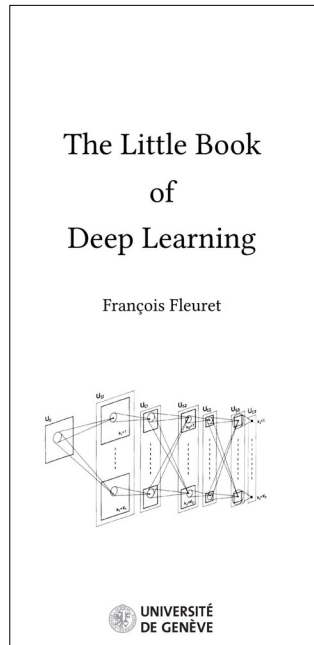


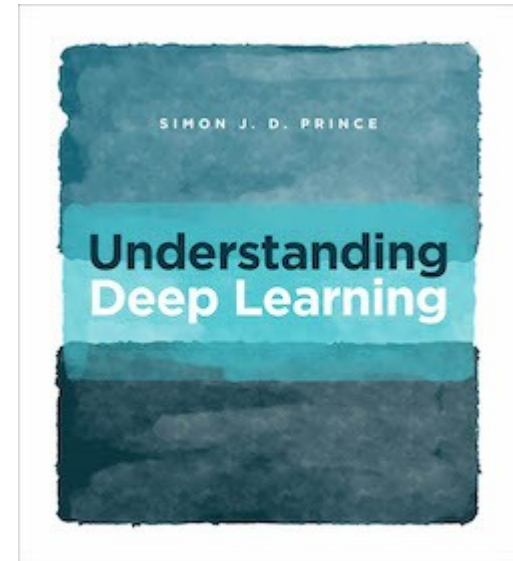
Introduction aux réseaux de neurones pour l'apprentissage supervisé

Guillaume Bourmaud

Livres



<https://fleuret.org/francois/lbdl.html>



<https://udlbook.github.io/udlbook/>

PLAN

I. Introduction

II. Apprentissage supervisé

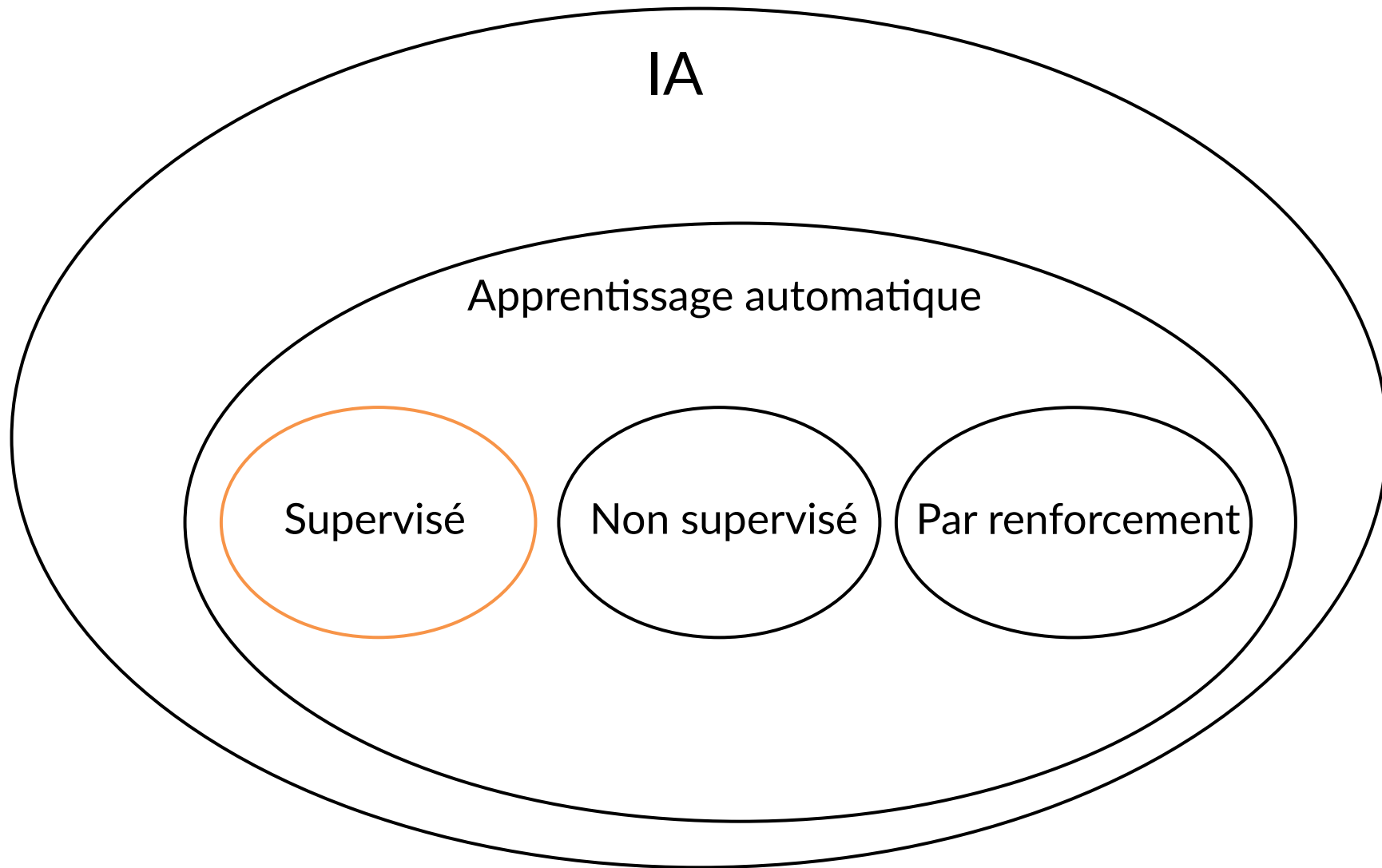
III. Approches paramétriques

IV. Réseaux de neurones

V. Risques

I) Introduction

1)



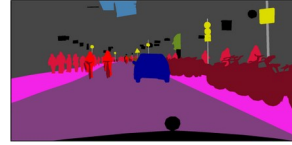
II) Apprentissage supervisé

II)

Apprentissage supervisé

donnée 1

$X_{\text{train},1}$

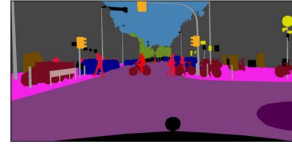


$Y_{\text{train},1}$

étiquette 1

donnée 2

$X_{\text{train},2}$



$Y_{\text{train},2}$

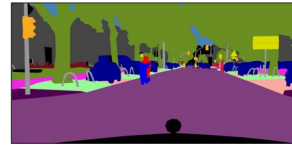
étiquette 2

⋮

⋮

donnée N

$X_{\text{train},N}$

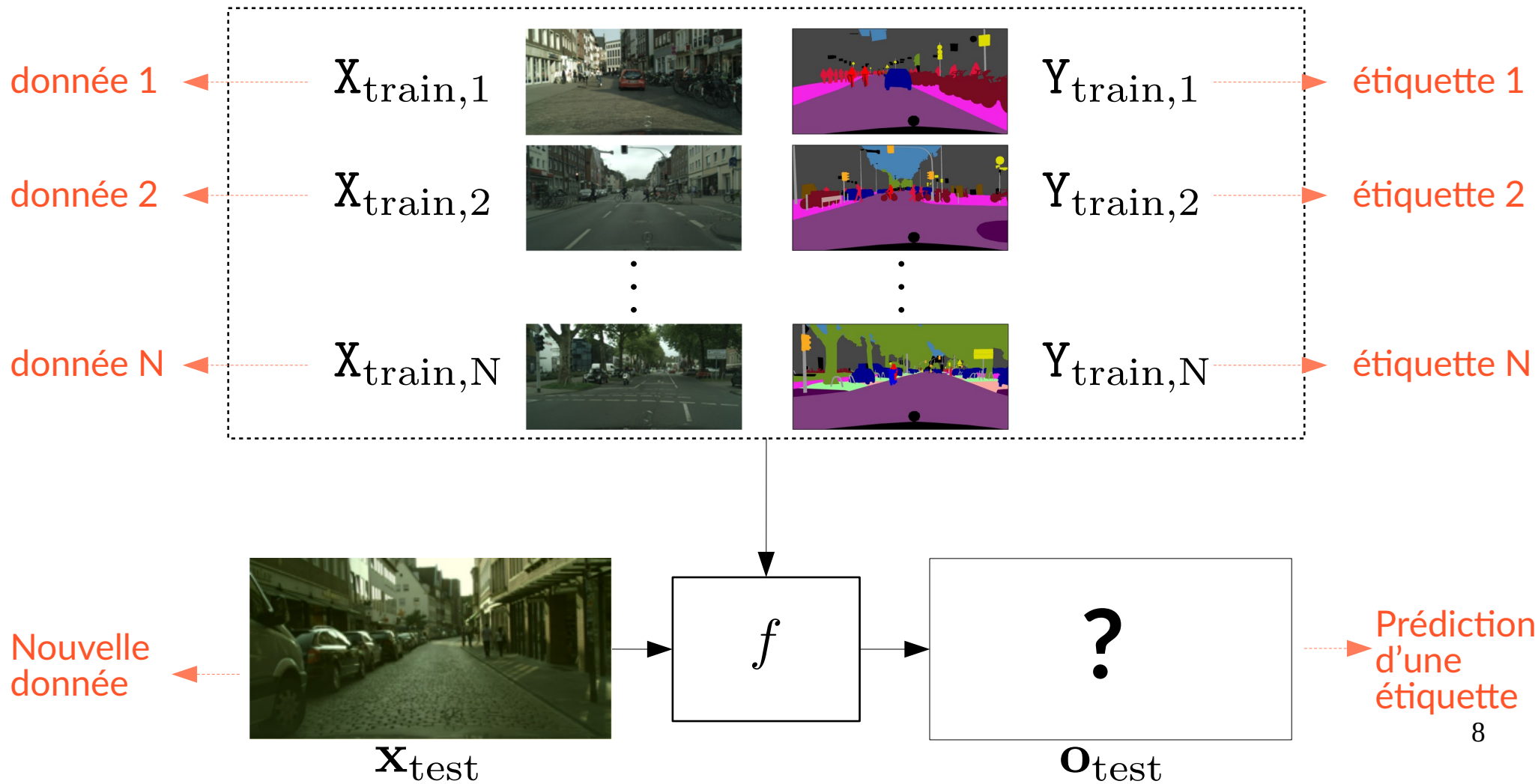


$Y_{\text{train},N}$

étiquette N

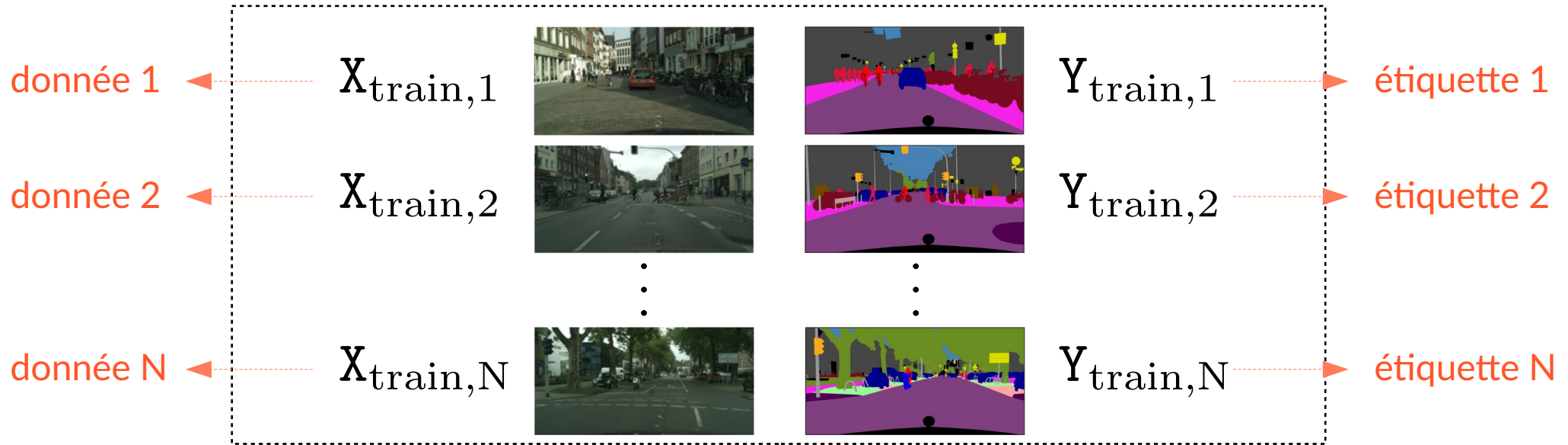
II)

Apprentissage supervisé

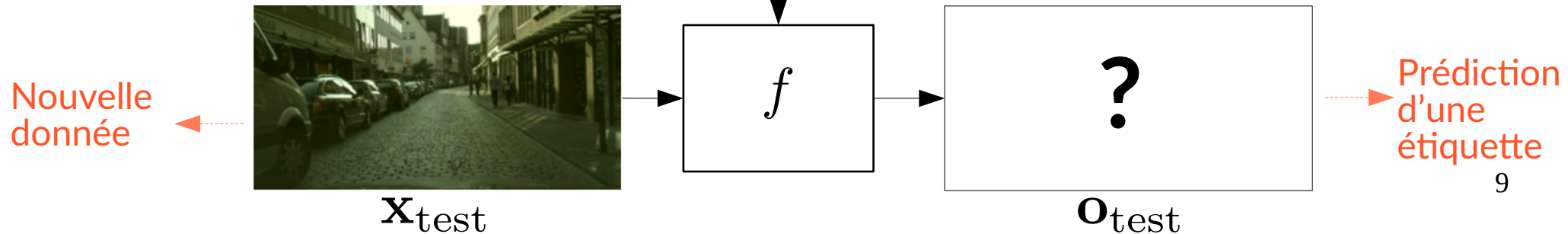


II)

Apprentissage supervisé

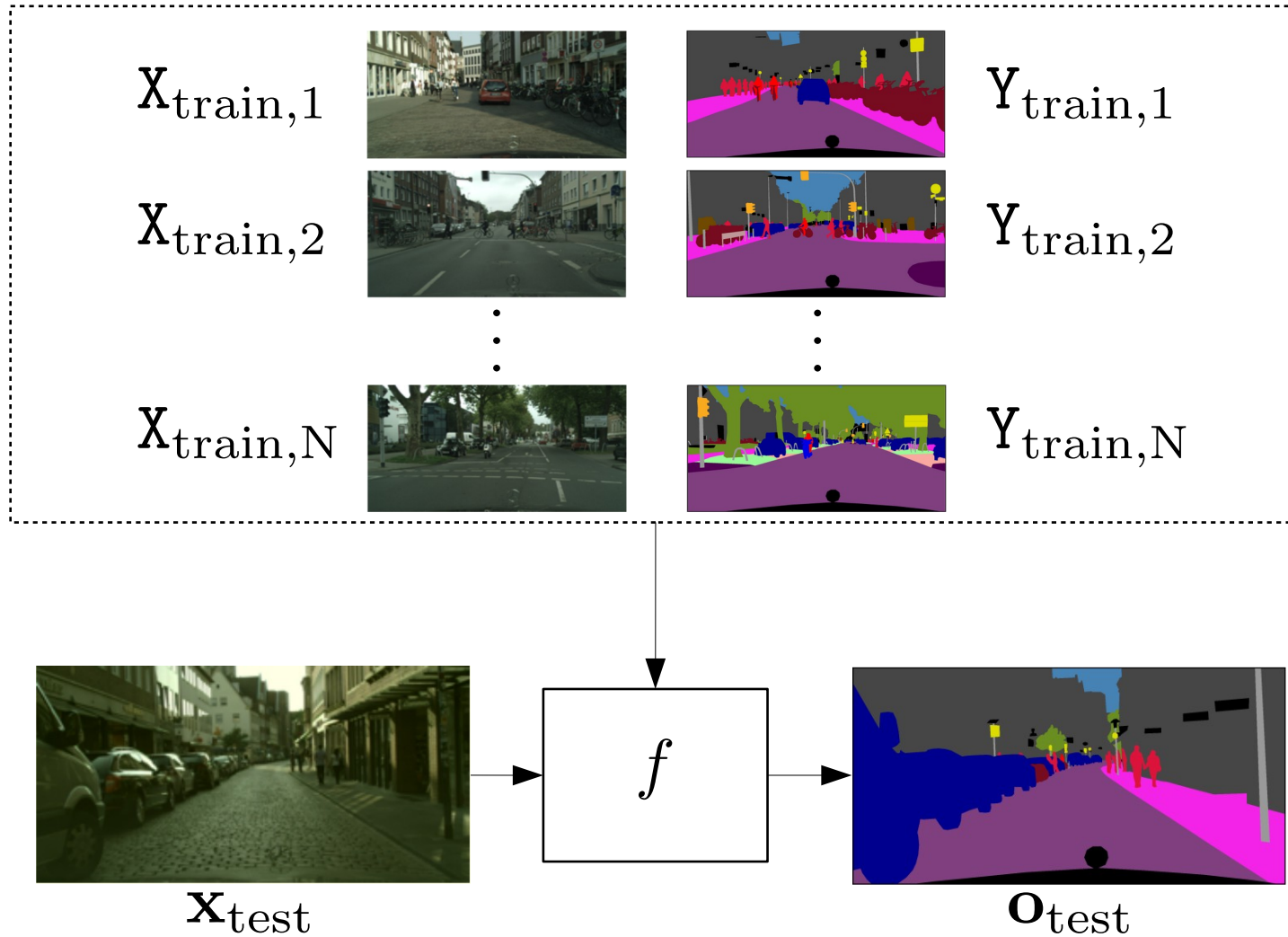


!!! À SAVOIR PAR CŒUR !!!



