space)** – et être capable de les reconstruire ces data.
 - ▶ On avait vu que l'**encodeur** pouvait être utilisé comme un **data compressor**...
 - ▶ et le **décodeur** comme un **data generator**.
 - ▶ Pour cela, il suffisait de sampler des points dans l'espace latent et de les décoder.

INTRODUCTION



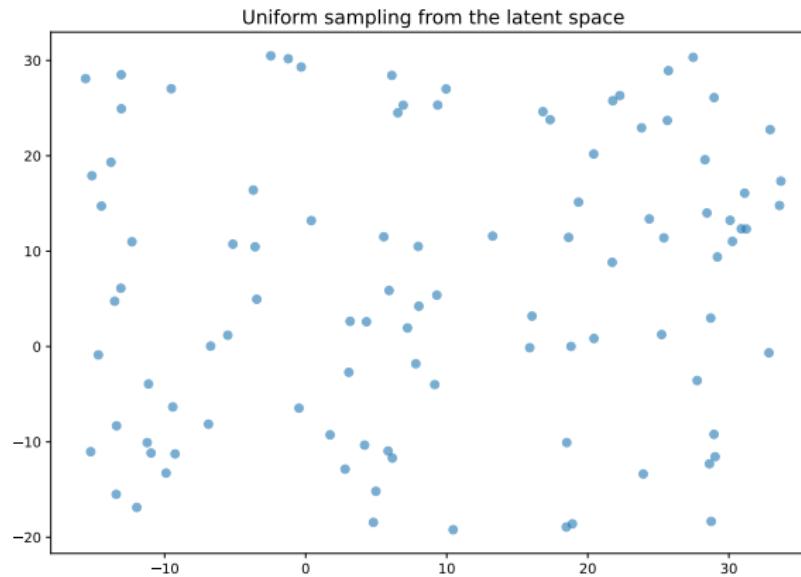
MOTIVATION

- ▶ Dans notre cas, on avait un espace latent de dimension 2, et lorsqu'on samplait dans cet espace, les data générées étaient d'assez bonne qualité (à l'oeil).



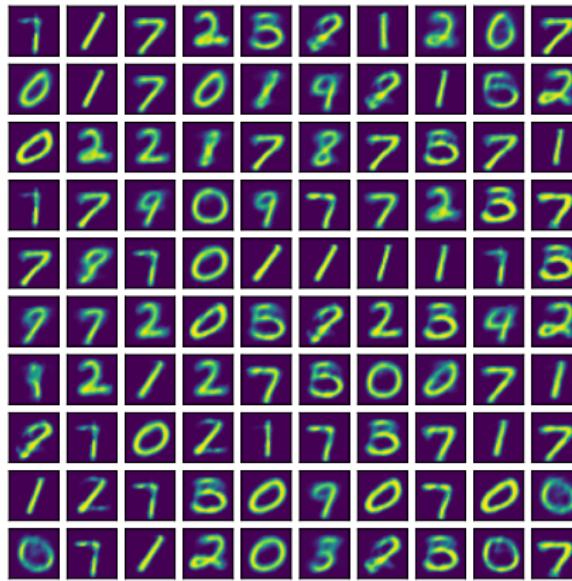
MOTIVATION

- ▶ Dans notre cas, on avait un espace latent de dimension 2, et lorsqu'on samplait dans cet espace, les data générées étaient d'assez bonne qualité (à l'oeil).



MOTIVATION

- ▶ Dans notre cas, on avait un espace latent de dimension 2, et lorsqu'on samplait dans cet espace, les data générées étaient d'assez bonne qualité (à l'oeil).



MOTIVATION

- ▶ Mais cette situation n'est pas représentative...
 - ▶ En pratique, on considère des espaces latents de dimensions plus grandes que 2 (on arrive rarement à compresser un signal en dimension 2).
 - ▶ Dans ce cas, le sampling dans l'espace latent génère des données beaucoup plus dégradées...

MOTIVATION

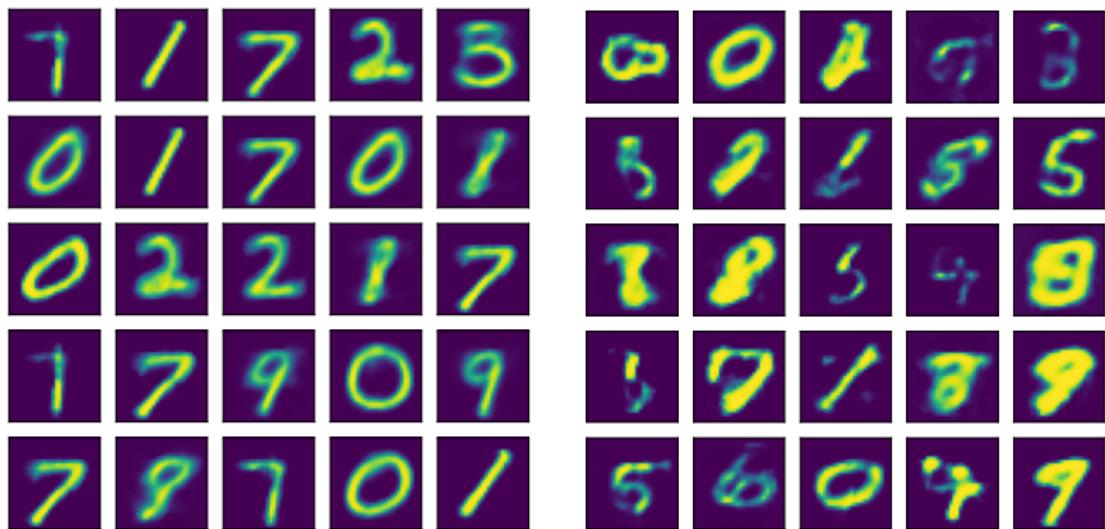
- ▶ Mais cette situation n'est pas représentative...
 - ▶ En pratique, on considère des espaces latents de dimensions plus grandes que 2 (on arrive rarement à compresser un signal en dimension 2).
 - ▶ Dans ce cas, le sampling dans l'espace latent génère des données beaucoup plus dégradées...

MOTIVATION

- ▶ Mais cette situation n'est pas représentative...
 - ▶ En pratique, on considère des espaces latents de dimensions plus grandes que 2 (on arrive rarement à compresser un signal en dimension 2).
 - ▶ Dans ce cas, le sampling dans l'espace latent génère des données beaucoup plus dégradées...

MOTIVATION

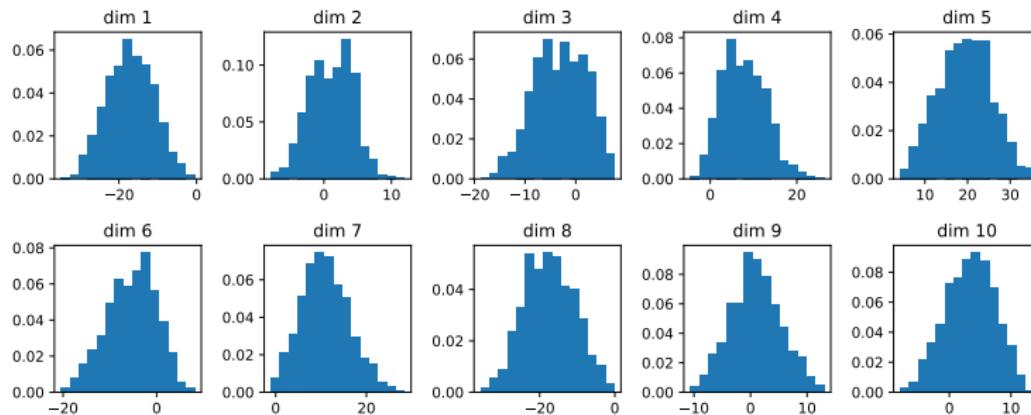
- ▶ Par exemple, si on répète l'opération avec un espace latent de dimension 10, et qu'on sample dans cet espace, les data générées sont bien moins bonnes.



Data generated from latent spaces of dim 2 and 10, respectively.

MOTIVATION

- ▶ De plus, la distribution des data dans l'espace latent est très déséquilibrée (non-centered, skewed, etc.).



Distribution of data in the latent space for each dimension.

MOTIVATION

- ▶ Pour un autoencodeur classique, qu'est-ce qui rend le processus de sampling de l'espace latent difficile?
 - ★ Distribution de l'espace latent non-continue: ne se voit pas forcément en 2D, mais en dimension supérieure, il y aura sûrement des "trous".
 - ★ Distribution de l'espace latent non-équilibrée et non-centrée.
- ▶ L'encodeur variationnel (variational autoencoder VAE) vise à pallier ces problèmes.
- ▶ Il force l'espace latent à suivre une certaine distribution, normale en général.

MOTIVATION

- ▶ Pour un autoencodeur classique, qu'est-ce qui rend le processus de sampling de l'espace latent difficile?
- ★ Distribution de l'espace latent non-continue: ne se voit pas forcément en 2D, mais en dimension supérieure, il y aura sûrement des "trous".
- ★ Distribution de l'espace latent non-équilibrée et non-centrée.
- ▶ L'encodeur variationnel (**variational autoencoder VAE**) vise à pallier ces problèmes.
- ▶ Il force l'espace latent à suivre une certaine distribution, normale en général.

MOTIVATION

- ▶ Pour un autoencodeur classique, qu'est-ce qui rend le processus de sampling de l'espace latent difficile?
- ★ Distribution de l'espace latent non-continue: ne se voit pas forcément en 2D, mais en dimension supérieure, il y aura sûrement des "trous".
- ★ Distribution de l'espace latent non-équilibrée et non-centrée.
- ▶ L'encodeur variationnel (**variational autoencoder VAE**) vise à pallier ces problèmes.
- ▶ Il force l'espace latent à suivre une certaine distribution, normale en général.

MOTIVATION

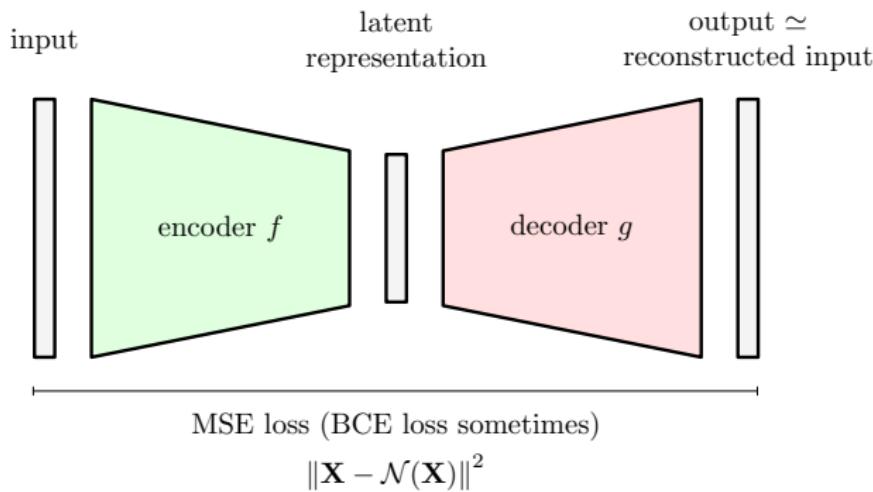
- ▶ Pour un autoencodeur classique, qu'est-ce qui rend le processus de sampling de l'espace latent difficile?
- ★ Distribution de l'espace latent non-continue: ne se voit pas forcément en 2D, mais en dimension supérieure, il y aura sûrement des "trous".
- ★ Distribution de l'espace latent non-équilibrée et non-centrée.
- ▶ **L'encodeur variationnel (variational autoencoder VAE) vise à pallier ces problèmes.**
- ▶ Il force l'espace latent à suivre une certaine distribution, normale en général.

MOTIVATION

- ▶ Pour un autoencodeur classique, qu'est-ce qui rend le processus de sampling de l'espace latent difficile?
- ★ Distribution de l'espace latent non-continue: ne se voit pas forcément en 2D, mais en dimension supérieure, il y aura sûrement des "trous".
- ★ Distribution de l'espace latent non-équilibrée et non-centrée.
- ▶ L'**encodeur variationnel (variational autoencoder VAE)** vise à pallier ces problèmes.
- ▶ Il force l'espace latent à suivre une certaine distribution, normale en général.

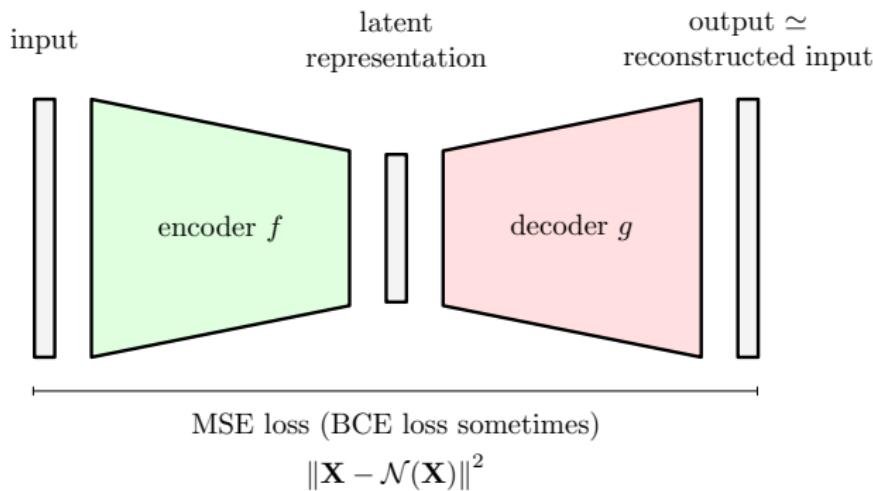
VARIATIONAL AUTOENCODER (VAE)

- ▶ **Idée générale:** Imposer une distribution sur l'espace latent et entraîner un décodeur de manière à reconstruire les data.
- ▶ On passe d'un bottleneck *déterministe* à un bottleneck *stochastique*.



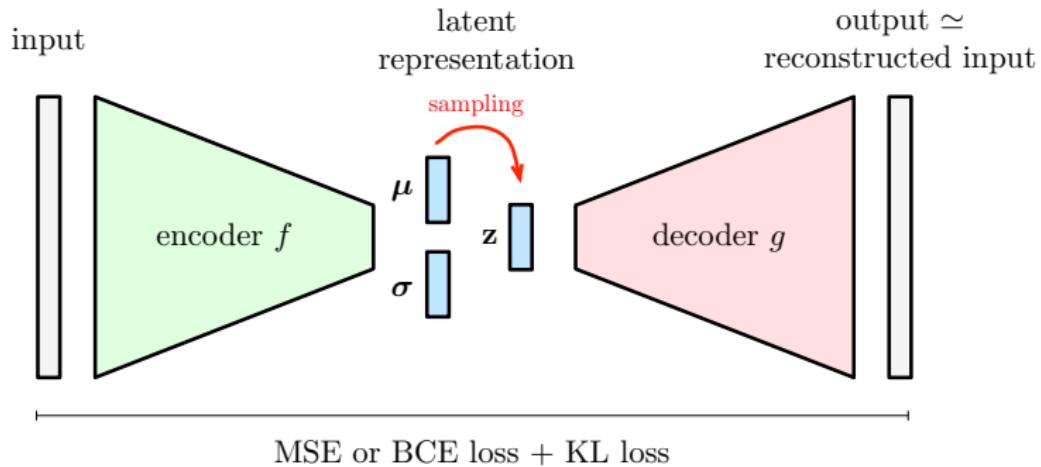
VARIATIONAL AUTOENCODER (VAE)

- ▶ **Idée générale:** Imposer une distribution sur l'espace latent et entraîner un décodeur de manière à reconstruire les data.
- ▶ On passe d'un bottleneck *déterministe* à un bottleneck *stochastique*.



