RCP211 – Auto-encodeurs variationnels

Principe – Théorie – Apprentissage

Nicolas Audebert `nicolas.audebert@lecnam.net`

Conservatoire national des arts & métiers

3 novembre 2021

Plan du cours

Auto-encodeurs (récapitulatif, 1/2)

Principe de l'auto-encodeur

Apprendre une *compression* avec les pertes les + faibles possibles.
Pour une variable aléatoire $\mathbf{X} \in \mathbb{R}^n$, un réseau auto-encodeur modélise $\mathcal{H} = \mathcal{D} \circ \mathcal{E}$ telle que :

$$\|\mathcal{H}(x) - x\| \leq \varepsilon$$

- $\mathcal{E}$ représente l'encodeur $\mathbb{R}^n \to \mathbb{R}^d$
- $\mathcal{D}$ représente le décodeur $\mathbb{R}^d \to \mathbb{R}^n$

L'auto-encodeur construit un espace latent $\mathcal{Z}$ qui est l'espace de dimension d contenant les codes $z = \mathcal{E}(x)$.

Auto-encodeurs (récapitulatif, 2/2)

Fonction de coût

On cherche les poids θ du réseau de neurones $\mathcal{H}$ tels que :

$$\theta^* = \arg\min_\theta \mathcal{L}(x, \hat{x}) = \|\mathcal{D}(\mathcal{E}(x)) - x\|$$

La fonction de coût de reconstruction dépend de la tâche (erreur quadratique, absolue, entropie croisée...).

Optimisation

Apprentissage classique comme n'importe quel réseau de neurones : algorithme de rétroprogation et descente de gradient stochastique. Le décodeur et l'encodeur sont appris *conjointement* (le gradient est rétropropagé du décodeur vers l'encodeur).

Auto-encodeurs (récapitulatif, 2/2)

Fonction de coût

On cherche les poids θ du réseau de neurones $\mathcal{H}$ tels que :

$$\theta^* = \arg\min_\theta \mathcal{L}(x, \hat{x}) = \|\mathcal{D}(\mathcal{E}(x)) - x\|$$

La fonction de coût de reconstruction dépend de la tâche (erreur quadratique, absolue, entropie croisée...).

Optimisation

Apprentissage classique comme n'importe quel réseau de neurones : algorithme de rétroprogation et descente de gradient stochastique.
Le décodeur et l'encodeur sont appris *conjointement* (le gradient est rétropropagé du décodeur vers l'encodeur).

Applications des auto-encodeurs : débruitage

Débruitage/restauration

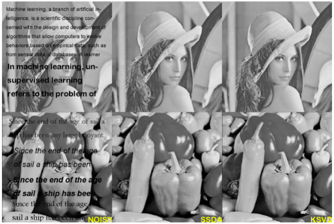
Fonction de coût :

$$\|\mathcal{H}(\tilde{x}) - x\| \leq \varepsilon$$

où $\tilde{x} = x + \delta$ une version bruitée de x.

Diverses applications

- Compression de signaux avec pertes
- Réduction de dimension non-linéaire



Applications des auto-encodeurs : débruitage

Débruitage/restauration

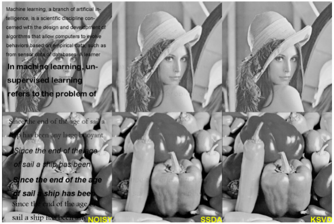
Fonction de coût :

$$\|\mathcal{H}(\tilde{x}) - x\| \leq \varepsilon$$

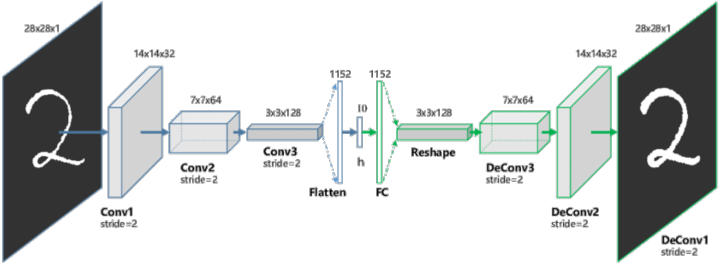
où $\tilde{x} = x + \delta$ une version bruitée de x.

Diverses applications

- Compression de signaux avec pertes
- Réduction de dimension non-linéaire



Applications des auto-encodeurs : AE convolutif



Guo et al., *Deep Clustering with Convolutional Autoencoders*, ICONIP 2017

$\rightarrow$ cf. séance de TP

L'auto-encodeur comme modèle génératif

Décoder = générer

Le décodeur $\mathcal{D}$ est un modèle génératif $\mathbb{P}(X|z)$.