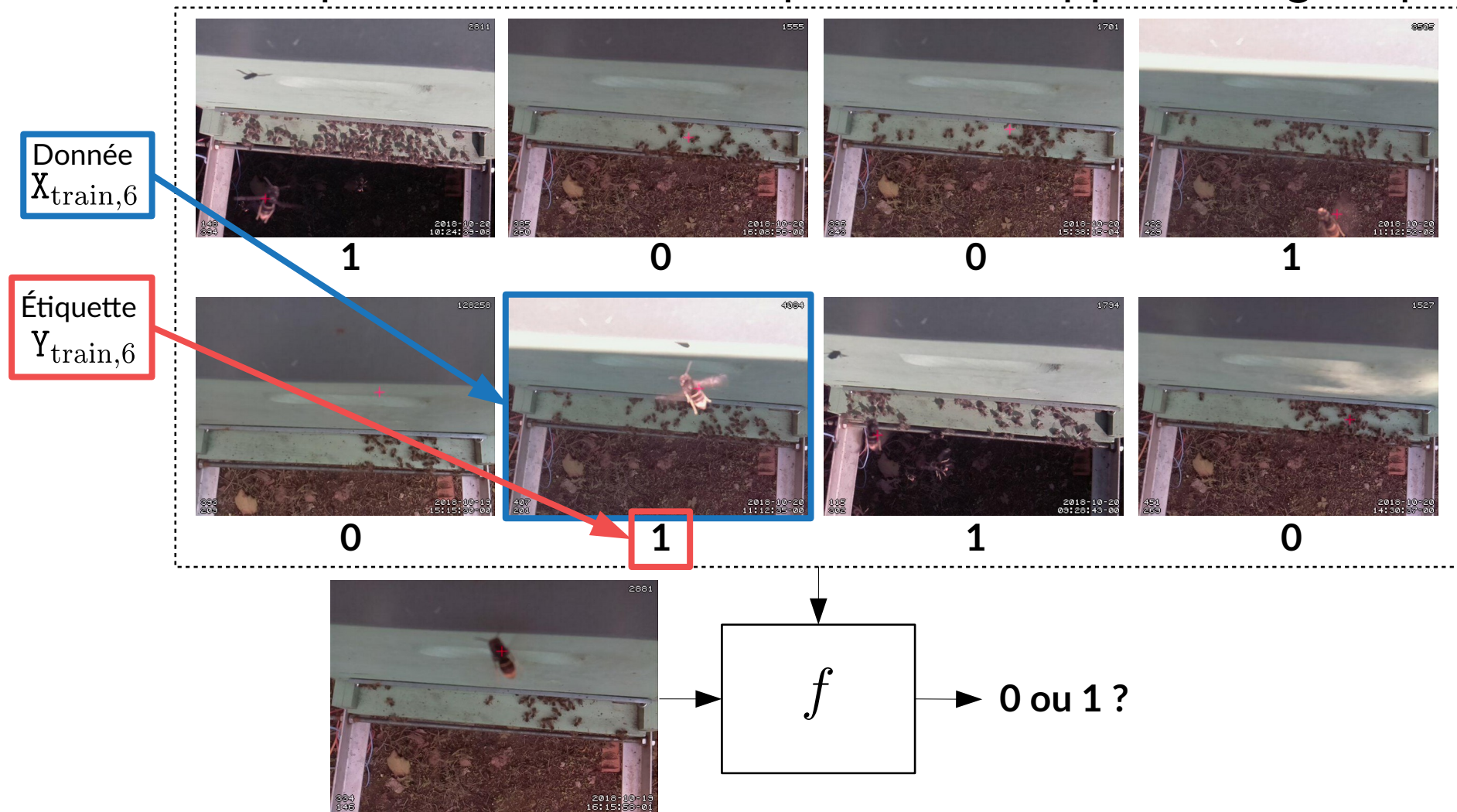
II)

Apprentissage supervisé



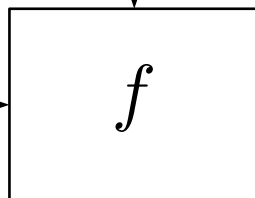
II)

Êtes-vous capable de résoudre ce problème d'apprentissage supervisé ?



II)

Mais de quel problème parle-t-on au juste ?



0 ou 1 ?

II)

Nous souhaitons que l'ordinateur apprenne à résoudre le problème suivant

Donnée
 $X_{\text{train},6}$



Présence d'au moins un frelon



Absence de frelon



Absence de frelon



Présence d'au moins un frelon



Absence de frelon



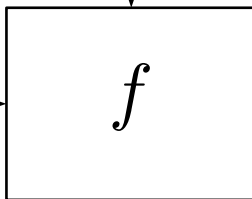
Présence d'au moins un frelon



Présence d'au moins un frelon



Absence de frelon



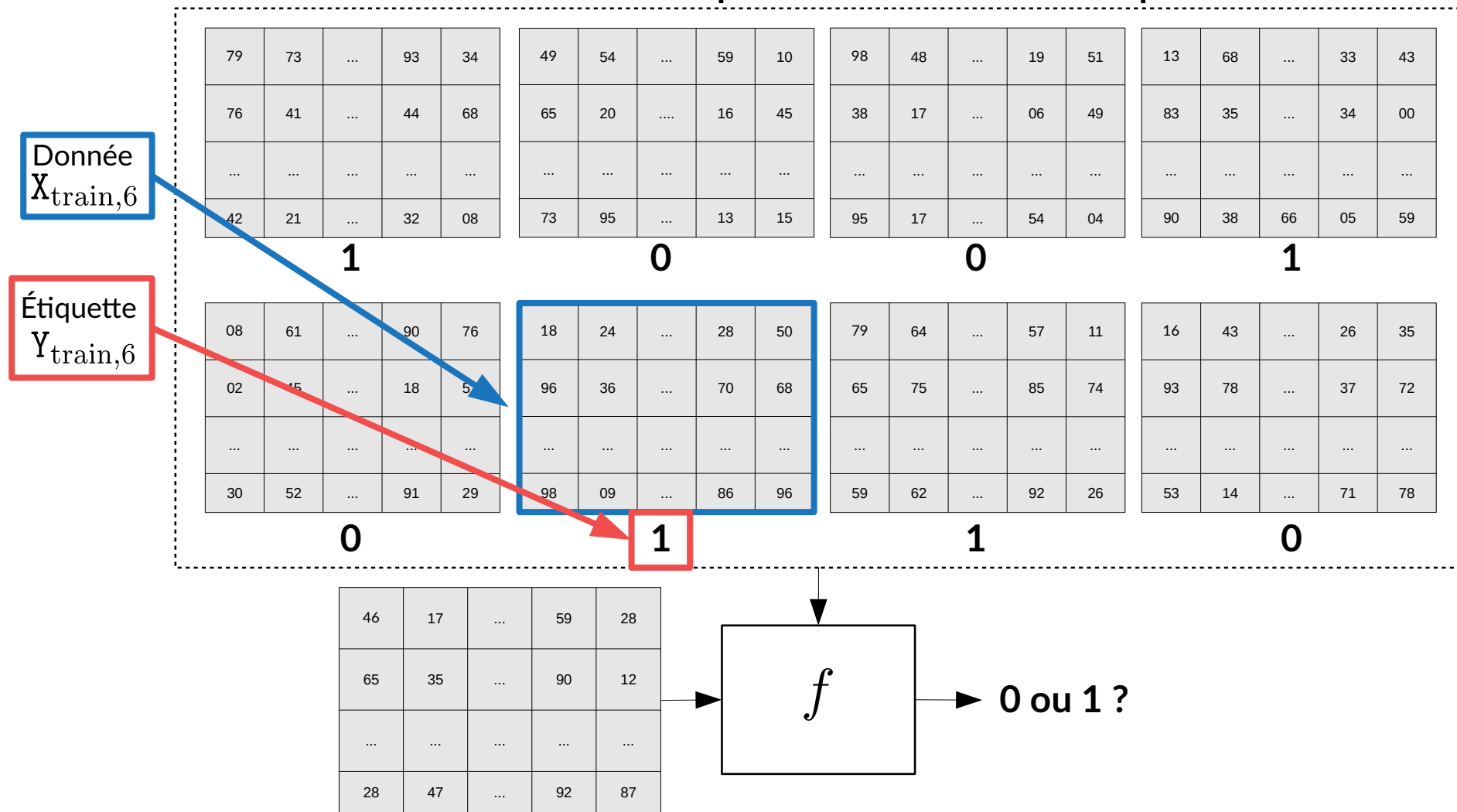
Présence d'au moins un frelon ?

ou

Absence de frelon ?

II)

Mais une fois mis en forme pour l'ordinateur le problème devient



II)

Résumé

- L'ordinateur apprend une fonction qui effectue des calculs sur des tableaux de valeurs numériques et produit une valeur numérique en sortie.
- Nous interprétons cette valeur numérique en sortie comme une réponse à la question : « Y a-t-il (au moins) un frelon présent dans l'image ? »
- Il ne s'agit que d'une interprétation...



Tesla said autopilot was activated during a fatal Model X crash last week in California.

III) Approches paramétriques

III)

Fonction paramétrique

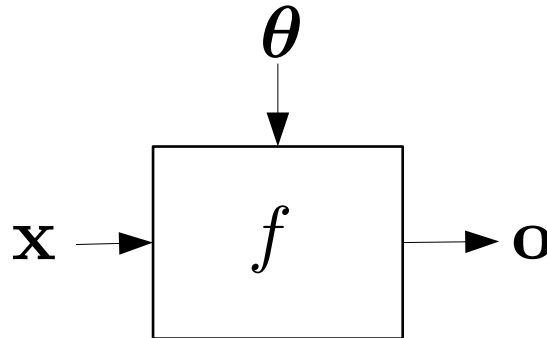
Mathématique

$$\mathbf{o} = f(\mathbf{x}; \boldsymbol{\theta})$$

Informatique
(Python)

```
def f(x, theta):  
    ...  
    ...  
    o = ...  
    return o
```

Graphique
(Graphe de calcul)



III) Exemple de fonction paramétrique : transformation affine

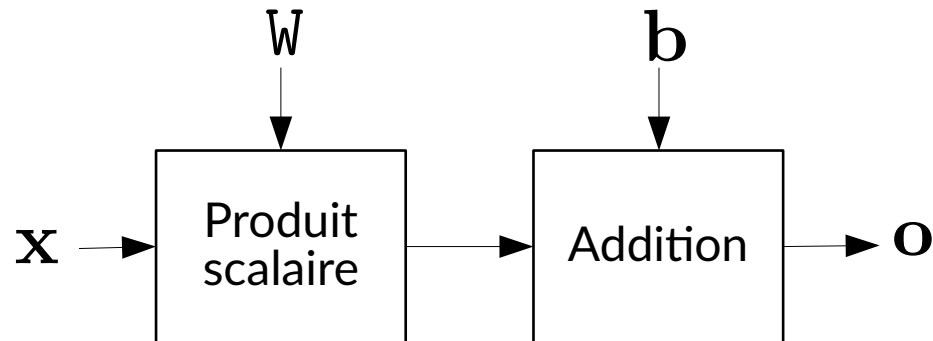
Mathématique

$$\mathbf{o} = f(\mathbf{x}; \boldsymbol{\theta} = \{W, \mathbf{b}\}) = W\mathbf{x} + \mathbf{b}$$

Informatique
(Python)

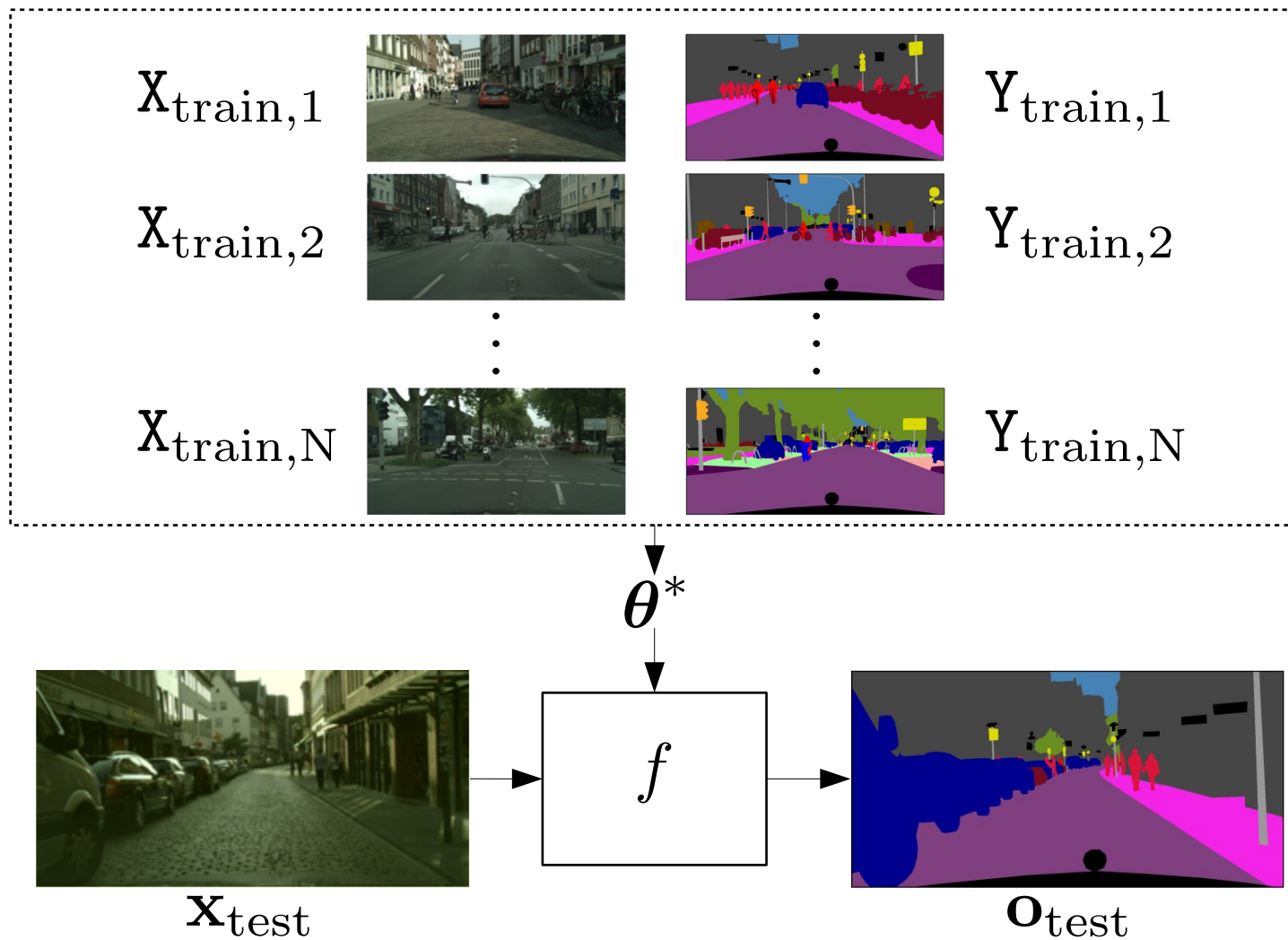
```
def f(x, W, b):  
    o = x.dot(W) + b  
    return o
```

Graphique
(Graphe de calcul)



III)

Approches paramétriques

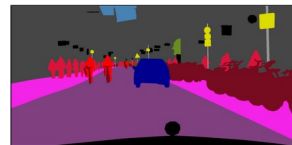


III)

Approches paramétriques

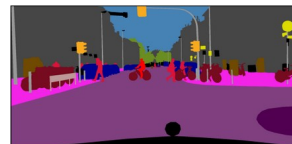
Apprentissage

$X_{\text{train},1}$



$Y_{\text{train},1}$

$X_{\text{train},2}$



$Y_{\text{train},2}$

\vdots

\vdots

$X_{\text{train},N}$

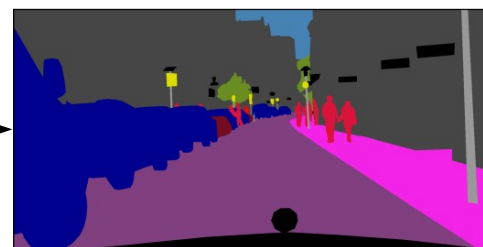
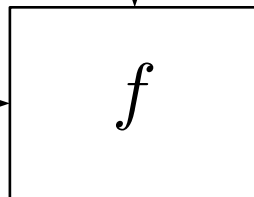