VARIATIONAL AUTOENCODER (VAE)

- ▶ **Idée générale:** Imposer une distribution sur l'espace latent et entraîner un décodeur de manière à reconstruire les data.
- ▶ On passe d'un bottleneck *déterministe* à un bottleneck *stochastique*.



VARIATIONAL AUTOENCODER (VAE)

- ▶ Régularisation des data: on impose des contraintes de *continuité* et de *complétude* sur l'espace latent.
- ▶ Contrainte de continuité: on aimerait que des points proches de l'espace latent génèrent des data similaires.
- On introduit un processus de décodage stochastique lors de l'entraînement: chaque data est samplée dans l'espace latent (voisinage) avant d'être décodée.
- ▶ Contrainte de complétude: on aimerait que tous les points de l'espace latent génèrent des data qui soient valables.
- Pour "ramasser les data" (éviter les trous), on force les data de l'espace latent à suivre une loi normale centrée réduite $\mathcal{N}(\mathbf{0}, \mathbf{I})$.

VARIATIONAL AUTOENCODER (VAE)

- ▶ Régularisation des data: on impose des contraintes de *continuité* et de *complétude* sur l'espace latent.
- ▶ **Contrainte de continuité:** on aimerait que des points proches de l'espace latent génèrent des data similaires.
 - On introduit un processus de décodage stochastique lors de l'entraînement: chaque data est samplée dans l'espace latent (voisinage) avant d'être décodée.
- ▶ **Contrainte de complétude:** on aimerait que tous les points de l'espace latent génèrent des data qui soient valables.
 - Pour "ramasser les data" (éviter les trous), on force les data de l'espace latent à suivre une loi normale centrée réduite $\mathcal{N}(\mathbf{0}, \mathbf{I})$.

VARIATIONAL AUTOENCODER (VAE)

- ▶ Régularisation des data: on impose des contraintes de *continuité* et de *complétude* sur l'espace latent.
- ▶ **Contrainte de continuité:** on aimerait que des points proches de l'espace latent génèrent des data similaires.
- On introduit un processus de décodage stochastique lors de l'entraînement: chaque data est samplée dans l'espace latent (voisinage) avant d'être décodée.
- ▶ **Contrainte de complétude:** on aimerait que tous les points de l'espace latent génèrent des data qui soient valables.
- Pour "ramasser les data" (éviter les trous), on force les data de l'espace latent à suivre une loi normale centrée réduite $\mathcal{N}(\mathbf{0}, \mathbf{I})$.

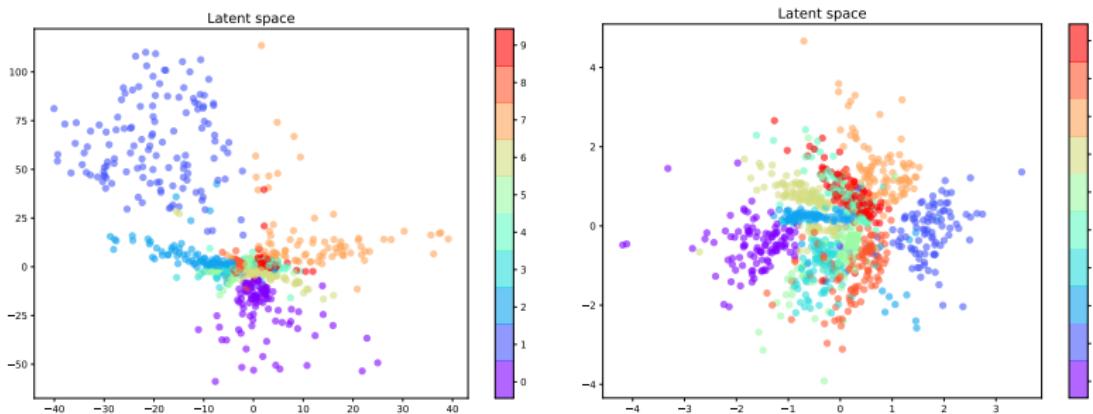
VARIATIONAL AUTOENCODER (VAE)

- ▶ Régularisation des data: on impose des contraintes de *continuité* et de *complétude* sur l'espace latent.
- ▶ **Contrainte de continuité:** on aimerait que des points proches de l'espace latent génèrent des data similaires.
- On introduit un processus de décodage stochastique lors de l'entraînement: chaque data est samplée dans l'espace latent (voisinage) avant d'être décodée.
- ▶ **Contrainte de complétude:** on aimerait que tous les points de l'espace latent génèrent des data qui soient valables.
- Pour "ramasser les data" (éviter les trous), on force les data de l'espace latent à suivre une loi normale centrée réduite $\mathcal{N}(\mathbf{0}, \mathbf{I})$.

VARIATIONAL AUTOENCODER (VAE)

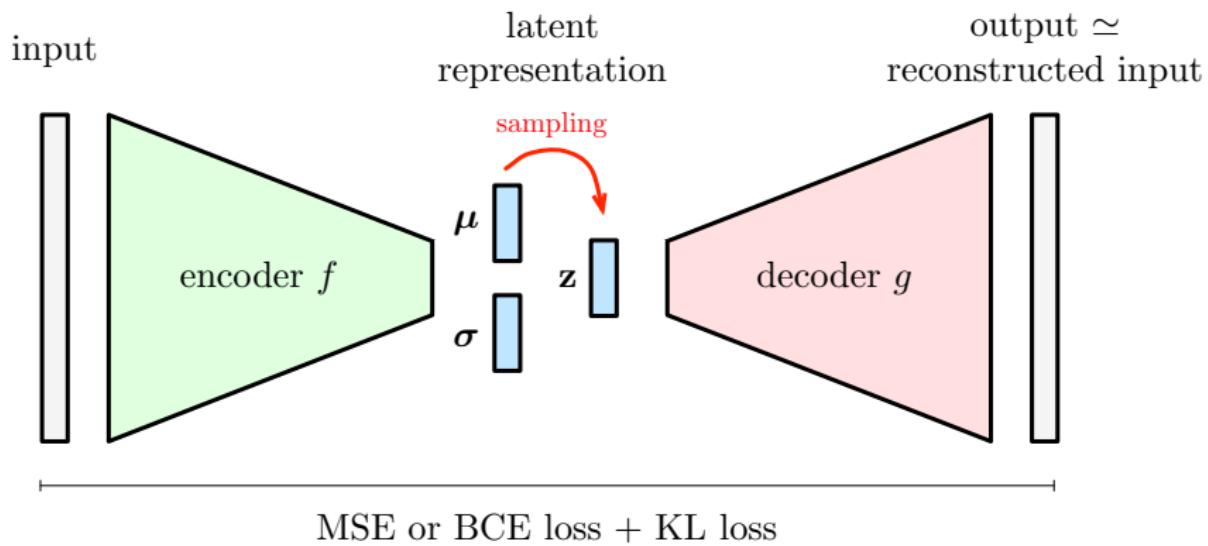
- ▶ Régularisation des data: on impose des contraintes de *continuité* et de *complétude* sur l'espace latent.
- ▶ **Contrainte de continuité:** on aimerait que des points proches de l'espace latent génèrent des data similaires.
- On introduit un processus de décodage stochastique lors de l'entraînement: chaque data est samplée dans l'espace latent (voisinage) avant d'être décodée.
- ▶ **Contrainte de complétude:** on aimerait que tous les points de l'espace latent génèrent des data qui soient valables.
- Pour “ramasser les data” (éviter les trous), on force les data de l'espace latent à suivre une loi normale centrée réduite $\mathcal{N}(\mathbf{0}, \mathbf{I})$.

VARIATIONAL AUTOENCODER (VAE)

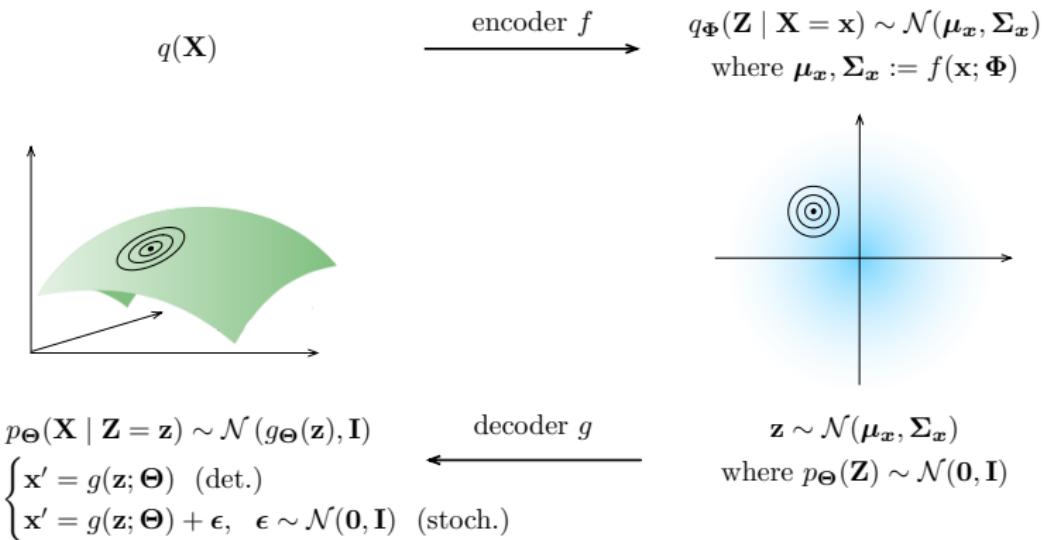


Left: less continuous and complete latent space; **Right:** more continuous and complete latent space.

VARIATIONAL AUTOENCODER (VAE)

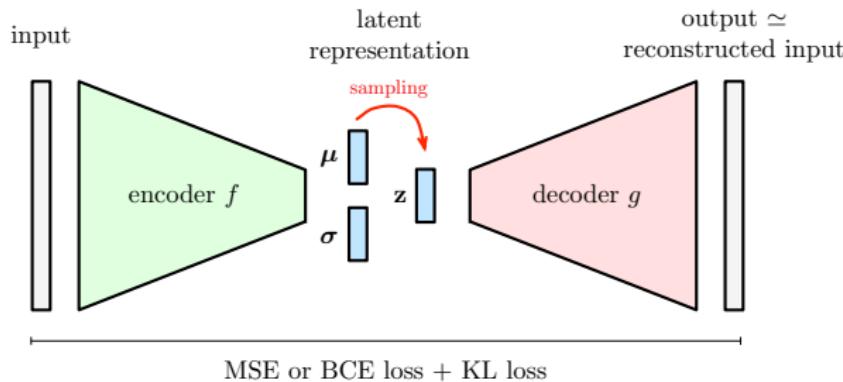


VARIATIONAL AUTOENCODER (VAE)



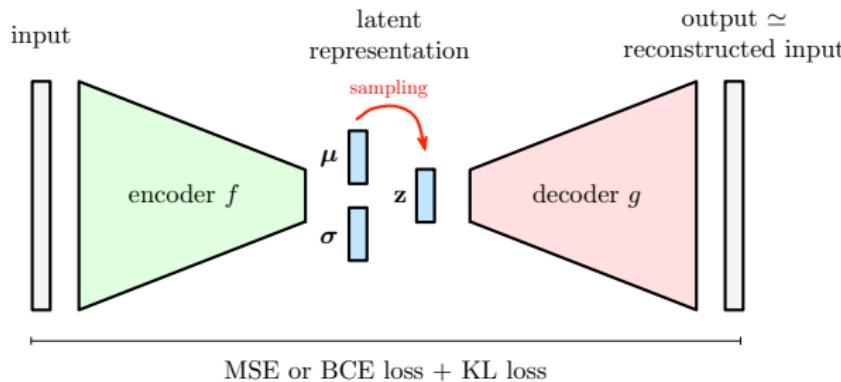
VARIATIONAL AUTOENCODER (VAE)

- ▶ L'**encodeur** est un deep neural network $f(\cdot; \Phi)$.
- ▶ Le **sampler** $s(\cdot; \mu, \Sigma)$ sample selon une loi normale $\mathcal{N}(\mu, \Sigma)$.
- ▶ Le **décodeur** est un deep neural network $g(\cdot; \Theta)$.
- ▶ La composition de $f(\cdot; \Phi)$, $s(\cdot; \mu, \Sigma)$ et $g(\cdot; \Theta)$ forme un réseau de neurones stochastique $\mathcal{N}(\cdot; \Phi, \Theta)$.



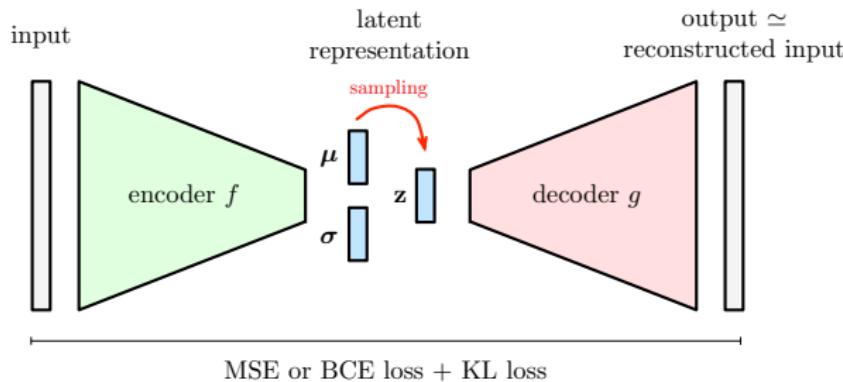
VARIATIONAL AUTOENCODER (VAE)

- ▶ L'**encodeur** est un deep neural network $f(\cdot; \Phi)$.
- ▶ Le **sampler** $s(\cdot; \mu, \Sigma)$ sample selon une loi normale $\mathcal{N}(\mu, \Sigma)$.
- ▶ Le **décodeur** est un deep neural network $g(\cdot; \Theta)$.
- ▶ La composition de $f(\cdot; \Phi)$, $s(\cdot; \mu, \Sigma)$ et $g(\cdot; \Theta)$ forme un réseau de neurones stochastique $\mathcal{N}(\cdot; \Phi, \Theta)$.



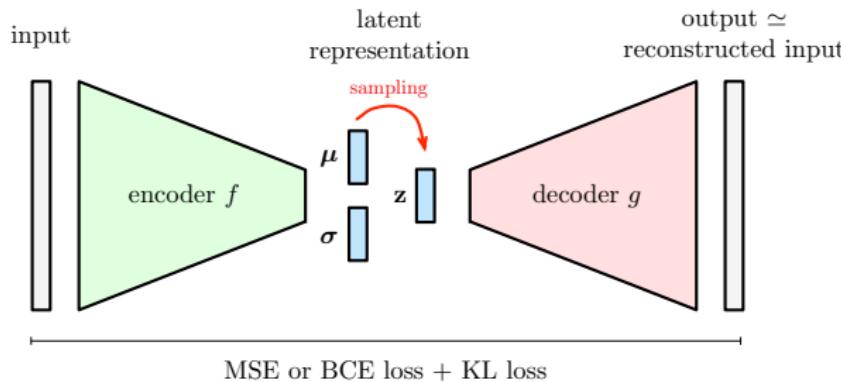
VARIATIONAL AUTOENCODER (VAE)

- ▶ L'**encodeur** est un deep neural network $f(\cdot; \Phi)$.
- ▶ Le **sampler** $s(\cdot; \mu, \Sigma)$ sample selon une loi normale $\mathcal{N}(\mu, \Sigma)$.
- ▶ Le **décodeur** est un deep neural network $g(\cdot; \Theta)$.
- ▶ La composition de $f(\cdot; \Phi)$, $s(\cdot; \mu, \Sigma)$ et $g(\cdot; \Theta)$ forme un réseau de neurones stochastique $\mathcal{N}(\cdot; \Phi, \Theta)$.



