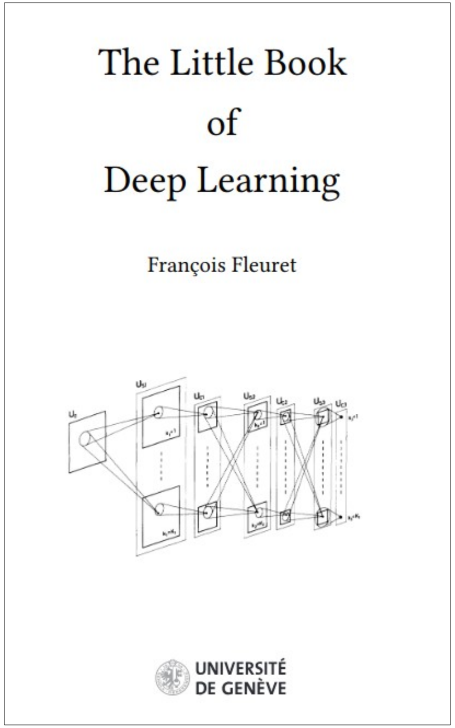
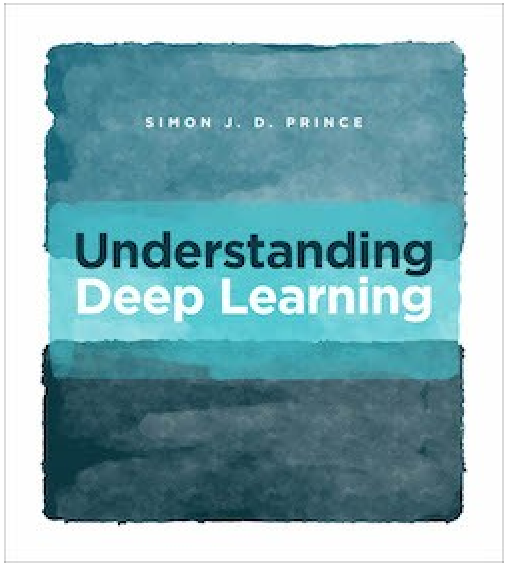
Introduction aux réseaux de neurones pour l'apprentissage supervisé

Guillaume Bourmaud

Livres conseillés





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. Enjeux

I) Introduction

Apprentissage automatique (« Machine Learning »)

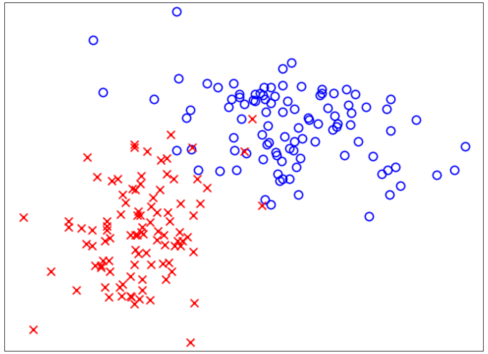
Laisser un ordinateur déduire des **« règles »** d'un ensemble de **données numériques.**

Ex. d'application : recommandation de produits

I)

Apprentissage automatique
(« Machine Learning »)

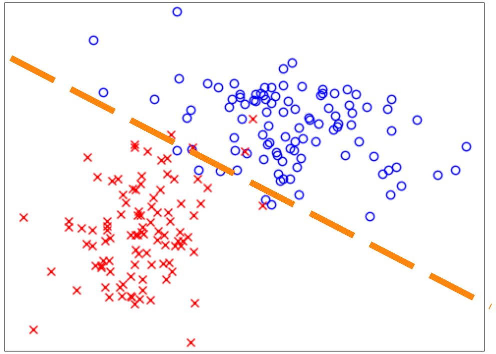
Supervisé



I)

Apprentissage automatique (« Machine Learning »)

Supervisé

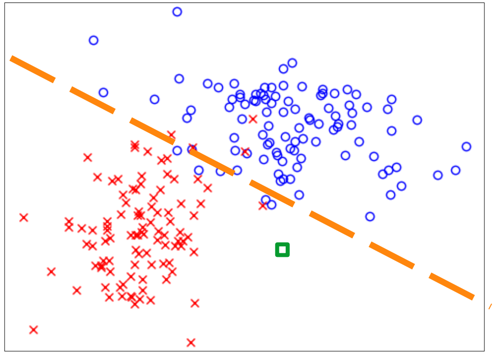


Déduire des « règles » associant une étiquette à une donnée.

I)

Apprentissage automatique
(« Machine Learning »)

Supervisé



Déduire des « règles » associant une étiquette à une donnée.

Utilisation la plus fréquente :
Classifier une nouvelle donnée

I)

Apprentissage automatique
(« Machine Learning »)

Supervisé

Comestible

Non comestible

I)

Apprentissage automatique
(« Machine Learning »)

Supervisé

Comestible

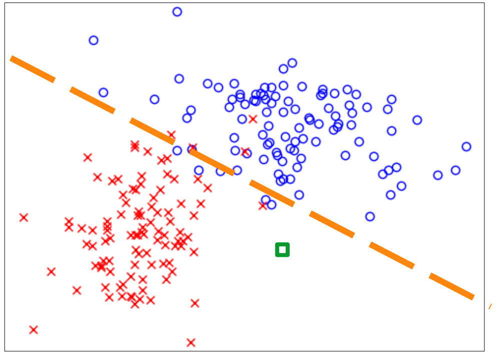
Non comestible

Comestible ?

I)

Apprentissage automatique
(« Machine Learning »)

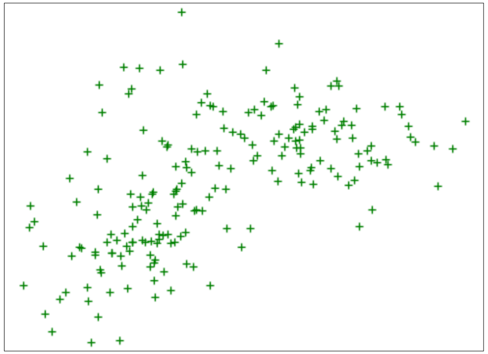
Supervisé



Déduire des « règles » associant une étiquette à une donnée.

Utilisation la plus fréquente :
Classifier une nouvelle donnée

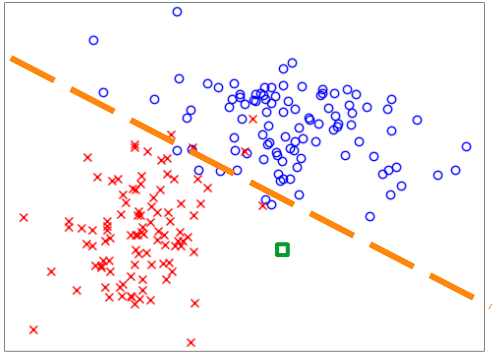
Non supervisé



I)

Apprentissage automatique
(« Machine Learning »)

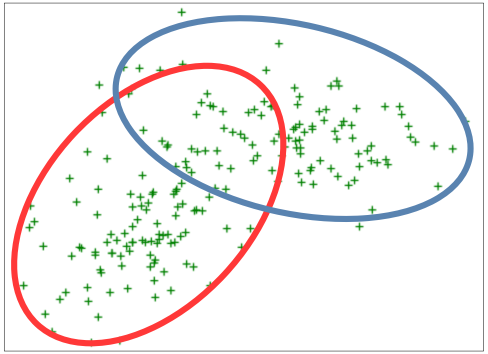
Supervisé



Déduire des « règles » associant une étiquette à une donnée.

Utilisation la plus fréquente :
Classifier une nouvelle donnée

Non supervisé

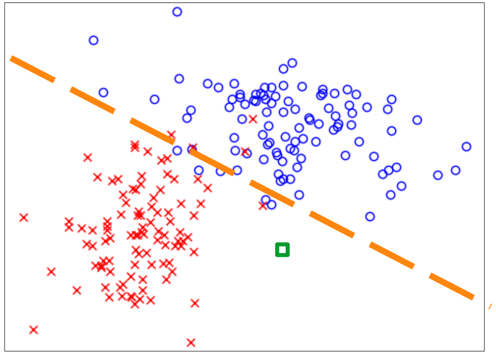


Déduire des « règles » structurant les données.

I)

Apprentissage automatique
(« Machine Learning »)

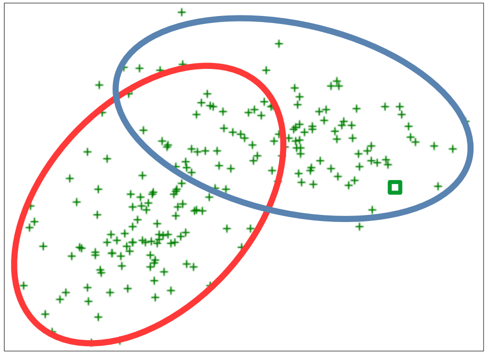
Supervisé



Déduire des « règles » associant une étiquette à une donnée.

Utilisation la plus fréquente :
Classifier une nouvelle donnée

Non supervisé

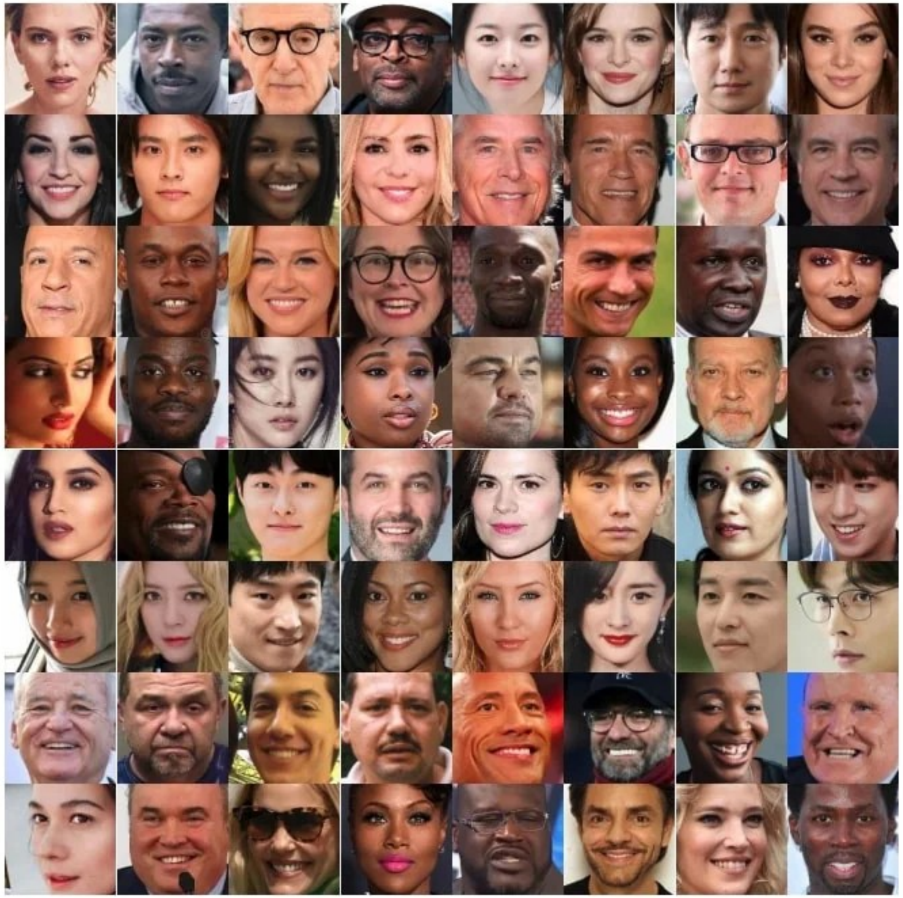


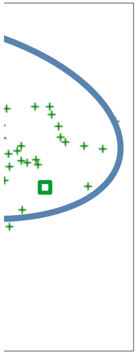
Déduire des « règles » structurant les données.

Ex. d'utilisation :
- Réduction de dimension
- Générer une nouvelle donnée

I)

Apprentissage automatique
(« Machine Learning »)





sé

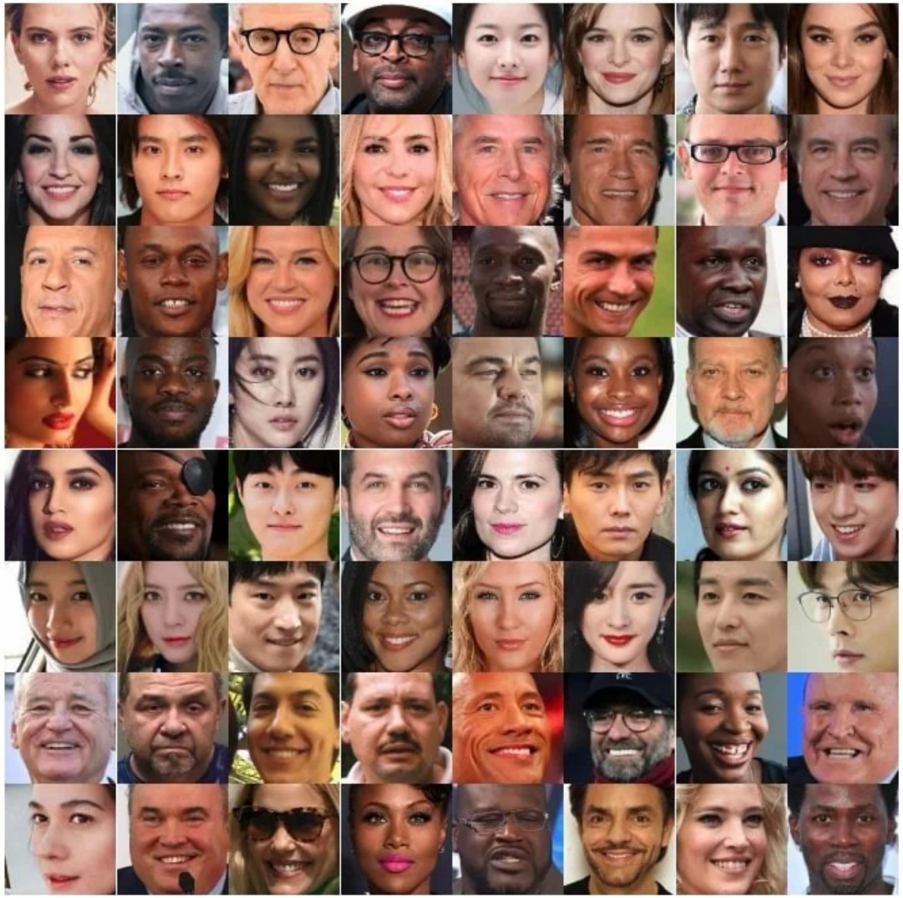
Dédui… »
associ… ées.
donné…

Utilisa…
Classif… e donnée

I)