1. Choisir au hasard un code latent z de dimension d
2. Décoder ce vecteur

Micro-quiz : à quoi peut-on s'attendre en pratique ?

1. Une observation synthétique mais plausible
2. Une reconstruction médiocre

Limites de cette approche

- Comment choisir z ? A priori on ne sait pas !
- Pas de régularité (= continuité) dans l'espace latent z
- $d << n$ mais d reste grand en pratique

L'auto-encodeur comme modèle génératif

Décoder = générer

Le décodeur $\mathcal{D}$ est un modèle génératif $\mathbb{P}(X|z)$.

1. Choisir au hasard un code latent z de dimension d
2. Décoder ce vecteur

Micro-quiz : à quoi peut-on s'attendre en pratique ?

1. Une observation synthétique mais plausible
2. Une reconstruction médiocre

Limites de cette approche

- Comment choisir z ? A priori on ne sait pas !
- Pas de régularité (= continuité) dans l'espace latent z
- $d << n$ mais d reste grand en pratique

L'auto-encodeur comme modèle génératif

Décoder = générer

Le décodeur $\mathcal{D}$ est un modèle génératif $\mathbb{P}(X|z)$.

1. Choisir au hasard un code latent z de dimension d
2. Décoder ce vecteur

Micro-quiz : à quoi peut-on s'attendre en pratique ?

1. Une observation synthétique mais plausible
2. Une reconstruction médiocre

Limites de cette approche

- Comment choisir z ? A priori on ne sait pas !
- Pas de régularité (= continuité) dans l'espace latent z
- $d << n$ mais d reste grand en pratique

Plan du cours

De l'auto-encodeur à l'auto-encodeur variationnel

Idée générale

une observation = une distribution

- une donnée x correspond à plusieurs z possibles,
- l'encodeur ne produit pas un code mais une *distribution* $p(z|x)$.

En général, on choisira une distribution gaussienne : l'encodeur produit μ_x, σ_x.

Avantages

- L'espace latent est moins clairsemé
- Échantillonnage plus aisé pour un x donné
- $p(z)$ plus facile à estimer

De l'auto-encodeur à l'auto-encodeur variationnel

Idée générale

une observation = une distribution

- une donnée x correspond à plusieurs z possibles,
- l'encodeur ne produit pas un code mais une *distribution* $p(z|x)$.

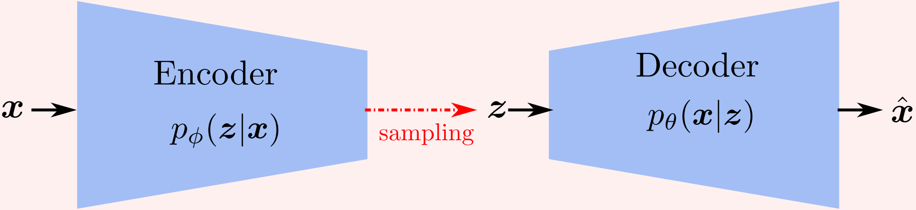
En général, on choisira une distribution gaussienne : l'encodeur produit μ_x, σ_x.

Avantages

- L'espace latent est moins clairsemé
- Échantillonnage plus aisé pour un x donné
- $p(z)$ plus facile à estimer

Schéma du VAE

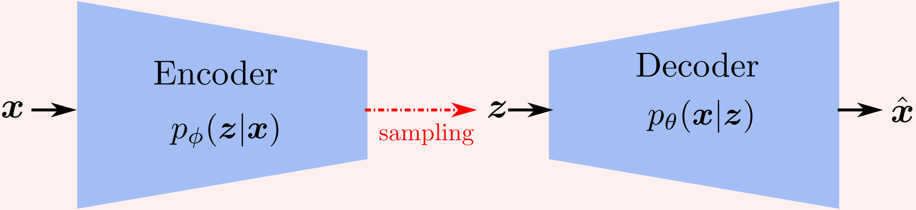
Schéma général



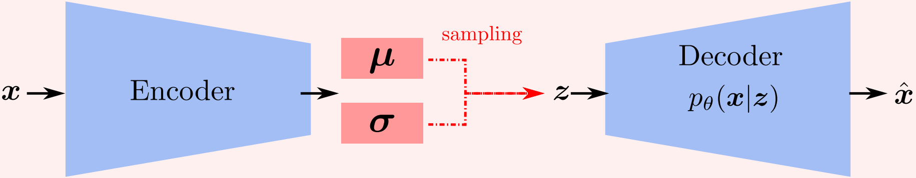
VAE gaussien

Schéma du VAE

Schéma général



VAE gaussien



$$p_\phi(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}\boldsymbol{I})$$

Générer des données

Récapitulons...