$Y_{\text{train},N}$

θ^*



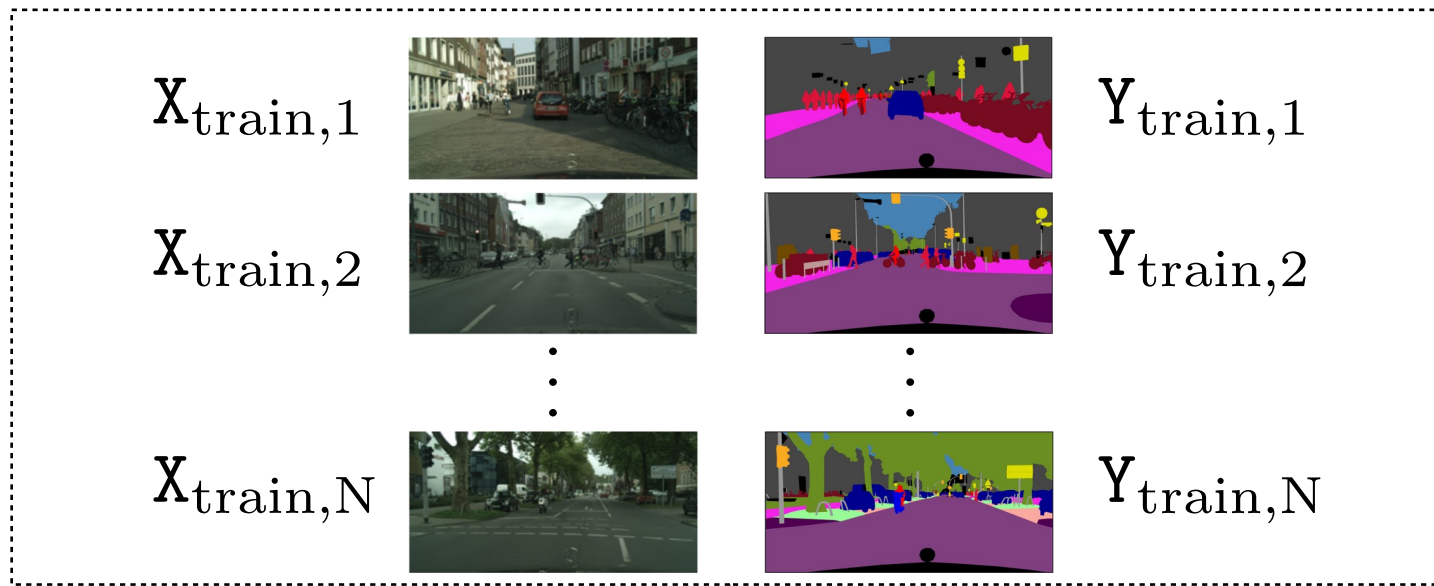
X_{test}



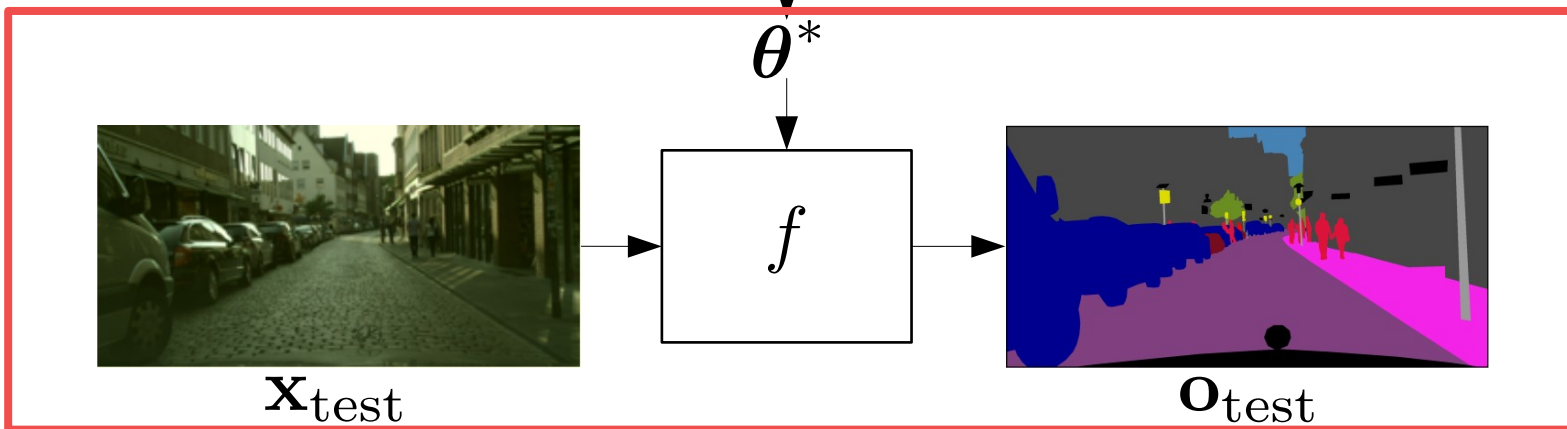
O_{test}

III)

Approches paramétriques

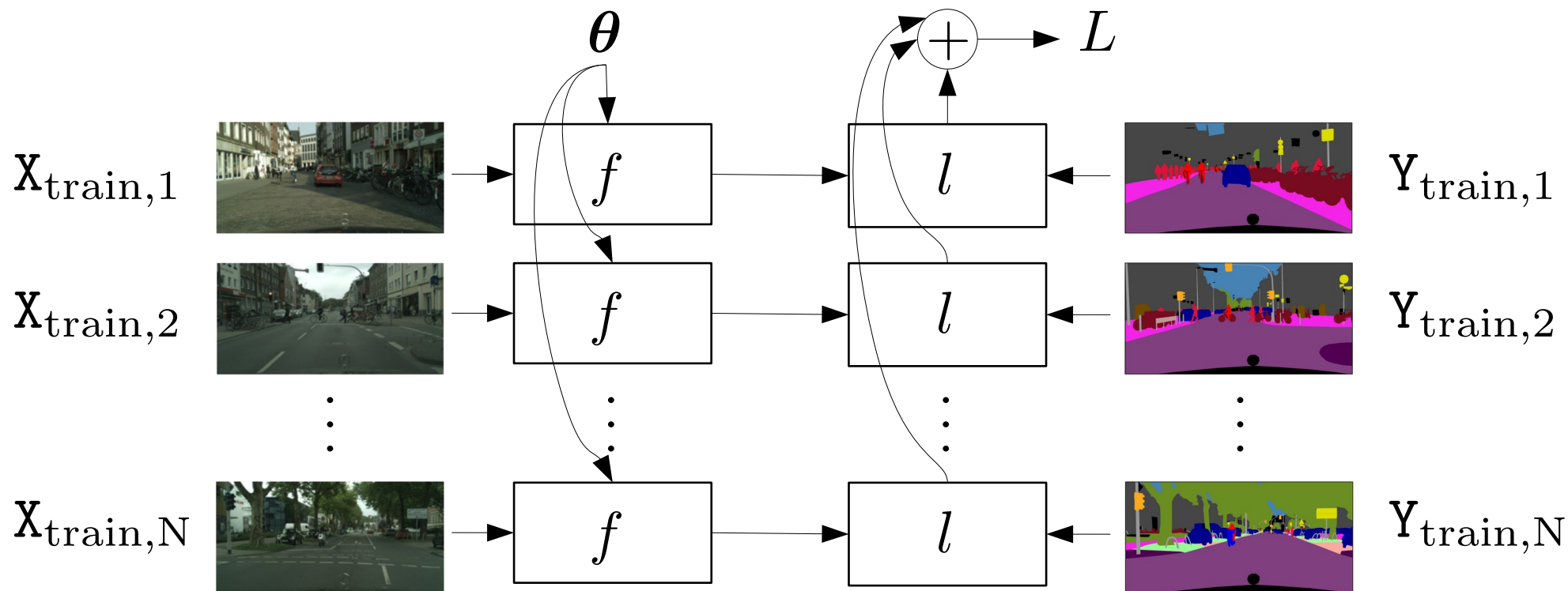


Inférence



III)

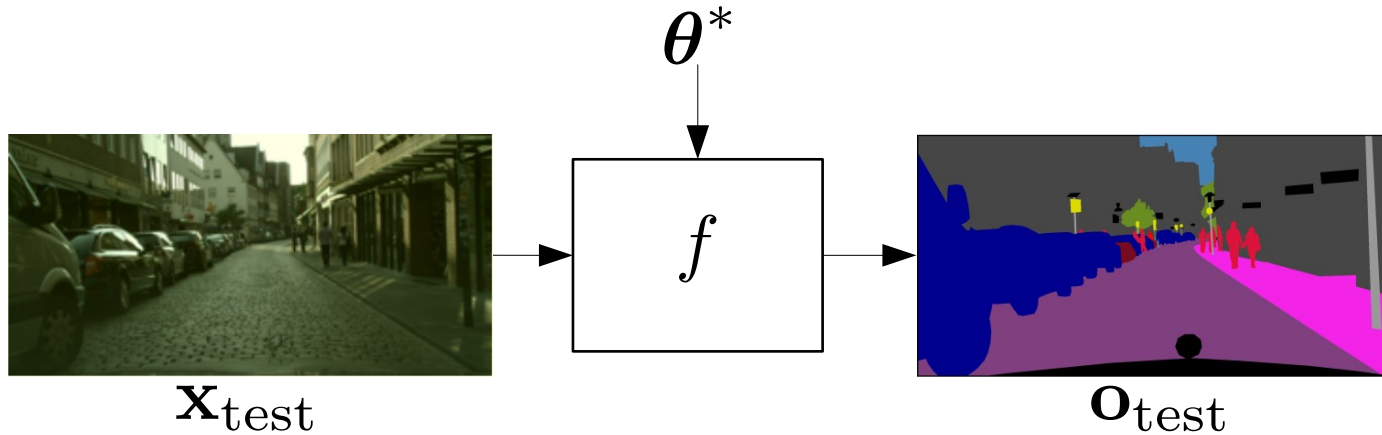
Étape d'apprentissage ("Training time")



$$\theta^* = \arg \min_{\theta} \sum_{i=1}^N l(Y_{\text{train},i}, f(X_{\text{train},i}; \theta))$$

III)

Étape d'inférence ("Test time")

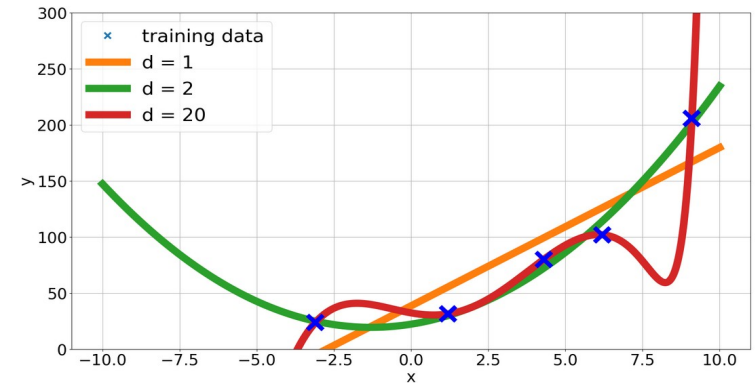


$$\mathbf{o}_{\text{test}} = f(\mathbf{x}_{\text{test}}; \theta^*)$$

III)

Exemple de régression : 5 données, $x \in \mathbb{R}$ et $y \in \mathbb{R}$

$$\begin{array}{lllll} X_{\text{train},1} = -3.1 & X_{\text{train},2} = 1.2 & X_{\text{train},3} = 4.3 & X_{\text{train},4} = 6.2 & X_{\text{train},5} = 9.1 \\ Y_{\text{train},1} = 23.7 & Y_{\text{train},2} = 31.3 & Y_{\text{train},3} = 79.9 & Y_{\text{train},4} = 101.9 & Y_{\text{train},5} = 205.5 \end{array}$$



III)

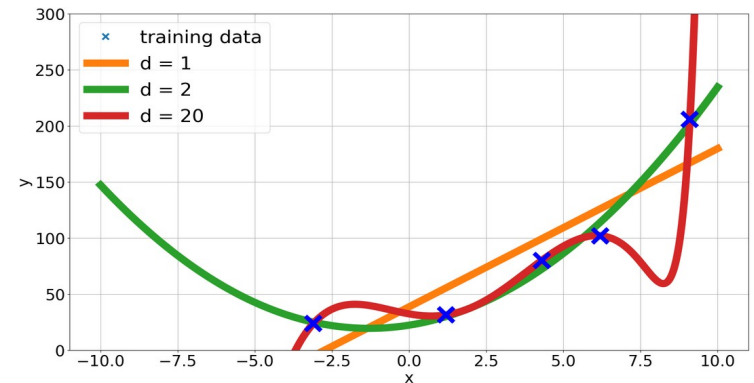
Exemple de régression : 5 données, $x \in \mathbb{R}$ et $y \in \mathbb{R}$

$$\begin{array}{ccccc} X_{\text{train},1} = -3.1 & X_{\text{train},2} = 1.2 & X_{\text{train},3} = 4.3 & X_{\text{train},4} = 6.2 & X_{\text{train},5} = 9.1 \\ Y_{\text{train},1} = 23.7 & Y_{\text{train},2} = 31.3 & Y_{\text{train},3} = 79.9 & Y_{\text{train},4} = 101.9 & Y_{\text{train},5} = 205.5 \end{array}$$

Choix de la fonction $f(x; \theta) = \theta^\top \phi(x)$

$$\text{où } \phi(x) = [1 \quad x \quad \dots \quad x^d]^\top$$

hyper-paramètre : d



III)

Exemple de régression : 5 données, $x \in \mathbb{R}$ et $y \in \mathbb{R}$

$$\begin{array}{ccccc} X_{\text{train},1} = -3.1 & X_{\text{train},2} = 1.2 & X_{\text{train},3} = 4.3 & X_{\text{train},4} = 6.2 & X_{\text{train},5} = 9.1 \\ Y_{\text{train},1} = 23.7 & Y_{\text{train},2} = 31.3 & Y_{\text{train},3} = 79.9 & Y_{\text{train},4} = 101.9 & Y_{\text{train},5} = 205.5 \end{array}$$

Choix de la fonction $f(x; \theta) = \theta^\top \phi(x)$

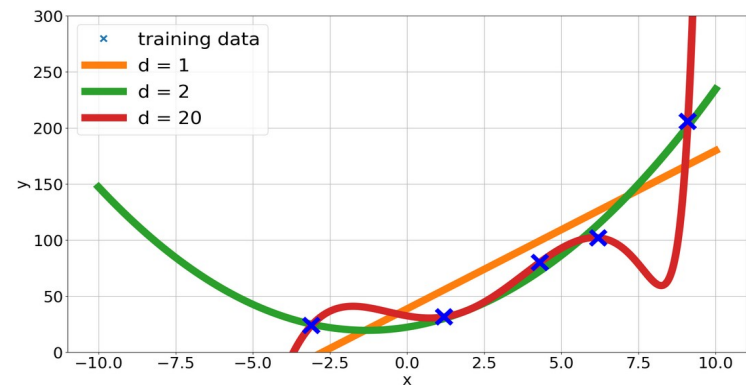
$$\text{où } \phi(x) = [1 \quad x \quad \dots \quad x^d]^\top$$

hyper-paramètre : d

Apprentissage

Choix du coût $l(y, o) = (y - o)^2$

Optimisation
$$\theta^* = \arg \min_{\theta} \sum_{i=1}^5 \left(Y_{\text{train},i} - \theta^\top \phi(X_{\text{train},i}) \right)^2$$



Exemple de régression : 5 données, $x \in \mathbb{R}$ et $y \in \mathbb{R}$

$$\begin{array}{ccccc} X_{\text{train},1} = -3.1 & X_{\text{train},2} = 1.2 & X_{\text{train},3} = 4.3 & X_{\text{train},4} = 6.2 & X_{\text{train},5} = 9.1 \\ Y_{\text{train},1} = 23.7 & Y_{\text{train},2} = 31.3 & Y_{\text{train},3} = 79.9 & Y_{\text{train},4} = 101.9 & Y_{\text{train},5} = 205.5 \end{array}$$