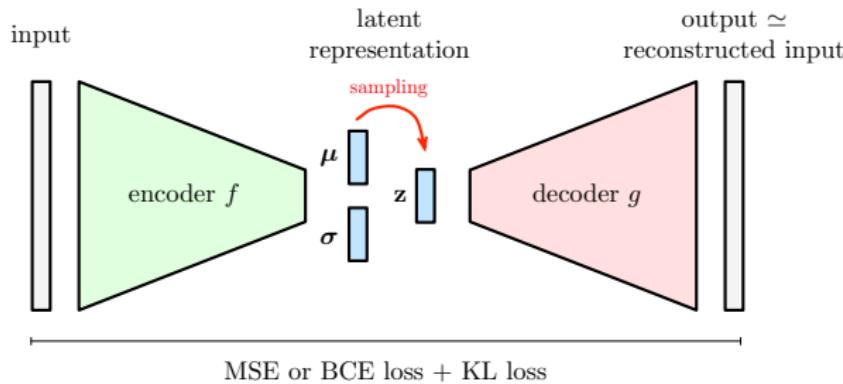
VARIATIONAL AUTOENCODER (VAE)

- ▶ L'**encodeur** est un deep neural network $f(\cdot; \Phi)$.
- ▶ Le **sampler** $s(\cdot; \mu, \Sigma)$ sample selon une loi normale $\mathcal{N}(\mu, \Sigma)$.
- ▶ Le **décodeur** est un deep neural network $g(\cdot; \Theta)$.
- ▶ La composition de $f(\cdot; \Phi)$, $s(\cdot; \mu, \Sigma)$ et $g(\cdot; \Theta)$ forme un **réseau de neurones stochastique** $\mathcal{N}(\cdot; \Phi, \Theta)$.



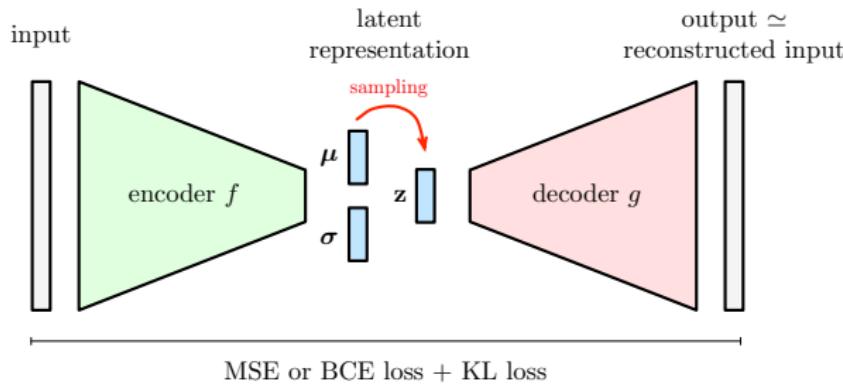
VARIATIONAL AUTOENCODER (VAE)

1. L'encodeur $f(\cdot; \Phi)$ encode une data \mathbf{x} en deux vecteurs μ_x et Σ_x de plus petite dimension.
2. Le sampler $s(\cdot; \mu_x, \Sigma_x)$ sample un point \mathbf{z} dans l'espace latent selon la loi normale $\mathcal{N}(\mu_x, \Sigma_x)$.
3. Le décodeur $g(\cdot; \Theta)$ reconstruit \mathbf{x} à partir de \mathbf{z} .



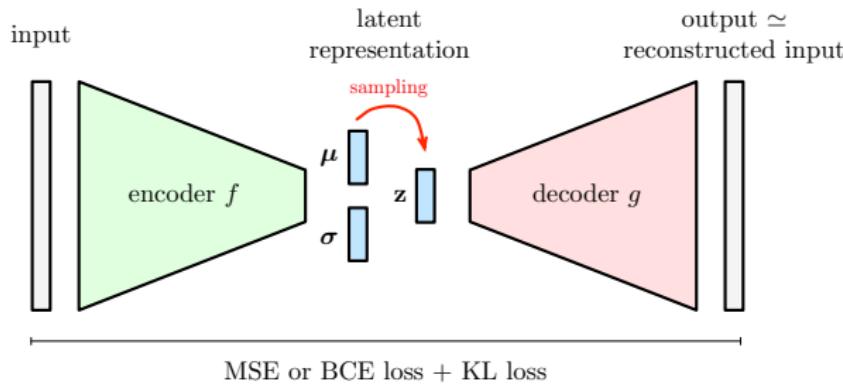
VARIATIONAL AUTOENCODER (VAE)

1. L'**encodeur** $f(\cdot; \Phi)$ encode une data \mathbf{x} en deux vecteurs μ_x et Σ_x de plus petite dimension.
2. Le **sampler** $s(\cdot; \mu_x, \Sigma_x)$ sample un point \mathbf{z} dans l'espace latent selon la loi normale $\mathcal{N}(\mu_x, \Sigma_x)$.
3. Le **décodeur** $g(\cdot; \Theta)$ reconstruit \mathbf{x} à partir de \mathbf{z} .



VARIATIONAL AUTOENCODER (VAE)

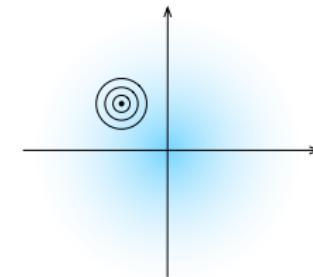
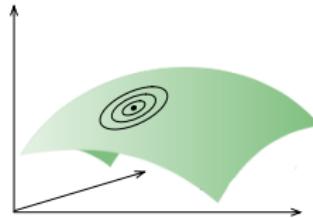
1. L'**encodeur** $f(\cdot; \Phi)$ encode une data \mathbf{x} en deux vecteurs μ_x et Σ_x de plus petite dimension.
2. Le **sampler** $s(\cdot; \mu_x, \Sigma_x)$ sample un point \mathbf{z} dans l'espace latent selon la loi normale $\mathcal{N}(\mu_x, \Sigma_x)$.
3. Le **décodeur** $g(\cdot; \Theta)$ reconstruit \mathbf{x} à partir de \mathbf{z} .



VARIATIONAL AUTOENCODER (VAE)

$$q(\mathbf{X}) \xrightarrow{\text{encoder } f} q_{\Phi}(\mathbf{Z} \mid \mathbf{X} = \mathbf{x}) \sim \mathcal{N}(\boldsymbol{\mu}_{\mathbf{x}}, \boldsymbol{\Sigma}_{\mathbf{x}})$$

where $\boldsymbol{\mu}_{\mathbf{x}}, \boldsymbol{\Sigma}_{\mathbf{x}} := f(\mathbf{x}; \Phi)$



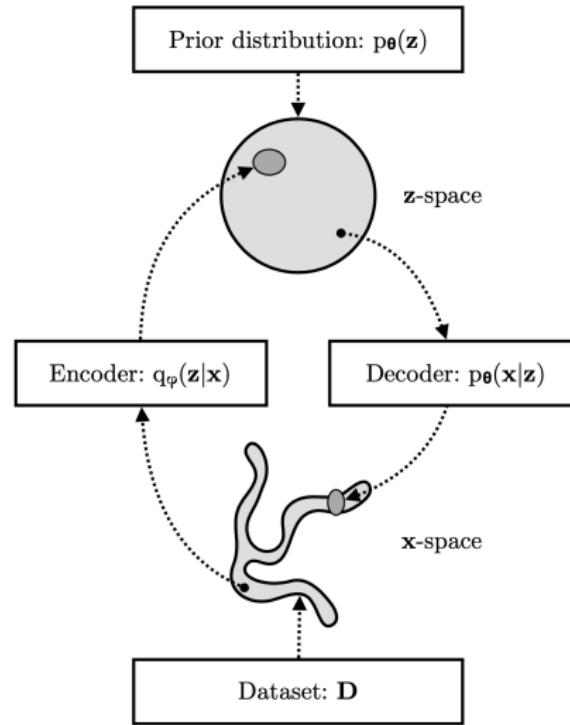
$$p_{\Theta}(\mathbf{X} \mid \mathbf{Z} = \mathbf{z}) \sim \mathcal{N}(g_{\Theta}(\mathbf{z}), \mathbf{I})$$
$$\begin{cases} \mathbf{x}' = g(\mathbf{z}; \Theta) \text{ (det.)} \\ \mathbf{x}' = g(\mathbf{z}; \Theta) + \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \text{ (stoch.)} \end{cases}$$

$$\xleftarrow{\text{decoder } g}$$

$$\mathbf{z} \sim \mathcal{N}(\boldsymbol{\mu}_{\mathbf{x}}, \boldsymbol{\Sigma}_{\mathbf{x}})$$

where $p_{\Theta}(\mathbf{Z}) \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

VARIATIONAL AUTOENCODER (VAE)



DISTRIBUTIONS

- ▶ $q(\mathbf{x}) \in \Delta(\mathcal{X})$: Distribution empirique des data.
- ▶ $q_{\Phi}(\mathbf{z} \mid \mathbf{x}) \in \Delta(\mathcal{Z})$: Distribution des data latentes étant donnée la data originale \mathbf{x} . Fonction de \mathbf{z} (variable \mathbf{X} fixé à \mathbf{x}).
→ Dépend de l'encodeur $f(\cdot; \Phi)$.
- ▶ $p_{\Theta}(\mathbf{x} \mid \mathbf{z}) \in \Delta(\mathcal{X})$: Distribution des data reconstruites étant donné la data latente \mathbf{z} . Fonction de \mathbf{x} (variable \mathbf{Z} fixé à \mathbf{z}).
→ Dépend du décodeur $g(\cdot; \Theta)$.
- ▶ $p_{\Theta}(\mathbf{x}) \in \Delta(\mathcal{X})$: Distribution des data reconstruites, induite par les distributions $p_{\Theta}(\mathbf{x} \mid \mathbf{z})$ pour tout \mathbf{z} .
- ▶ La reconstruction s'effectue correctement si $q(\mathbf{x}) \simeq p_{\Theta}(\mathbf{x})$.

DISTRIBUTIONS

- ▶ $q(\mathbf{x}) \in \Delta(\mathcal{X})$: Distribution empirique des data.
- ▶ $q_{\Phi}(\mathbf{z} \mid \mathbf{x}) \in \Delta(\mathcal{Z})$: Distribution des data latentes étant donnée la data originale \mathbf{x} . Fonction de \mathbf{z} (variable \mathbf{X} fixé à \mathbf{x}).
→ Dépend de l'encodeur $f(\cdot; \Phi)$.
- ▶ $p_{\Theta}(\mathbf{x} \mid \mathbf{z}) \in \Delta(\mathcal{X})$: Distribution des data reconstruites étant donné la data latente \mathbf{z} . Fonction de \mathbf{x} (variable \mathbf{Z} fixé à \mathbf{z}).
→ Dépend du décodeur $g(\cdot; \Theta)$.
- ▶ $p_{\Theta}(\mathbf{x}) \in \Delta(\mathcal{X})$: Distribution des data reconstruites, induite par les distributions $p_{\Theta}(\mathbf{x} \mid \mathbf{z})$ pour tout \mathbf{z} .
- ▶ La reconstruction s'effectue correctement si $q(\mathbf{x}) \simeq p_{\Theta}(\mathbf{x})$.

DISTRIBUTIONS

- ▶ $q(\mathbf{x}) \in \Delta(\mathcal{X})$: Distribution empirique des data.
- ▶ $q_{\Phi}(\mathbf{z} \mid \mathbf{x}) \in \Delta(\mathcal{Z})$: Distribution des data latentes étant donnée la data originale \mathbf{x} . Fonction de \mathbf{z} (variable \mathbf{X} fixé à \mathbf{x}).
→ Dépend de l'encodeur $f(\cdot; \Phi)$.
- ▶ $p_{\Theta}(\mathbf{x} \mid \mathbf{z}) \in \Delta(\mathcal{X})$: Distribution des data reconstruites étant donné la data latente \mathbf{z} . Fonction de \mathbf{x} (variable \mathbf{Z} fixé à \mathbf{z}).
→ Dépend du décodeur $g(\cdot; \Theta)$.
- ▶ $p_{\Theta}(\mathbf{x}) \in \Delta(\mathcal{X})$: Distribution des data reconstruites, induite par les distributions $p_{\Theta}(\mathbf{x} \mid \mathbf{z})$ pour tout \mathbf{z} .
- ▶ La reconstruction s'effectue correctement si $q(\mathbf{x}) \simeq p_{\Theta}(\mathbf{x})$.

DISTRIBUTIONS

- ▶ $q(\mathbf{x}) \in \Delta(\mathcal{X})$: Distribution empirique des data.
- ▶ $q_{\Phi}(\mathbf{z} \mid \mathbf{x}) \in \Delta(\mathcal{Z})$: Distribution des data latentes étant donnée la data originale \mathbf{x} . Fonction de \mathbf{z} (variable \mathbf{X} fixé à \mathbf{x}).
→ Dépend de l'encodeur $f(\cdot; \Phi)$.
- ▶ $p_{\Theta}(\mathbf{x} \mid \mathbf{z}) \in \Delta(\mathcal{X})$: Distribution des data reconstruites étant donné la data latente \mathbf{z} . Fonction de \mathbf{x} (variable \mathbf{Z} fixé à \mathbf{z}).
→ Dépend du décodeur $g(\cdot; \Theta)$.
- ▶ $p_{\Theta}(\mathbf{x}) \in \Delta(\mathcal{X})$: Distribution des data reconstruites, induite par les distributions $p_{\Theta}(\mathbf{x} \mid \mathbf{z})$ pour tout \mathbf{z} .
- ▶ La reconstruction s'effectue correctement si $q(\mathbf{x}) \simeq p_{\Theta}(\mathbf{x})$.

DISTRIBUTIONS

- ▶ $q(\mathbf{x}) \in \Delta(\mathcal{X})$: Distribution empirique des data.
- ▶ $q_{\Phi}(\mathbf{z} \mid \mathbf{x}) \in \Delta(\mathcal{Z})$: Distribution des data latentes étant donnée la data originale \mathbf{x} . Fonction de \mathbf{z} (variable \mathbf{X} fixé à \mathbf{x}).
→ Dépend de l'encodeur $f(\cdot; \Phi)$.
- ▶ $p_{\Theta}(\mathbf{x} \mid \mathbf{z}) \in \Delta(\mathcal{X})$: Distribution des data reconstruites étant donné la data latente \mathbf{z} . Fonction de \mathbf{x} (variable \mathbf{Z} fixé à \mathbf{z}).
→ Dépend du décodeur $g(\cdot; \Theta)$.
- ▶ $p_{\Theta}(\mathbf{x}) \in \Delta(\mathcal{X})$: Distribution des data reconstruites, induite par les distributions $p_{\Theta}(\mathbf{x} \mid \mathbf{z})$ pour tout \mathbf{z} .
- ▶ La reconstruction s'effectue correctement si $q(\mathbf{x}) \simeq p_{\Theta}(\mathbf{x})$.

