Modèle génératif basé sur la fonction de score pour la prédiction de la température des océans

Groupe A7- Genhack2

December 24, 2022

1 Introduction

On dispose des données des températures des océans prélevées à six stations différentes sur 9618 jours consécutifs entre 1981 et 2008. Le but est d'apprendre la distribution de ces données afin de générer les températures des océans à ces six mêmes stations sur la période de 2008 à 2016, pour exactement 3288 jours. On note $x = (x_1, ..., x_6)$ le vecteur représentant les températures aux six stations, et on suppose que x provient d'une dristribution de probabilité $p(x)$. On veut approcher $p(x)$ par un modèle paramétrique $p_\theta(x)$ où

$$p_\theta(x) = \frac{e^{-f_\theta(x)}}{Z_\theta},$$

θ étant le paramètre du modèle, $f_\theta(x)$ une certaine fonction d'énergie et Z_θ la constante de normalisation.

2 Fonction de score

Plutôt que d'apprendre directement le modèle de distribution $p_\theta(x)$, on va apprendre sa fonction de score

$$s_\theta(x) = \nabla_x \log p_\theta(x) = -\nabla_x f_\theta(x).$$

Comme on peut le voir, $s_\theta(x)$ à l'avantage de ne pas dépendre de la constante Z_θ de normalisation, ce qui élargit considérablement la famille de modèles que nous pouvons utiliser, puisque nous n'avons pas besoin d'architectures spéciales pour rendre la constante de normalisation maniable.

2.1 Apprentissage de la fonction de score et simulation des températures

Le modèle d'apprentissage de $s_\theta(x)$ est un réseau de neurones avec pour seule contrainte que le nombre de neurones d'entrée (6 dans notre cas) doit être égal au nombre de neurones de sortie. On entraîne le modèle en minimisant la divergence de Fisher

$$E_{p(x)}[\|s_\theta(x) - \nabla_x \log p(x)\|_2^2] = \int p(x)\|s_\theta(x) - \nabla_x \log p(x)\|_2^2 dx.$$

Une fois que le modèle de score $s_\theta(x)$ est entraîné, nous utilisons une procédure itérative, la dynamique de Langevin, pour tirer des échantillons x de loi $p_\theta(x)$.

La dynamique de Langevin fournit une procédure de monte-carlo par chaîne de markov (MCMC) pour échantillonner à partir de la distribution $p_\theta(x)$ en utilisant uniquement sa fonction de score $s_\theta(x)$. Plus précisément, on initialise la chaîne à partir d'une distribution à priori arbitraire $x_0 \sim \pi(x)$, puis on itère ce qui suit

$$x_{i+1} = x_i + \epsilon s_\theta(x) + \sqrt{2\epsilon} z_i, \quad i = 0, 1, ..., K,$$

où $z_i \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. Quand $\epsilon \to 0$ et $K \to \infty$, x_K obtenu à partir de la procédure converge vers un échantillon de $p_\theta(x)$ sous certaines conditions de régularité.

2.2 Détails importants sur l'entraînement du modèle de score

Minimiser la divergence de Fisher donnée en 2.1 suppose de connaître la fonction de score de la vraie distribution des données, information qu'on n'a pas. De plus, l'erreur en norme 2 dans l'intégrale est pondérée par la distribution $p(x)$, elle est donc largement ignorée dans les régions à faible densité où $p(x)$ est petit. Lors de l'échantillonnage avec la dynamique de Langevin, notre échantillon initial est dans une région à faible densité car les températures résident dans un espace de grande dimension (6). Par conséquent, avoir un modèle basé sur des scores inexactes fera dérailler la dynamique de Langevin dès le début de la procédure, l'empêchant de générer des échantillons de haute qualité représentatifs des données.

Comme solution, nous perturbons progressivement les données au travers du processus stochastique

$$dx = \sigma^t dw, \quad t \in [0, 1].$$

Notons $p_t(x)$ est la distribution de $x(t)$, $p_0(x)$ la distribution des températures de notre échantillon, et p_1 est la distribution bruitée finale obtenue en fin de processus.

En partant de $p_1(x)$, et en résolvant le processus inverse donné par l'équation

$$dx = -\sigma^{2t}\nabla_x \log p_t(x)dt + \sigma^t dw$$

par une méthode d'Euler-Maruyama, nous échantillonnons suivant la distribution de nos données.

Résoudre le processus inverse nécessite de connaître la fonction de score $\nabla_x \log p_t(x), \forall t$. Pour cela, nous entraînons un modèle de score dépendant du temps $s_\theta(x,t)$ en minimisant la fonction objective

$$\min_\theta E_{t\sim\mathcal{U}(0,1)}[\lambda(t)E_{x(0)\sim p_0(x)}E_{x(t)\sim p_{0t}(x(t)|x(0))}[\|s_\theta(x(t),t) - \nabla_{x(t)} \log p_{0t}(x(t)|x(0))\|_2^2]],$$

où $\mathcal{U}(0,1)$ est une distribution uniforme sur $[0,1]$, $p_{0t}(x(t)|x(0))$ désigne la distribution de $x(t)$ sachant $x(0)$, et $\lambda(t) \in R_+^*$ désigne une fonction de pondération positive.

Dans l'objectif, nous estimons l'espérance sur $x(0)$ avec des moyennes empiriques sur des échantillons de données à partir de p_0, l'espérance sur $x(t)$ en échantillonnant à partir de $p_{0t}(x(t) \mid x(0))$. La fonction de poids $\lambda(t)$ est choisie pour être inversement proportionnelle à

$$E[\|\nabla_x \log p_{0t}(x(t) \mid x(0))\|_2^2].$$

Il n'y a pas de restriction sur l'architecture du réseau du modèle de score dépendant du temps, à part que sa sortie à la même dimension que son entrée et qu'il est conditionné au temps. Nous avons intégré l'information temporelle en échantillonnant d'abord $Z \sim \mathcal{N}(0, s^2 I_6)$ qui est fixé pour le modèle (ne rentre pas dans l'apprentissage), puis pour un pas de temps t, nous évaluons le vecteur aléatoire gaussien

$$[\sin(2\pi Z[0:3]t); \cos(2\pi Z[3:6]t)],$$

de taille 6, concaténation de deux vecteurs de taille 3 (notation python). Ce vecteur sert d'encodage pour le pas de temps et est fourni en entrée au modèle.

2.3 Quelques formules utiles

$$p_{0t}(x(t)|x(0)) = \mathcal{N}\left(x(t); x(0), \frac{1}{2\log\sigma}(\sigma^{2t} - 1)I_6\right)$$

Nous avons choisi la fonction de pondération

$$\lambda(t) = \frac{1}{2\log\sigma}(\sigma^{2t} - 1)$$

.

Lorsque σ est grand, la distribution $p_{t=1}$ est

$$\int p_0(y)\mathcal{N}\left(x; y, \frac{1}{2\log\sigma}(\sigma^2 - 1)I_6\right)dy \approx \mathcal{N}\left(x; 0, \frac{1}{2\log\sigma}(\ sigma^2 - 1)I_6\right).$$

Elle a la particularité d'être facile à échantillonner.

Intuitivement, le processus de perturbation qu'on a choisi capture un continuum de perturbations gaussiennes avec la fonction de variance $\frac{1}{2\log\sigma}(\sigma^{2t} - 1)$. Ce continuum de perturbations nous a permis de transférer progressivement les échantillons de températures de la distribution p_0 vers une distribution gaussienne simple p_1.