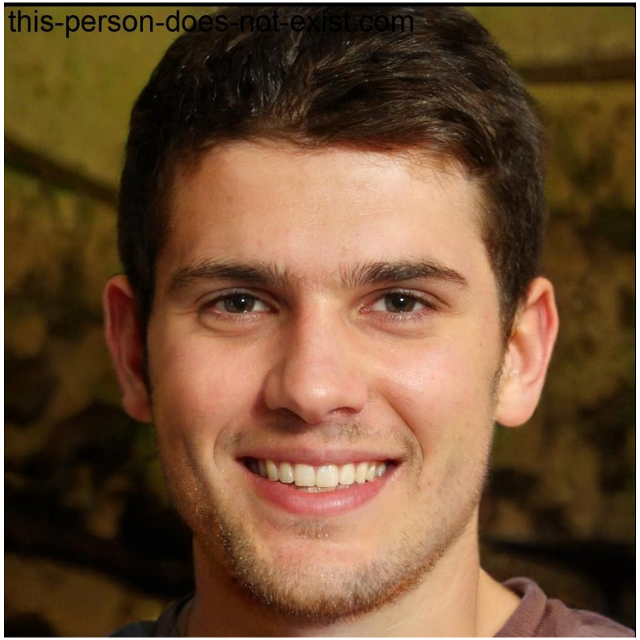
Apprentissage automatique
(« Machine Learning »)

sé



Dédui… »
associ… ées.
donné…

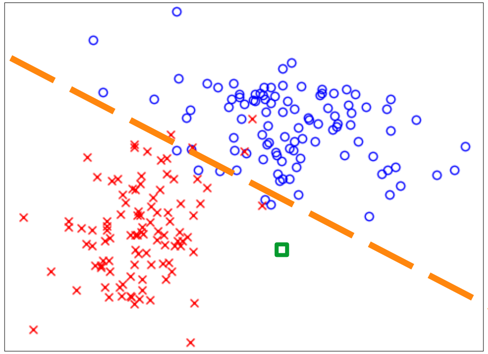
Utilisa…
Classif… e donnée



I)

Apprentissage automatique
(« Machine Learning »)

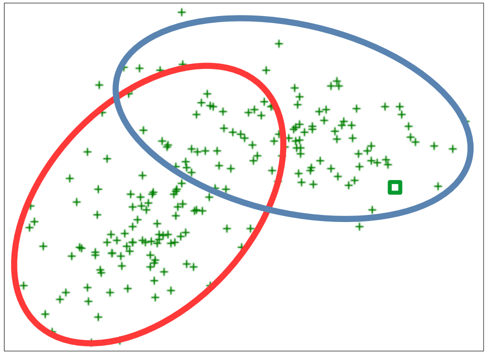
Supervisé



Déduire des « règles » associant une étiquette à une donnée.

Utilisation la plus fréquente :
Classifier une nouvelle donnée

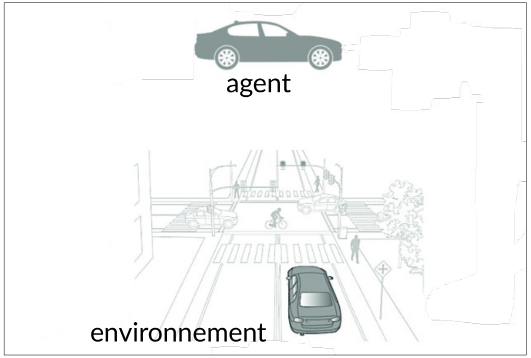
Non supervisé



Déduire des « règles » structurant les données.

Ex. d'utilisation :
- Réduction de dimension
- Générer une nouvelle donnée

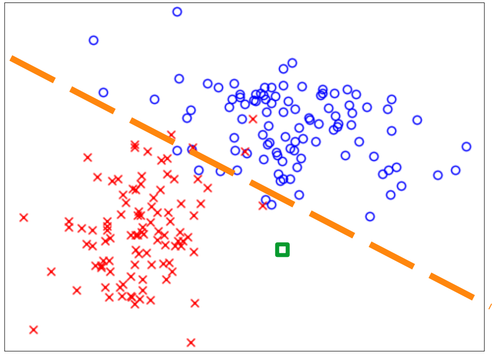
Par renforcement



I)

Apprentissage automatique
(« Machine Learning »)

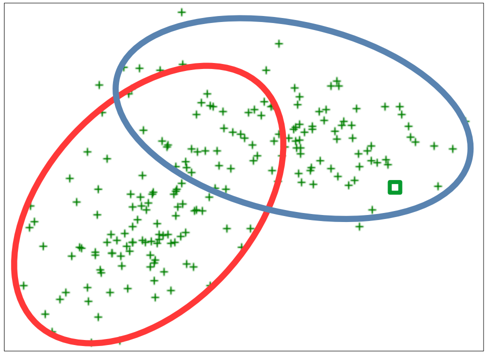
Supervisé



Déduire des « règles » associant une étiquette à une donnée.

Utilisation la plus fréquente :
Classifier une nouvelle donnée

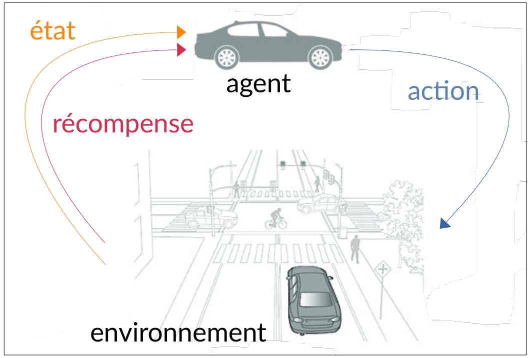
Non supervisé



Déduire des « règles » structurant les données.

Ex. d'utilisation :
- Réduction de dimension
- Générer une nouvelle donnée

Par renforcement



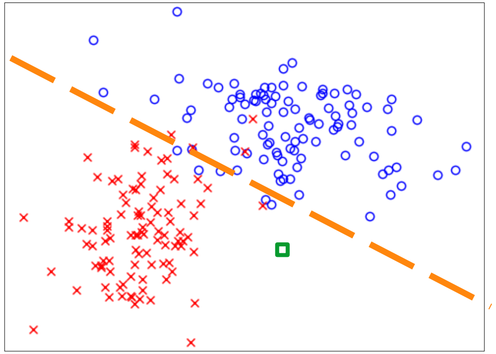
L'agent interagit avec l'environnement et déduit des « règles » (actions) permettant de maximiser une récompense.

Ex. d'application :
Obtenir un véhicule autonome

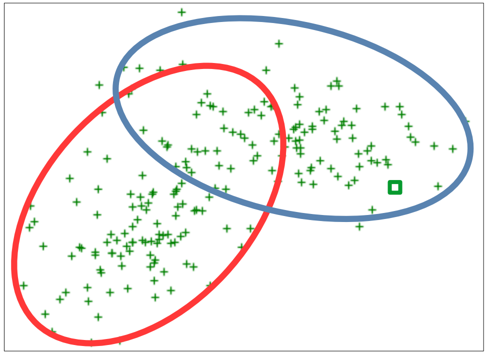
I)

Apprentissage automatique
(« Machine Learning »)

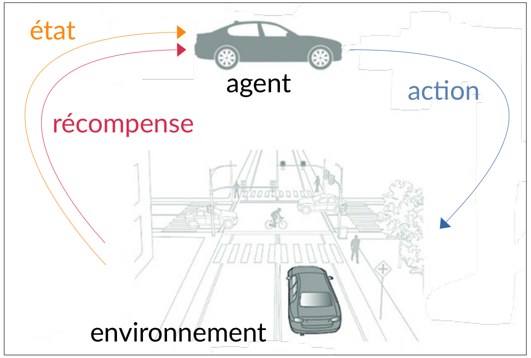
Supervisé



Non supervisé



Par renforcement



« Deep learning »

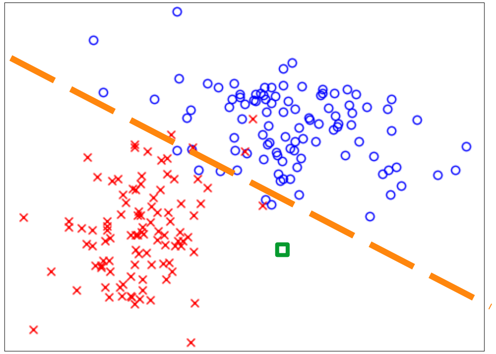
=

utilisation d'un réseau de neurones en apprentissage automatique

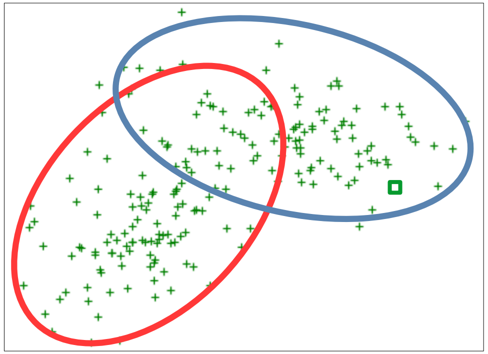
I)

Apprentissage automatique
(« Machine Learning »)

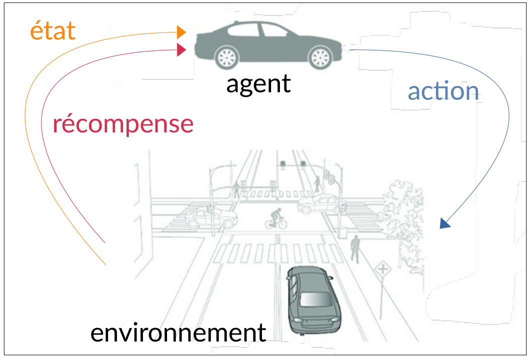
Supervisé



Non supervisé



Par renforcement



Aujourd'hui : le terme IA $\approx$ « Deep learning »

=

utilisation d'un réseau de neurones en apprentissage automatique

II) Apprentissage supervisé

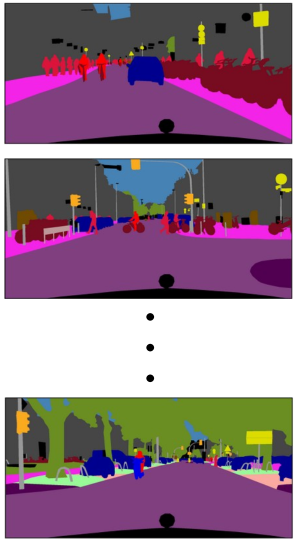
II)

Apprentissage supervisé

donnée 1 $\mathtt{X}_{\mathrm{train},1}$

donnée 2 $\mathtt{X}_{\mathrm{train},2}$

donnée N $\mathtt{X}_{\mathrm{train},N}$



$\mathtt{Y}_{\mathrm{train},1}$ étiquette 1

$\mathtt{Y}_{\mathrm{train},2}$ étiquette 2

$\mathtt{Y}_{\mathrm{train},N}$ étiquette N

Base de données étiquetées pour la *segmentation sémantique*

II)

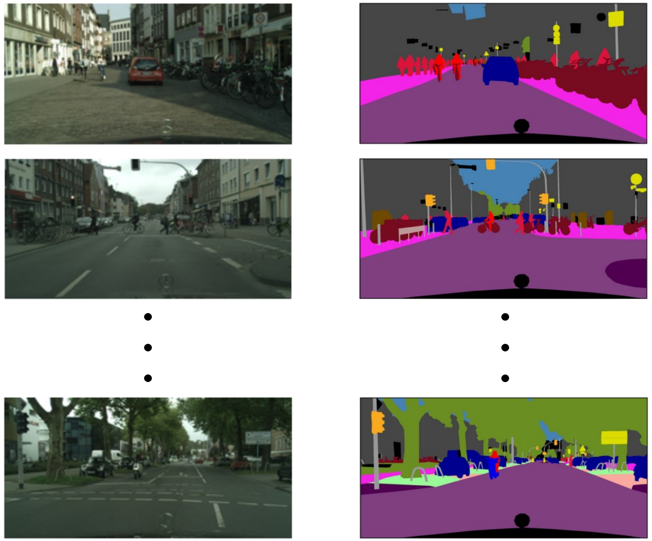
Apprentissage supervisé

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

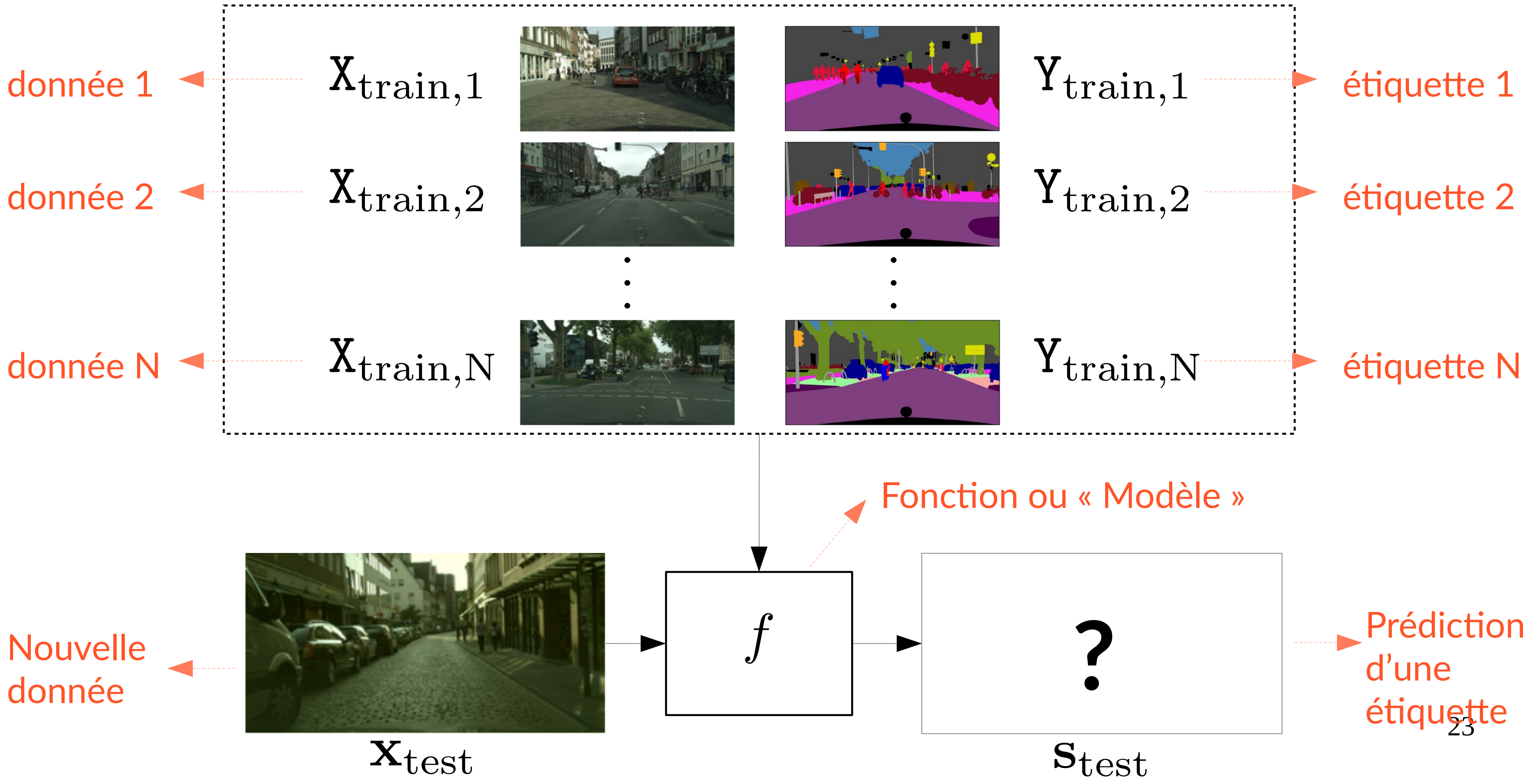


Base de données étiquetées pour la *segmentation sémantique*

Objectif : - déduire des « règles » de la base de données étiquetées
- appliquer ces « règles » à une nouvelle donnée pour prédire une étiquette