KL-DIVERGENCE ET LOG-LIKELIHOOD

- **Rappel:** On cherche à *apprendre* la distribution empirique des data $q(\mathbf{x})$, c'est à dire trouver $p_{\Theta}(\mathbf{x})$ telle que

$$q(\mathbf{x}) \simeq p_{\Theta}(\mathbf{x})$$

- Pour cela, on minimise la *divergence de Kullback–Leibler* entre $q(\mathbf{x})$ et $p_{\Theta}(\mathbf{x})$ définie par:

$$\begin{aligned} D_{\text{KL}}(q \parallel p_{\Theta}) &= \int q(\mathbf{x}) \log \frac{q(\mathbf{x})}{p_{\Theta}(\mathbf{x})} d\mathbf{x} \\ &= \int q(\mathbf{x}) (\log q(\mathbf{x}) - \log p_{\Theta}(\mathbf{x})) d\mathbf{x} \\ &= \mathbb{E}_{q(\mathbf{x})} [\log q(\mathbf{x})] - \mathbb{E}_{q(\mathbf{x})} [\log p_{\Theta}(\mathbf{x})] \quad (1) \end{aligned}$$

KL-DIVERGENCE ET LOG-LIKELIHOOD

- ▶ **Rappel:** On cherche à *apprendre* la distribution empirique des data $q(\mathbf{x})$, c'est à dire trouver $p_{\Theta}(\mathbf{x})$ telle que

$$q(\mathbf{x}) \simeq p_{\Theta}(\mathbf{x})$$

- ▶ Pour cela, on minimise la *divergence de Kullback–Leibler* entre $q(\mathbf{x})$ et $p_{\Theta}(\mathbf{x})$ définie par:

$$\begin{aligned} D_{\text{KL}}(q \parallel p_{\Theta}) &= \int q(\mathbf{x}) \log \frac{q(\mathbf{x})}{p_{\Theta}(\mathbf{x})} d\mathbf{x} \\ &= \int q(\mathbf{x}) (\log q(\mathbf{x}) - \log p_{\Theta}(\mathbf{x})) d\mathbf{x} \\ &= \mathbb{E}_{q(\mathbf{x})} [\log q(\mathbf{x})] - \mathbb{E}_{q(\mathbf{x})} [\log p_{\Theta}(\mathbf{x})] \quad (1) \end{aligned}$$

KL-DIVERGENCE ET LOG-LIKELIHOOD

$$\hat{\Theta} = \arg \min_{\Theta} D_{\text{KL}}(q \parallel p_{\Theta})$$

$$(1) = \arg \min_{\Theta} \left(\mathbb{E}_{q(\mathbf{x})} [\log q(\mathbf{x})] - \mathbb{E}_{q(\mathbf{x})} [\log p_{\Theta}(\mathbf{x})] \right)$$

$$(\text{terme indép. de } \Theta) = \arg \min_{\Theta} \left(-\mathbb{E}_{q(\mathbf{x})} [\log p_{\Theta}(\mathbf{x})] \right)$$

$$= \arg \max_{\Theta} \left(\mathbb{E}_{q(\mathbf{x})} [\log p_{\Theta}(\mathbf{x})] \right)$$

$$= \arg \max_{\Theta} \left(\int q(\mathbf{x}) \log p_{\Theta}(\mathbf{x}) d\mathbf{x} \right)$$

On peut donc écrire l'optimisation de $\hat{\Theta}$ comme une maximisation de l'espérance de log-likelihood :

KL-DIVERGENCE ET LOG-LIKELIHOOD

$$\hat{\Theta} = \arg \min_{\Theta} D_{\text{KL}}(q \parallel p_{\Theta})$$
$$(1) = \arg \min_{\Theta} \left(\mathbb{E}_{q(\mathbf{x})} [\log q(\mathbf{x})] - \mathbb{E}_{q(\mathbf{x})} [\log p_{\Theta}(\mathbf{x})] \right)$$

$$(\text{terme indép. de } \Theta) = \arg \min_{\Theta} \left(-\mathbb{E}_{q(\mathbf{x})} [\log p_{\Theta}(\mathbf{x})] \right)$$
$$= \arg \max_{\Theta} \left(\mathbb{E}_{q(\mathbf{x})} [\log p_{\Theta}(\mathbf{x})] \right)$$
$$= \arg \max_{\Theta} \left(\int q(\mathbf{x}) \log p_{\Theta}(\mathbf{x}) d\mathbf{x} \right)$$

On peut alors écrire l'objectif de l'entraînement d'un VAE comme :

$$\text{min}_{\Theta} \left(-\mathbb{E}_{q(\mathbf{x})} [\log p_{\Theta}(\mathbf{x})] + \text{constante} \right)$$

KL-DIVERGENCE ET LOG-LIKELIHOOD

$$\begin{aligned}\hat{\Theta} &= \arg \min_{\Theta} D_{\text{KL}}(q \parallel p_{\Theta}) \\ (1) &= \arg \min_{\Theta} \left(\mathbb{E}_{q(\mathbf{x})} [\log q(\mathbf{x})] - \mathbb{E}_{q(\mathbf{x})} [\log p_{\Theta}(\mathbf{x})] \right) \\ (\text{terme indép. de } \Theta) &= \arg \min_{\Theta} \left(-\mathbb{E}_{q(\mathbf{x})} [\log p_{\Theta}(\mathbf{x})] \right) \\ &= \arg \max_{\Theta} \left(\mathbb{E}_{q(\mathbf{x})} [\log p_{\Theta}(\mathbf{x})] \right) \\ &= \arg \max_{\Theta} \left(\int q(\mathbf{x}) \log p_{\Theta}(\mathbf{x}) d\mathbf{x} \right)\end{aligned}$$

On peut alors écrire l'objectif de l'entraînement d'un VAE comme :

$$\mathcal{L}(\Theta) = \mathbb{E}_{q(\mathbf{x})} [\log p_{\Theta}(\mathbf{x})] - \mathbb{E}_{q(\mathbf{x})} [\log q(\mathbf{x})]$$

KL-DIVERGENCE ET LOG-LIKELIHOOD

$$\begin{aligned}\hat{\Theta} &= \arg \min_{\Theta} D_{\text{KL}}(q \parallel p_{\Theta}) \\ (1) &= \arg \min_{\Theta} \left(\mathbb{E}_{q(\mathbf{x})} [\log q(\mathbf{x})] - \mathbb{E}_{q(\mathbf{x})} [\log p_{\Theta}(\mathbf{x})] \right) \\ (\text{terme indép. de } \Theta) &= \arg \min_{\Theta} \left(-\mathbb{E}_{q(\mathbf{x})} [\log p_{\Theta}(\mathbf{x})] \right) \\ &= \arg \max_{\Theta} \left(\mathbb{E}_{q(\mathbf{x})} [\log p_{\Theta}(\mathbf{x})] \right) \\ &= \arg \max_{\Theta} \left(\int q(\mathbf{x}) \log p_{\Theta}(\mathbf{x}) d\mathbf{x} \right)\end{aligned}$$

► Ainsi, minimiser la divergence de Kullback–Leibler entre q et p_{Θ} équivaut à maximiser la log-vraisemblance de $p_{\Theta}(\mathbf{x})$

$$\int q(\mathbf{x}) \log p_{\Theta}(\mathbf{x}) d\mathbf{x} = \mathbb{E}_q[\log p_{\Theta}(\mathbf{x})] \quad (2)$$

KL-DIVERGENCE ET LOG-LIKELIHOOD

$$\begin{aligned}\hat{\Theta} &= \arg \min_{\Theta} D_{\text{KL}}(q \parallel p_{\Theta}) \\ (1) &= \arg \min_{\Theta} \left(\mathbb{E}_{q(\mathbf{x})} [\log q(\mathbf{x})] - \mathbb{E}_{q(\mathbf{x})} [\log p_{\Theta}(\mathbf{x})] \right) \\ (\text{terme indép. de } \Theta) &= \arg \min_{\Theta} \left(-\mathbb{E}_{q(\mathbf{x})} [\log p_{\Theta}(\mathbf{x})] \right) \\ &= \arg \max_{\Theta} \left(\mathbb{E}_{q(\mathbf{x})} [\log p_{\Theta}(\mathbf{x})] \right) \\ &= \arg \max_{\Theta} \left(\int q(\mathbf{x}) \log p_{\Theta}(\mathbf{x}) d\mathbf{x} \right)\end{aligned}$$

► Ainsi, minimiser la divergence de Kullback–Leibler entre q et p_{Θ} équivaut à maximiser la log-vraisemblance de $p_{\Theta}(\mathbf{x})$

$$\int q(\mathbf{x}) \log p_{\Theta}(\mathbf{x}) d\mathbf{x} = \mathbb{E}_{q(\mathbf{x})} [\log p_{\Theta}(\mathbf{x})] \quad (2)$$

KL-DIVERGENCE ET LOG-LIKELIHOOD

$$\begin{aligned}\hat{\Theta} &= \arg \min_{\Theta} D_{\text{KL}}(q \parallel p_{\Theta}) \\ (1) &= \arg \min_{\Theta} \left(\mathbb{E}_{q(\mathbf{x})} [\log q(\mathbf{x})] - \mathbb{E}_{q(\mathbf{x})} [\log p_{\Theta}(\mathbf{x})] \right) \\ (\text{terme indép. de } \Theta) &= \arg \min_{\Theta} \left(-\mathbb{E}_{q(\mathbf{x})} [\log p_{\Theta}(\mathbf{x})] \right) \\ &= \arg \max_{\Theta} \left(\mathbb{E}_{q(\mathbf{x})} [\log p_{\Theta}(\mathbf{x})] \right) \\ &= \arg \max_{\Theta} \left(\int q(\mathbf{x}) \log p_{\Theta}(\mathbf{x}) d\mathbf{x} \right)\end{aligned}$$

- Ainsi, minimiser la *divergence de Kullback–Leibler* entre q et p_{Θ} équivaut à maximiser la *log-vraisemblance* de $p_{\Theta}(\mathbf{x})$

$$\int q(\mathbf{x}) \log p_{\Theta}(\mathbf{x}) d\mathbf{x} = \mathbb{E}_{q(\mathbf{x})} [\log p_{\Theta}(\mathbf{x})] \quad (2)$$

KL-DIVERGENCE ET LOG-LIKELIHOOD

- ▶ Dans la log-vraisemblance (2), le terme $p_{\Theta}(\mathbf{x})$ s'obtient en marginalisant sur la variable latente \mathbf{z} :

$$p_{\Theta}(\mathbf{x}) = \int p_{\Theta}(\mathbf{x}, \mathbf{z}) d\mathbf{z} = \int p_{\Theta}(\mathbf{x} \mid \mathbf{z}) p_{\Theta}(\mathbf{z}) d\mathbf{z}.$$

- ▶ Mais cette intégrale est **intractable**!
 - L'intégrale porte sur tout l'espace latent \mathbf{Z} (grande dim).
 - $p_{\Theta}(\mathbf{x} \mid \mathbf{z})$ et $p_{\Theta}(\mathbf{z})$ n'ont pas de forme analytique intégrable.

KL-DIVERGENCE ET LOG-LIKELIHOOD

- ▶ Dans la log-vraisemblance (2), le terme $p_{\Theta}(\mathbf{x})$ s'obtient en marginalisant sur la variable latente \mathbf{z} :

$$p_{\Theta}(\mathbf{x}) = \int p_{\Theta}(\mathbf{x}, \mathbf{z}) d\mathbf{z} = \int p_{\Theta}(\mathbf{x} \mid \mathbf{z}) p_{\Theta}(\mathbf{z}) d\mathbf{z}.$$

- ▶ Mais cette intégrale est **intractable**!
 - L'intégrale porte sur tout l'espace latent \mathbf{Z} (grande dim).
 - $p_{\Theta}(\mathbf{x} \mid \mathbf{z})$ et $p_{\Theta}(\mathbf{z})$ n'ont pas de forme analytique intégrable.

KL-DIVERGENCE ET LOG-LIKELIHOOD

- ▶ Dans la log-vraisemblance (2), le terme $p_{\Theta}(\mathbf{x})$ s'obtient en marginalisant sur la variable latente \mathbf{z} :

$$p_{\Theta}(\mathbf{x}) = \int p_{\Theta}(\mathbf{x}, \mathbf{z}) d\mathbf{z} = \int p_{\Theta}(\mathbf{x} \mid \mathbf{z}) p_{\Theta}(\mathbf{z}) d\mathbf{z}.$$

- ▶ Mais cette intégrale est **intractable**!
 - L'intégrale porte sur tout l'espace latent \mathbf{Z} (grande dim).
 - $p_{\Theta}(\mathbf{x} \mid \mathbf{z})$ et $p_{\Theta}(\mathbf{z})$ n'ont pas de forme analytique intégrable.

KL-DIVERGENCE ET LOG-LIKELIHOOD

- ▶ Dans la log-vraisemblance (2), le terme $p_{\Theta}(\mathbf{x})$ s'obtient en marginalisant sur la variable latente \mathbf{z} :

$$p_{\Theta}(\mathbf{x}) = \int p_{\Theta}(\mathbf{x}, \mathbf{z}) d\mathbf{z} = \int p_{\Theta}(\mathbf{x} \mid \mathbf{z}) p_{\Theta}(\mathbf{z}) d\mathbf{z}.$$

- ▶ Mais cette intégrale est **intractable**!
 - L'intégrale porte sur tout l'espace latent \mathbf{Z} (grande dim).
 - $p_{\Theta}(\mathbf{x} \mid \mathbf{z})$ et $p_{\Theta}(\mathbf{z})$ n'ont pas de forme analytique intégrable.

KL-DIVERGENCE ET LOG-LIKELIHOOD

- ▶ Autrement dit, en pratique, on approximerait

$$p_{\Theta}(\mathbf{x}) = \int p_{\Theta}(\mathbf{x} \mid \mathbf{z}) p_{\Theta}(\mathbf{z}) d\mathbf{z}$$

à partir des données $\{\mathbf{x}_i\}_{i=1}^N$ comme une somme discrète

$$p_{\Theta}(\mathbf{x}_i) \approx \frac{1}{M} \sum_{j=1}^M p_{\Theta}(\mathbf{x}_i \mid \mathbf{z}_j) p_{\Theta}(\mathbf{z}_j),$$

où M est un grand nombre d'échantillons latents (pour que l'approximation soit précise, surtout en haute dimension).

- ▶ Cette somme doit être calculée pour chaque \mathbf{x}_i , ce qui est computationnellement **intractable**.

KL-DIVERGENCE ET LOG-LIKELIHOOD

- ▶ Autrement dit, en pratique, on approximerait

$$p_{\Theta}(\mathbf{x}) = \int p_{\Theta}(\mathbf{x} \mid \mathbf{z}) p_{\Theta}(\mathbf{z}) d\mathbf{z}$$

à partir des données $\{\mathbf{x}_i\}_{i=1}^N$ comme une somme discrète

$$p_{\Theta}(\mathbf{x}_i) \approx \frac{1}{M} \sum_{j=1}^M p_{\Theta}(\mathbf{x}_i \mid \mathbf{z}_j) p_{\Theta}(\mathbf{z}_j),$$

où M est un grand nombre d'échantillons latents (pour que l'approximation soit précise, surtout en haute dimension).

- ▶ Cette somme doit être calculée pour chaque \mathbf{x}_i , ce qui est computationnellement **intractable**.