Choix de la fonction $f(x; \theta) = \theta^\top \phi(x)$

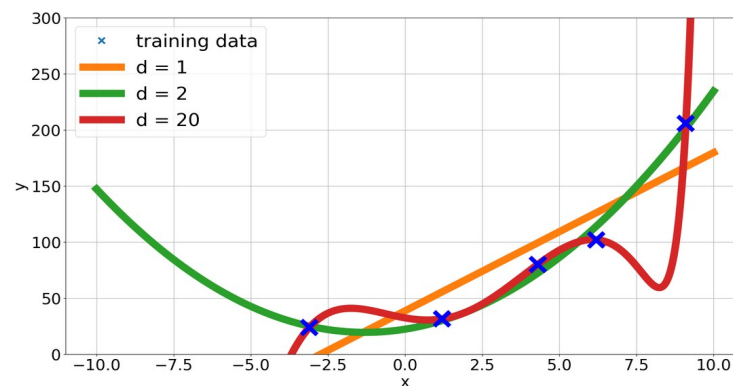
où $\phi(x) = [1 \quad x \quad \dots \quad x^d]^\top$

hyper-paramètre : d

Apprentissage

Choix du coût $l(y, o) = (y - o)^2$

Optimisation $\theta^* = \arg \min_{\theta} \sum_{i=1}^5 \left(Y_{\text{train},i} - \theta^\top \phi(X_{\text{train},i}) \right)^2$

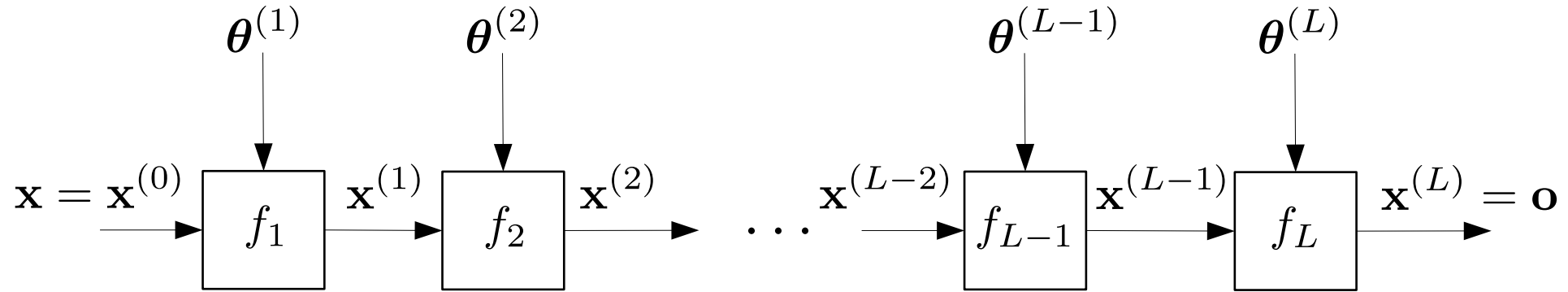


Inférence

$$o_{\text{test}} = \theta^{*\top} \phi(x_{\text{test}}) = \theta_0^* + \theta_1^* x_{\text{test}} + \dots + \theta_d^* x_{\text{test}}^d$$

IV) Réseaux de neurones

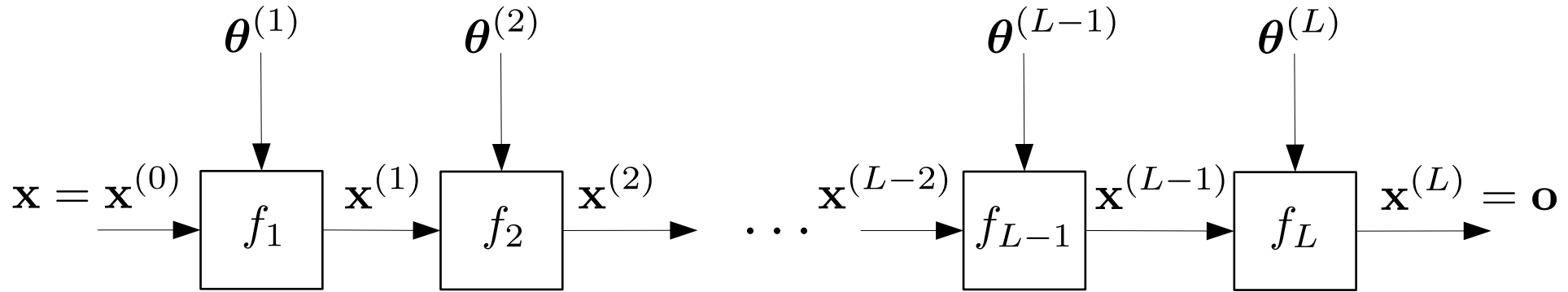
Choix de la fonction : Réseau de neurones



$$f(\mathbf{x}; \boldsymbol{\theta}) = f_L(f_{L-1}(\dots f_2(f_1(\mathbf{x}; \boldsymbol{\theta}_1); \boldsymbol{\theta}_2) \dots; \boldsymbol{\theta}_{L-1}); \boldsymbol{\theta}_L)$$

Réseau de neurones = Composition de fonctions paramétriques

Choix de la fonction : Réseau de neurones



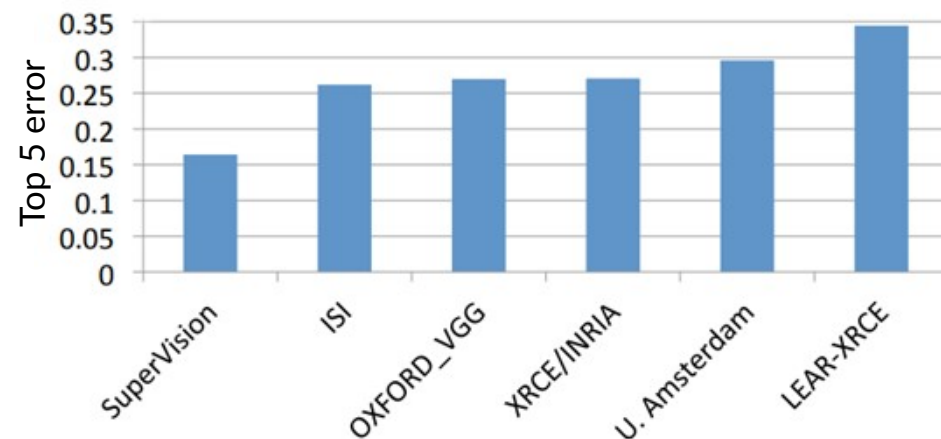
$$f(\mathbf{x}; \boldsymbol{\theta}) = f_L(f_{L-1}(\dots f_2(f_1(\mathbf{x}; \boldsymbol{\theta}_1); \boldsymbol{\theta}_2) \dots; \boldsymbol{\theta}_{L-1}); \boldsymbol{\theta}_L)$$

Réseau de neurones = Composition de fonctions paramétriques

Profondeur du réseau = nombre de couches

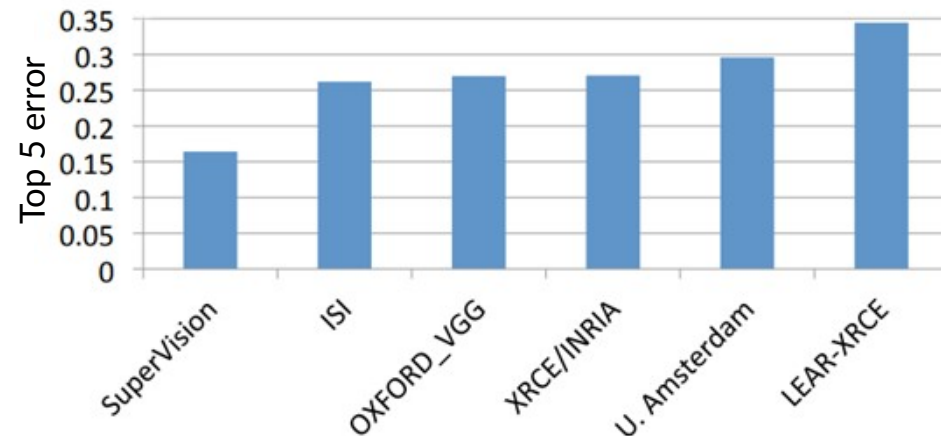
Réseau profond = « Deep Learning »

ImageNet Large Scale Visual Recognition Challenge 2012



- ImageNet : 1.2 millions d'images annotées, 1000 classes

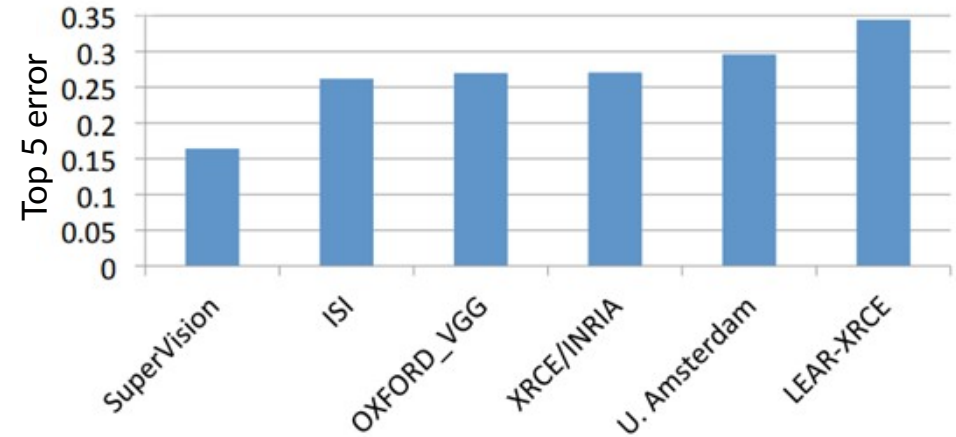
ImageNet Large Scale Visual Recognition Challenge 2012



- ImageNet : 1.2 millions d'images annotées, 1000 classes

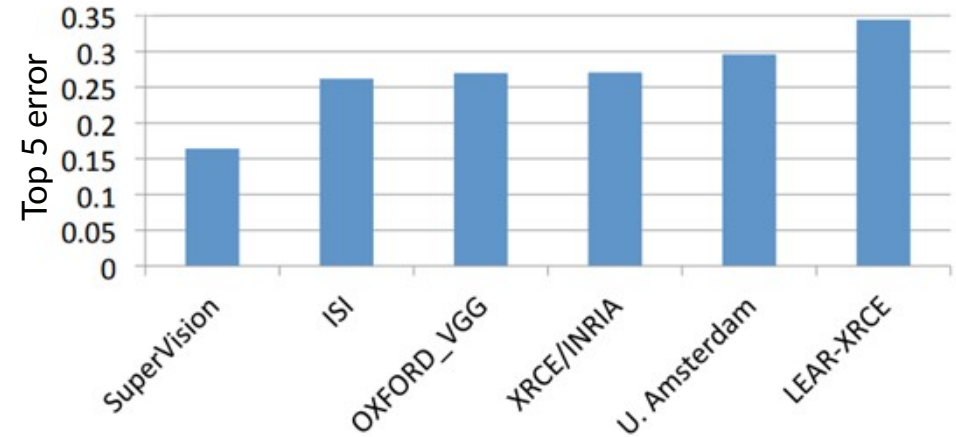
Ingrédient 1 : grande base de données étiquetées

ImageNet Large Scale Visual Recognition Challenge 2012



- ImageNet : 1.2 millions d'images annotées, 1000 classes
- SuperVision (A. Krizhevsky, I. Sutskever, G. Hinton, University of Toronto)
 - Réseau de neurones à convolution (AlexNet)
 - 62,3 millions de paramètres (pour information : GPT-3 → 175 000 millions de paramètres)

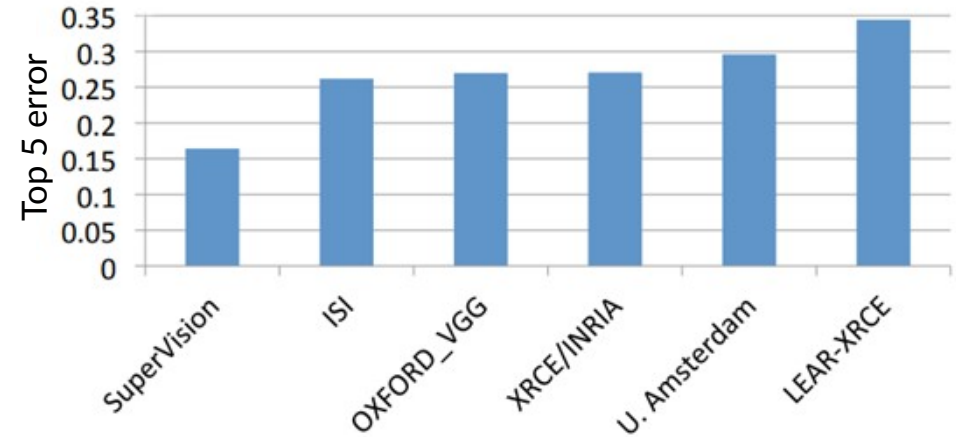
ImageNet Large Scale Visual Recognition Challenge 2012



- ImageNet : 1.2 millions d'images annotées, 1000 classes
- SuperVision (A. Krizhevsky, I. Sutskever, G. Hinton, University of Toronto)
 - Réseau de neurones à convolution (AlexNet)
 - 62,3 millions de paramètres (pour information : GPT-3 → 175 000 millions de paramètres)

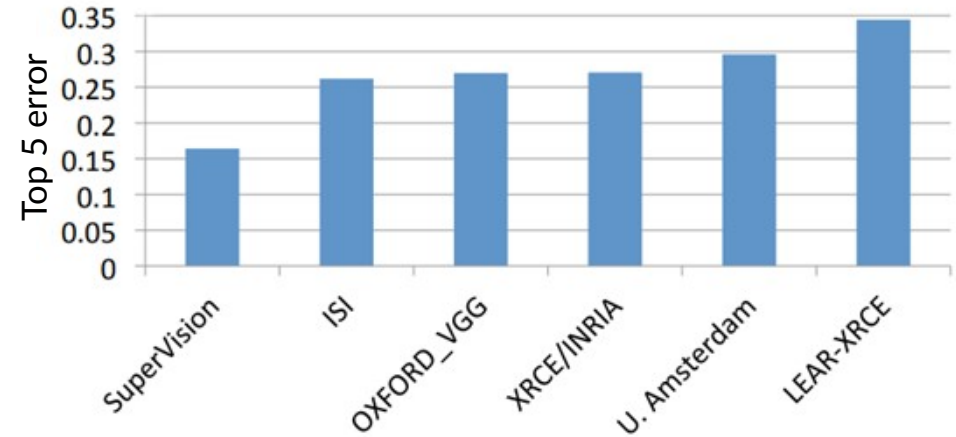
Ingrédient 2 : fonction paramétrique de type « réseau de neurones » 34

ImageNet Large Scale Visual Recognition Challenge 2012



- ImageNet : 1.2 millions d'images annotées, 1000 classes
- SuperVision (A. Krizhevsky, I. Sutskever, G. Hinton, University of Toronto)
 - Réseau de neurones à convolution (AlexNet)
 - 62,3 millions de paramètres (pour information : GPT-3 → 175 000 millions de paramètres)
 - 6 jours d'apprentissage sur 2 GPUs (GTX 580 3GB)

ImageNet Large Scale Visual Recognition Challenge 2012



- ImageNet : 1.2 millions d'images annotées, 1000 classes
- SuperVision (A. Krizhevsky, I. Sutskever, G. Hinton, University of Toronto)
 - Réseau de neurones à convolution (AlexNet)
 - 62,3 millions de paramètres (pour information : GPT-3 → 175 000 millions de paramètres)
 - 6 jours d'apprentissage sur 2 GPUs (GTX 580 3GB)

Ingrédient 3 : grande capacité de calculs en parallèle

Résumé des ingrédients du « Deep Learning »

- 1) Grande base de données étiquetées
- 2) « Bonne » architecture de réseau de neurones profond
- 3) Grande capacité de calculs en parallèle (GPU)

“Perceptron” multicouche (MLP)

Transformation affine = FC (“Fully Connected”)

$$\text{FC}(\mathbf{x}; \boldsymbol{\theta} = \{\mathbf{W}, \mathbf{b}\}) = \mathbf{W}\mathbf{x} + \mathbf{b}$$

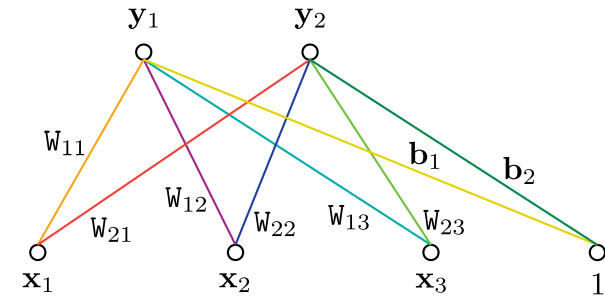
“Perceptron” multicouche (MLP)

Transformation affine = FC (“Fully Connected”)

$$\text{FC}(\mathbf{x}; \boldsymbol{\theta} = \{\mathbf{W}, \mathbf{b}\}) = \mathbf{W}\mathbf{x} + \mathbf{b}$$

Pourquoi « Fully Connected » ?