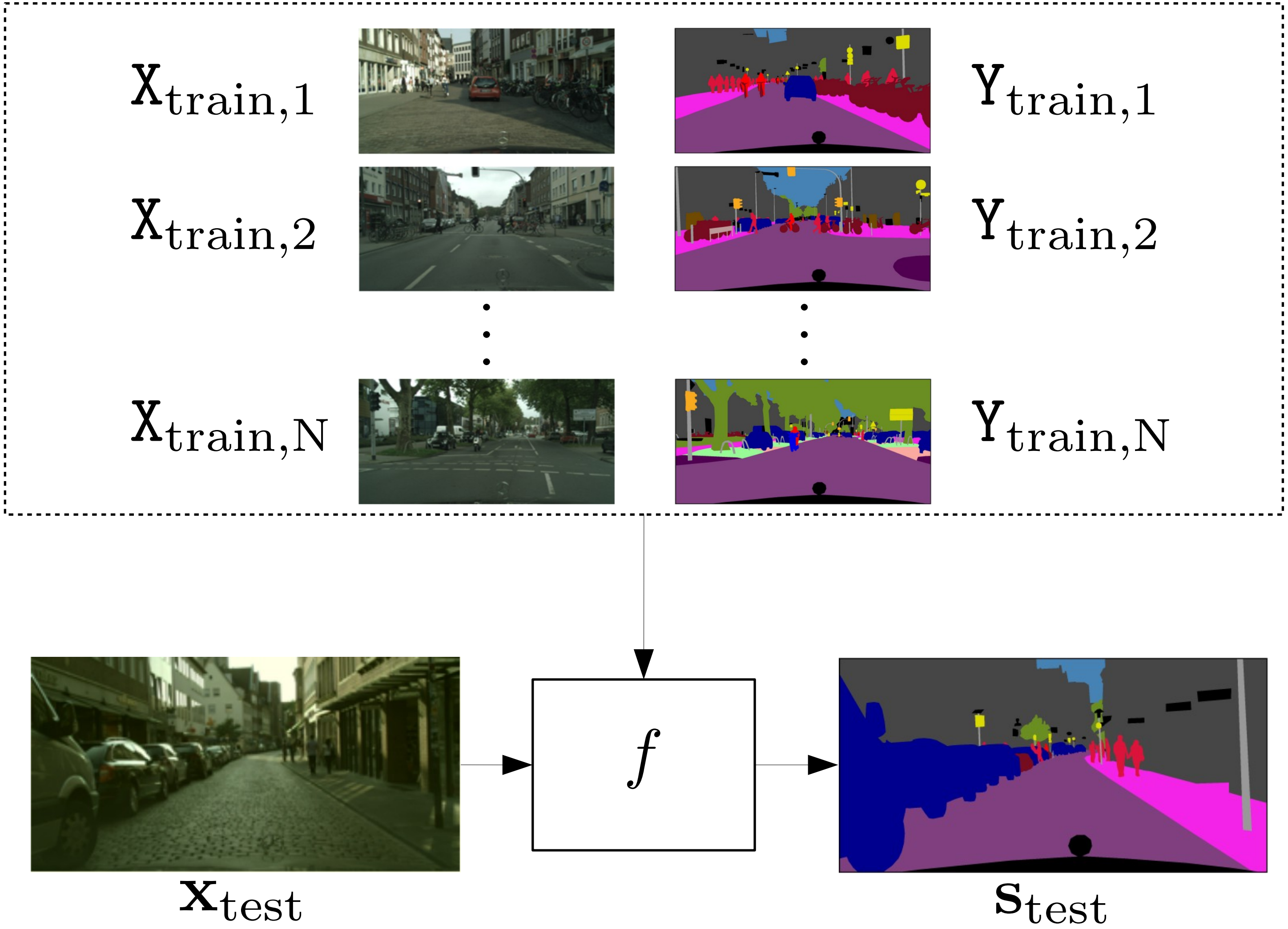
II)

Apprentissage supervisé



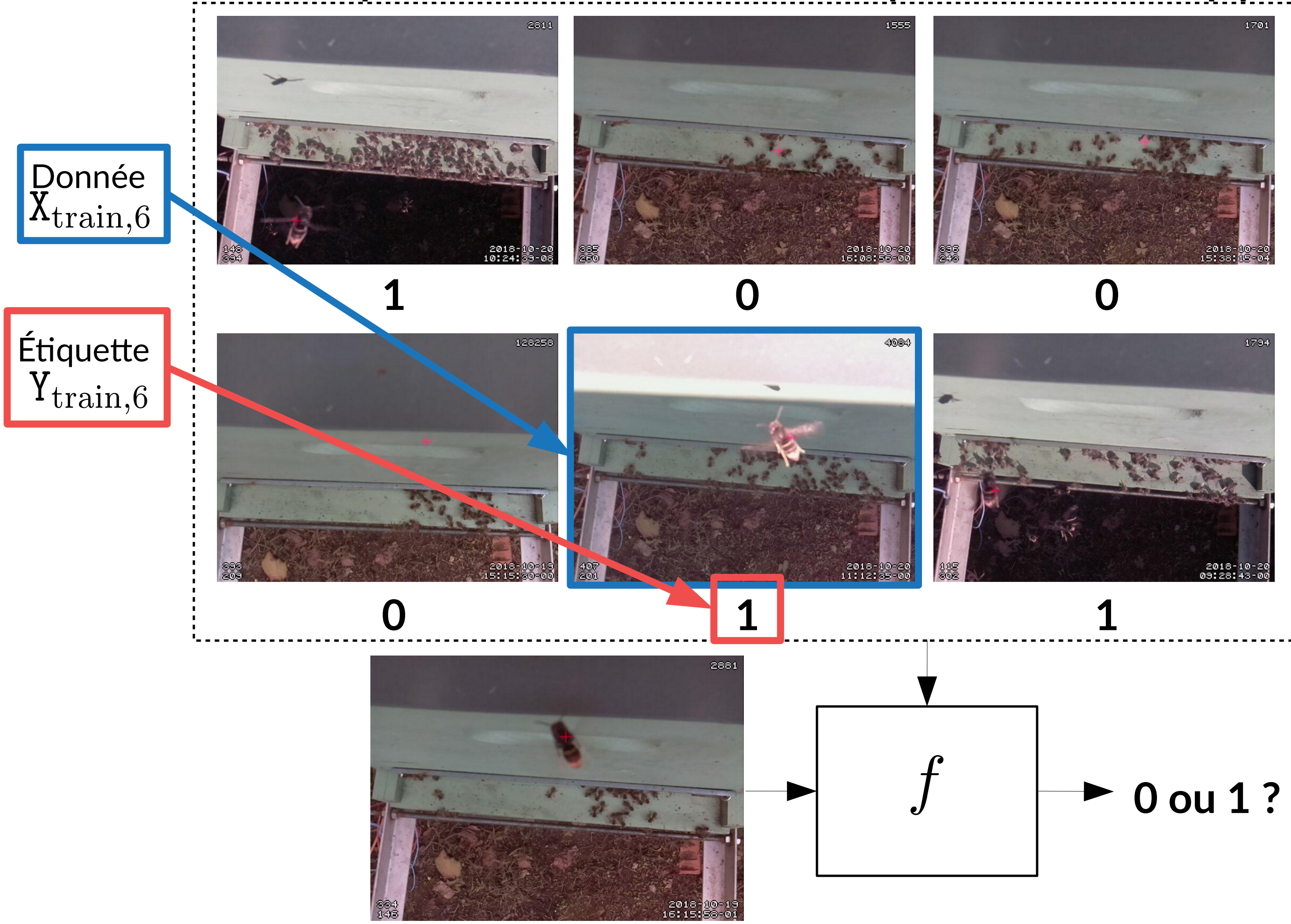
II)

Apprentissage supervisé



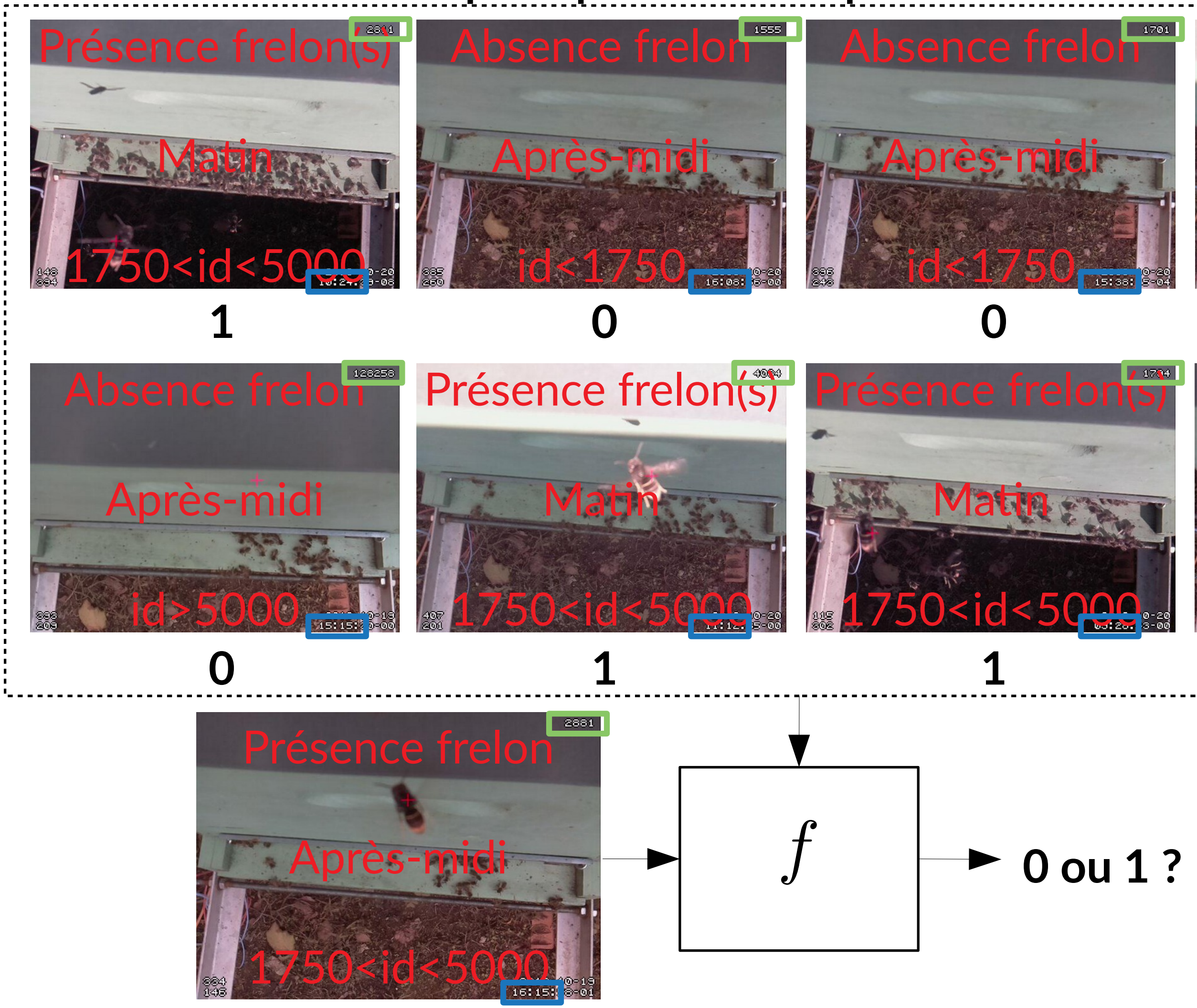
II)