LOSS FUNCTION

- Par contre, on peut montrer que (cf. notes manuscrites):

$$\begin{aligned} \log p_{\Theta}(\mathbf{x}) &\geq \mathbb{E}_{q_{\Phi}(\mathbf{z}|\mathbf{x})} \left[\log \underbrace{p_{\Theta}(\mathbf{x}|\mathbf{z})}_{\sim \mathcal{N}(g(\mathbf{z}; \Theta), \mathbf{I})} \right] - \\ &\quad \text{D}_{\text{KL}} \left(\underbrace{q_{\Phi}(\mathbf{z}|\mathbf{x})}_{\sim \mathcal{N}(\mu_{\mathbf{x}}, \Sigma_{\mathbf{x}})} \parallel \underbrace{p_{\Theta}(\mathbf{z})}_{\sim \mathcal{N}(\mathbf{0}, \mathbf{I})} \right) \end{aligned} \quad (3)$$

- En prenant l'espérance sur $q(\mathbf{x})$ à gauche et à droite, on a:

$$\begin{aligned} \mathbb{E}_{q(\mathbf{x})} [\log p_{\Theta}(\mathbf{x})] &\geq \mathbb{E}_{q(\mathbf{x})} \left[\mathbb{E}_{q_{\Phi}(\mathbf{z}|\mathbf{x})} \left[\log p_{\Theta}(\mathbf{x}|\mathbf{z}) \right] \right] - \\ &\quad \text{D}_{\text{KL}} \left(q_{\Phi}(\mathbf{z}|\mathbf{x}) \parallel p_{\Theta}(\mathbf{z}) \right) \end{aligned} \quad (4)$$

- Ces bornes inférieures (termes de droites) sont appelées **Evidence Lower Bounds (ELBOs)**.

LOSS FUNCTION

- Par contre, on peut montrer que (cf. notes manuscrites):

$$\begin{aligned} \log p_{\Theta}(\mathbf{x}) &\geq \mathbb{E}_{q_{\Phi}(\mathbf{z}|\mathbf{x})} \left[\log \underbrace{p_{\Theta}(\mathbf{x}|\mathbf{z})}_{\sim \mathcal{N}(g(\mathbf{z}; \Theta), \mathbf{I})} \right] - \\ &\quad \text{D}_{\text{KL}} \left(\underbrace{q_{\Phi}(\mathbf{z}|\mathbf{x})}_{\sim \mathcal{N}(\mu_{\mathbf{x}}, \Sigma_{\mathbf{x}})} \parallel \underbrace{p_{\Theta}(\mathbf{z})}_{\sim \mathcal{N}(\mathbf{0}, \mathbf{I})} \right) \end{aligned} \quad (3)$$

- En prenant l'espérance sur $q(\mathbf{x})$ à gauche et à droite, on a:

$$\begin{aligned} \mathbb{E}_{q(\mathbf{x})} [\log p_{\Theta}(\mathbf{x})] &\geq \mathbb{E}_{q(\mathbf{x})} \left[\mathbb{E}_{q_{\Phi}(\mathbf{z}|\mathbf{x})} \left[\log p_{\Theta}(\mathbf{x}|\mathbf{z}) \right] \right] - \\ &\quad \text{D}_{\text{KL}} \left(q_{\Phi}(\mathbf{z}|\mathbf{x}) \parallel p_{\Theta}(\mathbf{z}) \right) \end{aligned} \quad (4)$$

- Ces bornes inférieures (termes de droites) sont appelées **Evidence Lower Bounds (ELBOs)**.

LOSS FUNCTION

- Par contre, on peut montrer que (cf. notes manuscrites):

$$\begin{aligned} \log p_{\Theta}(\mathbf{x}) &\geq \mathbb{E}_{q_{\Phi}(\mathbf{z}|\mathbf{x})} \left[\log \underbrace{p_{\Theta}(\mathbf{x}|\mathbf{z})}_{\sim \mathcal{N}(g(\mathbf{z}; \Theta), \mathbf{I})} \right] - \\ &\quad D_{\text{KL}} \left(\underbrace{q_{\Phi}(\mathbf{z}|\mathbf{x})}_{\sim \mathcal{N}(\mu_{\mathbf{x}}, \Sigma_{\mathbf{x}})} \parallel \underbrace{p_{\Theta}(\mathbf{z})}_{\sim \mathcal{N}(\mathbf{0}, \mathbf{I})} \right) \end{aligned} \quad (3)$$

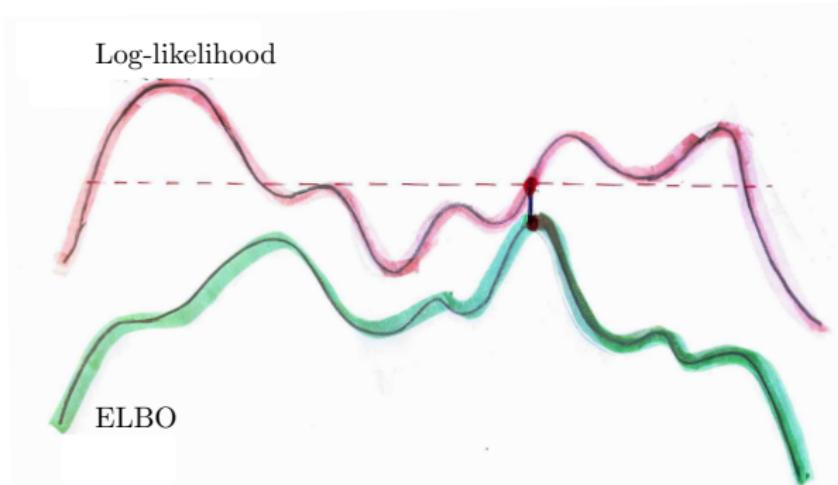
- En prenant l'espérance sur $q(\mathbf{x})$ à gauche et à droite, on a:

$$\begin{aligned} \mathbb{E}_{q(\mathbf{x})} [\log p_{\Theta}(\mathbf{x})] &\geq \mathbb{E}_{q(\mathbf{x})} \left[\mathbb{E}_{q_{\Phi}(\mathbf{z}|\mathbf{x})} \left[\log p_{\Theta}(\mathbf{x}|\mathbf{z}) \right] \right] - \\ &\quad D_{\text{KL}} \left(q_{\Phi}(\mathbf{z}|\mathbf{x}) \parallel p_{\Theta}(\mathbf{z}) \right) \end{aligned} \quad (4)$$

- Ces bornes inférieures (termes de droites) sont appelées **Evidence Lower Bounds (ELBOs)**.

LOSS FUNCTION

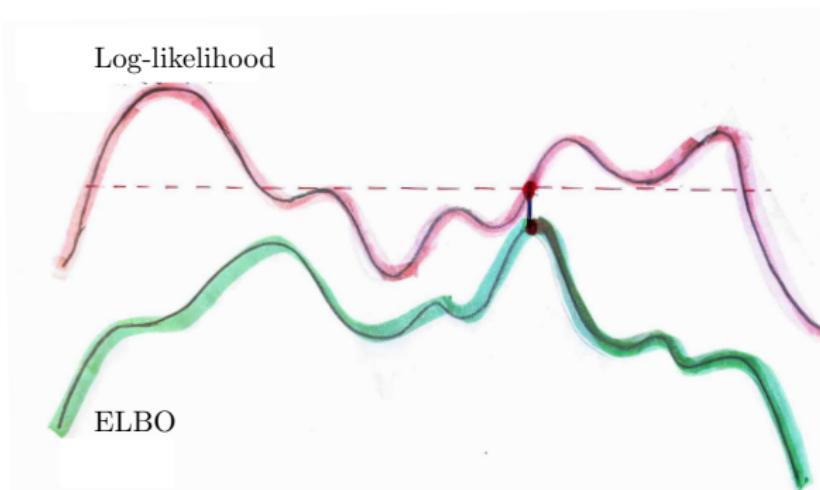
- ▶ Ainsi, on maximise l'ELBO (ou minimise son opposée) dans l'espoir de maximiser la log-vraisemblance (notre but).
- ▶ L'ELBO est un proxy optimisable de la log-vraisemblance, à propos de laquelle on ne sait pas grand chose...



Le max de l'ELBO ne correspond pas forcément au max de la log-vraisemblance.

LOSS FUNCTION

- ▶ Ainsi, on maximise l'ELBO (ou minimise son opposée) dans l'espoir de maximiser la log-vraisemblance (notre but).
- ▶ L'ELBO est un proxy optimisable de la log-vraisemblance, à propos de laquelle on ne sait pas grand chose...



Le max de l'ELBO ne correspond pas forcément au max de la log-vraisemblance.

LOSS FUNCTION

- On prend alors comme fonction de loss individuelle (cf. Eq. (3)):

$$\ell(\Phi, \Theta) = \underbrace{-\mathbb{E}_{q_\Phi(\mathbf{z}|\mathbf{x})} \left[\log p_\Theta(\mathbf{x} | \mathbf{z}) \right]}_{\text{reconstruction term}} + \underbrace{\text{D}_{\text{KL}} \left(q_\Phi(\mathbf{z} | \mathbf{x}) || p_\Theta(\mathbf{z}) \right)}_{\text{regularization term}} \quad (5)$$

- Et comme fonction de loss collective (cf. Eq. (4)):

$$\mathcal{L}(\Phi, \Theta) = \mathbb{E}_{q(\mathbf{x})} \left[\underbrace{-\mathbb{E}_{q_\Phi(\mathbf{z}|\mathbf{x})} \left[\log p_\Theta(\mathbf{x} | \mathbf{z}) \right]}_{\text{reconstruction term}} + \underbrace{\text{D}_{\text{KL}} \left(q_\Phi(\mathbf{z} | \mathbf{x}) || p_\Theta(\mathbf{z}) \right)}_{\text{regularization term}} \right] \quad (6)$$

LOSS FUNCTION

- On prend alors comme fonction de loss individuelle (cf. Eq. (3)):

$$\ell(\Phi, \Theta) = \underbrace{-\mathbb{E}_{q_\Phi(\mathbf{z}|\mathbf{x})} \left[\log p_\Theta(\mathbf{x} | \mathbf{z}) \right]}_{\text{reconstruction term}} + \underbrace{\text{D}_{\text{KL}} \left(q_\Phi(\mathbf{z} | \mathbf{x}) || p_\Theta(\mathbf{z}) \right)}_{\text{regularization term}} \quad (5)$$

- Et comme fonction de loss collective (cf. Eq. (4)):

$$\mathcal{L}(\Phi, \Theta) = \mathbb{E}_{q(\mathbf{x})} \left[\underbrace{-\mathbb{E}_{q_\Phi(\mathbf{z}|\mathbf{x})} \left[\log p_\Theta(\mathbf{x} | \mathbf{z}) \right]}_{\text{reconstruction term}} + \underbrace{\text{D}_{\text{KL}} \left(q_\Phi(\mathbf{z} | \mathbf{x}) || p_\Theta(\mathbf{z}) \right)}_{\text{regularization term}} \right] \quad (6)$$

LOSS FUNCTION

- Pour un batch de data $\{\mathbf{x}_i\}_{i=1}^B$, on a donc les *estimateurs* suivants des fonctions de loss (5) et (6):

$$\hat{\ell}(\Phi, \Theta) = -\log p_{\Theta}(\mathbf{x}_i | \mathbf{z}_i) + \hat{D}_{\text{KL}}\left(q_{\Phi}(\mathbf{z} | \mathbf{x}_i) \parallel p_{\Theta}(\mathbf{z})\right) \quad (7)$$

$$\hat{\mathcal{L}}(\Phi, \Theta) = \frac{1}{B} \sum_{i=1}^B \left[-\log p_{\Theta}(\mathbf{x}_i | \mathbf{z}_i) + \hat{D}_{\text{KL}}\left(q_{\Phi}(\mathbf{z} | \mathbf{x}_i) \parallel p_{\Theta}(\mathbf{z})\right) \right] \quad (8)$$

où $\mathbf{z}_i \sim \mathcal{N}(\boldsymbol{\mu}_{\mathbf{x}_i}, \boldsymbol{\Sigma}_{\mathbf{x}_i})$ et $\boldsymbol{\mu}_{\mathbf{x}_i}, \boldsymbol{\Sigma}_{\mathbf{x}_i} := f(\mathbf{x}_i; \Phi)$

- Il reste à exprimer ces loss en fonction des data \mathbf{x}_i ...

LOSS FUNCTION

- Pour un batch de data $\{\mathbf{x}_i\}_{i=1}^B$, on a donc les *estimateurs* suivants des fonctions de loss (5) et (6):

$$\hat{\ell}(\Phi, \Theta) = -\log p_{\Theta}(\mathbf{x}_i | \mathbf{z}_i) + \hat{D}_{\text{KL}}\left(q_{\Phi}(\mathbf{z} | \mathbf{x}_i) \parallel p_{\Theta}(\mathbf{z})\right) \quad (7)$$

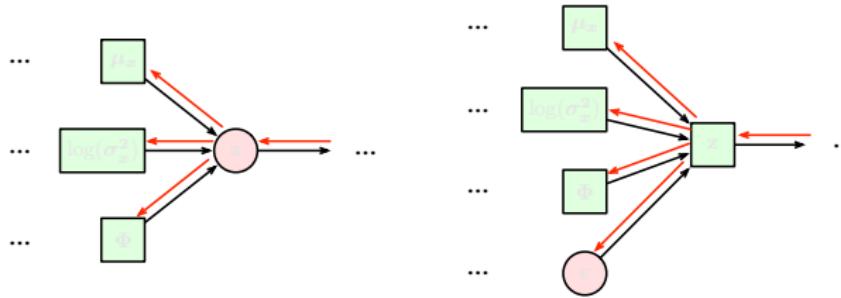
$$\hat{\mathcal{L}}(\Phi, \Theta) = \frac{1}{B} \sum_{i=1}^B \left[-\log p_{\Theta}(\mathbf{x}_i | \mathbf{z}_i) + \hat{D}_{\text{KL}}\left(q_{\Phi}(\mathbf{z} | \mathbf{x}_i) \parallel p_{\Theta}(\mathbf{z})\right) \right] \quad (8)$$

où $\mathbf{z}_i \sim \mathcal{N}(\boldsymbol{\mu}_{\mathbf{x}_i}, \boldsymbol{\Sigma}_{\mathbf{x}_i})$ et $\boldsymbol{\mu}_{\mathbf{x}_i}, \boldsymbol{\Sigma}_{\mathbf{x}_i} := f(\mathbf{x}_i; \Phi)$

- Il reste à exprimer ces loss en fonction des data \mathbf{x}_i ...

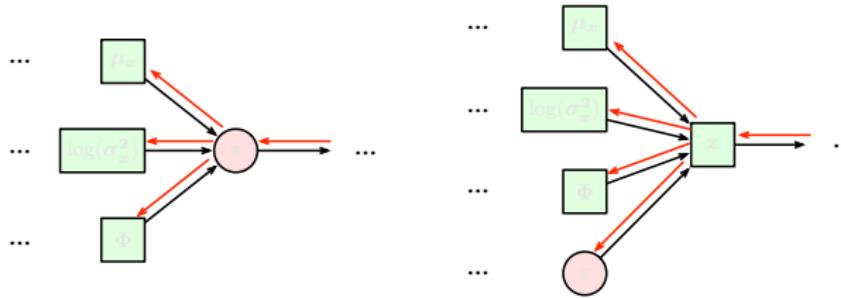
REPARAMETRIZATION TRICK

- ▶ Le processus de sampling n'est pas différentiable, ce qui pose problème pour l'algorithme de backpropagation.
- ▶ Pour remédier à cela, on recourt au "reparametrization trick" et au "log-var trick".
- ▶ Ce trick permet de sortir la stochasticité du graphe computationnel afin de préserver la différentiabilité par rapport à Φ .



