Deep Learning II

A. Rappels Monte Carlo

Soit $X_i \in \{0, 1\}$

$X = (X_1, \dots, X_p)$ de dimension p.

$$E(f(X)) = \sum_{x_i} f(x)\, P(X = x)$$

$$= \sum_{x_i} f(x_1, \dots, x_p) \cdot P(X = (x_1, \dots, x_p))$$

$$f : \mathbb{R}^p \longrightarrow \mathbb{R}$$

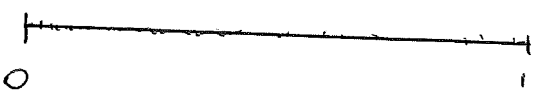
$\Rightarrow$ somme sur 2^p termes

Méthode de MC : $X^{(j)} \sim p(x)$

Alors $$\frac{1}{n} \sum_{j=1}^{n} f(X^{(j)}) \xrightarrow[n \to +\infty]{p.s} E(f(X))$$

(LGN)

Comment obtenir les $x^{(j)}$?



$\rightarrow$ tirage direct impossible.

Échantillonneur de Gibbs

Supposons que l'on connaisse $p(x_i \mid x_1, \dots x_{i-1}, x_{i+1}, \dots x_p)$ $\forall i$

Algorithme :

Initialisation : $x^{(0)} \sim q(x)$ $\left(\text{ex } q(x) = \prod_{i=1}^{p} q(x_i) \right.$

Pour $\ell = 1 : L$ $\left. q(x_i = 1) = 1/2 \right)$

$$x_1^{(\ell)} \sim p\left(x_1 \mid x_2^{(\ell-1)}, \dots x_p^{(\ell-1)}\right)$$

$\vdots$

$$x_i^{(\ell)} \sim p\left(x_i \mid x_1^{(\ell)}, \dots x_{i-1}^{(\ell)}, x_{i+1}^{(\ell-1)}, \dots x_p^{(\ell-1)}\right)$$

$\vdots$

$$x_p^{(\ell)} \sim p\left(x_p \mid x_1^{(\ell)}, \dots x_{p-1}^{(\ell)}\right)$$

fin

$$\Rightarrow \quad \frac{1}{L} \sum_{i=1}^{L} f\left(x^{(i)}\right) \xrightarrow[n \to +\infty]{p.s.\,?} E(f(X))$$

B) Restricted Boltzmann Machines

Soient $V_i \in \{0,1\} \quad 1 \leq i \leq p$

$H_j \in \{0,1\} \quad 1 \leq j \leq q$

Déf

$$p(v,h) = \frac{1}{Z} e^{-E(v,h)}$$

où $E(v,h) = -\sum_{i=1}^{p} a_i v_i - \sum_{j=1}^{q} b_j h_j - \sum_{i,j} w_{ij} v_i h_j$

. $Z = \sum_{v,h} e^{-E(v,h)}$

$$p(v) = \sum_{h} p(v,h)$$

Un RBM est caractérisé par :

$\Theta = (a_i, b_j, w_{ij})$ et par q

Propriétés :

. $p(h|v) = \dfrac{p(v,h)}{p(v)} = \dfrac{p(v,h)}{\sum_h p(v,h)}$

$$= \frac{e^{-E(v,h)}}{\sum_h e^{-E(v,h)}}$$

$$= \frac{\cancel{e^{+\sum_i a_i v_i}} \cdot e^{\sum_j b_j h_j + \sum_{i,j} w_{ij} v_i h_j}}{\cancel{e^{\sum_i a_i v_i}} \sum_h e^{\sum_j b_j h_j + \sum_{i,j} w_{ij} v_i h_j}}$$

$$= \frac{\prod_j e^{b_j h_j + \sum_i w_{ij} v_i h_j}}{\sum_h \prod_j e^{b_j h_j + \sum_i w_{ij} v_i h_j}}$$

$$= \frac{\prod_j e^{b_j h_j + \sum_i w_{ij} v_i h_j}}{\prod_j \sum_{h_j} e^{b_j h_j + \sum_i w_{ij} v_i h_j}}$$

$$= \prod_{j=1}^{q} \frac{e^{b_j h_j + \sum_i w_{ij} v_i h_j}}{1 + e^{b_j + \sum_i w_{ij} v_i}}$$