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



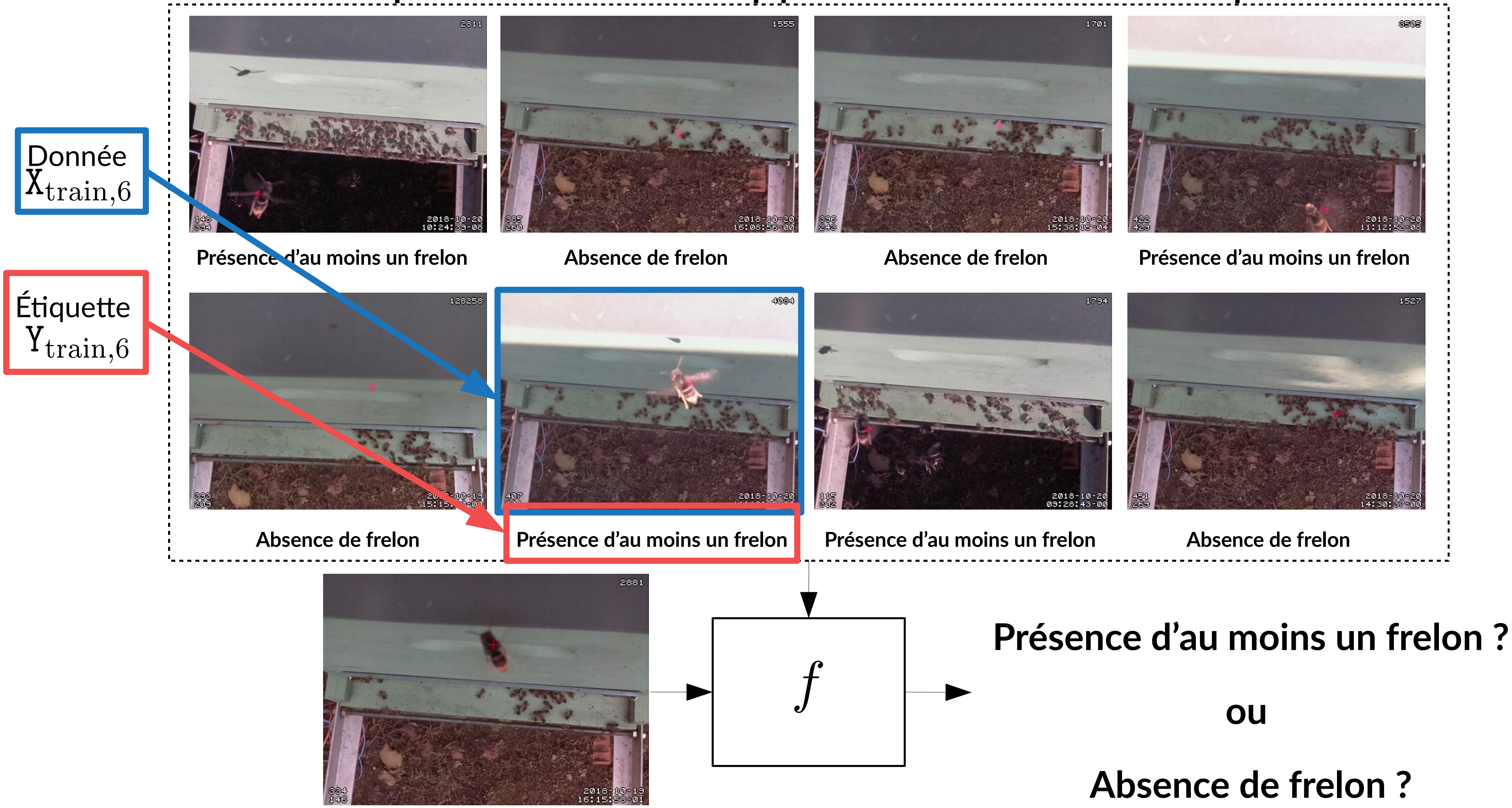
II)

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



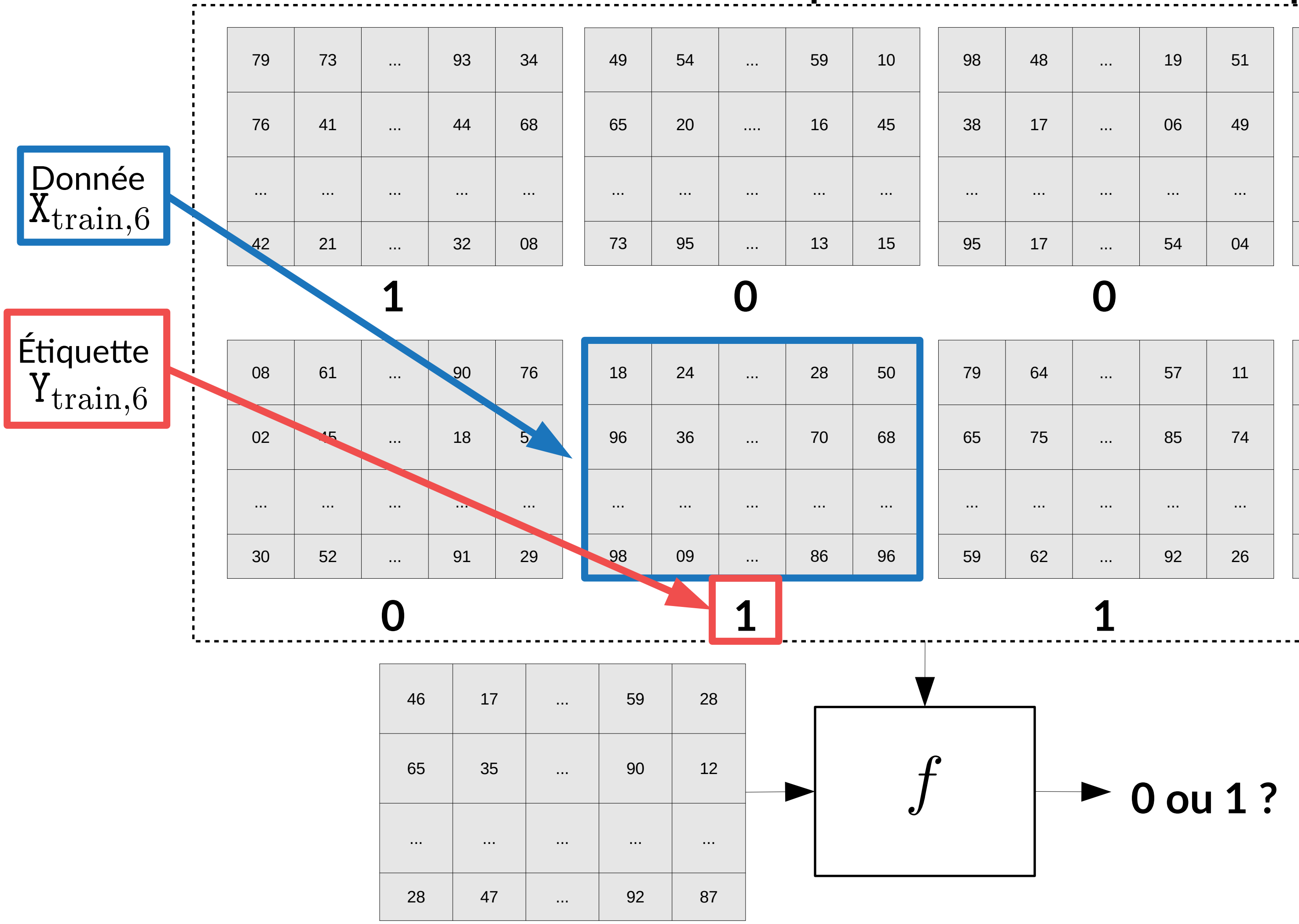
II)

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



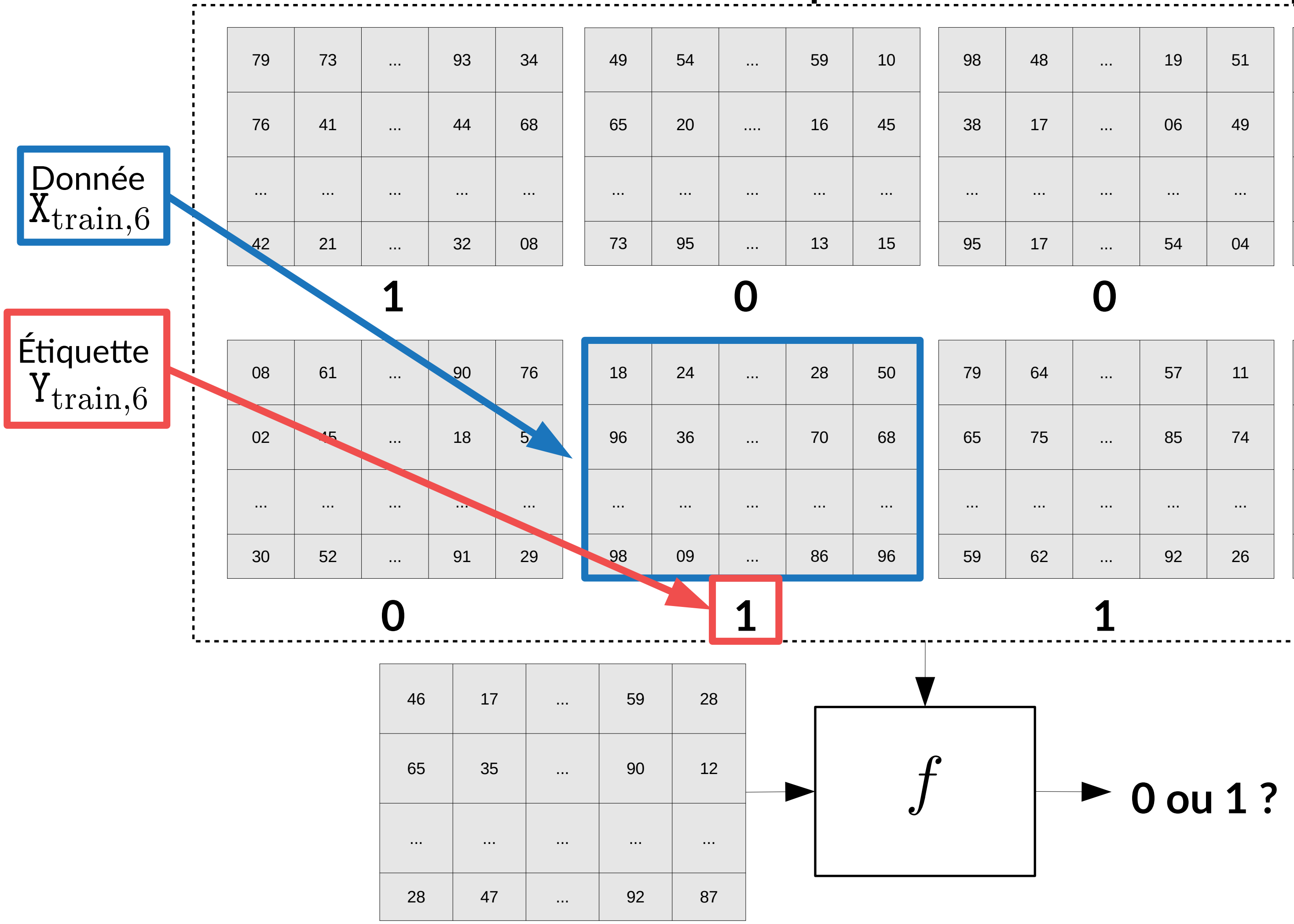
II)

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



II)

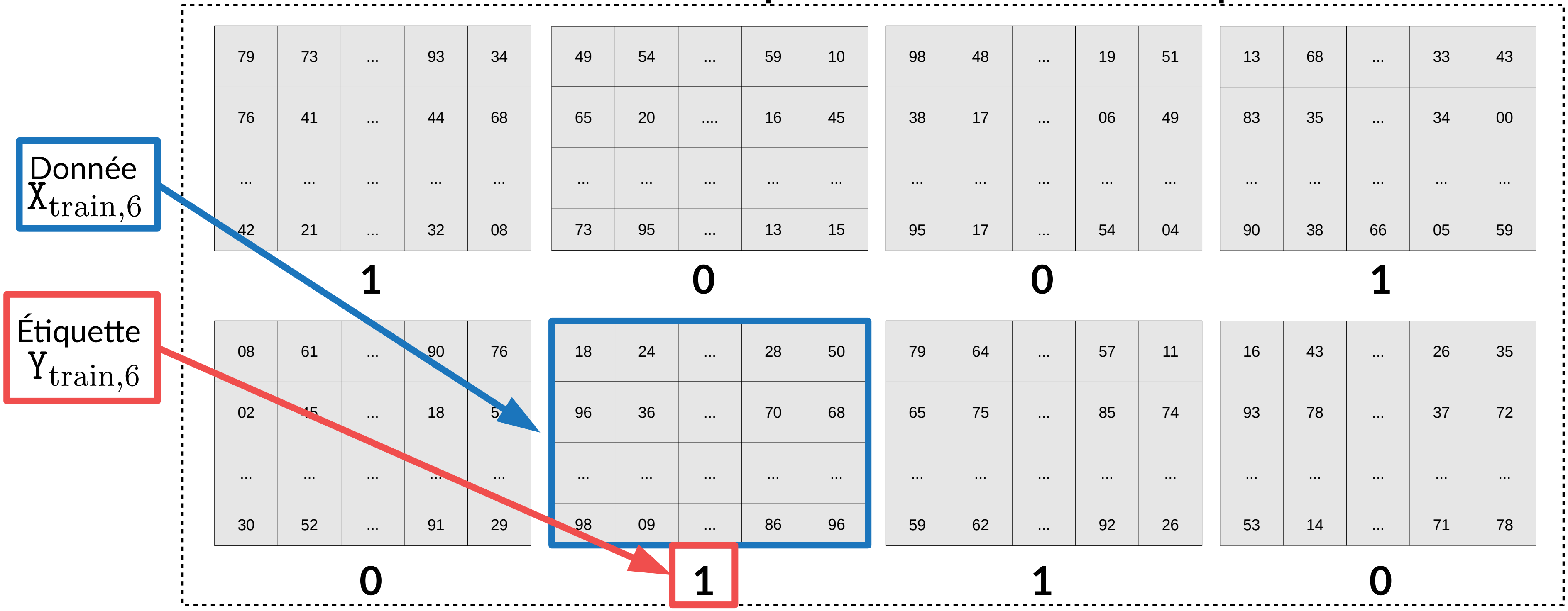
Mais une fois mis en forme pour l'ordinateur le problème devient

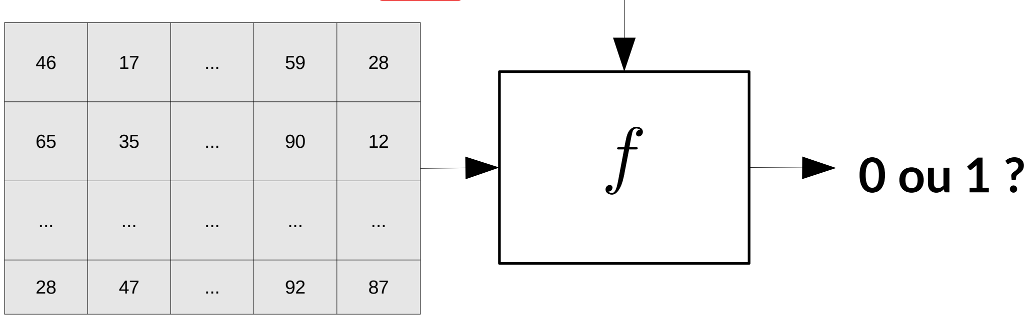


Les « **règles** » déduites ne seront (probablement) **pas interprétables.**
→ « Boîte noire »

II)

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





C'est nous qui **interprétons** :
« 0 » = présence frelon
« 1 » = absence frelon

II)

Résumé

- Des « règles » (souvent) non interprétables sont déduites de tableaux de valeurs numériques et d’étiquettes numériques.