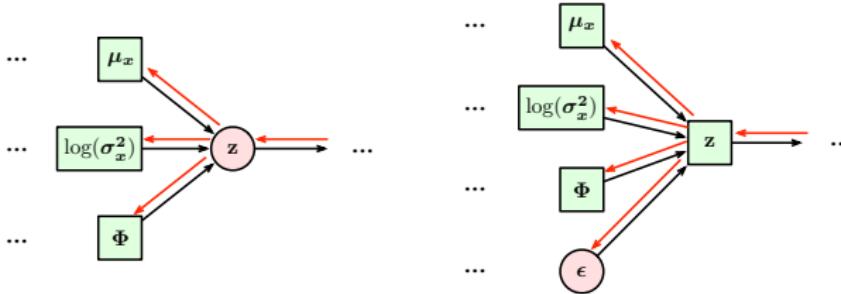
REPARAMETRIZATION TRICK

- ▶ Le processus de sampling n'est pas différentiable, ce qui pose problème pour l'algorithme de backpropagation.
- ▶ Pour remédier à cela, on recourt au "reparametrization trick" et au "log-var trick".
- ▶ Ce trick permet de sortir la stochasticité du graphe computationnel afin de préserver la différentiabilité par rapport à Φ .



REPARAMETRIZATION TRICK

- ▶ Le processus de sampling n'est pas différentiable, ce qui pose problème pour l'algorithme de backpropagation.
- ▶ Pour remédier à cela, on recourt au "reparametrization trick" et au "log-var trick".
- ▶ Ce trick permet de sortir la stochasticité du graphe computationnel afin de préserver la différentiabilité par rapport à Φ .



REPARAMETRIZATION TRICK

- ▶ Au lieu de sampler de z comme suit:

$$\begin{cases} \mu_x, \Sigma_x := f(x; \Phi) \\ z \sim q_\Phi(Z | X = x) = \mathcal{N}(\mu_x, \Sigma_x) \end{cases}$$

on procède de la manière suivante:

$$\begin{cases} \mu_x, \Sigma_x := f(x; \Phi) \\ z = \mu_x + \sigma_x \odot \epsilon \text{ où } \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \text{ et } \sigma_x = \text{diag}(\Sigma_x)^{\frac{1}{2}} \end{cases}$$

REPARAMETRIZATION TRICK

- ▶ En pratique, pour des raisons de stabilité, on apprend les vecteurs μ_x et $\log(\sigma_x^2)$ au lieu de μ_x et σ_x^2 .
 - Garantit automatiquement que $\sigma_x^2 = e^{\frac{1}{2}\log(\sigma_x^2)} > 0$, sans imposer de contraintes supplémentaires.
 - Meilleure stabilité numérique et convergence.
- ▶ On a alors

$$\begin{cases} \mu_x, \log(\sigma_x^2) := f(x; \Phi) \\ z = \mu_x + \sigma_x \odot \epsilon = \mu_x + e^{\frac{1}{2}\log(\sigma_x^2)} \odot \epsilon \quad \text{où } \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \end{cases}$$

REPARAMETRIZATION TRICK

- ▶ En pratique, pour des raisons de stabilité, on apprend les vecteurs μ_x et $\log(\sigma_x^2)$ au lieu de μ_x et σ_x^2 .
 - Garantit automatiquement que $\sigma_x^2 = e^{\frac{1}{2}\log(\sigma_x^2)} > 0$, sans imposer de contraintes supplémentaires.
 - Meilleure stabilité numérique et convergence.
- ▶ On a alors

$$\begin{cases} \mu_x, \log(\sigma_x^2) := f(x; \Phi) \\ z = \mu_x + \sigma_x \odot \epsilon = \mu_x + e^{\frac{1}{2}\log(\sigma_x^2)} \odot \epsilon \quad \text{où } \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \end{cases}$$

REPARAMETRIZATION TRICK

- ▶ En pratique, pour des raisons de stabilité, on apprend les vecteurs μ_x et $\log(\sigma_x^2)$ au lieu de μ_x et σ_x^2 .
 - Garantit automatiquement que $\sigma_x^2 = e^{\frac{1}{2}\log(\sigma_x^2)} > 0$, sans imposer de contraintes supplémentaires.
 - Meilleure stabilité numérique et convergence.
- ▶ On a alors

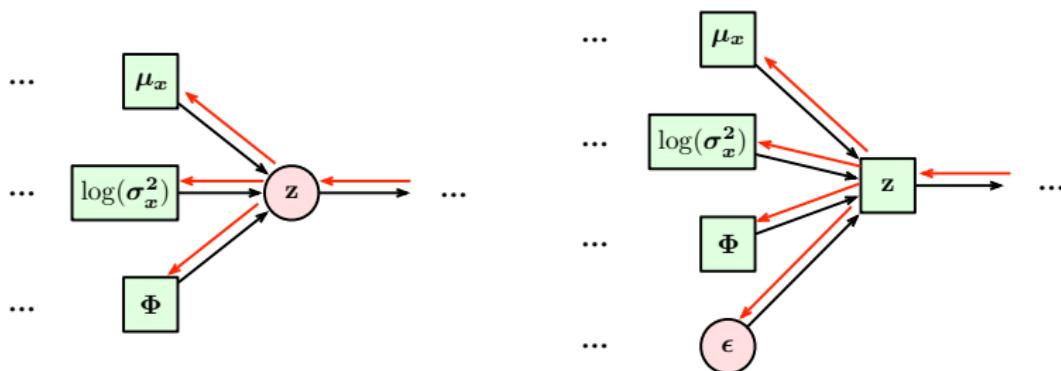
$$\begin{cases} \mu_x, \log(\sigma_x^2) := f(x; \Phi) \\ z = \mu_x + \sigma_x \odot \epsilon = \mu_x + e^{\frac{1}{2}\log(\sigma_x^2)} \odot \epsilon \quad \text{où } \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \end{cases}$$

REPARAMETRIZATION TRICK

- ▶ En pratique, pour des raisons de stabilité, on apprend les vecteurs μ_x et $\log(\sigma_x^2)$ au lieu de μ_x et σ_x^2 .
 - Garantit automatiquement que $\sigma_x^2 = e^{\frac{1}{2}\log(\sigma_x^2)} > 0$, sans imposer de contraintes supplémentaires.
 - Meilleure stabilité numérique et convergence.
- ▶ On a alors

$$\begin{cases} \mu_x, \log(\sigma_x^2) := f(x; \Phi) \\ z = \mu_x + \sigma_x \odot \epsilon = \mu_x + e^{\frac{1}{2}\log(\sigma_x^2)} \odot \epsilon \quad \text{où} \quad \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \end{cases}$$

REPARAMETERIZATION TRICK



Computational graphs without and with the reparametrization trick. Green and red nodes are deterministic and stochastic nodes, respectively. Red arrows illustrate backpropagation. The gradients w.r.t Φ can be computed deterministically.

FINAL LOSS

- Après le reparametrization trick, la loss collective (8) pour un batch $\{\mathbf{x}_i\}_{i=1}^B$ s'écrit:

$$\hat{\mathcal{L}}(\Phi, \Theta) = \frac{1}{B} \sum_{i=1}^B \left[-\log \underbrace{p_{\Theta}(\mathbf{x}_i | \mathbf{z}_i)}_{\sim \mathcal{N}(g(\mathbf{z}_i; \Theta), \mathbf{I})} + \hat{D}_{\text{KL}} \left(\underbrace{q_{\Phi}(\mathbf{z} | \mathbf{x}_i)}_{\sim \mathcal{N}(\boldsymbol{\mu}_{\mathbf{x}_i}, \boldsymbol{\Sigma}_{\mathbf{x}_i})} \parallel \underbrace{p_{\Theta}(\mathbf{z})}_{\sim \mathcal{N}(\mathbf{0}, \mathbf{I})} \right) \right]$$

avec $\boldsymbol{\mu}_{\mathbf{x}_i}, \log(\boldsymbol{\sigma}_{\mathbf{x}_i}^2) := f(\mathbf{x}_i; \Phi)$ (9)

$$\mathbf{z}_i = \boldsymbol{\mu}_{\mathbf{x}_i} + \boldsymbol{\sigma}_{\mathbf{x}_i} \odot \boldsymbol{\epsilon}, \quad \boldsymbol{\sigma}_{\mathbf{x}_i} = e^{\frac{1}{2} \log \boldsymbol{\sigma}_{\mathbf{x}_i}^2}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

FINAL LOSS

- ▶ On rappelle les expressions de densité de Gaussiennes multivariées (à covariance diagonale):

$$\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = (2\pi)^{-\frac{d}{2}} |\boldsymbol{\Sigma}|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{x} - \boldsymbol{\mu})\right)$$

- ▶ On rappelle les *hypothèses gaussiennes* sur les distributions:

$$q_\Phi(\mathbf{z} \mid \mathbf{x}_i) = \mathcal{N}(\boldsymbol{\mu}_{\mathbf{x}_i}, \text{diag}(\sigma_{\mathbf{x}_i}^2))$$

$$p_\Theta(\mathbf{z}) = \mathcal{N}(\mathbf{0}, \mathbf{I})$$

$$p_\Theta(\mathbf{x}_i \mid \mathbf{z}_i) = \mathcal{N}(g(\mathbf{z}_i; \Theta), \mathbf{I})$$

FINAL LOSS

- ▶ On rappelle les expressions de densité de Gaussiennes multivariées (à covariance diagonale):

$$\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = (2\pi)^{-\frac{d}{2}} |\boldsymbol{\Sigma}|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{x} - \boldsymbol{\mu})\right)$$

- ▶ On rappelle les **hypothèses gaussiennes** sur les distributions:

$$q_{\Phi}(\mathbf{z} \mid \mathbf{x}_i) = \mathcal{N}(\boldsymbol{\mu}_{x_i}, \text{diag}(\boldsymbol{\sigma}_{x_i}^2))$$

$$p_{\Theta}(\mathbf{z}) = \mathcal{N}(\mathbf{0}, \mathbf{I})$$

$$p_{\Theta}(\mathbf{x}_i \mid \mathbf{z}_i) = \mathcal{N}(g(\mathbf{z}_i; \Theta), \mathbf{I})$$

FINAL LOSS

- ▶ Pour ces gaussiennes, la distribution des data reconstruites et la KL-divergence s'approximent par:

$$-\log p_{\Theta}(\mathbf{x}_i \mid \mathbf{z}_i) = \frac{1}{2} \|\mathbf{x}_i - g(\mathbf{z}_i; \Theta)\|^2 + \text{cste}$$

$$\hat{D}_{\text{KL}}\left(q_{\Phi}(\mathbf{z} \mid \mathbf{x}_i) \parallel p_{\Theta}(\mathbf{z})\right) = \frac{1}{2} \sum_{j=1}^d \left(\mu_{x_i, j}^2 + \sigma_{x_i, j}^2 - 1 - \log \sigma_{x_i, j}^2 \right)$$

- ▶ Ainsi, la loss (9) exprimée à partir des data $\{\mathbf{x}_i\}_{i=1}^B$ est:

$$\hat{\mathcal{L}}(\Phi, \Theta) = \frac{1}{B} \sum_{i=1}^B \left[\underbrace{\frac{1}{2} \|\mathbf{x}_i - g(\mathbf{z}_i; \Theta)\|^2}_{\text{reconstruction}} + \underbrace{\frac{1}{2} \sum_{j=1}^d \left(\mu_{x_i, j}^2 + \sigma_{x_i, j}^2 - 1 - \log \sigma_{x_i, j}^2 \right)}_{\text{regularization (KL)}} \right]$$

avec $\mu_{\mathbf{x}_i}, \log(\sigma_{\mathbf{x}_i}^2) := f(\mathbf{x}_i; \Phi)$

$$\mathbf{z}_i = \mu_{\mathbf{x}_i} + \sigma_{\mathbf{x}_i} \odot \epsilon, \quad \sigma_{\mathbf{x}_i} = e^{\frac{1}{2} \log \sigma_{\mathbf{x}_i}^2}, \quad \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

FINAL LOSS

- ▶ Pour ces gaussiennes, la distribution des data reconstruites et la KL-divergence s'approximent par:

$$-\log p_{\Theta}(\mathbf{x}_i \mid \mathbf{z}_i) = \frac{1}{2} \|\mathbf{x}_i - g(\mathbf{z}_i; \Theta)\|^2 + \text{cste}$$

$$\hat{D}_{\text{KL}}\left(q_{\Phi}(\mathbf{z} \mid \mathbf{x}_i) \parallel p_{\Theta}(\mathbf{z})\right) = \frac{1}{2} \sum_{j=1}^d \left(\mu_{x_i, j}^2 + \sigma_{x_i, j}^2 - 1 - \log \sigma_{x_i, j}^2 \right)$$

- ▶ Ainsi, la loss (9) exprimée à partir des data $\{\mathbf{x}_i\}_{i=1}^B$ est:

$$\hat{\mathcal{L}}(\Phi, \Theta) = \frac{1}{B} \sum_{i=1}^B \left[\underbrace{\frac{1}{2} \|\mathbf{x}_i - g(\mathbf{z}_i; \Theta)\|^2}_{\text{reconstruction}} + \underbrace{\frac{1}{2} \sum_{j=1}^d \left(\mu_{x_i, j}^2 + \sigma_{x_i, j}^2 - 1 - \log \sigma_{x_i, j}^2 \right)}_{\text{regularization (KL)}} \right]$$

avec $\mu_{\mathbf{x}_i}, \log(\sigma_{\mathbf{x}_i}^2) := f(\mathbf{x}_i; \Phi)$

$$\mathbf{z}_i = \mu_{\mathbf{x}_i} + \sigma_{\mathbf{x}_i} \odot \epsilon, \quad \sigma_{\mathbf{x}_i} = e^{\frac{1}{2} \log \sigma_{\mathbf{x}_i}^2}, \quad \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

FINAL LOSS

- ▶ Pour ces gaussiennes, la distribution des data reconstruites et la KL-divergence s'approximent par:

$$-\log p_{\Theta}(\mathbf{x}_i \mid \mathbf{z}_i) = \frac{1}{2} \|\mathbf{x}_i - g(\mathbf{z}_i; \Theta)\|^2 + \text{cste}$$

$$\hat{D}_{\text{KL}}\left(q_{\Phi}(\mathbf{z} \mid \mathbf{x}_i) \parallel p_{\Theta}(\mathbf{z})\right) = \frac{1}{2} \sum_{j=1}^d (\mu_{x_i, j}^2 + \sigma_{x_i, j}^2 - 1 - \log \sigma_{x_i, j}^2)$$

- ▶ Ainsi, la loss (9) exprimée à partir des data $\{\mathbf{x}_i\}_{i=1}^B$ est:

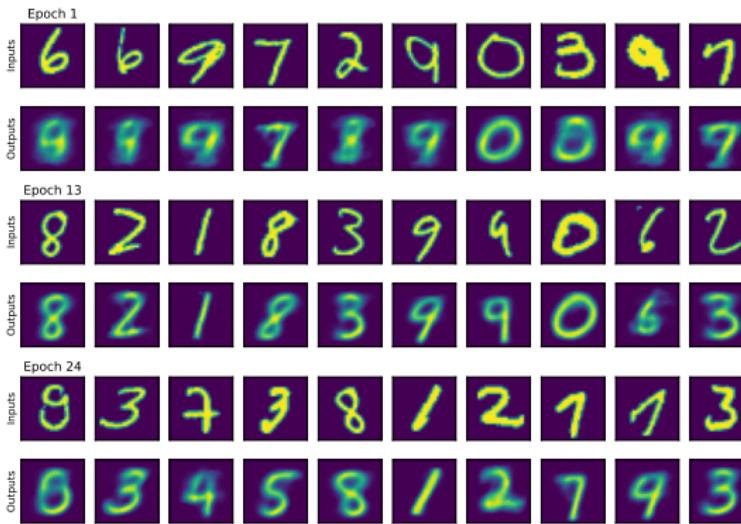
$$\hat{\mathcal{L}}(\Phi, \Theta) = \frac{1}{B} \sum_{i=1}^B \left[\underbrace{\frac{1}{2} \|\mathbf{x}_i - g(\mathbf{z}_i; \Theta)\|^2}_{\text{reconstruction}} + \underbrace{\frac{1}{2} \sum_{j=1}^d (\mu_{x_i, j}^2 + \sigma_{x_i, j}^2 - 1 - \log \sigma_{x_i, j}^2)}_{\text{regularization (KL)}} \right]$$

avec $\mu_{\mathbf{x}_i}, \log(\sigma_{\mathbf{x}_i}^2) := f(\mathbf{x}_i; \Phi)$

$$\mathbf{z}_i = \mu_{\mathbf{x}_i} + \sigma_{\mathbf{x}_i} \odot \epsilon, \quad \sigma_{\mathbf{x}_i} = e^{\frac{1}{2} \log \sigma_{\mathbf{x}_i}^2}, \quad \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

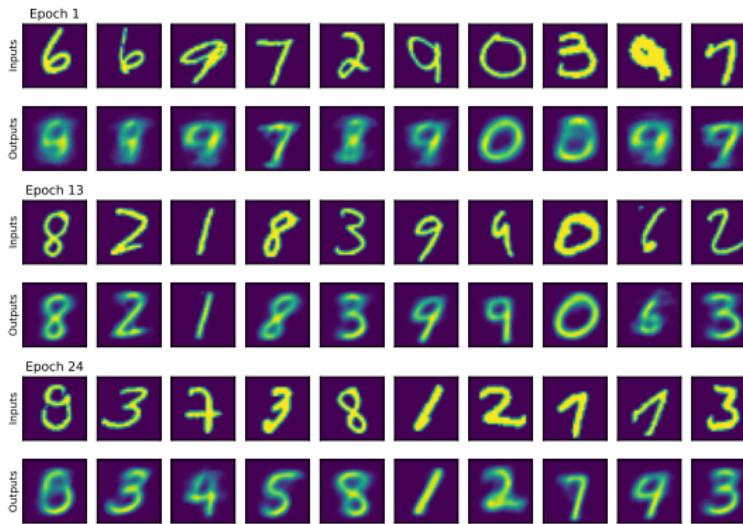
VAE LINÉAIRE: ESPACE LATENT DE DIMENSION 2

- ▶ L'espace latent est réparti de manière bien plus gaussienne et les capacités de génération sont censées être améliorées (poids des termes de reconstruction et régularisation: 10 et 0.001).
- ▶ Pas si clair sur un espace latent de dim 2 (cf. slides suivants)...



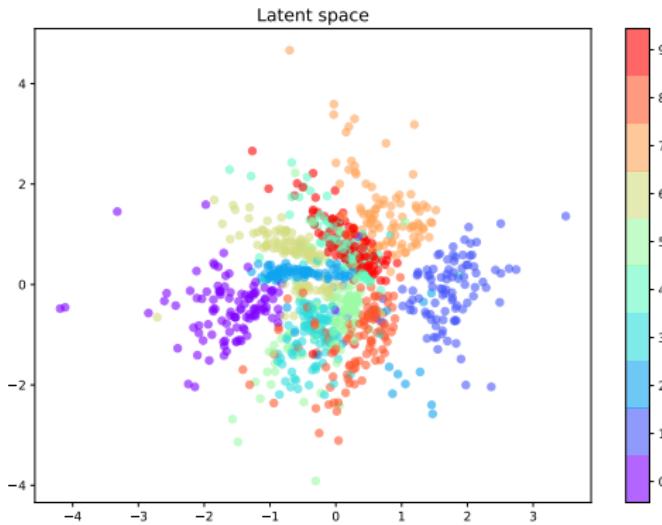
VAE LINÉAIRE: ESPACE LATENT DE DIMENSION 2

- ▶ L'espace latent est réparti de manière bien plus gaussienne et les capacités de génération sont censées être améliorées (poids des termes de reconstruction et régularisation: 10 et 0.001).
- ▶ Pas si clair sur un espace latent de dim 2 (cf. slides suivants)...



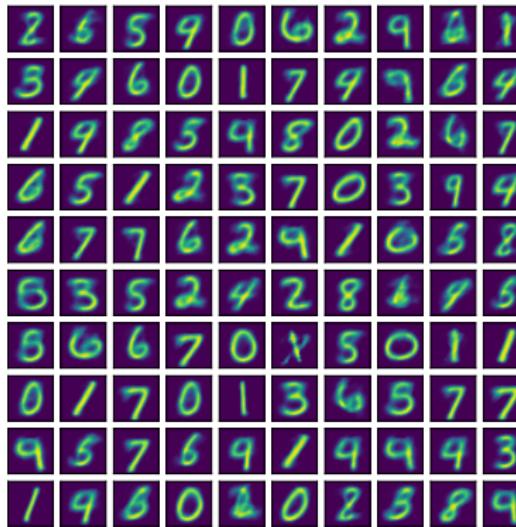
VAE LINÉAIRE: ESPACE LATENT DE DIMENSION 2

- ▶ L'espace latent est réparti de manière bien plus gaussienne et les capacités de génération sont censées être améliorées (poids des termes de reconstruction et régularisation: 10 et 0.001).
- ▶ Pas si clair sur un espace latent de dim 2 (cf. slides suivants)...



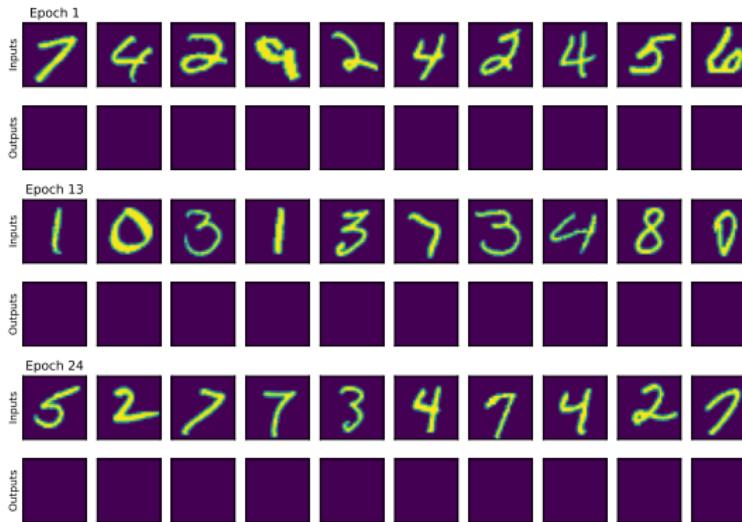
VAE LINÉAIRE: ESPACE LATENT DE DIMENSION 2

- ▶ L'espace latent est réparti de manière bien plus gaussienne et les capacités de génération sont censées être améliorées (poids des termes de reconstruction et régularisation: 10 et 0.001).
- ▶ Pas si clair sur un espace latent de dim 2 (cf. slides suivants)...



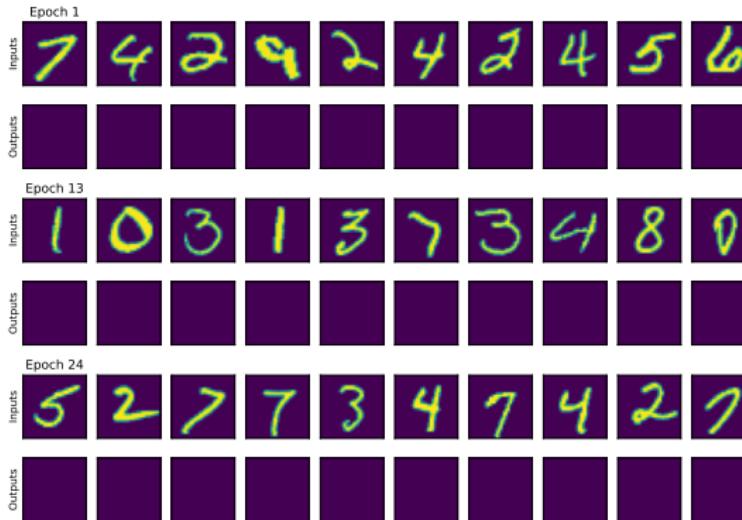
VAE LINÉAIRE: ESPACE LATENT DE DIMENSION 2

- ▶ Si on annule le terme de reconstruction, les points de l'espace latent sont distribués de manière parfaitement normale...
- ▶ mais le réseau n'a rien appris à reconstruire.



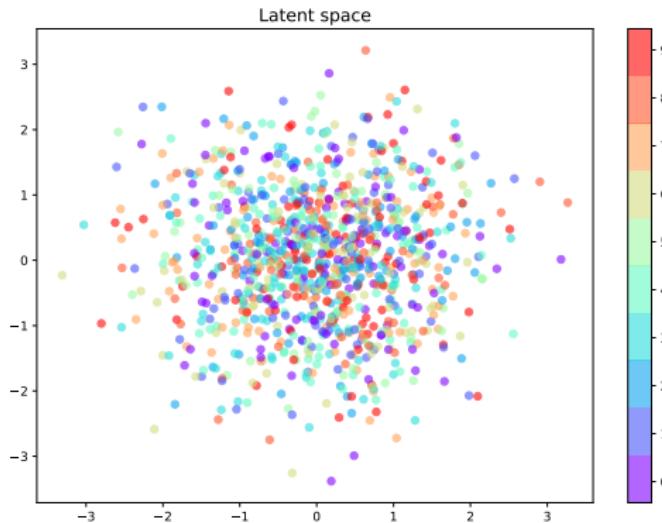
VAE LINÉAIRE: ESPACE LATENT DE DIMENSION 2

- ▶ Si on annule le terme de reconstruction, les points de l'espace latent sont distribués de manière parfaitement normale...
- ▶ mais le réseau n'a rien appris à reconstruire.



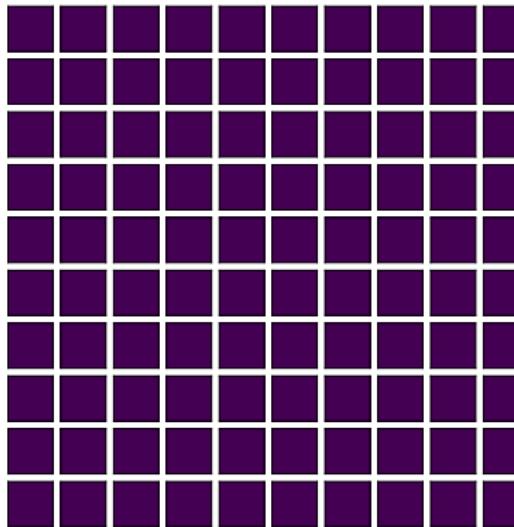
VAE LINÉAIRE: ESPACE LATENT DE DIMENSION 2

- ▶ Si on annule le terme de reconstruction, les points de l'espace latent sont distribués de manière parfaitement normale...
- ▶ mais le réseau n'a rien appris à reconstruire.



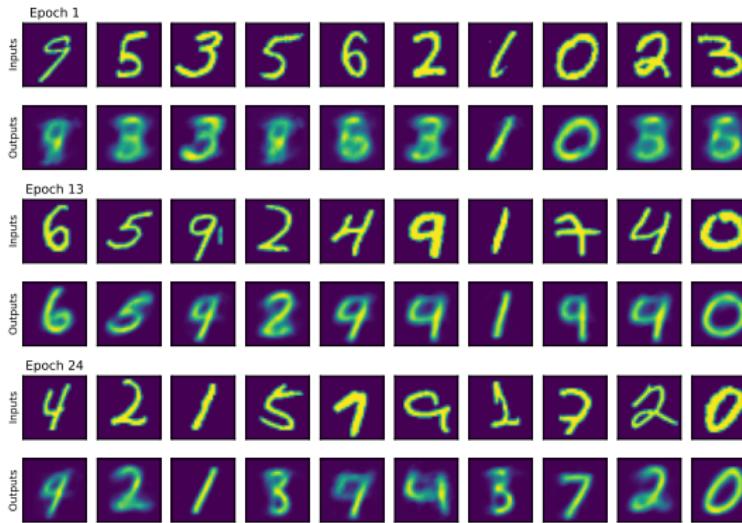
VAE LINÉAIRE: ESPACE LATENT DE DIMENSION 2

- ▶ Si on annule le terme de reconstruction, les points de l'espace latent sont distribués de manière parfaitement normale...
- ▶ mais le réseau n'a rien appris à reconstruire.



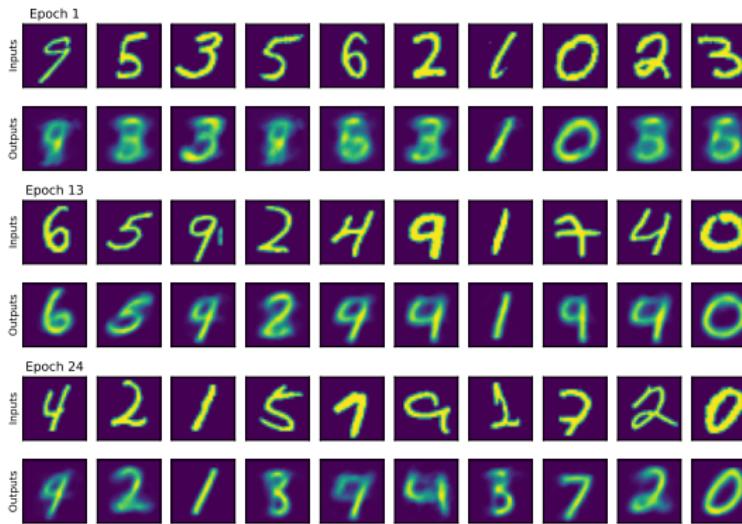
VAE LINÉAIRE: ESPACE LATENT DE DIMENSION 2

- ▶ Si, au contraire, on annule le terme de régularisation, les points de l'espace latent ne sont plus distribués de manière normale...
- ▶ le réseau a appris à reconstruire des data, mais en samplant dans $\mathcal{N}(0, 1)$, on génère des data assez similaires.



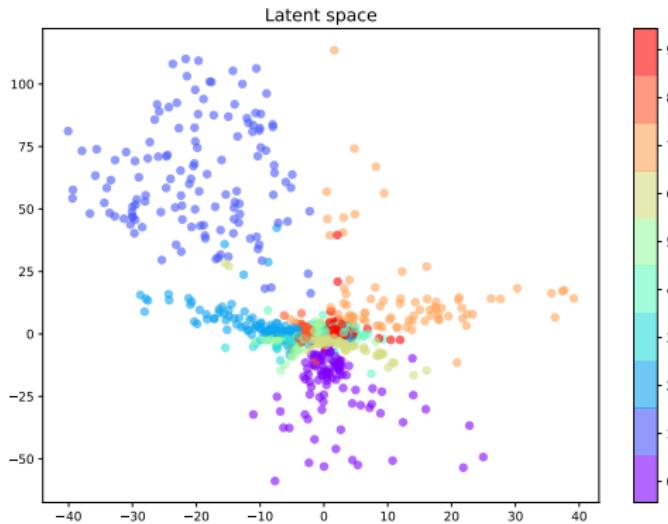
VAE LINÉAIRE: ESPACE LATENT DE DIMENSION 2

- ▶ Si, au contraire, on annule le terme de régularisation, les points de l'espace latent ne sont plus distribués de manière normale...
- ▶ le réseau a appris à reconstruire des data, mais en samplant dans $\mathcal{N}(0, 1)$, on génère des data assez similaires.