- L'encodeur produit une distribution $p_\phi(z|x)$
  - Typiquement, $p_\phi(z|x) = \mathcal{N}(\mu_x, \sigma_x)$
- On échantillonne $p_\phi(z|x)$ pour trouver un (ou plusieurs) code z
- Le décodeur reconstruit x à partir de z

Échantillonnage lors de l'inférence

À l'inférence, comment choisir z ?

- Option 1 : échantillonner sur la loi a posteriori
  - $p(z) = \frac{1}{n}\sum_{i=1}^{n} p_\phi(z|x) = \frac{1}{n}\sum_{i=1}^{n} \mathcal{N}(\mu_{x_i}, \sigma_{x_i})$
  - approximation mauvaise si peu de données, dimension de z grande, etc.
  - cher à calculer si beaucoup de données (n grand)
- Option 2 : forcer $p(z)$ à suivre une loi a priori

Générer des données

Récapitulons...

- L'encodeur produit une distribution $p_\phi(z|x)$
  - Typiquement, $p_\phi(z|x) = \mathcal{N}(\mu_x, \sigma_x)$
- On échantillonne $p_\phi(z|x)$ pour trouver un (ou plusieurs) code z
- Le décodeur reconstruit x à partir de z

Échantillonnage lors de l'inférence

À l'inférence, comment choisir z ?

- Option 1 : échantillonner sur la loi a posteriori
  - $p(z) = \frac{1}{n}\sum_{i=1}^{n} p_\phi(z|x) = \frac{1}{n}\sum_{i=1}^{n} \mathcal{N}(\mu_{x_i}, \sigma_{x_i})$
  - approximation mauvaise si peu de données, dimension de z grande, etc.
  - cher à calculer si beaucoup de données (n grand)
- Option 2 : forcer $p(z)$ à suivre une loi a priori

Générer des données

Récapitulons...

- L'encodeur produit une distribution $p_\phi(z|x)$
  - Typiquement, $p_\phi(z|x) = \mathcal{N}(\mu_x, \sigma_x)$
- On échantillonne $p_\phi(z|x)$ pour trouver un (ou plusieurs) code z
- Le décodeur reconstruit x à partir de z

Échantillonnage lors de l'inférence

À l'inférence, comment choisir z ?

- Option 1 : échantillonner sur la loi a posteriori
  - $p(z) = \frac{1}{n} \sum_{i=1}^{n} p_\phi(z|x) = \frac{1}{n} \sum_{i=1}^{n} \mathcal{N}(\mu_{x_i}, \sigma_{x_i})$
  - approximation mauvaise si peu de données, dimension de z grande, etc.
  - cher à calculer si beaucoup de données (n grand)
- **Option 2 : forcer $p(z)$ à suivre une loi a priori**

Régularisation

La divergence de Kullback-Leibler

Mesure de dissimilarité entre deux distributions $P(x)$ et $Q(x)$

- $D_{\text{KL}}(P|Q) = \sum_i P(x_i) \log \frac{P(x_i)}{Q(x_i)}$

A priori gaussien dans les VAE

Objectif : on exige que $p(z)$ suive approximativement la loi normale $\in \mathbb{R}^d$

- on impose :

$$D_{\text{KL}}\Big(\underbrace{q_\phi(z|x)}_{\text{approximation du postérieur par l'encodeur}} \Big| \underbrace{p(z)}_{\text{la loi a priori}}\Big) \leq \varepsilon$$

- avec $q_\phi(z|x) = \mathcal{N}(\mu_x, \sigma_x)$
- on choisit comme *prior* : $p(z) = \mathcal{N}(0, \mathbf{I})$

Régularisation

La divergence de Kullback-Leibler

Mesure de dissimilarité entre deux distributions $P(x)$ et $Q(x)$

- $D_{\mathsf{KL}}(P|Q) = \sum_i P(x_i) \log \frac{P(x_i)}{Q(x_i)}$

A priori gaussien dans les VAE

Objectif : on exige que $p(z)$ suive approximativement la loi normale $\in \mathbb{R}^d$