$$= \prod_{j=1}^{q} g(h_j)$$

$$\text{où } g(h_j = 1) = \frac{1}{1 + e^{-(b_j + \sum w_{ij} v_i)}}$$

$$= \text{sigm}\left(b_j + \sum_i w_{ij} v_i\right)$$

⑤

$\Rightarrow \quad p(h|v) = \prod_{j=1}^{q} p(h_j | v)$

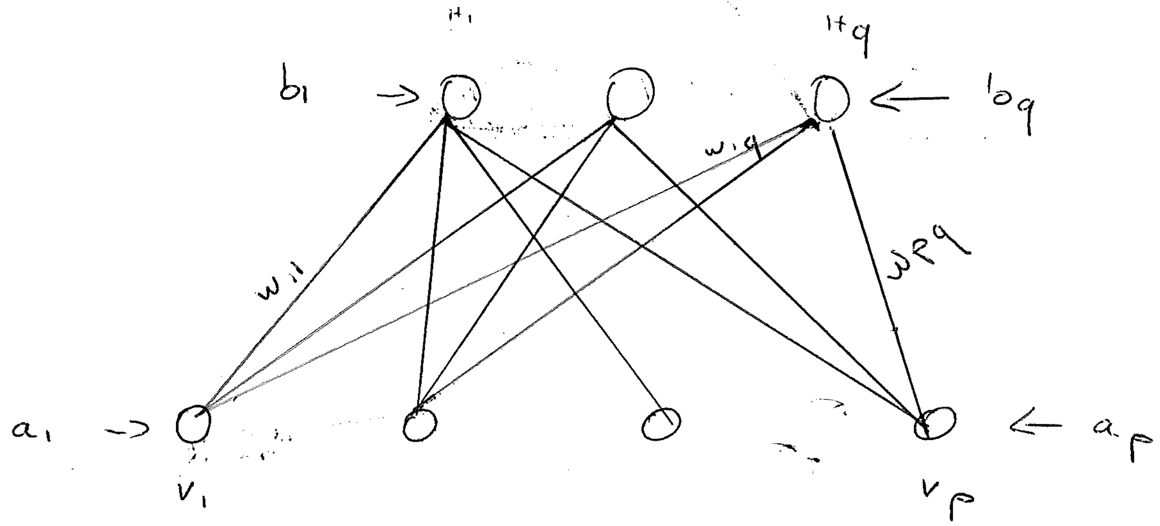
où $p(h_j = 1 | v) = \mathrm{sign}(b_j + \sum_i w_{ij} v_i)$

$\Rightarrow$ Cond à $V = v$, H sont independants.

De même $p(v|h) = \prod_{i=1}^{p} p(v_i | h)$

où $p(v_i = 1 | h) = \mathrm{sign}(a_i + \sum_j w_{ij} h_j)$

$\Rightarrow$ Représentation graphique d'un RBM :



$\Rightarrow$ 1 réseau de neurones probabiste à 1 couche

Applications

[LOTR, Retour vers le futur, Bridget Jones, Titanic, Star Wars, Twilight]

$U_1 = [1\ 1\ 0\ 0, 1, 0] \rightarrow$

$U_2 = [0\ 0\ 1\ 1, 0, 1]$

$U_3 = [0\ 0\ 1\ 0, 0, 1]$

$U_4 : [0\ 0\ 0\ 0\ 0\ 0]$

$U_5 = [1, 1, 1, 1, 1, 1]$

Sentimental, cinéphile, Cinéphobe, Geek

- Réduction de dimension

- Générer des datas.

Tirer $p(v)$?

On a $\begin{cases} p(v \mid h) \rightarrow \text{facile de simuler} \\ p(h \mid v) \rightarrow \text{———} \end{cases}$

échantillonneur de Gibbs.

<u>Pourquoi ce modèle ?</u>

Thm d'universalité : $\sum_v p(v) \log \frac{p(v)}{p_\theta(v)} \geq 0$

$$D_{KL}(\underset{\substack{\uparrow \\ \text{vrai loi}}}{p(v)}, \underset{\substack{\uparrow \\ \text{loi du RBM}}}{p_\theta(v)}) \leq \varepsilon$$

C. Apprentissage d'un RBM.

Estimer θ à partir de n réalisations

$$x^{(1)}, \dots, x^{(n)} \overset{iid}{\sim} p(v)$$

$$x^{(i)} = (x_1^{(i)}, \dots, x_p^{(i)})$$

$\Rightarrow$ Estimateur du MV

$$\mathcal{L}(x^{(1)} \dots x^{(n)}, \theta) = \prod_{i=1}^{n} \mathcal{L}(x^{(i)}, \theta)$$

AVEC

$$\mathcal{L}(x; \theta) = p_\theta(x)$$

$$= p_\theta(v = x)$$

$$= \sum_h p_\theta(x, h)$$

Maximisons

$$\log \mathcal{L} = \sum_{i=1}^{n} \log \mathcal{L}(x^{(i)}, \theta)$$

$$\nabla_\theta \log \mathcal{L} = \sum_{i=1}^{n} \nabla_\theta \log \mathcal{L}(x^{(i)}, \theta)$$