$$\begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{bmatrix} = \begin{bmatrix} W_{11} & W_{12} & W_{13} & \mathbf{b}_1 \\ W_{21} & W_{22} & W_{23} & \mathbf{b}_2 \end{bmatrix} \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_3 \\ 1 \end{bmatrix}$$



“Perceptron” multicouche (MLP)

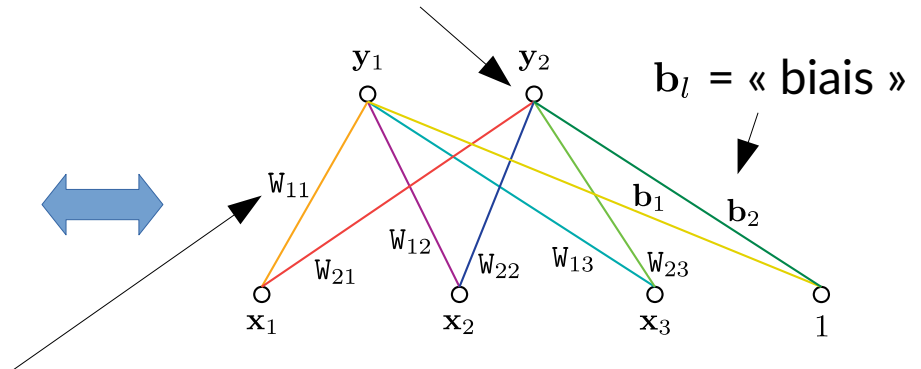
Transformation affine = FC (“Fully Connected”)

$$\text{FC}(\mathbf{x}; \boldsymbol{\theta} = \{\mathbf{W}, \mathbf{b}\}) = \mathbf{W}\mathbf{x} + \mathbf{b}$$

Pourquoi « Fully Connected » ?

$$\begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{bmatrix} = \begin{bmatrix} W_{11} & W_{12} & W_{13} & \mathbf{b}_1 \\ W_{21} & W_{22} & W_{23} & \mathbf{b}_2 \end{bmatrix} \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_3 \\ 1 \end{bmatrix}$$

Élément d'un vecteur = « neurone »



W_{ij} = « poids » d'une connexion entre deux « neurones »

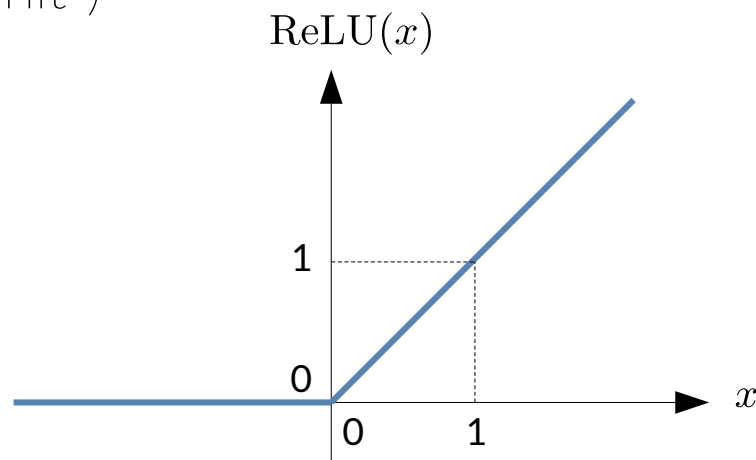
“Perceptron” multicouche (MLP)

Transformation affine = FC (“Fully Connected”)

$$\text{FC}(\mathbf{x}; \boldsymbol{\theta} = \{\mathbf{W}, \mathbf{b}\}) = \mathbf{W}\mathbf{x} + \mathbf{b}$$

Non-linéarité = ReLU (“Rectified Linear Unit”)

$$\text{ReLU}(x) = \max(0, x)$$



“Perceptron” multicouche (MLP)

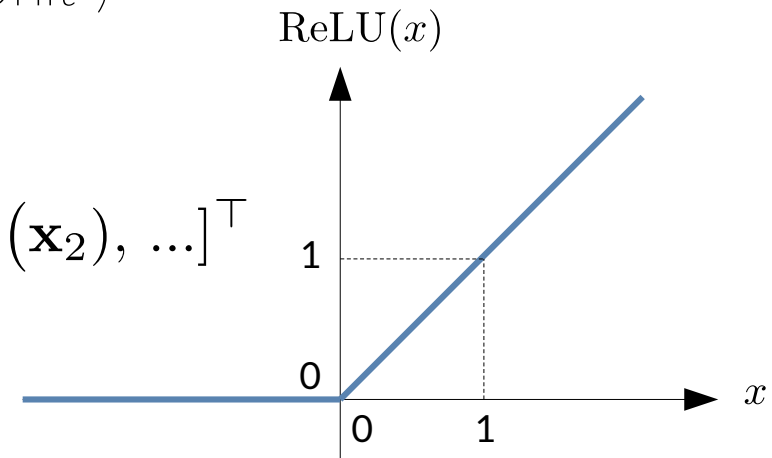
Transformation affine = FC (“Fully Connected”)

$$\text{FC}(\mathbf{x}; \boldsymbol{\theta} = \{\mathbf{W}, \mathbf{b}\}) = \mathbf{W}\mathbf{x} + \mathbf{b}$$

Non-linéarité = ReLU (“Rectified Linear Unit”)

$$\text{ReLU}(x) = \max(0, x)$$

Abus de notation : $\text{ReLU}(\mathbf{x}) = [\text{ReLU}(\mathbf{x}_1), \text{ReLU}(\mathbf{x}_2), \dots]^\top$



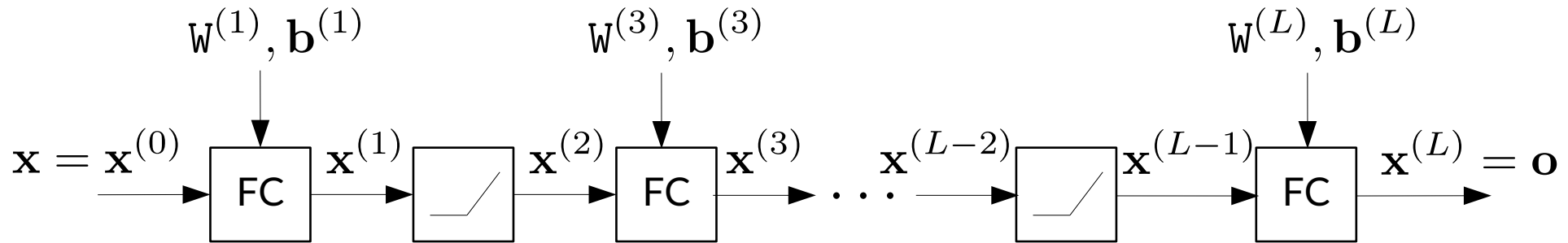
“Perceptron” multicouche (MLP)

Transformation affine = FC (“Fully Connected”)

$$\text{FC}(\mathbf{x}; \boldsymbol{\theta} = \{\mathbf{W}, \mathbf{b}\}) = \mathbf{W}\mathbf{x} + \mathbf{b}$$

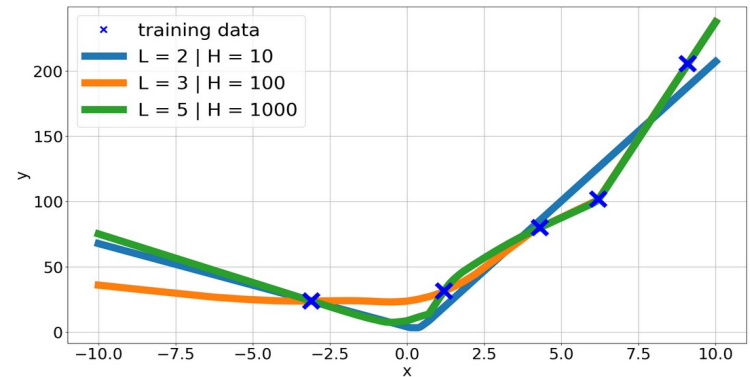
Non-linéarité = ReLU (“Rectified Linear Unit”)

$$\text{ReLU}(x) = \max(0, x)$$



Exemple de régression : 5 données, $X \in \mathbb{R}$ et $Y \in \mathbb{R}$

$X_{\text{train},1} = -3.1$ $X_{\text{train},2} = 1.2$ $X_{\text{train},3} = 4.3$ $X_{\text{train},4} = 6.2$ $X_{\text{train},5} = 9.1$
 $Y_{\text{train},1} = 23.7$ $Y_{\text{train},2} = 31.3$ $Y_{\text{train},3} = 79.9$ $Y_{\text{train},4} = 101.9$ $Y_{\text{train},5} = 205.5$

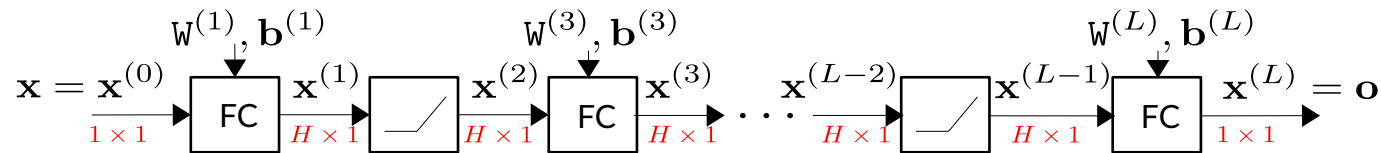


Exemple de régression : 5 données, $x \in \mathbb{R}$ et $y \in \mathbb{R}$

$$x_{\text{train},1} = -3.1 \quad x_{\text{train},2} = 1.2 \quad x_{\text{train},3} = 4.3 \quad x_{\text{train},4} = 6.2 \quad x_{\text{train},5} = 9.1$$

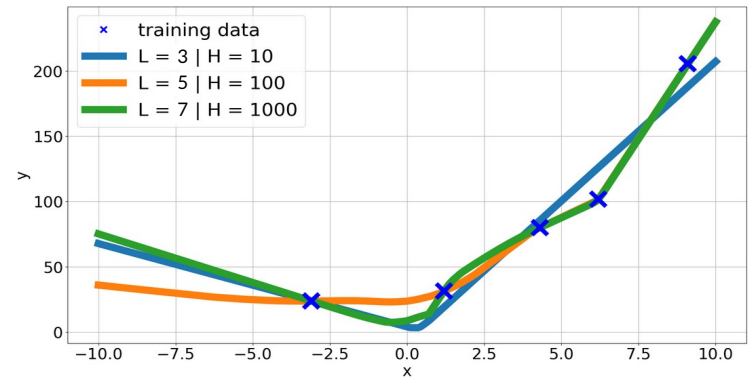
$$y_{\text{train},1} = 23.7 \quad y_{\text{train},2} = 31.3 \quad y_{\text{train},3} = 79.9 \quad y_{\text{train},4} = 101.9 \quad y_{\text{train},5} = 205.5$$

Choix de la fonction



$$f(x; \theta) = \text{MLP} \left(x; \theta = \{w^{(l)}, b^{(l)}\}_l \right)$$

hyper-paramètres : L, H

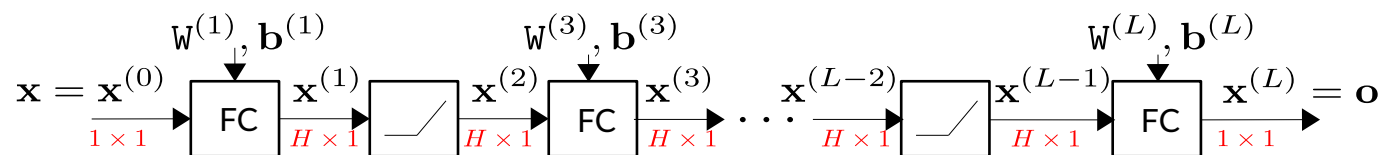


Exemple de régression : 5 données, $x \in \mathbb{R}$ et $y \in \mathbb{R}$

$$x_{\text{train},1} = -3.1 \quad x_{\text{train},2} = 1.2 \quad x_{\text{train},3} = 4.3 \quad x_{\text{train},4} = 6.2 \quad x_{\text{train},5} = 9.1$$

$$y_{\text{train},1} = 23.7 \quad y_{\text{train},2} = 31.3 \quad y_{\text{train},3} = 79.9 \quad y_{\text{train},4} = 101.9 \quad y_{\text{train},5} = 205.5$$

Choix de la fonction



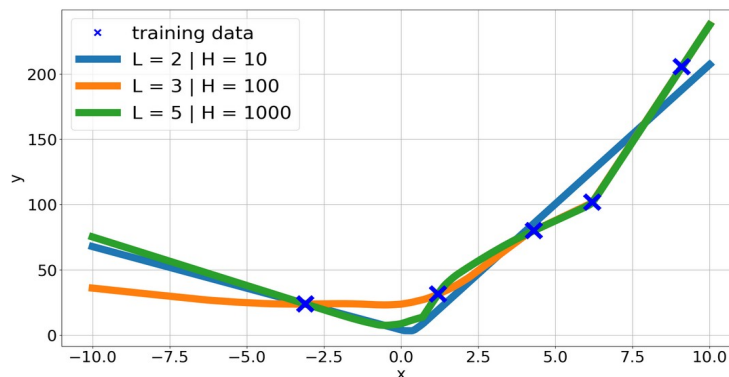
$$f(x; \theta) = \text{MLP} \left(x; \theta = \{w^{(l)}, b^{(l)}\}_l \right)$$

hyper-paramètres : L, H

Apprentissage

Choix du coût $l(y, o) = (y - o)^2$

Optimisation $\theta^* = \arg \min_{\theta} \sum_{i=1}^5 (y_{\text{train},i} - \text{MLP}(x_{\text{train},i}; \theta))^2$

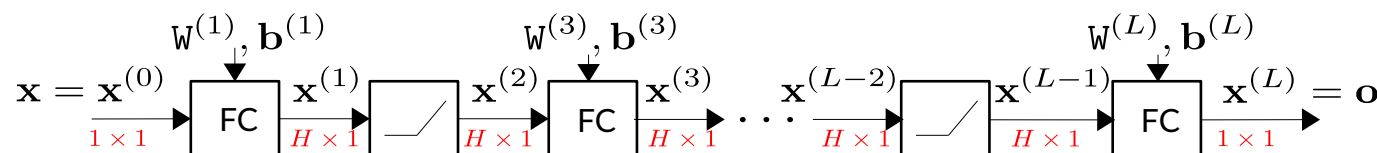


Exemple de régression : 5 données, $x \in \mathbb{R}$ et $y \in \mathbb{R}$

$$x_{\text{train},1} = -3.1 \quad x_{\text{train},2} = 1.2 \quad x_{\text{train},3} = 4.3 \quad x_{\text{train},4} = 6.2 \quad x_{\text{train},5} = 9.1$$

$$y_{\text{train},1} = 23.7 \quad y_{\text{train},2} = 31.3 \quad y_{\text{train},3} = 79.9 \quad y_{\text{train},4} = 101.9 \quad y_{\text{train},5} = 205.5$$

Choix de la fonction



$$f(x; \theta) = \text{MLP} \left(x; \theta = \{w^{(l)}, b^{(l)}\}_l \right)$$

hyper-paramètres : L, H

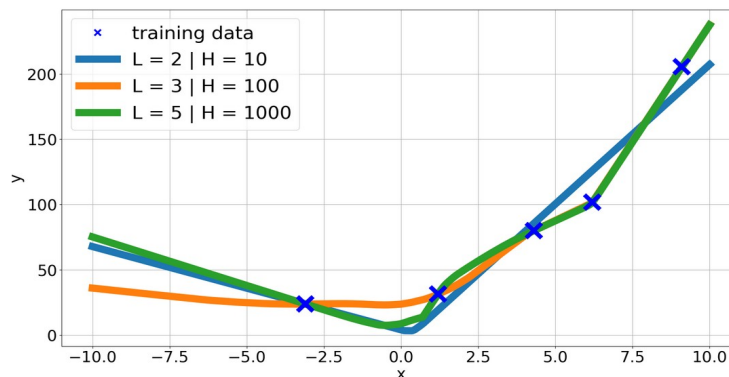
Apprentissage

Choix du coût $l(y, o) = (y - o)^2$

Optimisation
$$\theta^* = \arg \min_{\theta} \sum_{i=1}^5 (y_{\text{train},i} - \text{MLP}(x_{\text{train},i}; \theta))^2$$

Inférence

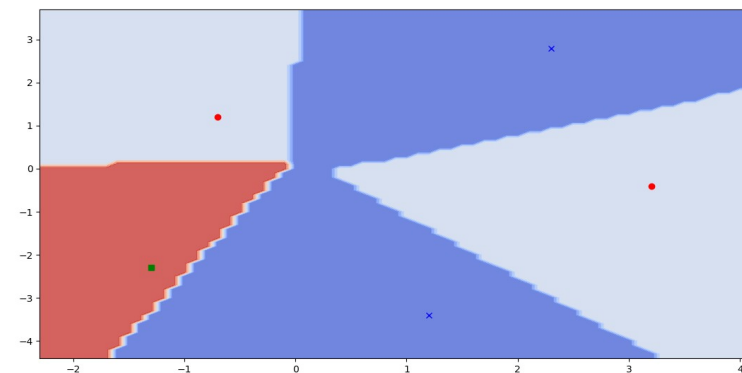
$$o_{\text{test}} = \text{MLP}(x_{\text{test}}; \theta^*)$$