- on impose :

$$D_{\mathsf{KL}}(\underbrace{q_\phi(z|x)}_{\text{approximation du postérieur par l'encodeur}} \mid \underbrace{p(z)}_{\text{la loi a priori}}) \leq \varepsilon$$

- avec $q_\phi(z|x) = \mathcal{N}(\mu_x, \sigma_x)$
- on choisit comme *prior* : $p(z) = \mathcal{N}(0, \mathbf{I})$

Fonction de coût

Fonction de coût du VAE

$$\mathcal{L}(\theta, \phi; x) = \mathbb{E}_{q_\phi(z|x)}[||\hat{x} - x||] + D_{\mathsf{KL}}(q_\phi(z|x)|p(z))$$

- **Reconstruction** : “similarité” entre l’entrée et la sortie
  - dépend du problème (MSE, MAE, BCE...)
- **Régularisation** : divergence-KL (contrainte de proximité au *prior*)
  - Expression analytique dans le cas où tout est gaussien :

$$D_{\mathsf{KL}}(q_\phi(z|x)|p(z)) = \frac{1}{2}(\mathrm{tr}(\sigma_x) + \mu_x^t \mu_x - d - \log\det(\sigma_x)$$

Plan du cours

Théorie

Cadre formel

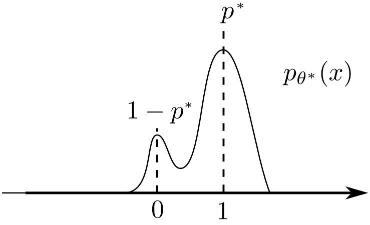
- $\mathcal{D} = \{x_1, \dots, x_n\}$ un jeu de données i.i.d. issues d'un processus génératif $p_{\mathcal{D}}$.
- On cherche le modèle génératif qui permet d'approcher $p_{\mathcal{D}}$.
- Soit $z \in \mathcal{Z}$ une variable latente qui suit une distribution *a priori* $p_{\theta^*}(z)$, paramétrée par θ^*.
  - on suppose que x_i s'obtient par la réalisation de $p_{\theta^*}(x_i|z_i)$
  - en pratique, on ne connaît ni θ^*, ni les variables latentes z_i

$$\boxed{\boldsymbol{z}} \longrightarrow \boxed{\boldsymbol{x}}$$

$$\sim p_{\theta^*}(\boldsymbol{z}) \qquad \sim p_{\theta^*}(\boldsymbol{x} \mid \boldsymbol{z})$$

Trouver la distribution latente

$$p_{\theta^*}(z) = \mathrm{Bern}(p) \qquad p_{\theta^*}(x \mid z) = \mathcal{N}(z, \sigma)$$



$$\theta = \{p\}$$

Modèle génératif

On cherche θ qui maximise la (log-)vraisemblance sur $\mathcal{D}$:

$$\hat{\theta} = \arg\max_{\theta} \mathbb{E}_{p_{\mathcal{D}}}[\log p_\theta(x)]$$

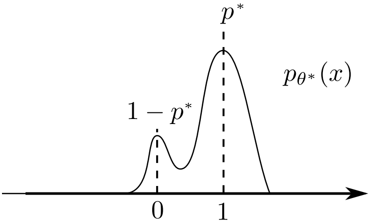
Problèmes

- $p_\theta(x)$ est difficile à calculer...
- $p_\theta(z|x)$ est inconnu...

On remplace $p_\theta(z|x)$ par une distribution approchée $q_\phi(z|x)$ de paramètres ϕ

Trouver la distribution latente

$p_{\theta^*}(z) = \mathrm{Bern}(p) \qquad p_{\theta^*}(x \mid z) = \mathcal{N}(z, \sigma)$



$$\theta = \{p\}$$

$$\begin{aligned} \mathrm{p}_\theta(x) =& (1-p)\mathrm{p}_\theta(x|0,\sigma) \\ &+ p\,\mathrm{p}_\theta(x|1,\sigma) \end{aligned}$$

sampling

Modèle génératif

On cherche θ qui maximise la (log-)vraisemblance sur $\mathcal{D}$:

$$\hat{\theta} = \arg\max_\theta \mathbb{E}_{p_{\mathcal{D}}}[\log p_\theta(x)]$$

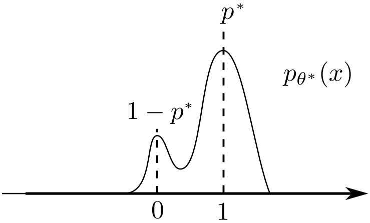
Problèmes

- $p_\theta(x)$ est difficile à calculer...
- $p_\theta(z|x)$ est inconnu...