II)

Résumé

- Des « règles » (souvent) non interprétables sont déduites de tableaux de valeurs numériques et d'étiquettes numériques.

- Ces « règles » sont appliquées à un nouveau tableau de valeurs numériques pour produire une étiquette numérique.

II)

Résumé

- Des « règles » (souvent) non interprétables sont déduites de tableaux de valeurs numériques et d’étiquettes numériques.

- Ces « règles » sont appliquées à un nouveau tableau de valeurs numériques pour produire une étiquette numérique.

- Nous interprétons cette étiquette numérique comme une étiquette « sémantique ».

II)

Résumé

- Des « règles » (souvent) non interprétables sont déduites de tableaux de valeurs numériques et d’étiquettes numériques.

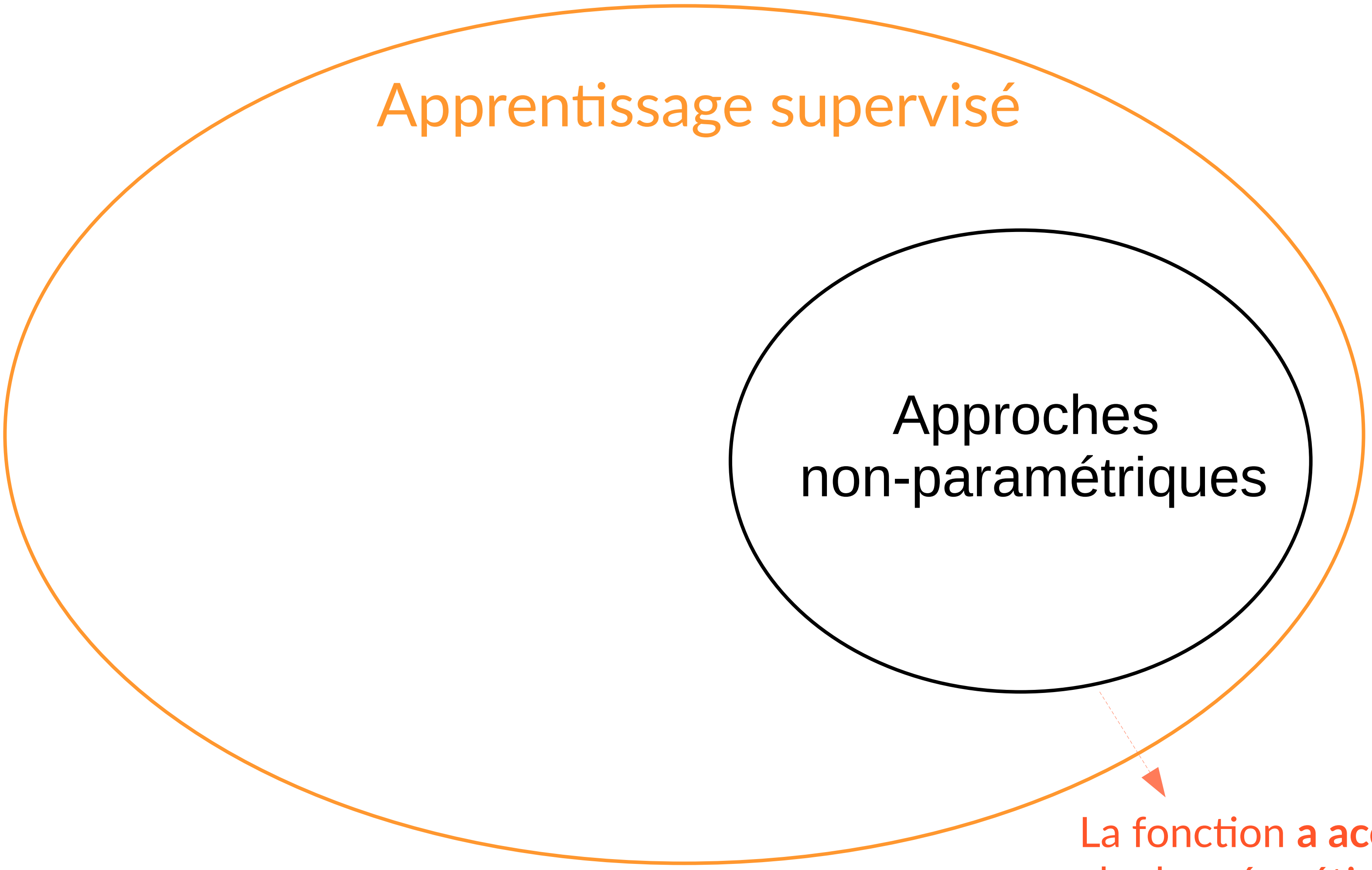
- Ces « règles » sont appliquées à un nouveau tableau de valeurs numériques pour produire une étiquette numérique.

- Nous interprétons cette étiquette numérique comme une étiquette « sémantique ».

- Il ne s’agit que d’une interprétation…

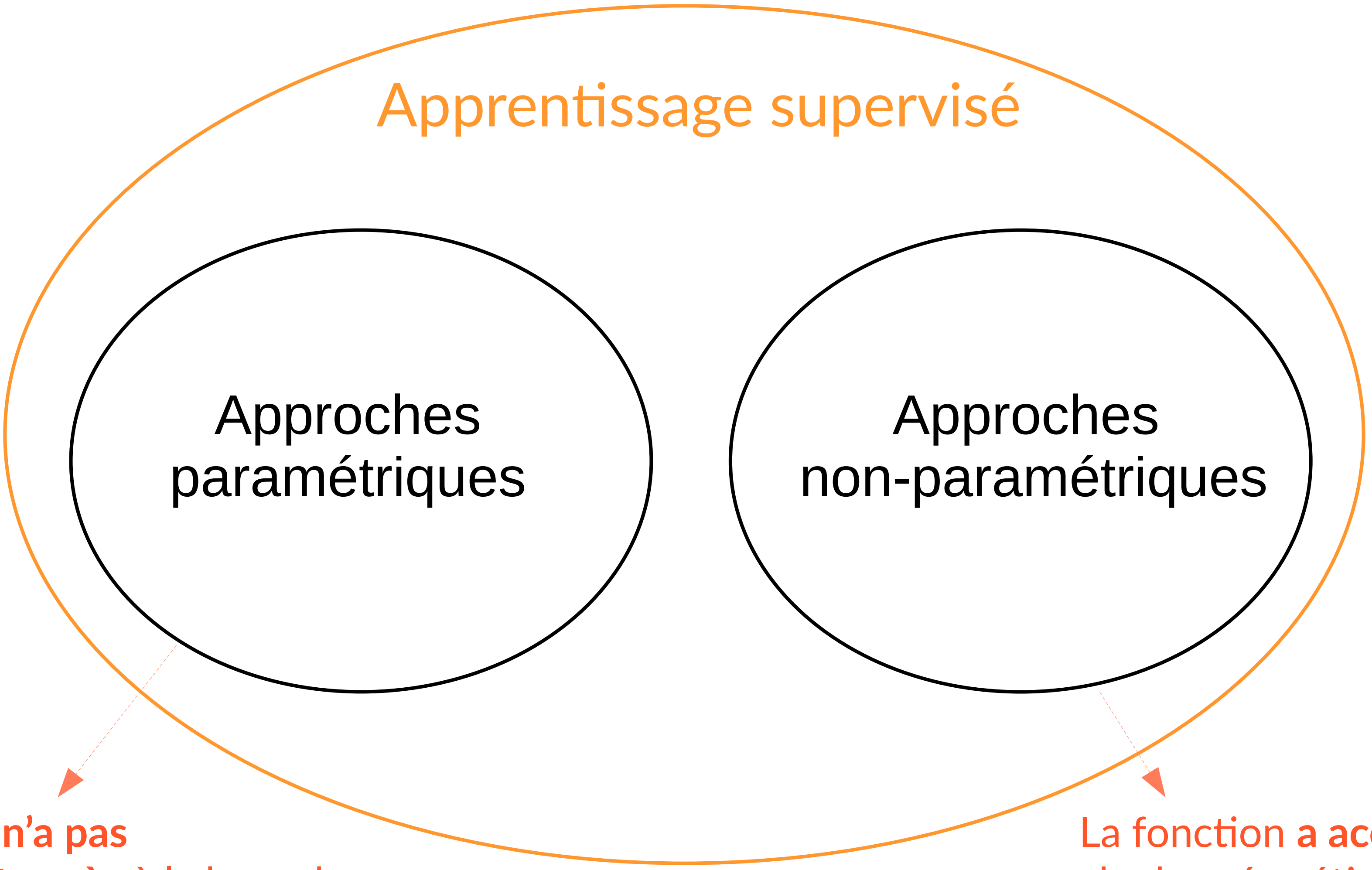
Tesla said autopilot was activated during a fatal Model X crash last week in California.

II)



La fonction **a accès** à la base de données étiquetées pour traiter une nouvelle donnée (ex : processus gaussien).

II)



La fonction **n'a pas directement accès** à la base de données étiquetées pour traiter une nouvelle donnée.

La fonction **a accès** à la base de données étiquetées pour traiter une nouvelle donnée (ex : processus gaussien).

III) Approches paramétriques

III)

Représentation d'une fonction paramétrique

Paramètres

Mathématique $$\mathbf{s} = f\left(\mathbf{x}; \boldsymbol{\theta}\right)$$

III)

Représentation d’une fonction paramétrique

Paramètres

Mathématique

$$\mathbf{s} = f\left(\mathbf{x}; \boldsymbol{\theta}\right)$$

Informatique
(Python)

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

III)

Représentation d'une fonction paramétrique

Paramètres

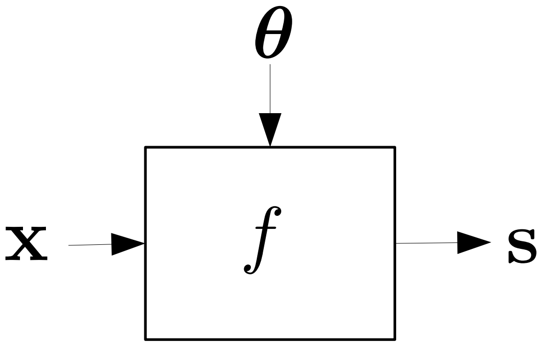
Mathématique

$$\mathbf{s} = f\left(\mathbf{x}; \boldsymbol{\theta}\right)$$

Informatique
(Python)

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

Graphique
(Graphe de calcul)



III)

Exemple de fonction paramétrique : transformation affine

Mathématique

$$\mathbf{s} = f\left(\mathbf{x}; \boldsymbol{\theta} = \{\mathtt{W}, \mathbf{b}\}\right) = \mathtt{W}\mathbf{x} + \mathbf{b}$$

III)

Exemple de fonction paramétrique : transformation affine

Mathématique

$$\mathbf{s} = f\left(\mathbf{x}; \boldsymbol{\theta} = \{\mathtt{W}, \mathbf{b}\}\right) = \mathtt{W}\mathbf{x} + \mathbf{b}$$

Informatique
(Python)

```python
def f(x, W, b):
    s = x @ W +b
    return s
```

III) Exemple de fonction paramétrique : transformation affine

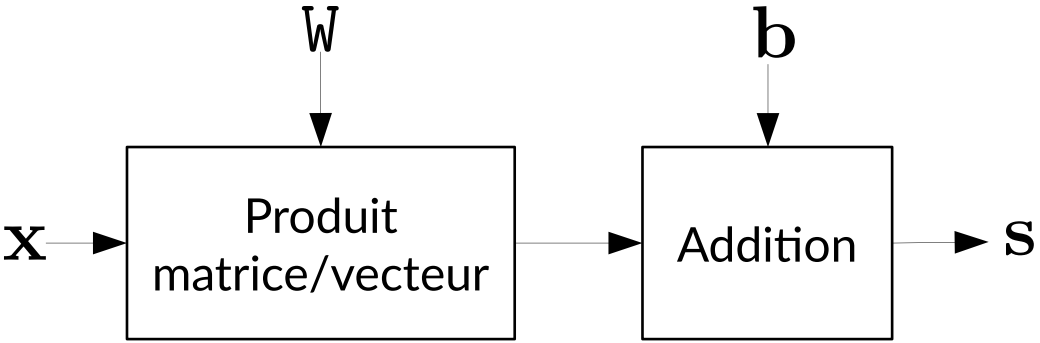
Mathématique

$$\mathbf{s} = f\left(\mathbf{x}; \boldsymbol{\theta} = \{\mathtt{W}, \mathbf{b}\}\right) = \mathtt{W}\mathbf{x} + \mathbf{b}$$

Informatique
(Python)

```python
def f(x, W, b):
    s = x @ W +b
    return s
```

Graphique
(Graphe de calcul)



III)

Exemple de fonction paramétrique : modèle polynomial

Mathématique $$s = f(x;\boldsymbol{\theta}) = \sum_{i=0}^{d} \boldsymbol{\theta}_i x^i = \boldsymbol{\theta}^\top \phi(x)$$ où $$\phi(x) = \begin{bmatrix} 1 & x & \dots & x^d \end{bmatrix}^\top$$

III)

Exemple de fonction paramétrique : modèle polynomial

Mathématique $s = f(x; \boldsymbol{\theta}) = \sum_{i=0}^{d} \boldsymbol{\theta}_i x^i = \boldsymbol{\theta}^\top \phi(x)$ où $\phi(x) = \begin{bmatrix} 1 & x & \dots & x^d \end{bmatrix}^\top$

Informatique (Python)

```python
def poly(x, theta):
    phi_x = phi(x)
    s = phi_x @ theta
    return s
```

III)