IV)

Exemple de classification : 5 données, $\mathbf{X} \in \mathbb{R}^2$ et $\mathbf{Y} \in \{0, 1, 2\}$

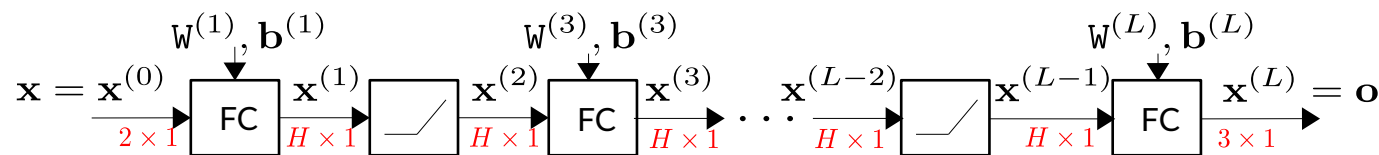
$$\begin{array}{lllll} \mathbf{x}_{\text{train},1} = [1.2, -3.4] & \mathbf{x}_{\text{train},2} = [2.3, 2.8] & \mathbf{x}_{\text{train},3} = [-0.7, 1.2] & \mathbf{x}_{\text{train},4} = [3.2, -0.4] & \mathbf{x}_{\text{train},5} = [-1.3, 2.3] \\ \mathbf{y}_{\text{train},1} = 0 & \mathbf{y}_{\text{train},2} = 0 & \mathbf{y}_{\text{train},3} = 1 & \mathbf{y}_{\text{train},4} = 1 & \mathbf{y}_{\text{train},5} = 2 \end{array}$$



Exemple de classification : 5 données, $\mathbf{x} \in \mathbb{R}^2$ et $\mathbf{y} \in \{0, 1, 2\}$

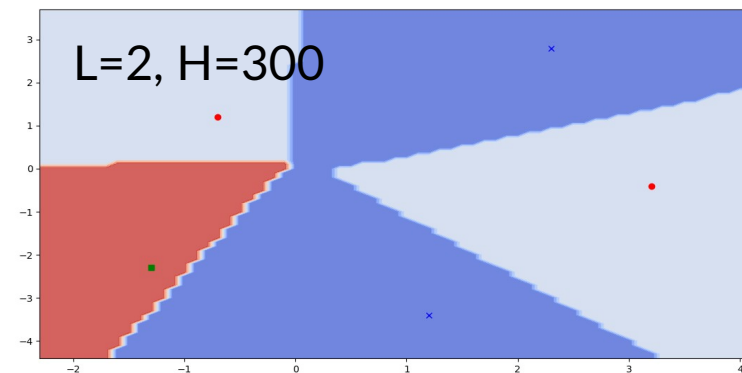
$$\begin{array}{lllll} \mathbf{x}_{\text{train},1} = [1.2, -3.4] & \mathbf{x}_{\text{train},2} = [2.3, 2.8] & \mathbf{x}_{\text{train},3} = [-0.7, 1.2] & \mathbf{x}_{\text{train},4} = [3.2, -0.4] & \mathbf{x}_{\text{train},5} = [-1.3, 2.3] \\ \mathbf{y}_{\text{train},1} = 0 & \mathbf{y}_{\text{train},2} = 0 & \mathbf{y}_{\text{train},3} = 1 & \mathbf{y}_{\text{train},4} = 1 & \mathbf{y}_{\text{train},5} = 2 \end{array}$$

Choix de la fonction



$$f(x; \theta) = \text{MLP} \left(x; \theta = \{w^{(l)}, b^{(l)}\}_l \right)$$

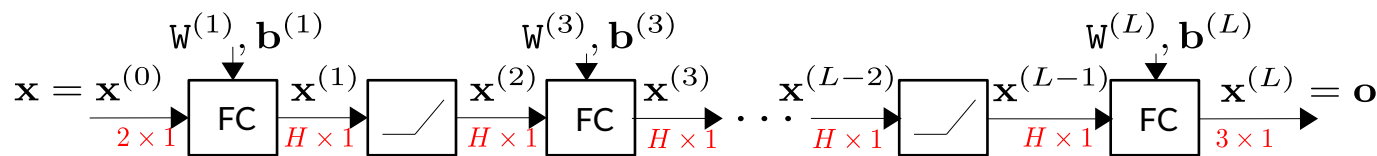
hyper-paramètres : L, H



Exemple de classification : 5 données, $\mathbf{x} \in \mathbb{R}^2$ et $\mathbf{y} \in \{0, 1, 2\}$

$$\begin{array}{lllll} \mathbf{x}_{\text{train},1} = [1.2, -3.4] & \mathbf{x}_{\text{train},2} = [2.3, 2.8] & \mathbf{x}_{\text{train},3} = [-0.7, 1.2] & \mathbf{x}_{\text{train},4} = [3.2, -0.4] & \mathbf{x}_{\text{train},5} = [-1.3, 2.3] \\ \mathbf{y}_{\text{train},1} = 0 & \mathbf{y}_{\text{train},2} = 0 & \mathbf{y}_{\text{train},3} = 1 & \mathbf{y}_{\text{train},4} = 1 & \mathbf{y}_{\text{train},5} = 2 \end{array}$$

Choix de la fonction



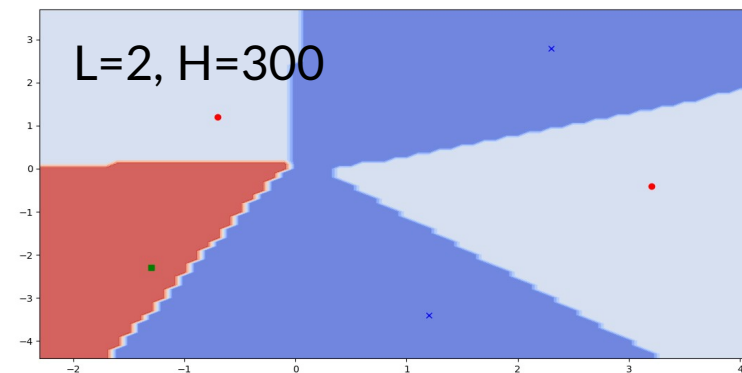
$$f(x; \theta) = \text{MLP} \left(x; \theta = \{\mathbf{w}^{(l)}, \mathbf{b}^{(l)}\}_l \right)$$

hyper-paramètres : L, H

Apprentissage

Choix du coût $l(y, \mathbf{o}) = -\ln(\text{softmax}(\mathbf{o})[y])$

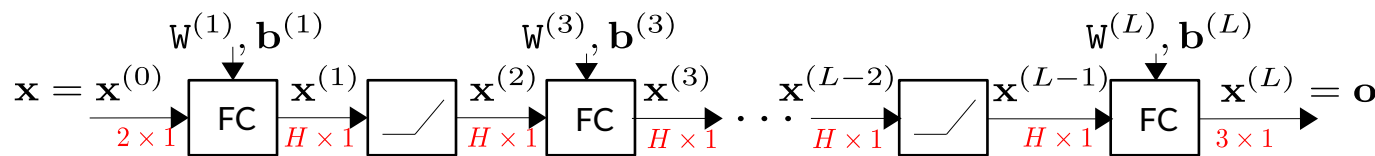
Optimisation $\theta^* = \arg \min_{\theta} \sum_{i=1}^5 -\ln(\text{softmax}(\text{MLP}(\mathbf{x}_{\text{train},i}; \theta))[\mathbf{y}_{\text{train},i}])$



Exemple de classification : 5 données, $\mathbf{X} \in \mathbb{R}^2$ et $\mathbf{Y} \in \{0, 1, 2\}$

$$\begin{array}{lllll} \mathbf{x}_{\text{train},1} = [1.2, -3.4] & \mathbf{x}_{\text{train},2} = [2.3, 2.8] & \mathbf{x}_{\text{train},3} = [-0.7, 1.2] & \mathbf{x}_{\text{train},4} = [3.2, -0.4] & \mathbf{x}_{\text{train},5} = [-1.3, 2.3] \\ \mathbf{y}_{\text{train},1} = 0 & \mathbf{y}_{\text{train},2} = 0 & \mathbf{y}_{\text{train},3} = 1 & \mathbf{y}_{\text{train},4} = 1 & \mathbf{y}_{\text{train},5} = 2 \end{array}$$

Choix de la fonction



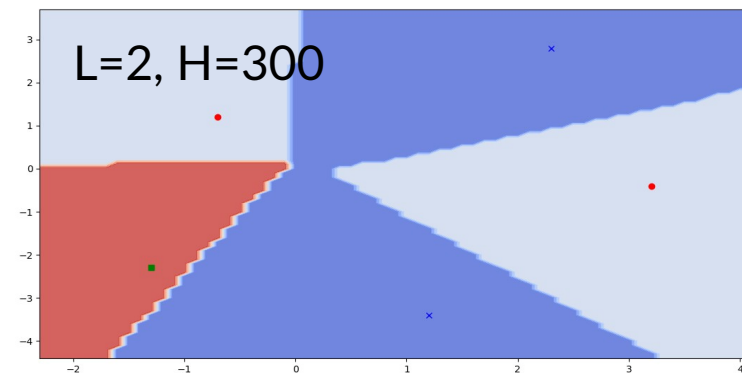
$$f(x; \theta) = \text{MLP} \left(x; \theta = \{w^{(l)}, b^{(l)}\}_l \right)$$

hyper-paramètres : L, H

Apprentissage

Choix du coût $l(y, \mathbf{o}) = -\ln(\text{softmax}(\mathbf{o})[y])$

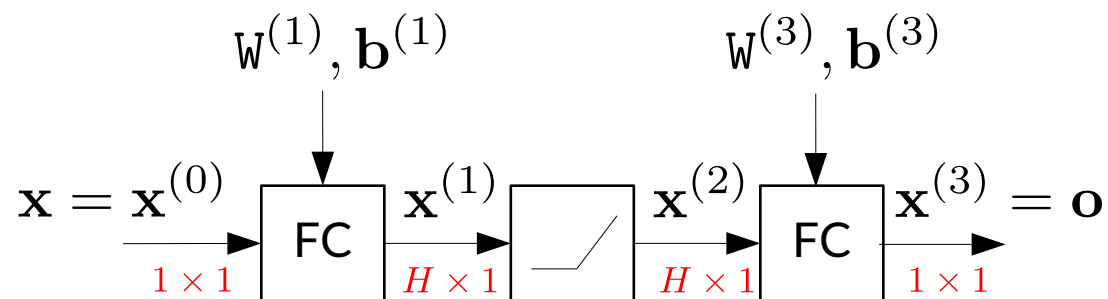
Optimisation $\theta^* = \arg \min_{\theta} \sum_{i=1}^5 -\ln(\text{softmax}(\text{MLP}(\mathbf{x}_{\text{train},i}; \theta))[\mathbf{y}_{\text{train},i}])$



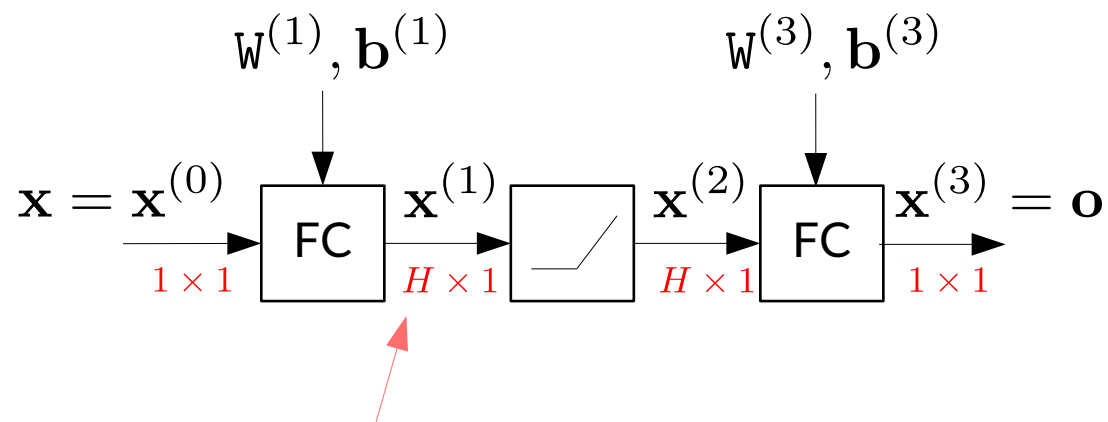
Inférence

$\mathbf{o}_{\text{test}} = \text{MLP}(x_{\text{test}}; \theta^*)$ Classe prédite : $\arg \max \mathbf{o}_{\text{test}}$

Étude du MLP à une couche cachée en 1D

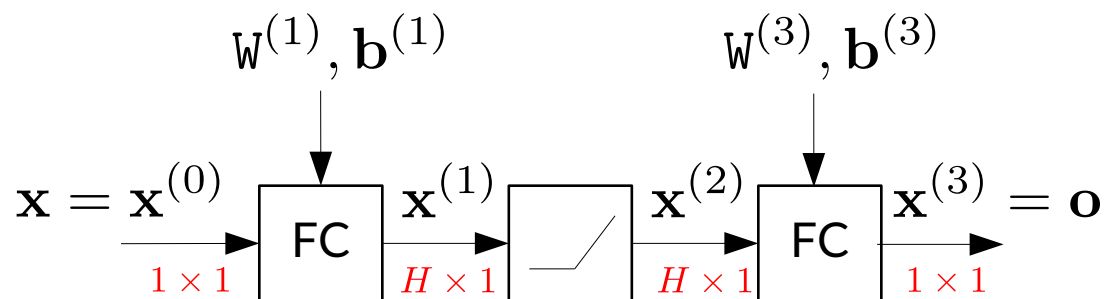


Étude du MLP à une couche cachée en 1D



H contrôle la largeur (« width ») du réseau

Étude du MLP à une couche cachée en 1D



$\mathbf{W}^{(1)} \doteq \mathbf{w}_1$ Vecteur colonne $H \times 1$

$\mathbf{b}^{(1)} \doteq \mathbf{b}_1$ Vecteur colonne $H \times 1$

$\mathbf{W}^{(3)} \doteq \mathbf{w}_3^\top$ Vecteur ligne $1 \times H$

$\mathbf{b}^{(3)} \doteq b_3$ Scalaire 1×1

$$f(x) = \mathbf{w}_3^\top (\text{ReLU}(\mathbf{w}_1 x + \mathbf{b}_1)) + b_3$$

Étude du MLP à une couche cachée en 1D (suite)

$$\begin{aligned} f(x) &= \mathbf{w}_3^\top \text{ReLU}(\mathbf{w}_1 x + \mathbf{b}_1) + b_3 \\ &= \sum_{j=1}^H \mathbf{w}_{3,j} \text{ReLU}(\mathbf{w}_{1,j} x + \mathbf{b}_{1,j}) + b_3 \end{aligned}$$

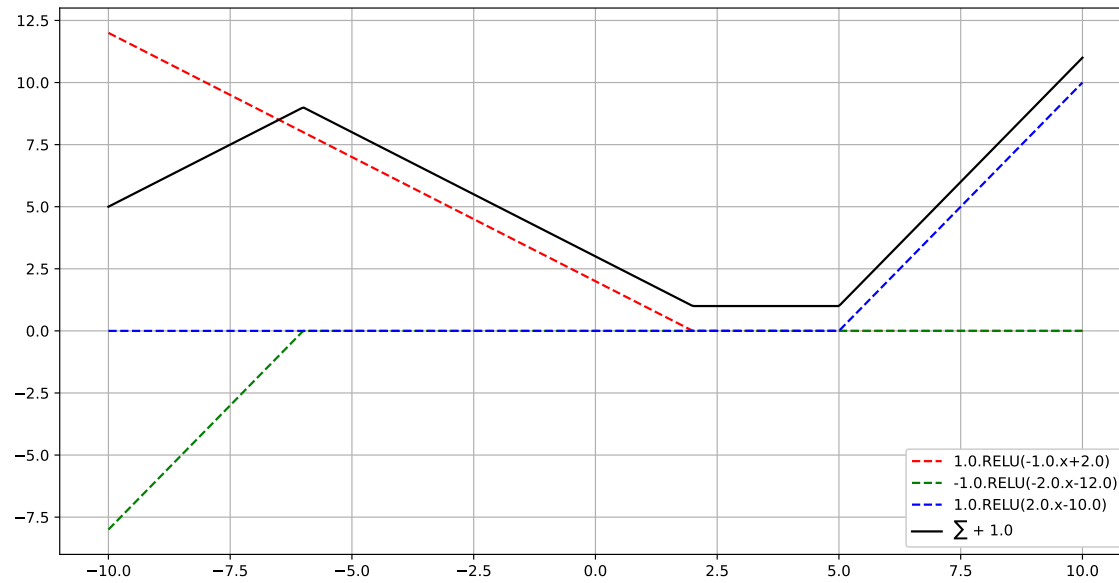


Somme pondérée de fonctions ReLU !