On remplace $p_\theta(z|x)$ par une distribution approchée $q_\phi(z|x)$ de paramètres ϕ

Trouver la distribution latente

$p_{\theta^*}(z) = \mathrm{Bern}(p) \qquad p_{\theta^*}(x \mid z) = \mathcal{N}(z, \sigma)$



$\theta = \{p\}$

$$\begin{aligned}\mathrm{p}_\theta(x) = &(1-p)\mathrm{p}_\theta(x|0,\sigma) \\ &+ p\,\mathrm{p}_\theta(x|1,\sigma)\end{aligned}$$

sampling

Modèle génératif

On cherche θ qui maximise la (log-)vraisemblance sur $\mathcal{D}$:

$$\hat{\theta} = \arg\max_\theta \mathbb{E}_{p_\mathcal{D}}[\log p_\theta(x)]$$

Problèmes

- $p_\theta(x)$ est difficile à calculer...
- $p_\theta(z|x)$ est inconnu...

On remplace $p_\theta(z|x)$ par une distribution approchée $q_\phi(z|x)$ de paramètres ϕ

Approximation du postérieur

Comment choisir $q_\phi(z|x)$?

On cherche à être proche du véritable postérieur $p_\theta(z|x)$:

$$\hat{\phi} = \arg\min_{\phi} \mathrm{KL}(q_\phi(z|x)|p_\theta(z|x))$$

On peut montrer que :

$$\begin{aligned}\mathrm{KL}(q_\phi(z|x)|p_\theta(z|x)) &= -\mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] - \mathrm{KL}(q_\phi(z|x)|p_\theta(z)) + \log p_\theta(x) \\ &= -\mathcal{L}(\theta, \phi; x) + \log p_\theta(x)\end{aligned}$$

Minimiser la divergence KL revient à **maximiser** $\mathcal{L}$.

ELBO

Définition

ELBO (*evidence lower-bound*) est définie par :

$$\mathcal{L}(\theta, \phi; x) = \mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x|z)\right] - \mathrm{KL}\left(q_\phi(z|x) \,||\, p_\theta(z)\right)$$

Minimiser la divergence KL revient à **maximiser** l'ELBO.

Fonction objectif de l'inférence variationnelle

- On cherche θ qui vérifie $\hat{\theta} = \arg\max_\theta \mathbb{E}_{p_\mathcal{D}}\left[\log p_\theta(x)\right]$
- Or, $\log p_\theta(x) = \mathrm{KL}\left(q_\phi(z|x) | p_\theta(z|x)\right) + \mathcal{L}(\theta, \phi; x)$
- Comme la divergence KL est positive, il vient :

$$\log p_\theta(x) \geq \mathcal{L}(\theta, \phi; x)$$

Maximiser l'ELBO revient donc à **maximiser** la vraisemblance.

ELBO

Définition

ELBO (*evidence lower-bound*) est définie par :

$$\mathcal{L}(\theta, \phi; x) = \mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x|z)\right] - \mathsf{KL}\left(q_\phi(z|x) \,||\, p_\theta(z)\right)$$

Minimiser la divergence KL revient à **maximiser** l'ELBO.

Fonction objectif de l'inférence variationnelle

- On cherche θ qui vérifie $\hat{\theta} = \arg\max_\theta \mathbb{E}_{p_\mathcal{D}}\left[\log p_\theta(x)\right]$
- Or, $\log p_\theta(x) = \mathsf{KL}\left(q_\phi(z|x) | p_\theta(z|x)\right) + \mathcal{L}(\theta, \phi; x)$
- Comme la divergence KL est positive, il vient :

$$\log p_\theta(x) \geq \mathcal{L}(\theta, \phi; x)$$

Maximiser l'ELBO revient donc à **maximiser** la vraisemblance.

Lien avec les VAE

$$\mathcal{L}(\theta, \phi; x) = \underbrace{\mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x|z)\right]}_{\text{Espérance de la vraisemblance}} - \underbrace{\mathrm{KL}\left(q_\phi(z|x) \,||\, p_\theta(z)\right)}_{\text{Écart au prior}}$$

Quelles fonctions choisir comme postérieur et *prior* ?