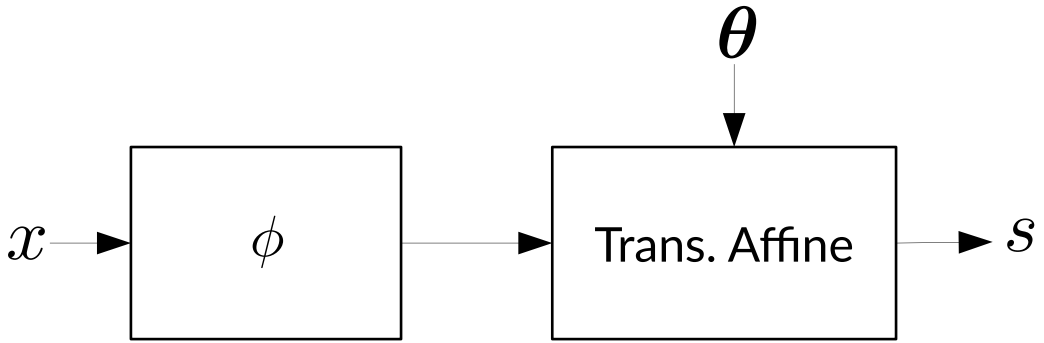
Exemple de fonction paramétrique : modèle polynomial

Mathématique $s = f\left(x;\boldsymbol{\theta}\right) = \sum_{i=0}^{d} \boldsymbol{\theta}_i x^i = \boldsymbol{\theta}^\top \phi\left(x\right)$ où $\phi\left(x\right) = \begin{bmatrix} 1 & x & \dots & x^d \end{bmatrix}^\top$

Informatique
(Python)

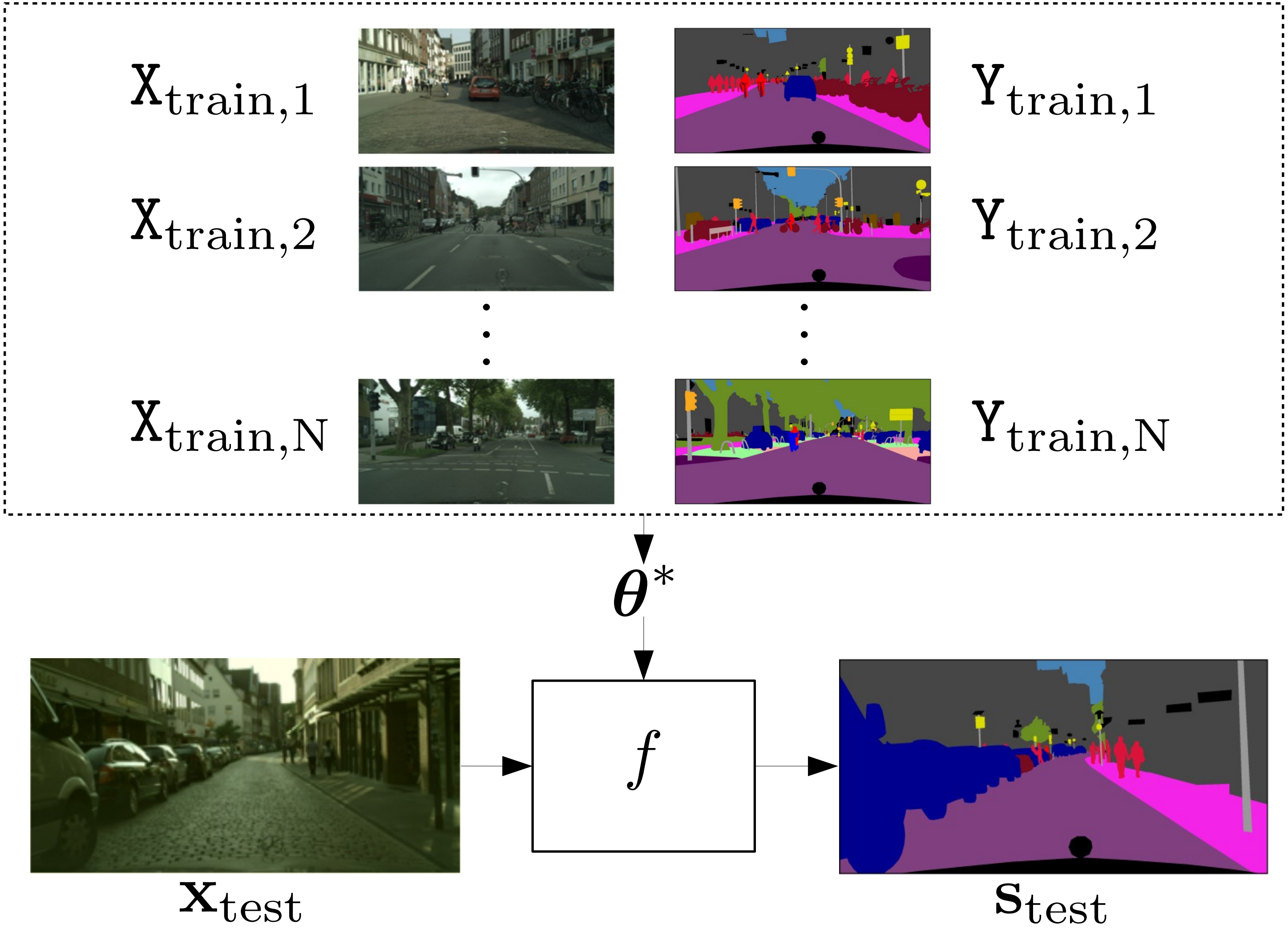
```python
def poly(x, theta):
    phi_x = phi(x)
    s = phi_x @ theta
    return s
```

Graphique
(Graphe de calcul)



III)

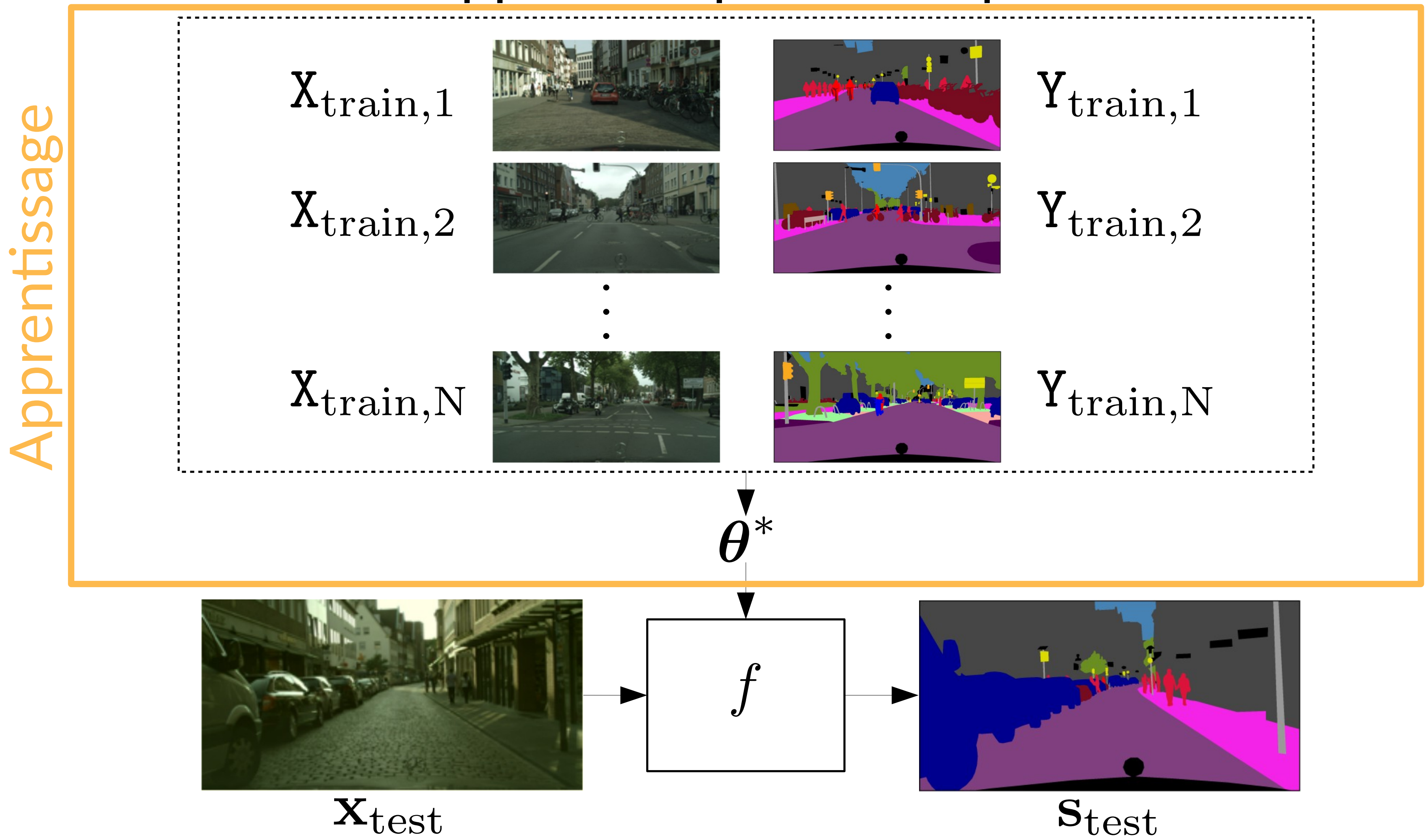
Approches paramétriques



Les « règles » déduites de la base de données étiquetées sont (en partie) encodées dans la valeur des paramètres $\boldsymbol{\theta}^*$.

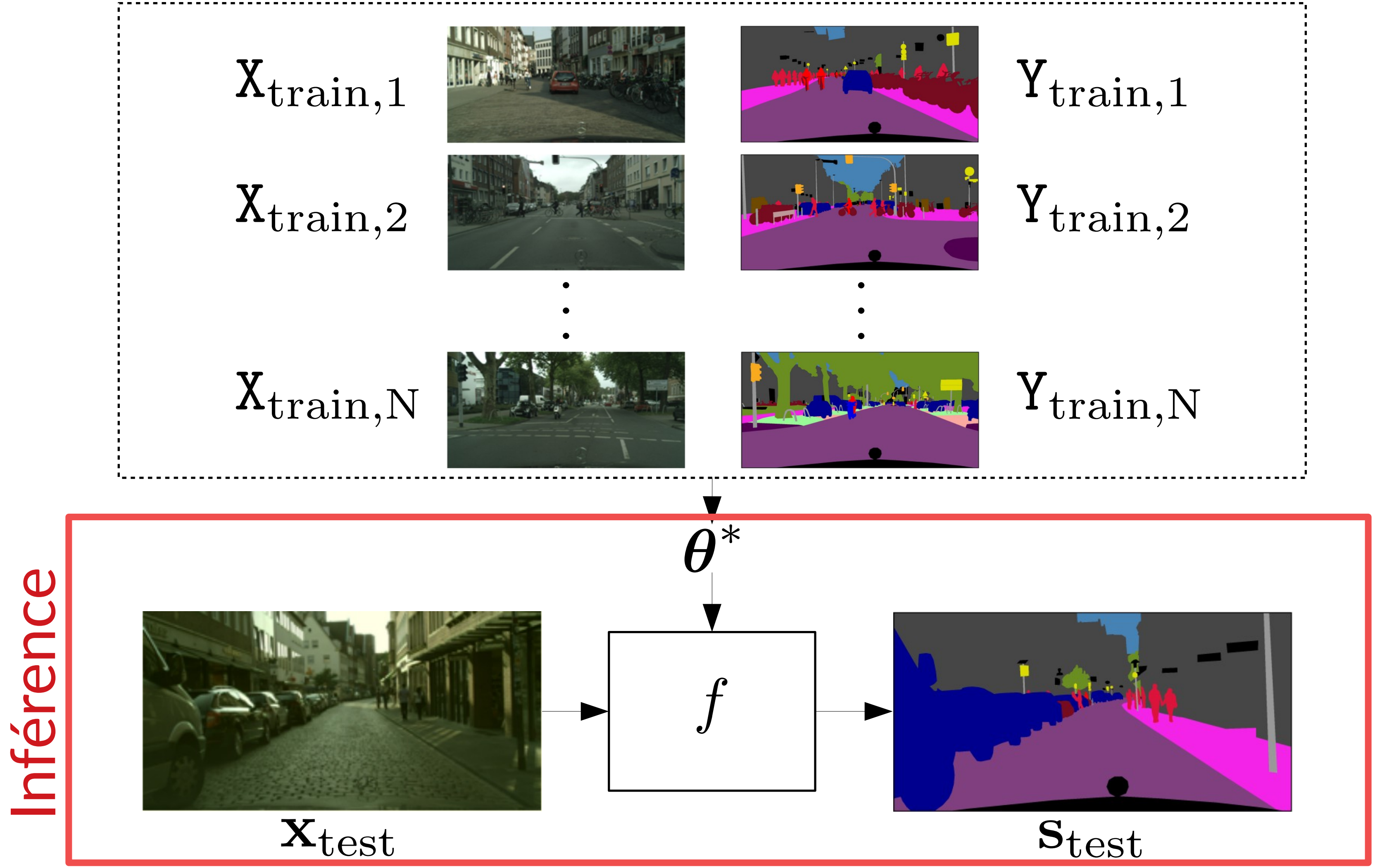
III)

Approches paramétriques



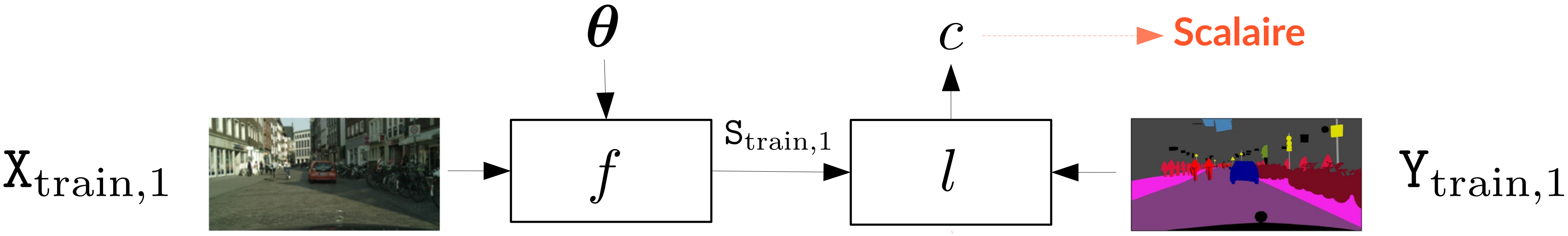
III)