Étude du MLP à une couche cachée en 1D (suite)

$$\begin{aligned} f(x) &= \mathbf{w}_3^\top \text{ReLU}(\mathbf{w}_1 x + \mathbf{b}_1) + b_3 \\ &= \sum_{j=1}^H \mathbf{w}_{3,j} \text{ReLU}(\mathbf{w}_{1,j} x + \mathbf{b}_{1,j}) + b_3 \end{aligned}$$

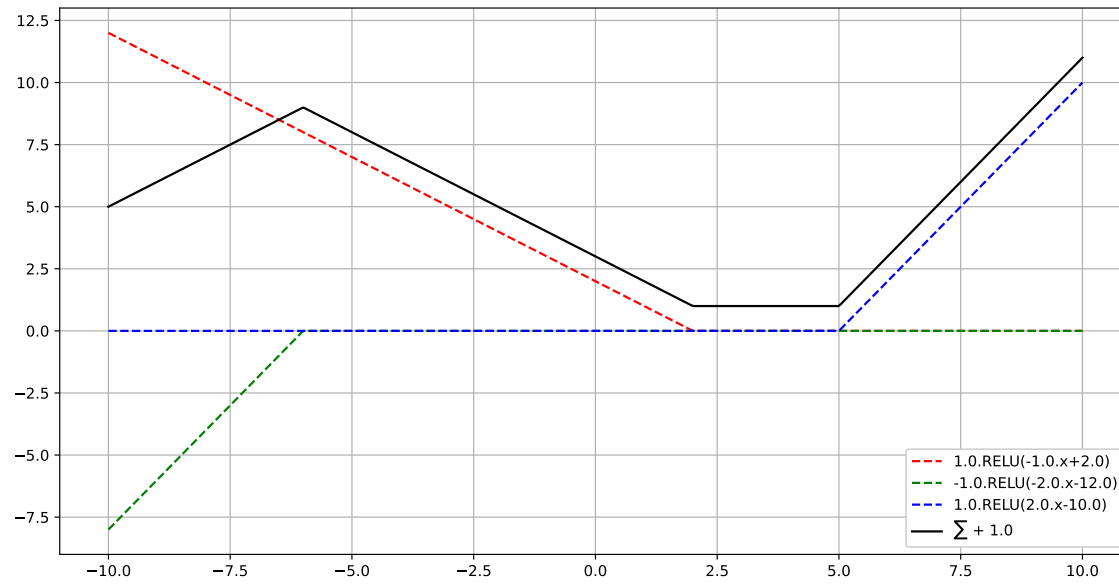
Exemple
numérique avec
 $H=3$



Étude du MLP à une couche cachée en 1D (suite)

$$\begin{aligned} f(x) &= \mathbf{w}_3^\top \text{ReLU}(\mathbf{w}_1 x + \mathbf{b}_1) + b_3 \\ &= \sum_{j=1}^H \mathbf{w}_{3,j} \text{ReLU}(\mathbf{w}_{1,j} x + \mathbf{b}_{1,j}) + b_3 \end{aligned}$$

Exemple
numérique avec
 $H=3$



Fonction
continue et
affine par
morceaux

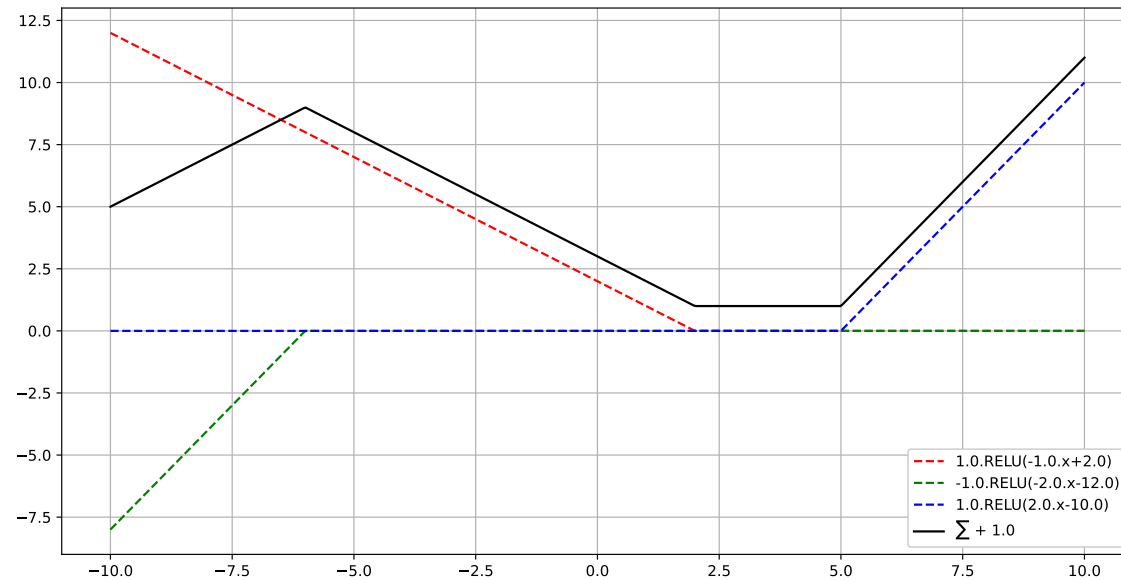
IV)

Étude du MLP à une couche cachée en 1D (suite)

$$f(x) = \mathbf{w}_3^\top \text{ReLU}(\mathbf{w}_1 x + \mathbf{b}_1) + b_3$$

$$= \sum_{j=1}^H \mathbf{w}_{3,j} \text{ReLU}(\mathbf{w}_{1,j} x + \mathbf{b}_{1,j}) + b_3$$

Exemple
numérique avec
H=3

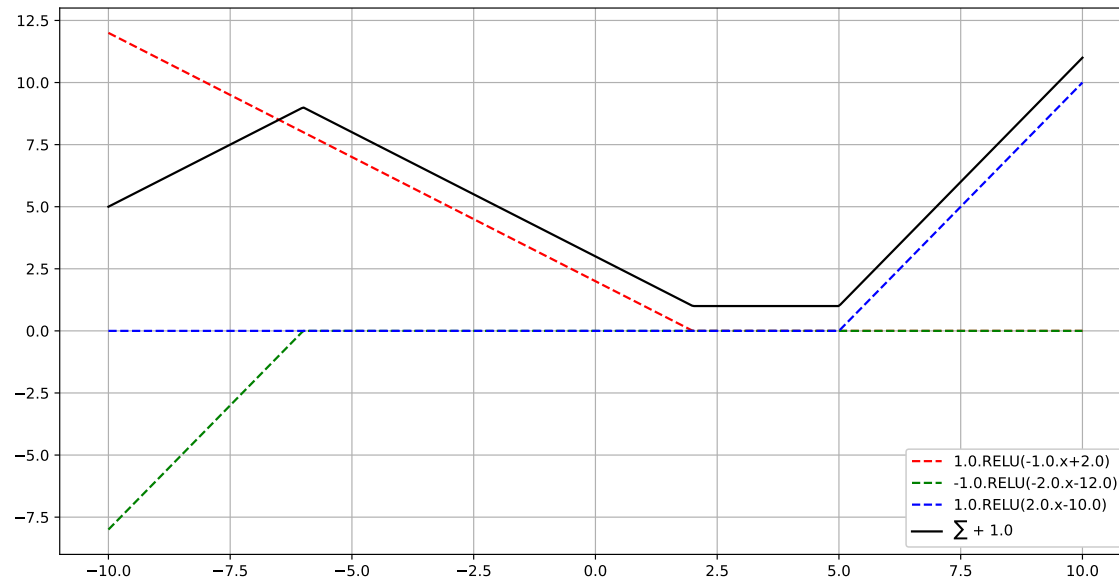


H
changements
de pente
(un par ReLU)
=
H+1
morceaux

Étude du MLP à une couche cachée en 1D (suite)

$$\begin{aligned} f(x) &= \mathbf{w}_3^\top \text{ReLU}(\mathbf{w}_1 x + \mathbf{b}_1) + b_3 \\ &= \sum_{j=1}^H \mathbf{w}_{3,j} \text{ReLU}(\mathbf{w}_{1,j} x + \mathbf{b}_{1,j}) + b_3 \end{aligned}$$

Exemple
numérique avec
 $H=3$



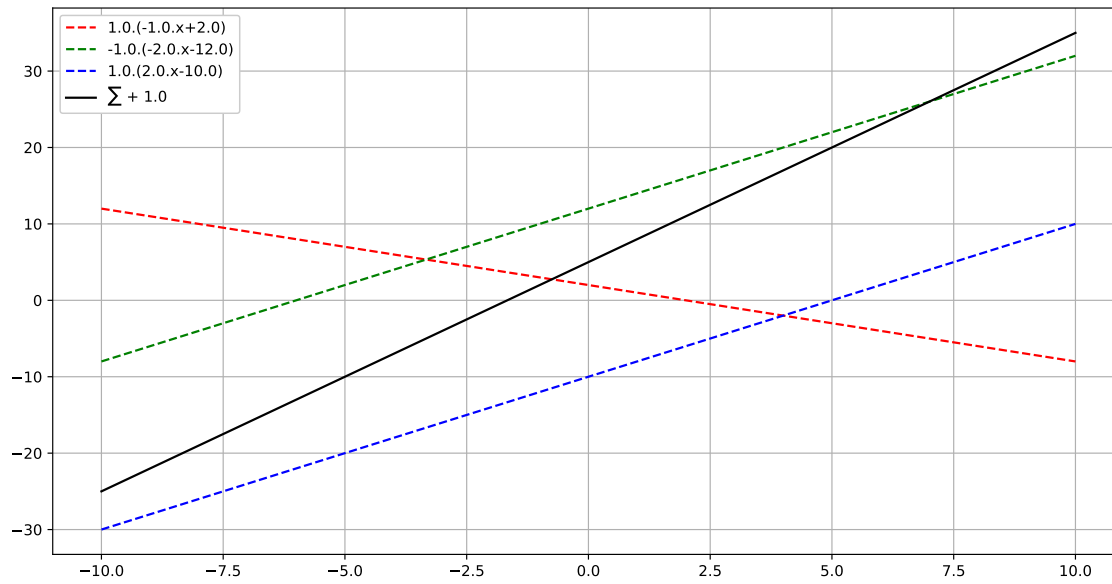
Augmenter H
accroît la
capacité du
réseau.

Étude du MLP à une couche cachée en 1D (suite)

$$f(x) = \mathbf{w}_3^\top \cancel{\text{ReLU}}(\mathbf{w}_1 x + \mathbf{b}_1) + b_3$$

$$= \sum_{j=1}^H \mathbf{w}_{3,j} \cancel{\text{ReLU}}(\mathbf{w}_{1,j} x + \mathbf{b}_{1,j}) + b_3$$

Et sans ReLU ?



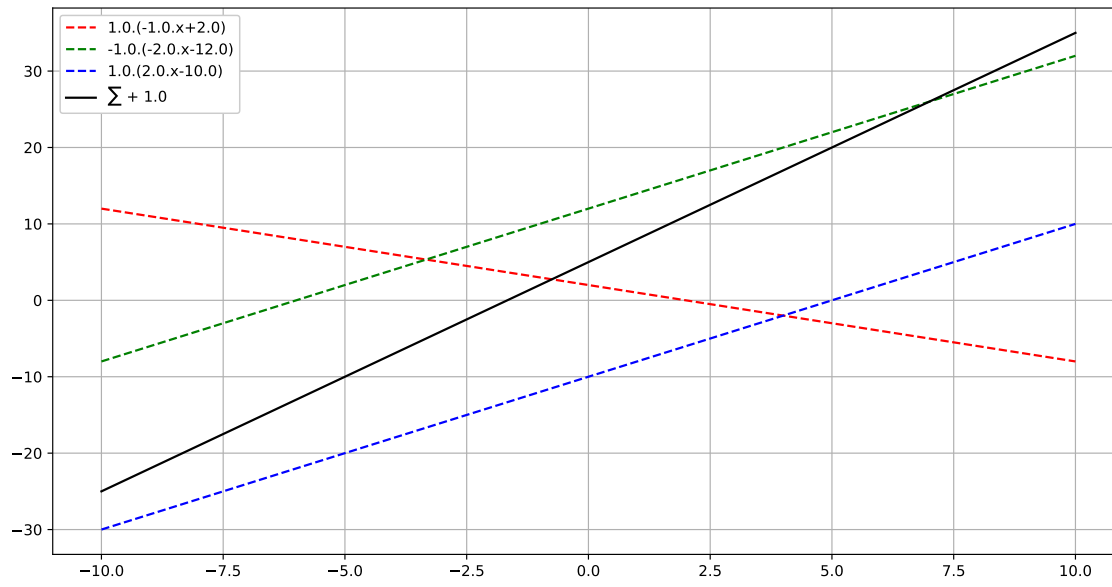
Une somme
de fonctions
linéaires est
une fonction
linéaire...

Étude du MLP à une couche cachée en 1D (suite)

$$f(x) = \mathbf{w}_3^\top \cancel{\text{ReLU}}(\mathbf{w}_1 x + \mathbf{b}_1) + b_3$$

$$= \sum_{j=1}^H \mathbf{w}_{3,j} \cancel{\text{ReLU}}(\mathbf{w}_{1,j} x + \mathbf{b}_{1,j}) + b_3$$

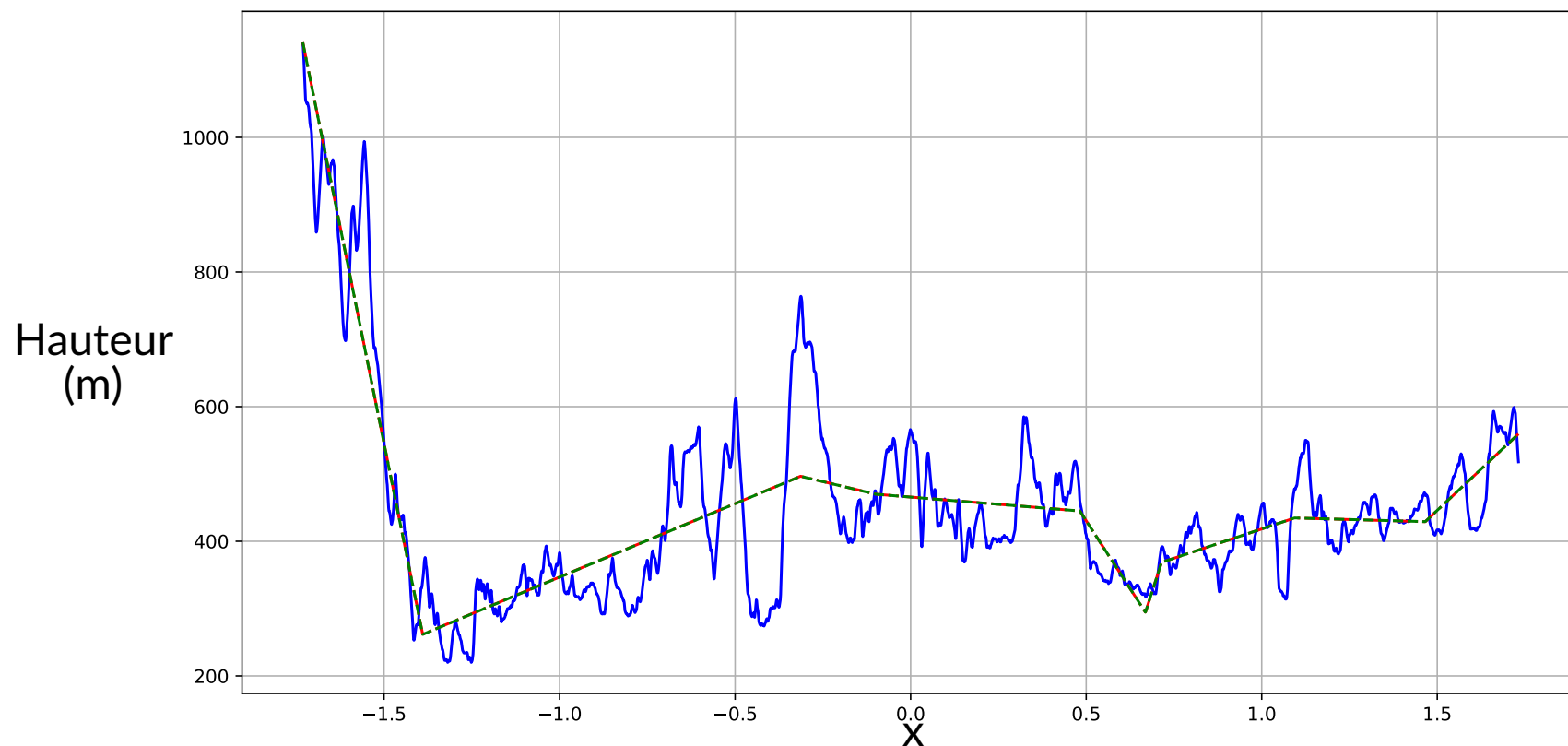
Et sans ReLU ?