- Un choix naturel pour $q_\phi(z|x)$: une gaussienne $\mathcal{N}(\mu, \sigma)$
  - ses paramètres dépendent de x
- Un choix naturel pour $p_\theta(z)$: la loi normale $\mathcal{N}(0, \mathbf{I})$
  - structure simple et facile à échantillonner

Vraisemblance

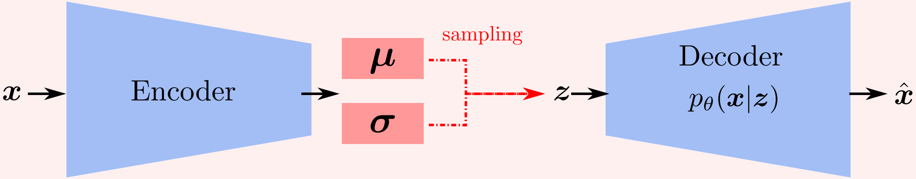
Dans notre cas, en notant f le décodeur :

$$\arg\max \mathbb{E}_{q_\phi(z|x)}[\log p(x|z)] = \arg\min \mathbb{E}_{q_\phi(z|x)} ||x - f(z)||$$

Plan du cours

Entraînement en pratique par descente de gradient

Schéma du VAE

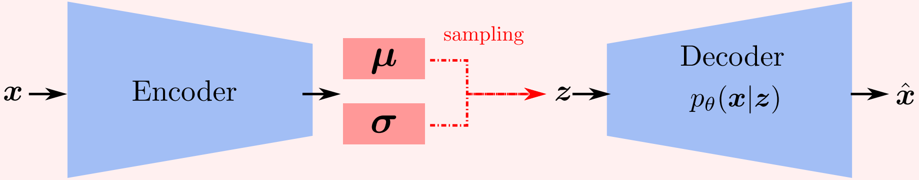


$$p_\phi(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma I})$$

1. Création d'un *batch* $\{x_i\}$
2. Passage dans l'encodeur
   - Calcul de mu_{x_i}, σ_{x_i} pour chaque i
3. Échantillonnage de $z_i \sim \mathcal{N}(\mu_{x_i}, \sigma_{x_i}) \leftarrow$ **non dérivable !**
4. Passage dans le décodeur
   - Calcul de $\hat{x}_i$
5. Calcul de la fonction de coût
6. Calcul du gradient, rétropagation

Entraînement en pratique par descente de gradient

Schéma du VAE

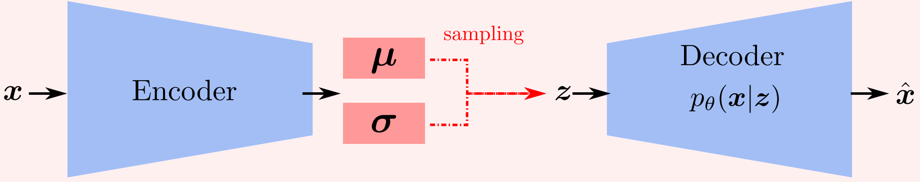


$$p_\phi(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma I})$$

1. Création d'un *batch* $\{x_i\}$
2. Passage dans l'encodeur
   - Calcul de mu_{x_i}, σ_{x_i} pour chaque i
3. Échantillonnage de $z_i \sim \mathcal{N}(\mu_{x_i}, \sigma_{x_i})$ $\leftarrow$ **non dérivable !**
4. Passage dans le décodeur
   - Calcul de $\hat{x}_i$
5. Calcul de la fonction de coût
6. Calcul du gradient, rétropagation

Entraînement en pratique par descente de gradient

Schéma du VAE

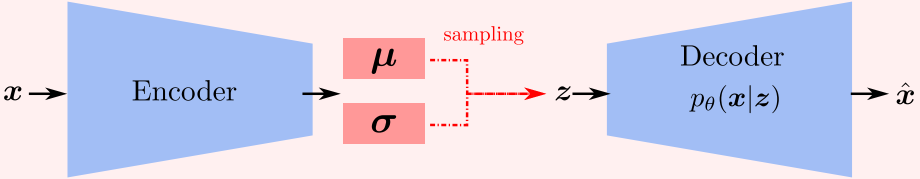


$$p_\phi(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}\boldsymbol{I})$$

1. Création d'un *batch* $\{x_i\}$
2. Passage dans l'encodeur
   - Calcul de mu_{x_i}, σ_{x_i} pour chaque i
3. Échantillonnage de $z_i \sim \mathcal{N}(\mu_{x_i}, \sigma_{x_i})$ $\leftarrow$ **non dérivable !**
4. Passage dans le décodeur
   - Calcul de $\hat{x}_i$
5. Calcul de la fonction de coût
6. Calcul du gradient, rétropagation