Calculons $\nabla_{w_{ij}} \log \mathcal{L}(x; \theta)$

$$\nabla_{w_{ij}} \log \sum_h p_\theta(x, h)$$

- $$\frac{\partial}{\partial w_{ij}} \log \sum_h e^{-E_\theta(x,h)} - \log Z_\theta$$

$$= \frac{\partial}{\partial w_{ij}} \log \sum_h e^{-E(x,h)} - \log \sum_{v,h} e^{-E(v,h)}$$

$$= \frac{\sum_h \frac{d}{dw_{ij}} e^{-E(x,h)}}{\sum_h e^{-E(x,h)}} - \frac{\sum_{v,h} \frac{d}{dw_{ij}} e^{-E(v,h)}}{\sum_{v,h} e^{-E(v,h)}}$$

$$= \frac{\sum_h x_i h_j \, e^{-E(x,h)}}{\underbrace{\sum_h e^{-E(x,h)}}_{=}} - \sum_{v,h} v_i h_j \underbrace{\frac{e^{-E(v,h)}}{\sum_{v,h} e^{-E(v,h)}}}_{= p(v,h)}$$

$$= \sum_h x_i h_j \, p(h|x) - \sum_{v,h} v_i h_j \, p(v,h)$$

$$= \sum_{h_j} x_i h_j \, p(h_j|x) - \sum_{v,h} v_i h_j \, p(v,h)$$

$$= x_i \times p(h_j = 1 | x) - E(f_{ij}(V,H))$$

où $f_{ij}(V,H) = V_i H_j$

$$E(V_i H_j) = \cancel{E(E(H_j V_i | H_j))}$$

$$= E(E(V_i H_j | V))$$

$$= \sum_V v_i \left[\sum_{h_j} h_j \, p(h_j | v) \right] p(v)$$

$$= E(V_i \cdot p(h_j = 1 | V))$$

Approchons $E(V_i \, p(h_j = 1 | V))$ via Gibbs :

$v^{(0)} = x$

$h^{(0)} \sim p(h \mid v^{(0)})$

$v^{(1)} \sim p(v \mid h^{(0)})$

$\Rightarrow \mathbb{E}(v_i H_j) \approx v_i^{(1)} \cdot p(h_j = 1 \mid v^{(1)})$

On a donc $\boxed{\dfrac{\partial}{\partial w_{ij}} \log \mathcal{L}(x;\theta) \approx x_i\, p(h_j = 1 \mid x) - v_i^{(1)} \cdot p(h_j = 1 \mid v^{(1)})}$

De même
$$\frac{\partial}{\partial a_i} \log \mathcal{L}(x,\theta) \approx x_i - v_i^{(1)}$$
$$\frac{\partial}{\partial b_j} \log \mathcal{L}(x;\theta) \approx p(h_j = 1 \mid x) - p(h_j = 1 \mid v^{(1)})$$

Algorithme contrastive . Divergence - 1

Input: . Pour une donnée x

- $\underset{p \times q}{W}$, $\underset{1 \times p}{a}$, $\underset{1 \times q}{b}$

$v^{(0)} = p(x)$

$h^{(0)} \sim p(h \mid v^{(0)})$

$v^{(1)} \sim p(v \mid h^{(0)})$

$\nabla_{w_{ij}} \log \mathcal{L} = \cancel{x_i}^{(0)} p(h_j = 1 \mid \cancel{x}^{(0)}) - v_i^{(1)} p(h_j = 1 \mid v^{(1)})$

$\nabla_{a_i} \log \mathcal{L} = v_i^{(0)} - v_i^{(1)}$

$\nabla_{b_j} \log \mathcal{L} = p(h_j = 1 \mid v^{(0)}) - p(h_j = 1 \mid v^{(1)})$

$$w_{ij} \leftarrow w_{ij} + \varepsilon \nabla_{w_{ij}} \ell\mathcal{L}$$

$$a_i \leftarrow a_i + \varepsilon \nabla_{a_i} \ell\mathcal{L}$$

$$b_j \leftarrow b_j + \varepsilon \nabla_{b_j} \ell\mathcal{L}$$

- Evaluer l'erreur de reconstruction
  - $x \rightarrow p(h|x) \rightarrow p(x \mid p(h|x))$

    $x \rightarrow h \rightarrow x'$

    $(x' - x)^2 \rightarrow$ diminue au cours des itérations.

En pratique

- $\text{DATA} = x = n \updownarrow \underbrace{\begin{pmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \end{pmatrix}}_{p}$

- RBM.
  - RBM.W $\rightarrow$ matrice $p \times q$
  - RBM.a $\rightarrow$ vecteur $1 \times p$
  - RBM.b $\rightarrow$ $1 \times q$

- Entree_sortie $(\underbrace{\text{data}}_{m \times p}, \text{RBM}) \rightarrow$ doit ressortir data_h $\rightarrow m \times q$

- sortie_entree $(\underbrace{\text{data\_h}}_{m \times q}, \text{RBM}) \rightarrow$ data_v $\rightarrow m \times p$

- Init_RBM (p, q)

- train_RBM (x, RBM, epochs, learning_rate, batch_size)

  -> doit retourné RBM entrainé

Écrire le pseudo code de gradient par mini-batch pour entrainer le RBM via CD-1. Boucle for réservé aux epochs et aux batchs.