Augmenter H
n'accroît pas
la capacité du
réseau.

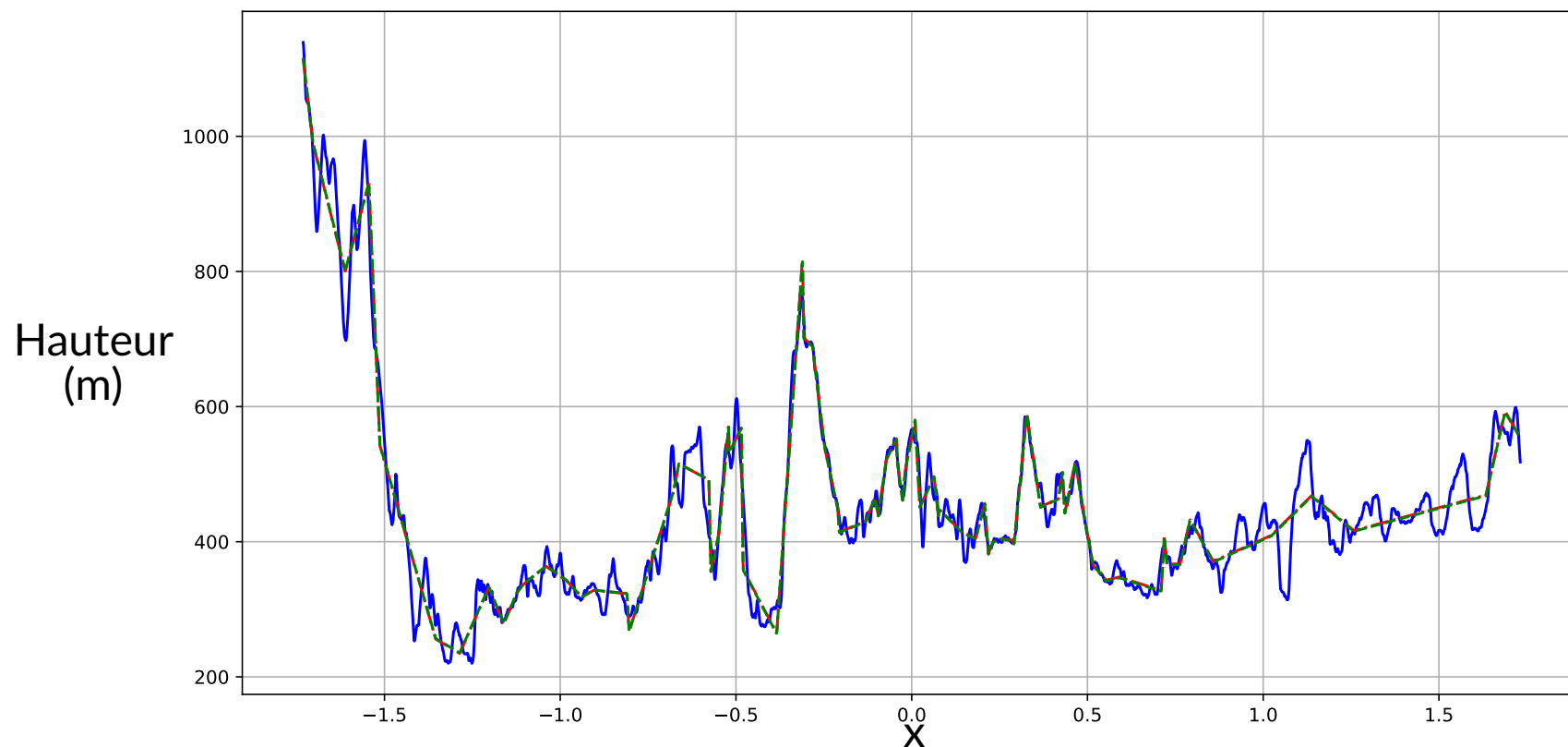
Étude du MLP à une couche cachée en 1D (suite bis)

Optimisation d'un MLP sur un profil de terrain : $H=10$



Étude du MLP à une couche cachée en 1D (suite bis)

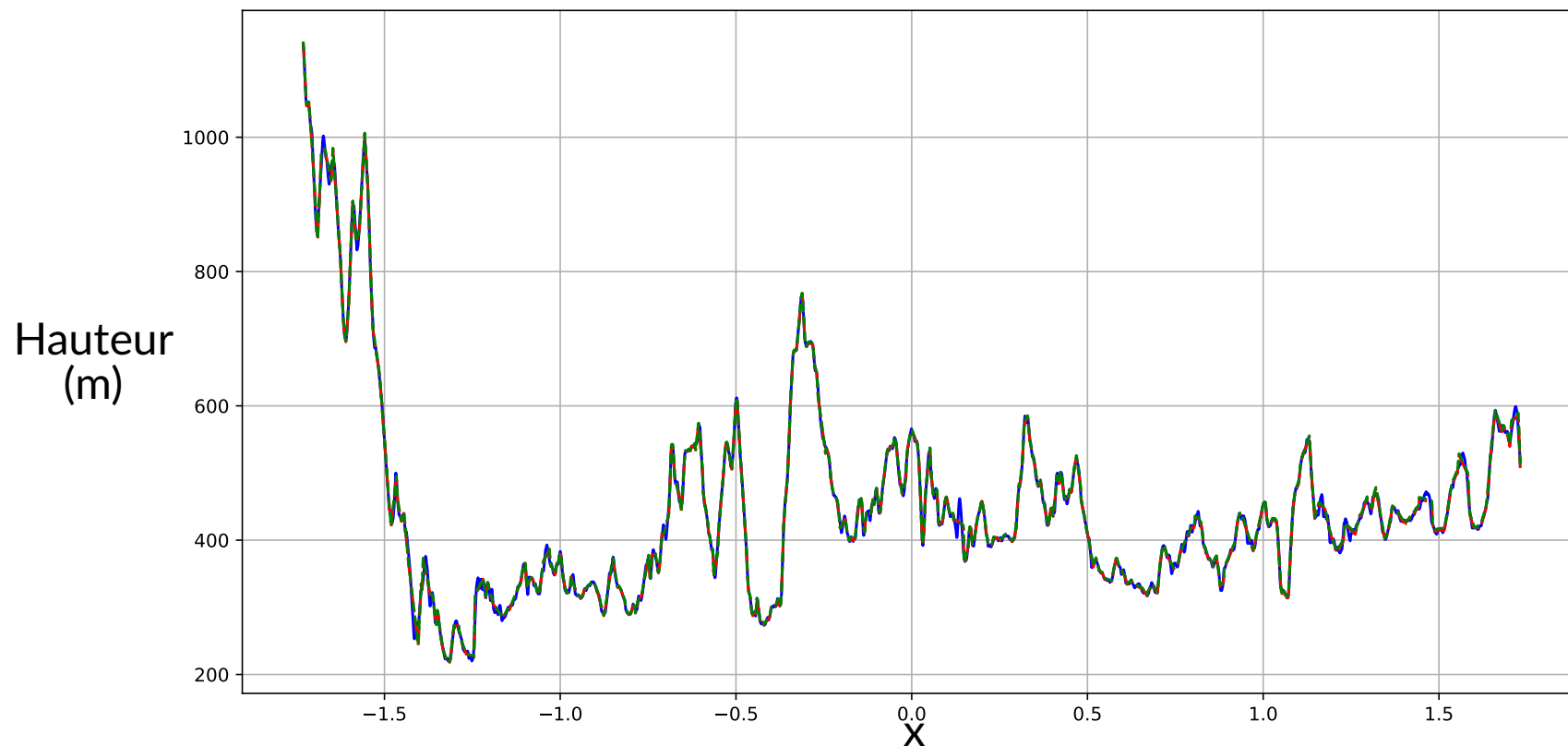
Optimisation d'un MLP sur un profil de terrain : $H=100$



IV)

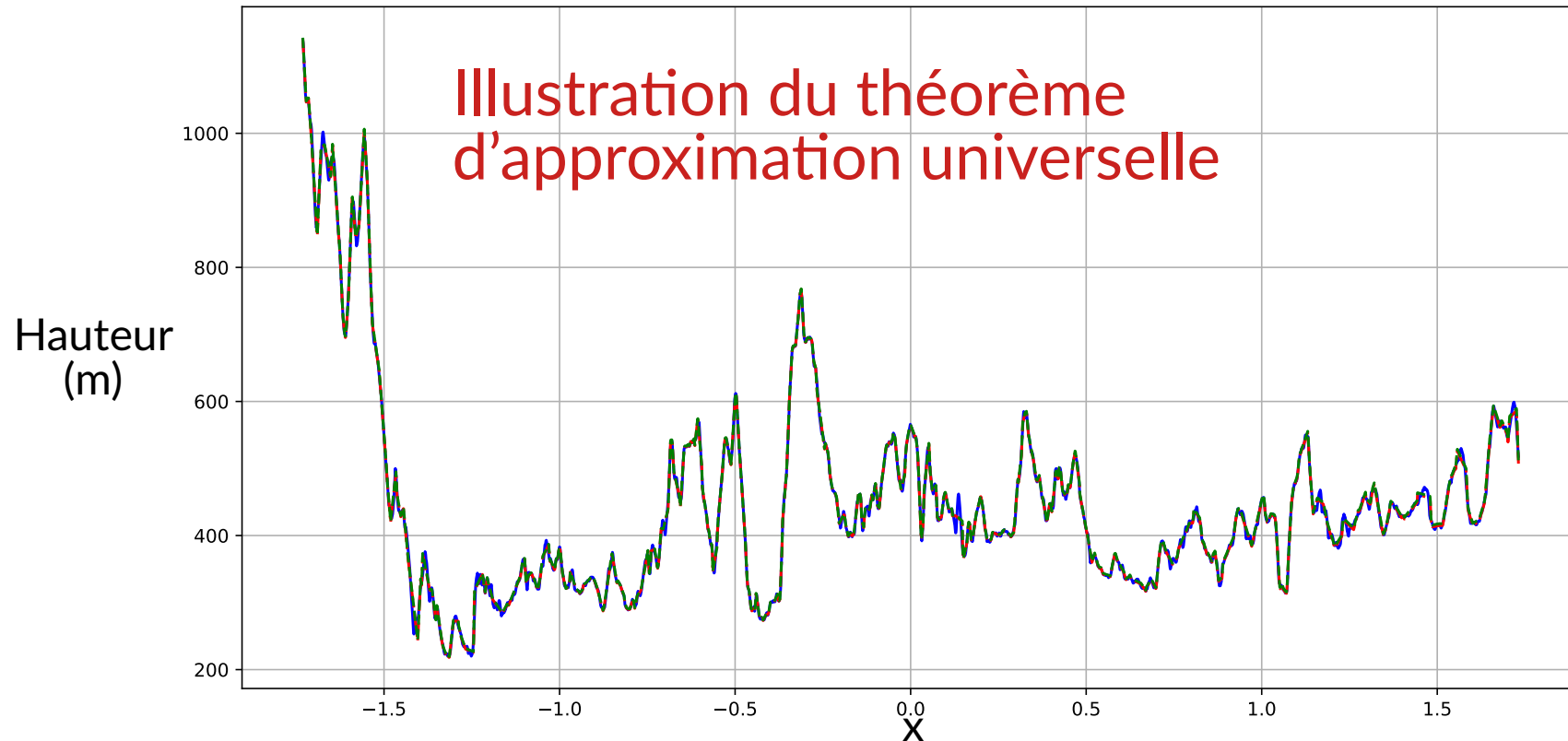
Étude du MLP à une couche cachée en 1D (suite bis)

Optimisation d'un MLP sur un profil de terrain : $H=1000$



Étude du MLP à une couche cachée en 1D (suite bis)

Optimisation d'un MLP sur un profil de terrain : $H=1000$



V) Risques

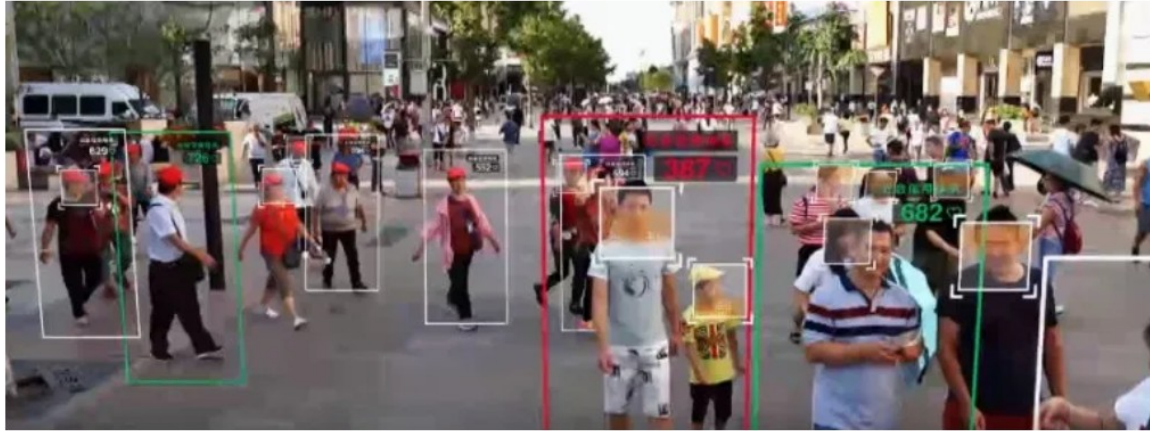
IA et vie privée

Are you ready? Here is all the data
Facebook and Google have on you
Dylan Curran

- Google knows where you've been
- Google knows everything you've ever searched – and deleted
- Google has an advertisement profile of you
- Google knows all the apps you use
- Google has all of your YouTube history
- The data Google has on you can fill millions of Word documents

<https://www.theguardian.com/commentisfree/2018/mar/28/all-the-data-facebook-google-has-on-you-privacy>

IA et vie privée (suite)



The Chinese state wants to control its citizens via a system of social scoring that punishes behavior it doesn't approve of. Image Credit: Telecoms

Le projet de smart city d'Alphabet à Toronto suscite l'inquiétude des experts

VU AILLEURS Le projet de smart city lancé à Toronto en 2017 par Sidewalk Labs, la branche d'innovation urbaine d'Alphabet, est vivement critiqué par la population locale et certains experts consultants du projet. La protection de la vie privée ne serait pas respectée.

La machine au service de l'homme ?



Mme Tang Yu, PDG du chinois NetDragon Websoft et de ses 6000 employés, est le premier robot à être nommé à la tête d'une société. Disponible H24, elle ne touche aucun salaire. *NetDragon Websoft*

IA et consommation énergétique

Common carbon footprint benchmarks

in lbs of CO2 equivalent

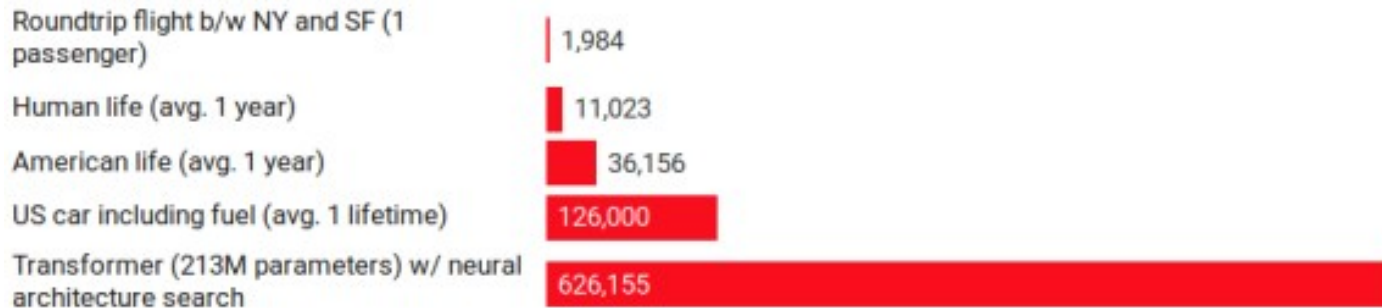


Chart: MIT Technology Review • Source: Strubell et al. • Created with Datawrapper

Conseils de lecture :

L'Enfer Numérique, Guillaume Pitron, 2021

Training a single AI model can emit as much carbon as five cars in their lifetimes, MIT Press

Energy and Policy Considerations for Deep Learning in NLP, Strubell et al., 2019

IA et transhumanisme



Conseil de lecture : Leurre et malheur du transhumanisme, Olivier Rey