Entraînement en pratique par descente de gradient

Schéma du VAE

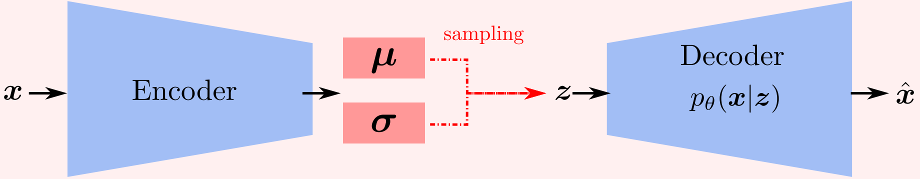


$$p_\phi(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma I})$$

1. Création d'un *batch* $\{x_i\}$
2. Passage dans l'encodeur
   - Calcul de mu_{x_i}, σ_{x_i} pour chaque i
3. Échantillonnage de $z_i \sim \mathcal{N}(\mu_{x_i}, \sigma_{x_i})$ $\leftarrow$ **non dérivable !**
4. Passage dans le décodeur
   - Calcul de $\hat{x}_i$
5. Calcul de la fonction de coût
6. Calcul du gradient, rétropagation

Entraînement en pratique par descente de gradient

Schéma du VAE

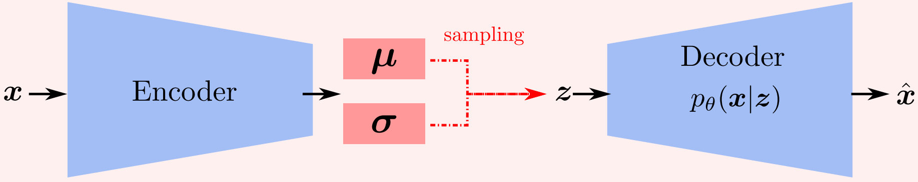


$$p_\phi(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}\boldsymbol{I})$$

1. Création d'un *batch* $\{x_i\}$
2. Passage dans l'encodeur
   - Calcul de mu_{x_i}, σ_{x_i} pour chaque i
3. Échantillonnage de $z_i \sim \mathcal{N}(\mu_{x_i}, \sigma_{x_i})$ $\leftarrow$ **non dérivable !**
4. Passage dans le décodeur
   - Calcul de $\hat{x}_i$
5. Calcul de la fonction de coût
6. Calcul du gradient, rétropagation

Entraînement en pratique par descente de gradient

Schéma du VAE

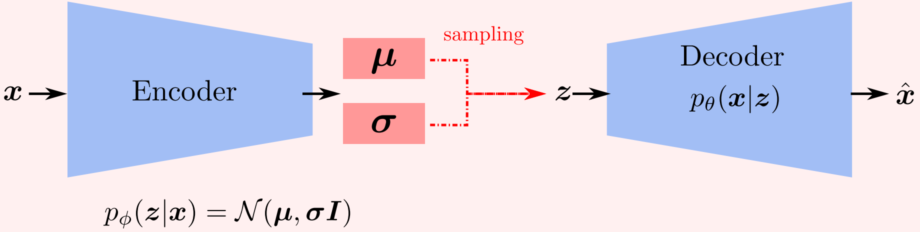


$p_\phi(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma I})$

1. Création d'un *batch* $\{x_i\}$
2. Passage dans l'encodeur
   - Calcul de mu_{x_i}, σ_{x_i} pour chaque i
3. Échantillonnage de $z_i \sim \mathcal{N}(\mu_{x_i}, \sigma_{x_i})$ $\leftarrow$ **non dérivable !**
4. Passage dans le décodeur
   - Calcul de $\hat{x}_i$
5. Calcul de la fonction de coût
6. Calcul du gradient, rétropagation