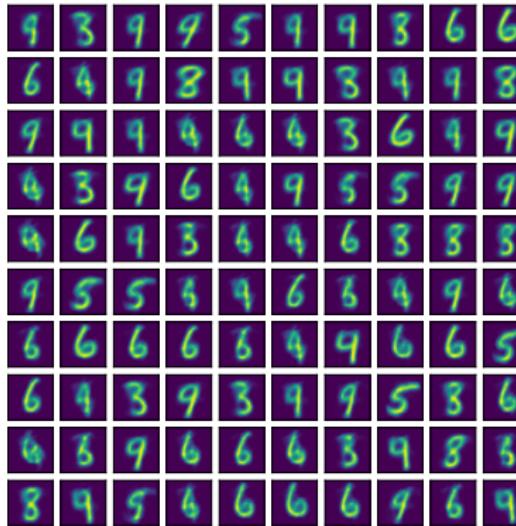
VAE LINÉAIRE: ESPACE LATENT DE DIMENSION 2

- ▶ Si, au contraire, on annule le terme de régularisation, les points de l'espace latent ne sont plus distribués de manière normale...
- ▶ le réseau a appris à reconstruire des data, mais en samplant dans $\mathcal{N}(0, 1)$, on génère des data assez similaires.



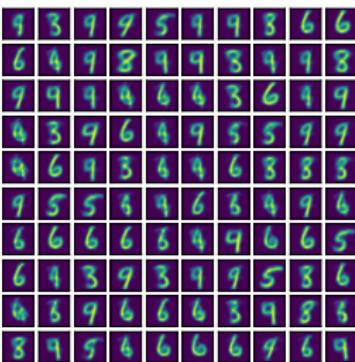
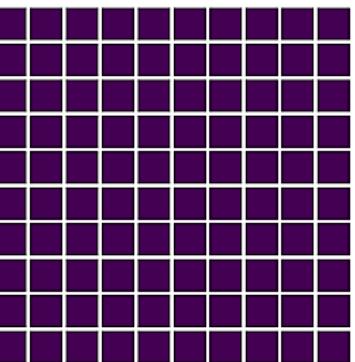
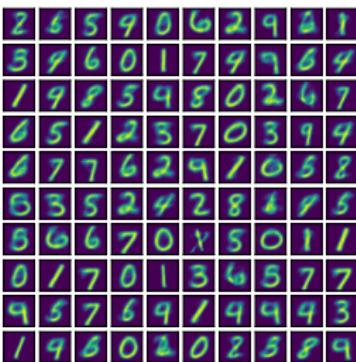
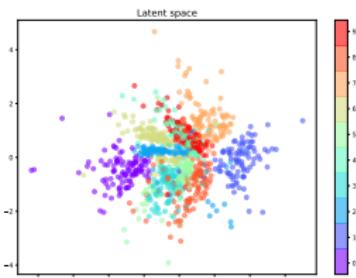
VAE LINÉAIRE: ESPACE LATENT DE DIMENSION 2

- ▶ Si, au contraire, on annule le terme de régularisation, les points de l'espace latent ne sont plus distribués de manière normale...
- ▶ le réseau a appris à reconstruire des data, mais en samplant dans $\mathcal{N}(0, 1)$, on génère des data assez similaires.



VAE LINÉAIRE: ESPACE LATENT DE DIMENSION 2

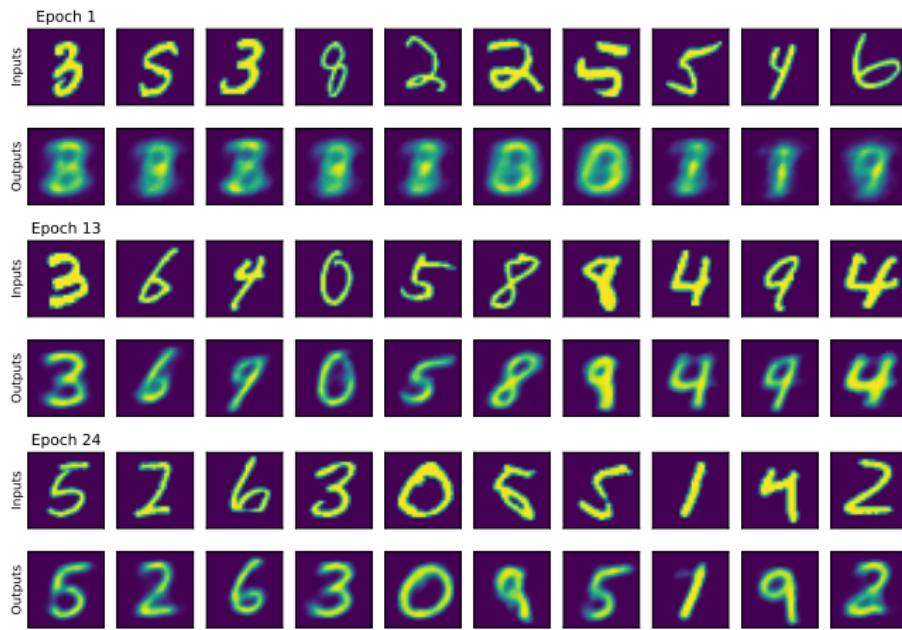
- Comparaison entre les trois situations.



Left: combination of MSE and KL losses; **Center:** only KL loss; **Right:** only MSE loss.

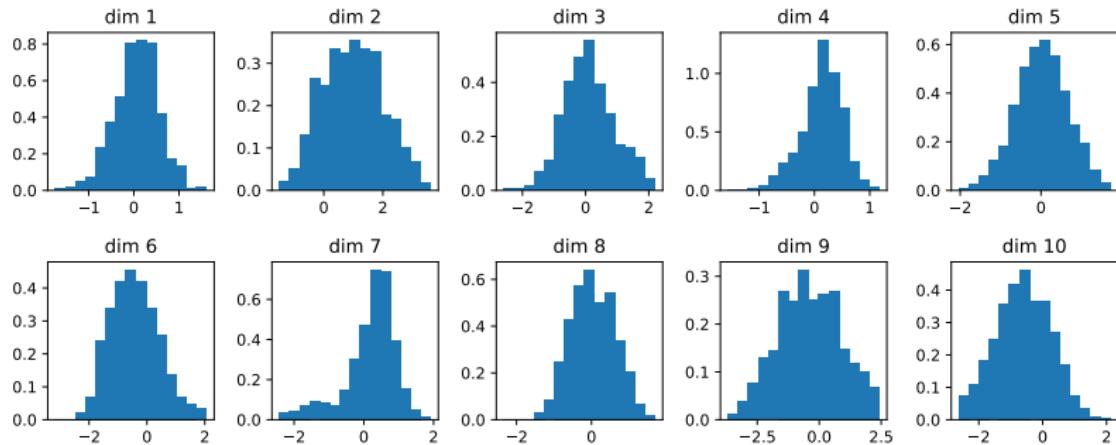
VAE LINÉAIRE: ESPACE LATENT DE DIMENSION 10

- ▶ En augmentant la dimension de l'espace latent, la génération de data est de meilleure qualité.



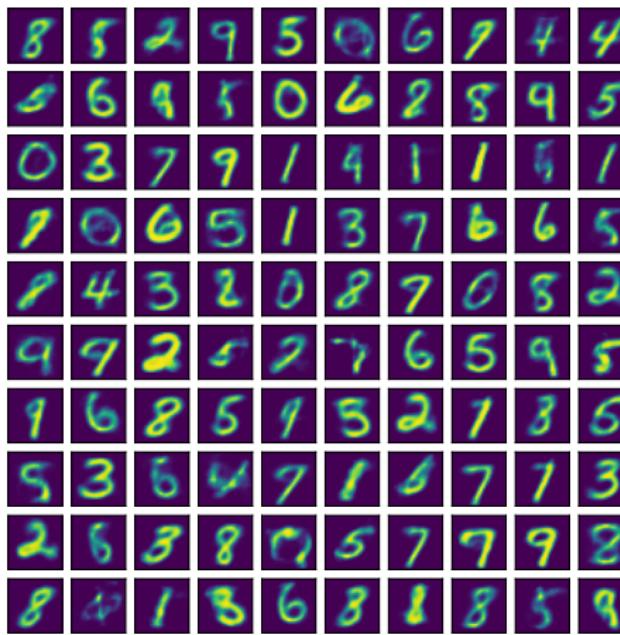
VAE LINÉAIRE: ESPACE LATENT DE DIMENSION 10

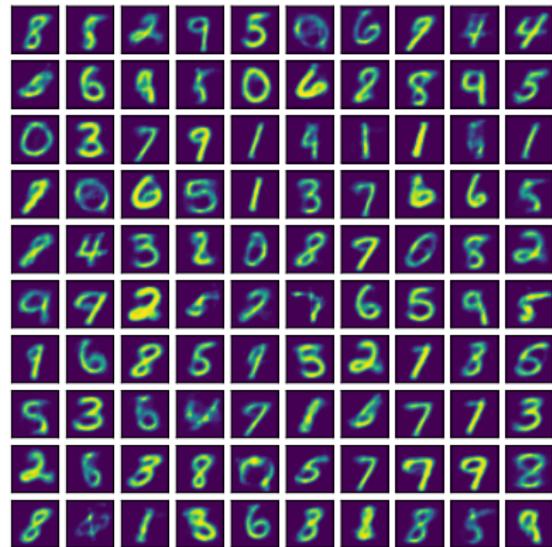
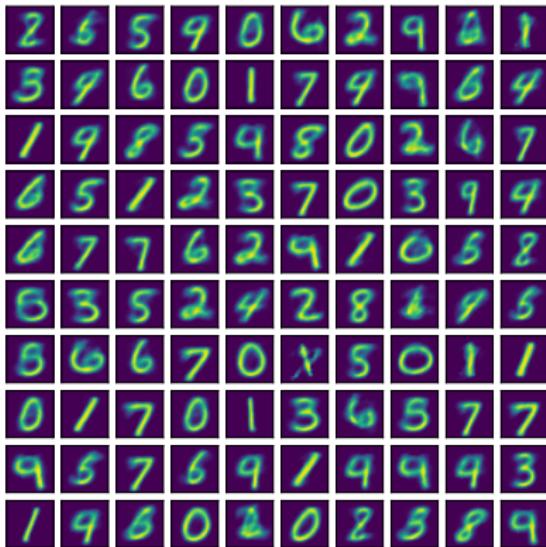
- ▶ En augmentant la dimension de l'espace latent, la génération de data est de meilleure qualité.



VAE LINÉAIRE: ESPACE LATENT DE DIMENSION 10

- ▶ En augmentant la dimension de l'espace latent, la génération de data est de meilleure qualité.

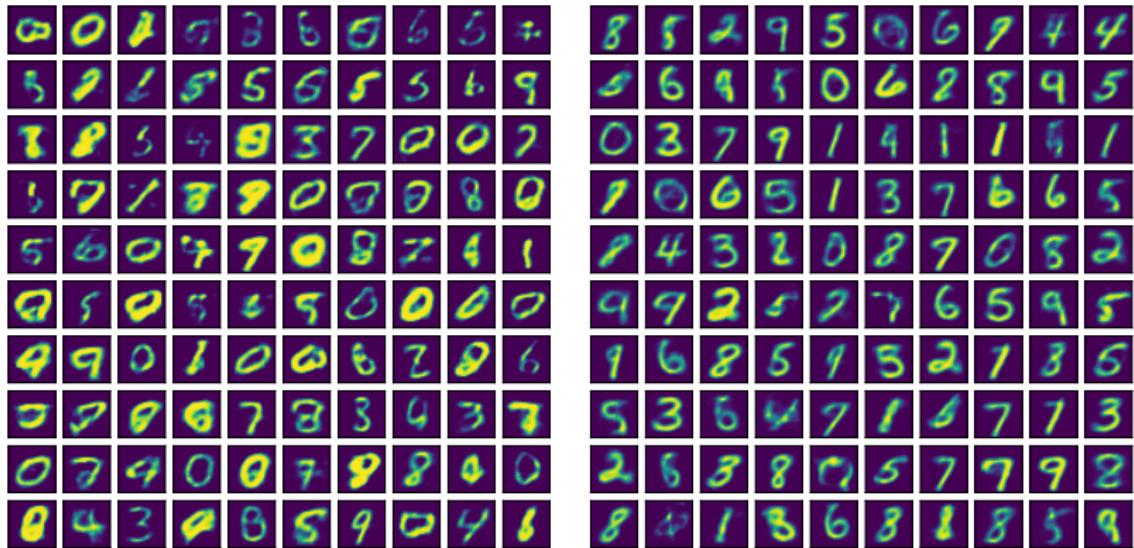


VAE LINÉAIRE: ESPACES LATENTS DE DIMENSIONS 2
ET 10

Data generation of two VAEs with latent spaces of dimensions 2 (left) and 10 (right), respectively.

AE VS VAE: ESPACE LATENT DE DIMENSION 10

- Génération de data: AE vs VAE.



Data generation of an AE (left) and a VAE (right) with latent space of dimension 10.

CONCLUSION

- ▶ Les autoencodeurs variationnels (VAEs) améliorent les capacités de génération de data par rapport aux autoencodeurs (AEs).
- ▶ L'ajout de stochasticité et l'imposition d'une distribution sur l'espace latent répond à des contraintes de continuité et de complétude sur ce espace.
- ▶ La continuité de l'espace latent rend les variables latentes interprétables: on comprend bien l'effet induit par la modification d'une variable alors que les autres restent fixes.

CONCLUSION

- ▶ Les autoencodeurs variationnels (VAEs) améliorent les capacités de génération de data par rapport aux autoencodeurs (AEs).
- ▶ L'ajout de stochasticité et l'imposition d'une distribution sur l'espace latent répond à des contraintes de continuité et de complétude sur ce espace.
- ▶ La continuité de l'espace latent rend les variables latentes interprétables: on comprend bien l'effet induit par la modification d'une variable alors que les autres restent fixes.

CONCLUSION

- ▶ Les autoencodeurs variationnels (VAEs) améliorent les capacités de génération de data par rapport aux autoencodeurs (AEs).
- ▶ L'ajout de stochasticité et l'imposition d'une distribution sur l'espace latent répond à des contraintes de continuité et de complétude sur ce espace.
- ▶ La continuité de l'espace latent rend les variables latentes interprétables: on comprend bien l'effet induit par la modification d'une variable alors que les autres restent fixes.

BIBLIOGRAPHIE

-  Alexander Amini (2022).
Alexander amini: Youtube channel.
-  CNRS - Formation FIDLE (2022).
Cnrs - formation fidle: Youtube channel.
-  Fleuret, F. (2022).
Deep Learning Course.
-  Kingma, D. P. and Welling, M. (2014).
Auto-encoding variational bayes.
In Bengio, Y. and LeCun, Y., editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.
-  Kingma, D. P. and Welling, M. (2019).
An introduction to variational autoencoders.
Found. Trends Mach. Learn., 12(4):307–392.
-  Sebastian Raschka (2022).
Sebastian raschka: Youtube channel.
-  Stephen Odaibo (2020).
Stephen odaibo : Meduim post.