Approches paramétriques



III)

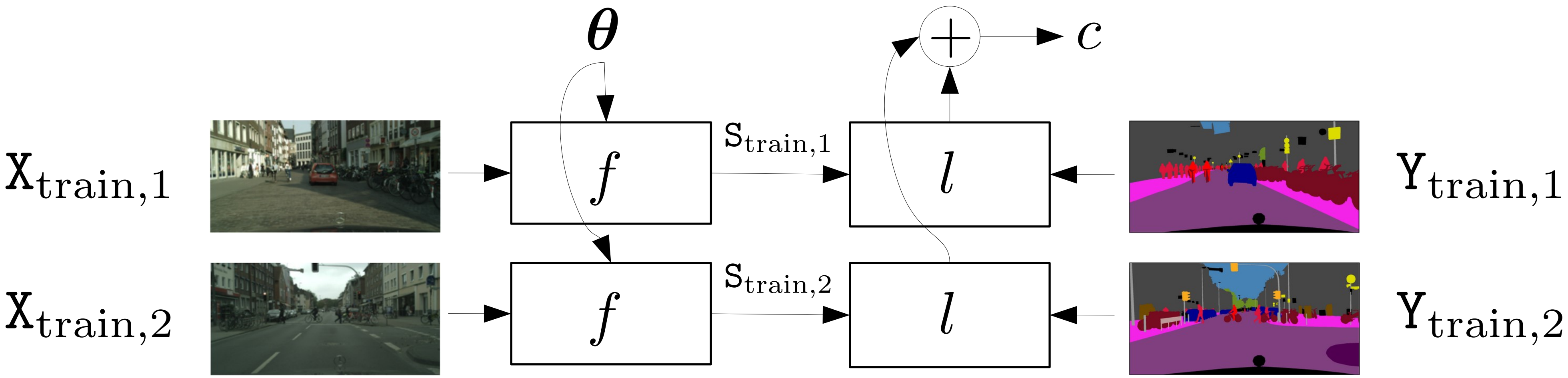
Étape d'apprentissage ("Training time")



Fonction de coût (« loss function ») : permet de comparer la prédiction du réseau à l'étiquette.

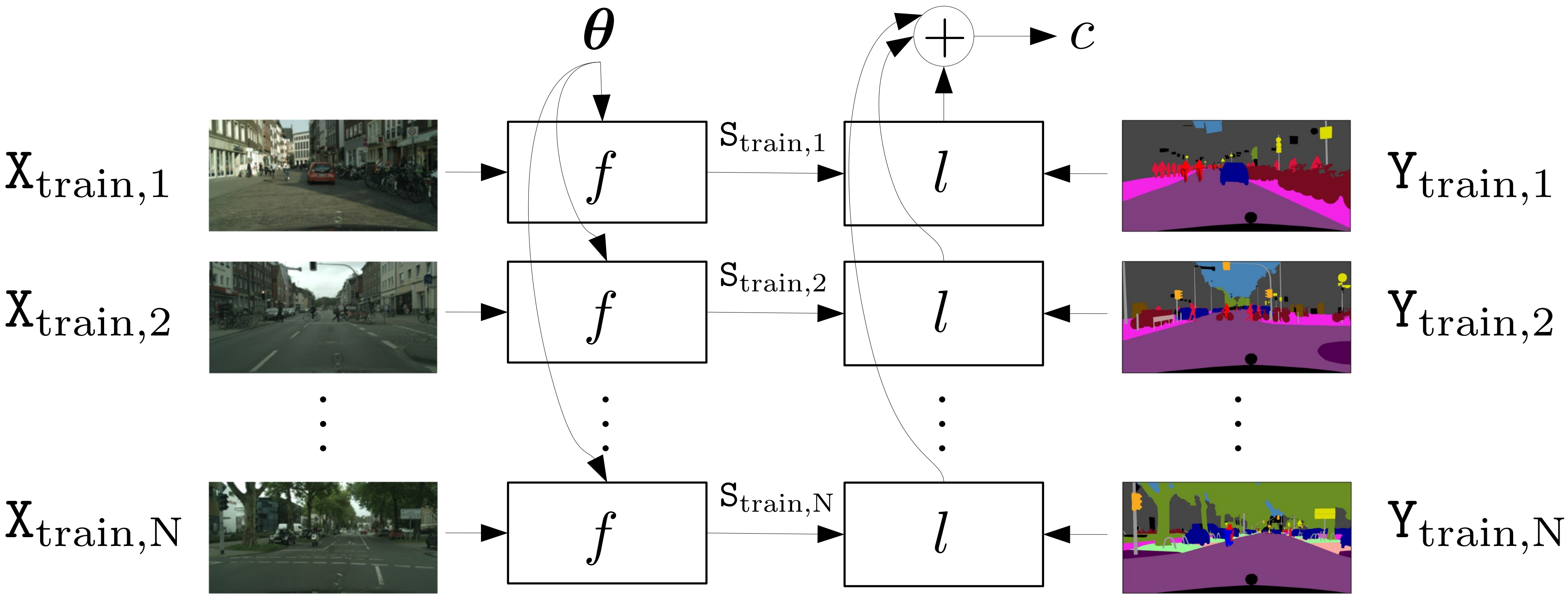
III)

Étape d'apprentissage ("Training time")



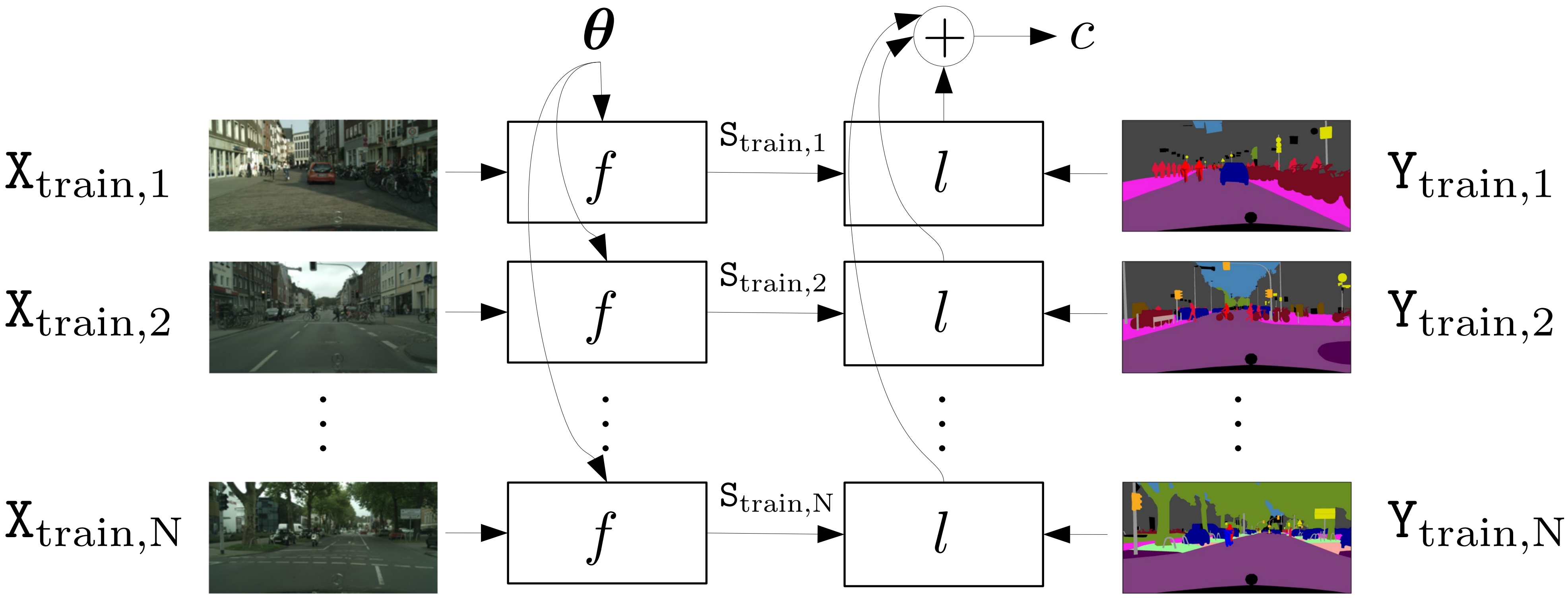
III)

Étape d'apprentissage ("Training time")



III)

Étape d'apprentissage (“Training time”)

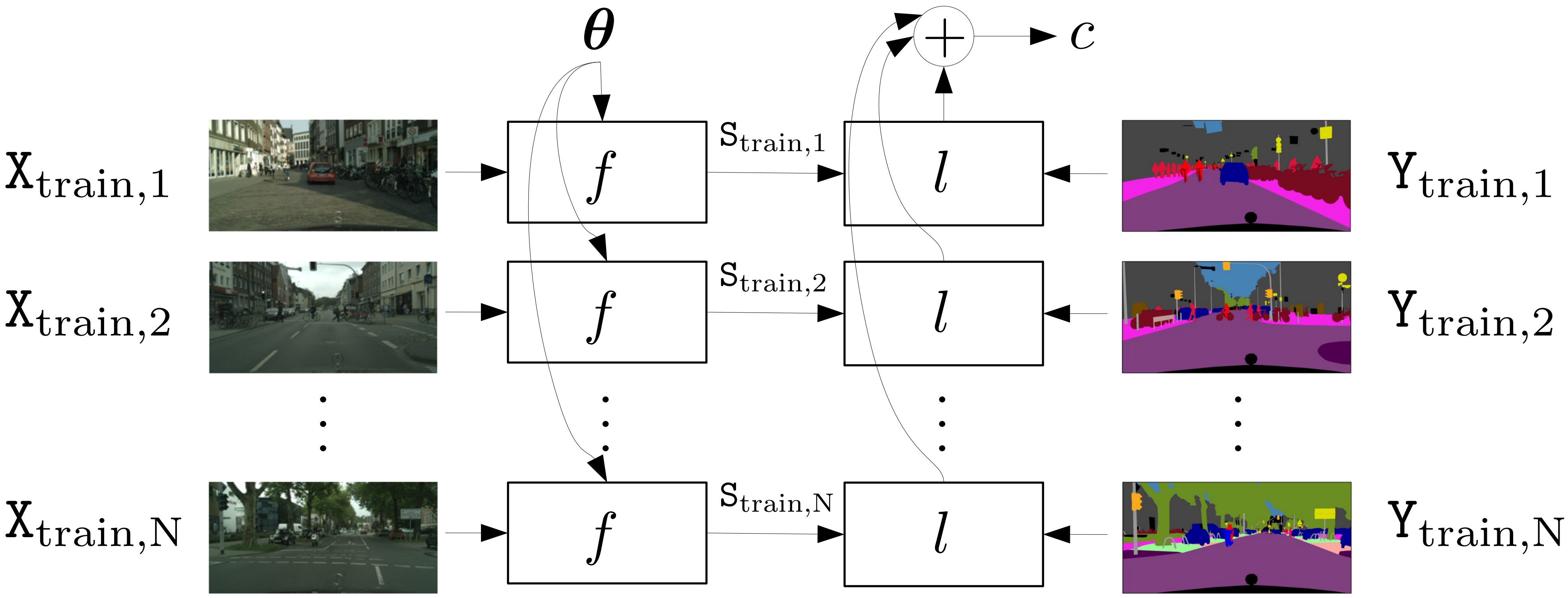


Coût d'entraînement
(« Training loss »)

$$\sum_{i=1}^{N} l\left(\mathtt{Y}_{\text{train},i}, f\left(\mathtt{X}_{\text{train},i}; \boldsymbol{\theta}\right)\right)$$

III)

Étape d'apprentissage ("Training time")

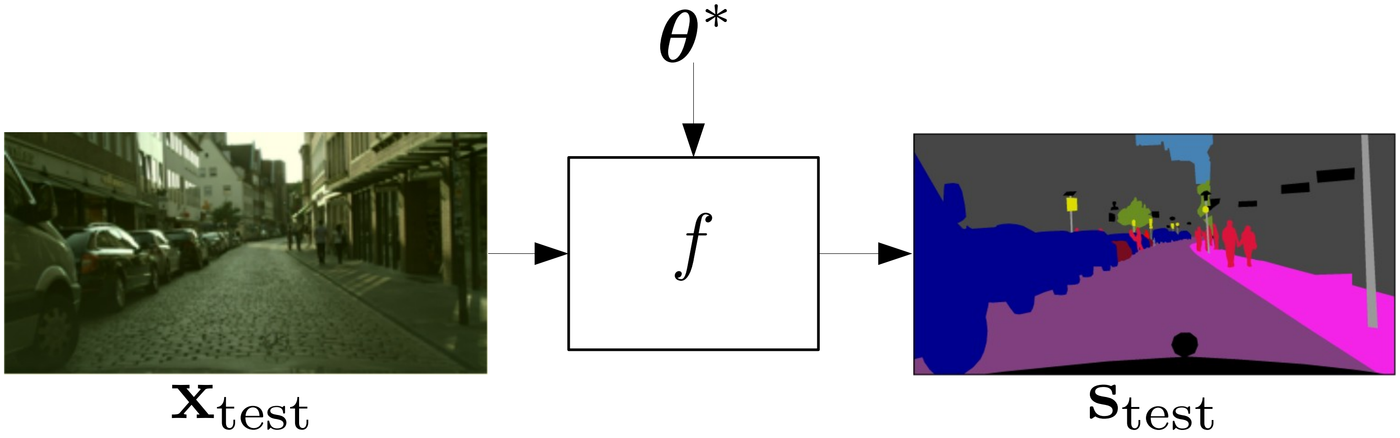


Problème d'optimisation

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{N} l\left(\mathtt{Y}_{\mathtt{train},i}, f\left(\mathtt{X}_{\mathtt{train},i}; \boldsymbol{\theta}\right)\right)$$

III)

Étape d’inférence (“Test time”)