Entraînement en pratique par descente de gradient

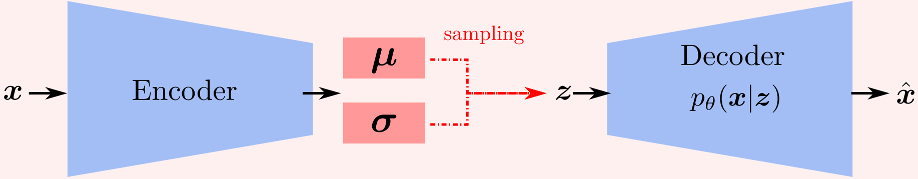
Schéma du VAE



1. Création d'un *batch* $\{x_i\}$
2. Passage dans l'encodeur
   - Calcul de mu_{x_i}, σ_{x_i} pour chaque i
3. Échantillonnage de $z_i \sim \mathcal{N}(\mu_{x_i}, \sigma_{x_i})$ $\leftarrow$ **non dérivable !**
4. Passage dans le décodeur
   - Calcul de $\hat{x}_i$
5. Calcul de la fonction de coût
6. Calcul du gradient, rétropagation

Entraînement en pratique par descente de gradient

Schéma du VAE

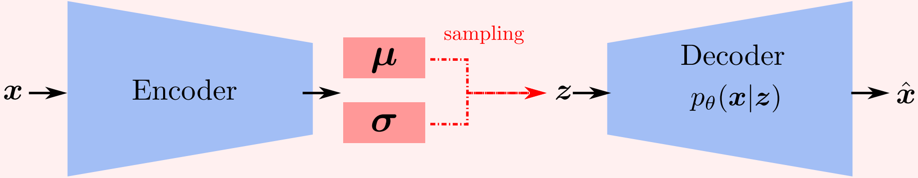


$$p_\phi(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma I})$$

1. Création d'un *batch* $\{x_i\}$
2. Passage dans l'encodeur
   - Calcul de mu_{x_i}, σ_{x_i} pour chaque i
3. Échantillonnage de $z_i \sim \mathcal{N}(\mu_{x_i}, \sigma_{x_i})$ $\leftarrow$ **non dérivable !**
4. Passage dans le décodeur
   - Calcul de $\hat{x}_i$
5. Calcul de la fonction de coût
6. Calcul du gradient, rétropagation

Entraînement en pratique par descente de gradient

Schéma du VAE



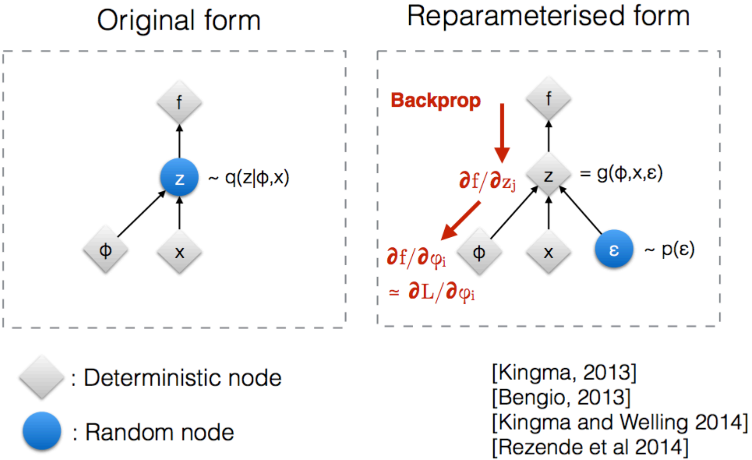
$$p_\phi(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma I})$$

1. Création d'un *batch* $\{x_i\}$
2. Passage dans l'encodeur
   - Calcul de mu_{x_i}, σ_{x_i} pour chaque i
3. Échantillonnage de $z_i \sim \mathcal{N}(\mu_{x_i}, \sigma_{x_i}) \leftarrow$ **non dérivable !**
4. Passage dans le décodeur
   - Calcul de $\hat{x}_i$
5. Calcul de la fonction de coût
6. Calcul du gradient, rétropagation

Reparametrization trick

Astuce

Ne pas échantillonner z directement en réécrivant : $z = \mu + \sigma \odot \varepsilon$ avec ε un bruit gaussien aléatoire $\sim \mathcal{N}(0, \mathbf{I})$.



β-VAE

$$\mathcal{L}(\theta, \phi; x) = \underbrace{\mathbb{E}_{q_\phi(z|x)} \|\hat{x} - x\|}_{\text{Erreur de reconstruction}} + \beta \cdot \underbrace{\mathrm{KL}\left(q_\phi(z|x) \,||\, p_\theta(z)\right)}_{\text{Écart au prior}}$$

β contrôle le dilemme reconstruction/structure de l'espace latent :

- $\beta = 1$: VAE classique,
- $\beta > 1$: encodage plus efficace et plus proche du *prior* mais reconstruction moins bonne,
- $\beta < 1$: meilleure reconstruction mais espace latent moins bien structuré.