Apprentissage statistique et réseaux de neurones

Chapitre 4: Auto-encodeurs.

Frédéric Richard
frederic.richard@univ-amu.fr
Master Mathématiques appliquées, statistique (1ère année), Parcours Data Science

2025

Auto-encodeur : définition.

- Données : for $i \in [\![1, n]\!]$, $x_i \in \mathcal{X}$, $\dim(\mathcal{X}) = m$.
- Encodeur : fonction paramétrées $E_{\boldsymbol{\theta}}$ de $\mathcal{X}$ à valeurs dans $\mathcal{Z}$ espace de dimension p.
- Décodeur : fonction paramétrées $D_{\boldsymbol{\psi}}$ de $\mathcal{Z}$ à valeurs dans $\mathcal{X}$.
- Auto-encodeur : $H_{\boldsymbol{\theta},\boldsymbol{\psi}} = D_{\boldsymbol{\psi}} \circ E_{\boldsymbol{\theta}}$ où, le plus souvent,

$$(\boldsymbol{\theta}, \boldsymbol{\psi}) \in \arg\min_{\boldsymbol{\Theta}\times\boldsymbol{\Psi}} \sum_{i=1}^{n} S(x_i, D_{\boldsymbol{\psi}} \circ E_{\boldsymbol{\theta}}(x_i)) + R(\boldsymbol{\theta}, \boldsymbol{\psi}).$$

  - S mesure de similarité, par exemple $S(x, y) = |x - y|^2$ (MSE).
  - R termes de régularisation.
- Reproduire les données x en passant par un espace latent $\mathcal{Z}$.
- Lorsque $p < m$: compression des données,
  - $z = E_{\boldsymbol{\theta}}(x)$ représentation compacte et "essentielle" de x.
  - $\tilde{x} = D_{\boldsymbol{\psi}}(z)$: reconstruction de x à partir de sa représentation.

Auto-encodeur : exemple de l'ACP.

- Données : for $i \in [\![1, n]\!]$, $x \in \mathbb{R}^m$.
- $u_j \in \mathbb{R}^m$ vecteurs propres associés aux valeurs propres λ_j de la covariance empirique $\widehat{\Sigma}_n = \frac{1}{n}\sum_{i=1}^n (x_i - \bar{x}_n)(x_i - \bar{x}_n)^T$.
- $\lambda_1 \geq \cdots \geq \lambda_m$ et p, $\frac{\sum_{j=1}^p \lambda_j}{\sum_{j=1}^n \lambda_j} > r$ (ex. $r = 0.95$).
- On peut définir

$$z = E(x) = \begin{pmatrix} u_1^T \\ \hline \\ \vdots \\ \hline \\ u_p^T \end{pmatrix} x \quad \text{et} \quad \tilde{x} = D(z) = \left(\begin{array}{c|c|c} u_1 & \cdots & u_p \end{array} \right) z.$$

- $\tilde{x} = D \circ E(x) = \sum_{j=1}^p \langle x, u_j \rangle u_j$ projection orthogonale de x sur $\mathrm{vect}(u_1, \cdots, u_p)$.

Auto-encodeur par réseau de neurones dense.

- Données : for $i \in [\![1, n]\!]$, $x \in \mathbb{R}^m$. Modèle :

$$z = E(x) = \varphi_e(Mx) \quad \text{et} \quad \tilde{x} = D(z) = \varphi_d(Pz).$$

  - M matrice de taille $m \times p$ (couche dense à p cellules)
  - P matrice de taille $p \times m$ (couche dense à m cellules),
  - φ_e, fonction d'activation de $\mathbb{R}^p$ dans $\mathbb{R}^p$.
  - φ_d, fonction d'activation de $\mathbb{R}^m$ dans $\mathbb{R}^m$.
- cas où φ_e et φ_d sont égales à l'identité : équivalent à l'ACP.
- Paramètres :
  - $\theta = M$ (pondération de la couche dense de l'encodeur)
  - $\psi = P$ (pondération de la couche dense du décodeur)
- Apprentissage en minimisant le MSE.

Auto-encodeur dense multicouche.

- Modèle :
  - Encodeur : $z = E(x) = u^{(K)}$, où $u^{(1)} = x$ et

$$\forall k \in [\![1, K-1]\!], u^{(k+1)} = \varphi_e^{(k)}(M^{(k)}u^{(k)}).$$

  - Décodeur : $\tilde{x} = D(z) = \tilde{v}^{(L)}$ où $v^{(1)} = z$ et

$$\forall L \in [\![1, L-1]\!], v^{(l+1)} = \varphi_d^{(l)}(P^{(l)}v^{(l)}).$$

- Conditions de compatibilité sur les tailles de matrice :
  - $M^{(k)}$ matrice de taille $m_k \times m_{k+1}$, $m_1 = m$, $m_K = p$.
  - $P^{(l)}$ matrice de taille $p_l \times p_{l+1}$, $p_1 = p$, $p_L = m$.
- Nombre de paramètres :
  - Encodeur : $\sum_{k=1}^{K-1} m_k \times m_{k+1}$.
  - Décodeur : $\sum_{l=1}^{L-1} p_l \times p_{l+1}$.

Auto-encodeur convolutionnel : encodeur.

- Encodeur : $z = E(x) = u^{(K)}$, où $u^{(1)} = x$ et

$$\forall k \in [\![1, K-1]\!], u^{(k+1,j)} = \mathrm{Pool}_{Q_k}\left(\varphi^{(k)}\left(C(u^{(k,\kappa(k,j))}; w^{(k,j)})\right)\right).$$

- A l'entrée de l'encodeur, x est de dimension $(n_{x,1}, n_{x,2}, 1)$.
- A l'issue de l'encodeur, z est de dimension $(n_{z,1}, n_{z,2}, n_{z,3})$.
- Pour que la couche cachée z soit une représentation compacte de x, on fait en sorte que sa taille soit inférieure à celle de la couche d'entrée :

$$n_{z,1}\, n_{z,2} < n_{z,3}\, n_{x,1}\, n_{x,2}.$$

Auto-encodeur convolutionnel : décodeur.

- A l'entrée de l'encodeur, x est de dimension $(n_{x,1}, n_{x,2}, 1)$.
- A l'issue de l'encodeur, z est de dimension $(n_{z,1}, n_{z,2}, n_{z,3})$.
- Décodeur : Comment reconstruire une image $\tilde{x}$ de la même dimension que z à partir de $u^{(K)}$?
- Il faut
  - des opérations pour réduire le nombre de canaux de $n_{z,3}$ à 1.
  - des opérations pour augmenter la taille des images de $(n_{z,1}, n_{z,2})$ à $(n_{x,1}, n_{x,2}$.

Convolution transposée.

- Convolution 1D de f avec un noyau v de taille M (et zero-padding) :
  - Forme indicielle :
    $\forall\, n \in [\![0, N-1]\!],\; g[n] = f * v[n] = \sum_{k=0}^{M-1} v[k]x[n-k].$
  - Forme matricielle : $g = Vf$.
- Convolution transposée :
  - Forme matricielle : $h = V^T g$ (ne donne pas f !).
  - Forme indicielle :
    $\forall\, n \in [\![0, N-1]\!],\; h[n] = g * v[n] = \sum_{k=0}^{M-1} v[k]g[n+k].$
- En tant que tel, n'augmente pas la dimension du signal.
- On l'applique à un signal sur-échantillonné, par exemple d'un facteur 2 :

$$\tilde{g} = (g[0], 0, g[1], 0, \cdots, g[N-1], 0).$$