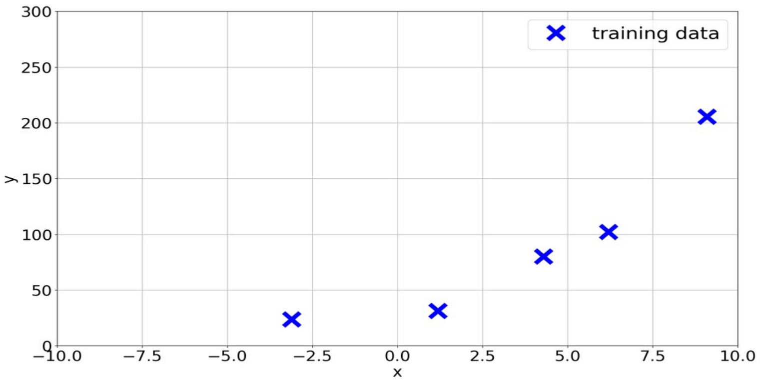


$$\mathbf{s}_{\text{test}} = f\left(\mathbf{x}_{\text{test}};\, \boldsymbol{\theta}^*\right)$$

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

$\mathtt{X}_{\text{train},1} = -3.1$ $\mathtt{X}_{\text{train},2} = 1.2$ $\mathtt{X}_{\text{train},3} = 4.3$ $\mathtt{X}_{\text{train},4} = 6.2$ $\mathtt{X}_{\text{train},5} = 9.1$

$\mathtt{Y}_{\text{train},1} = 23.7$ $\mathtt{Y}_{\text{train},2} = 31.3$ $\mathtt{Y}_{\text{train},3} = 79.9$ $\mathtt{Y}_{\text{train},4} = 101.9$ $\mathtt{Y}_{\text{train},5} = 205.5$



III)

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

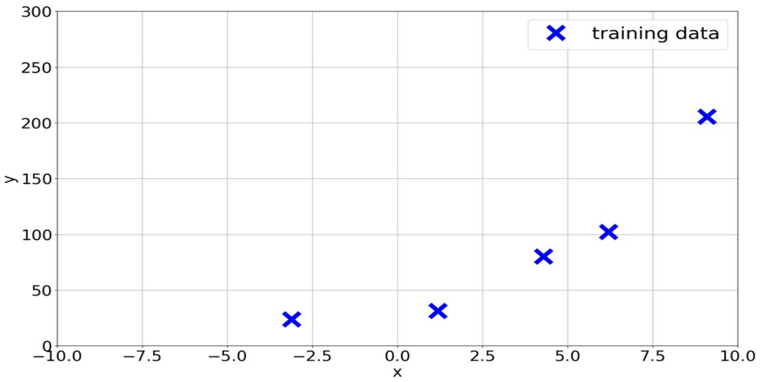
$\mathtt{X}_{\text{train},1} = -3.1 \quad \mathtt{X}_{\text{train},2} = 1.2 \quad \mathtt{X}_{\text{train},3} = 4.3 \quad \mathtt{X}_{\text{train},4} = 6.2 \quad \mathtt{X}_{\text{train},5} = 9.1$

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

Choix de la fonction $\quad f(x; \boldsymbol{\theta}) = \sum_{i=0}^{d} \boldsymbol{\theta}_i x^i = \boldsymbol{\theta}^\top \phi(x)$

où $\quad \phi(x) = \begin{bmatrix} 1 & x & \dots & x^d \end{bmatrix}^\top$

hyper-paramètre : d ► **degré du polynôme**



III)

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

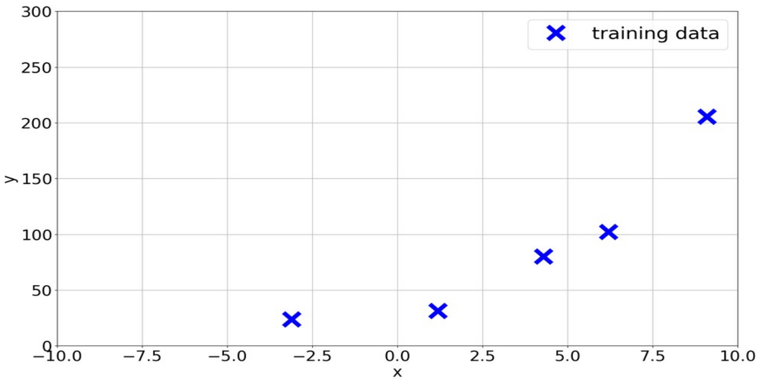
$$\mathtt{X}_{\text{train},1} = -3.1 \quad \mathtt{X}_{\text{train},2} = 1.2 \quad \mathtt{X}_{\text{train},3} = 4.3 \quad \mathtt{X}_{\text{train},4} = 6.2 \quad \mathtt{X}_{\text{train},5} = 9.1$$

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

Choix de la fonction $f(x;\boldsymbol{\theta}) = \sum_{i=0}^{d} \boldsymbol{\theta}_i x^i = \boldsymbol{\theta}^\top \phi(x)$

où $\phi(x) = \begin{bmatrix} 1 & x & \ldots & x^d \end{bmatrix}^\top$

hyper-paramètre : d ► **degré du polynôme**



Quel est le nombre de paramètres du « modèle » ?

III)

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

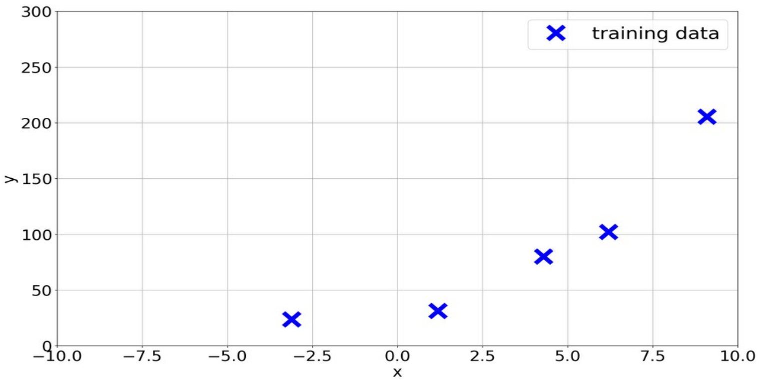
$\mathtt{X}_{\text{train},1} = -3.1$ $\mathtt{X}_{\text{train},2} = 1.2$ $\mathtt{X}_{\text{train},3} = 4.3$ $\mathtt{X}_{\text{train},4} = 6.2$ $\mathtt{X}_{\text{train},5} = 9.1$

$\mathtt{Y}_{\text{train},1} = 23.7$ $\mathtt{Y}_{\text{train},2} = 31.3$ $\mathtt{Y}_{\text{train},3} = 79.9$ $\mathtt{Y}_{\text{train},4} = 101.9$ $\mathtt{Y}_{\text{train},5} = 205.5$

Choix de la fonction $f(x;\boldsymbol{\theta}) = \sum_{i=0}^{d} \boldsymbol{\theta}_i x^i = \boldsymbol{\theta}^\top \phi(x)$

où $\phi(x) = \begin{bmatrix} 1 & x & \dots & x^d \end{bmatrix}^\top$

hyper-paramètre : d ► **degré du polynôme**



Apprentissage

Choix du coût $l(y,s) = (y-s)^2$

III)

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

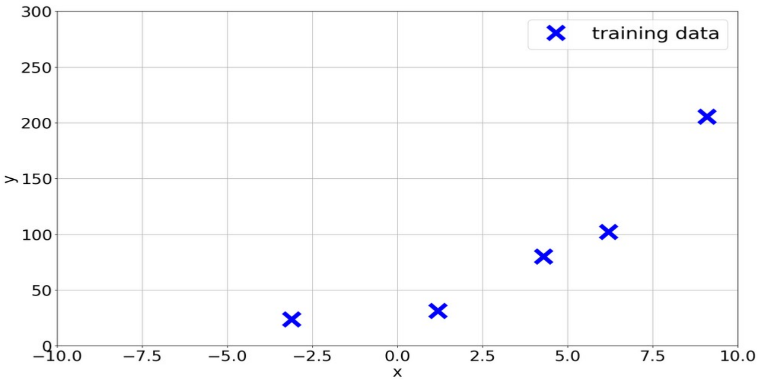
$\mathtt{X}_{\mathrm{train},1} = -3.1$ $\mathtt{X}_{\mathrm{train},2} = 1.2$ $\mathtt{X}_{\mathrm{train},3} = 4.3$ $\mathtt{X}_{\mathrm{train},4} = 6.2$ $\mathtt{X}_{\mathrm{train},5} = 9.1$

$\mathtt{Y}_{\mathrm{train},1} = 23.7$ $\mathtt{Y}_{\mathrm{train},2} = 31.3$ $\mathtt{Y}_{\mathrm{train},3} = 79.9$ $\mathtt{Y}_{\mathrm{train},4} = 101.9$ $\mathtt{Y}_{\mathrm{train},5} = 205.5$

Choix de la fonction $$f(x;\boldsymbol{\theta}) = \sum_{i=0}^{d} \boldsymbol{\theta}_i x^i = \boldsymbol{\theta}^\top \phi(x)$$

où $$\phi(x) = \begin{bmatrix}1 & x & \dots & x^d\end{bmatrix}^\top$$

hyper-paramètre : d → **degré du polynôme**



Apprentissage

Choix du coût $$l(y,s) = (y-s)^2$$

Optimisation $$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{5} \left(\mathtt{Y}_{\mathrm{train},i} - \boldsymbol{\theta}^\top \phi(\mathtt{X}_{\mathrm{train},i})\right)^2$$ → **Moindres carrés linéaires**

III)

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

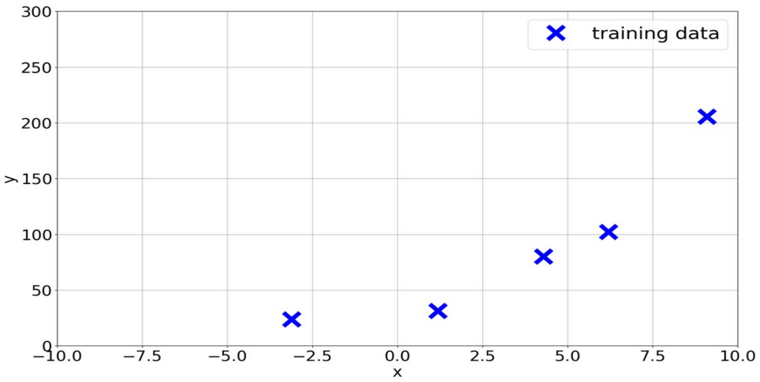
$\mathtt{X}_{\mathrm{train},1} = -3.1$ $\mathtt{X}_{\mathrm{train},2} = 1.2$ $\mathtt{X}_{\mathrm{train},3} = 4.3$ $\mathtt{X}_{\mathrm{train},4} = 6.2$ $\mathtt{X}_{\mathrm{train},5} = 9.1$

$\mathtt{Y}_{\mathrm{train},1} = 23.7$ $\mathtt{Y}_{\mathrm{train},2} = 31.3$ $\mathtt{Y}_{\mathrm{train},3} = 79.9$ $\mathtt{Y}_{\mathrm{train},4} = 101.9$ $\mathtt{Y}_{\mathrm{train},5} = 205.5$

Choix de la fonction $$f(x;\boldsymbol{\theta}) = \sum_{i=0}^{d} \boldsymbol{\theta}_i x^i = \boldsymbol{\theta}^\top \phi(x)$$

où $\phi(x) = \begin{bmatrix} 1 & x & \dots & x^d \end{bmatrix}^\top$

hyper-paramètre : d ► **degré du polynôme**



Apprentissage

Choix du coût $l(y,s) = (y-s)^2$

Optimisation $$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{5} \left(\mathtt{Y}_{\mathrm{train},i} - \boldsymbol{\theta}^\top \phi(\mathtt{X}_{\mathrm{train},i})\right)^2$$ ► **Moindres carrés linéaires**

Inférence $$s_{\mathrm{test}} = f(x;\boldsymbol{\theta}^*) = {\boldsymbol{\theta}^*}^\top \phi(x_{\mathrm{test}})$$

III)

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

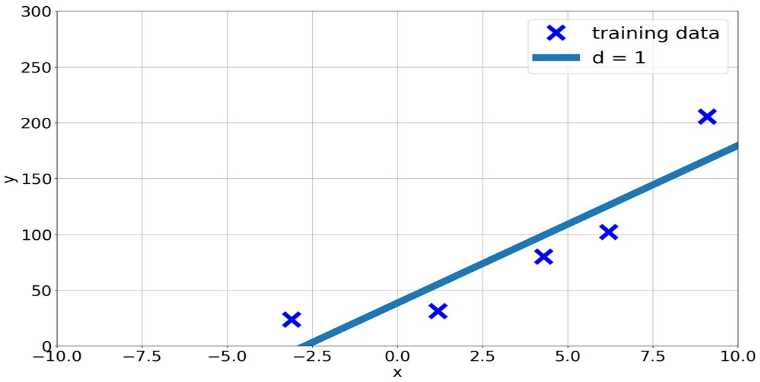
$\mathtt{X}_{\mathtt{train},1} = -3.1$ $\quad \mathtt{X}_{\mathtt{train},2} = 1.2$ $\quad \mathtt{X}_{\mathtt{train},3} = 4.3$ $\quad \mathtt{X}_{\mathtt{train},4} = 6.2$ $\quad \mathtt{X}_{\mathtt{train},5} = 9.1$

$\mathtt{Y}_{\mathtt{train},1} = 23.7$ $\quad \mathtt{Y}_{\mathtt{train},2} = 31.3$ $\quad \mathtt{Y}_{\mathtt{train},3} = 79.9$ $\quad \mathtt{Y}_{\mathtt{train},4} = 101.9$ $\quad \mathtt{Y}_{\mathtt{train},5} = 205.5$

Choix de la fonction $\quad f(x;\boldsymbol{\theta}) = \sum_{i=0}^{d} \boldsymbol{\theta}_i x^i = \boldsymbol{\theta}^\top \phi(x)$

où $\quad \phi(x) = \begin{bmatrix} 1 & x & \dots & x^d \end{bmatrix}^\top$

hyper-paramètre : d → **degré du polynôme**



Apprentissage

Choix du coût $\quad l(y,o) = (y-o)^2$

Optimisation $\quad \boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{5} \left(\mathtt{Y}_{\mathtt{train},i} - \boldsymbol{\theta}^\top \phi\left(\mathtt{X}_{\mathtt{train},i}\right) \right)^2$ → **Moindres carrés linéaires**

Inférence $\quad s_{\text{test}} = f(x;\boldsymbol{\theta}^*) = {\boldsymbol{\theta}^*}^\top \phi(x_{\text{test}})$

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

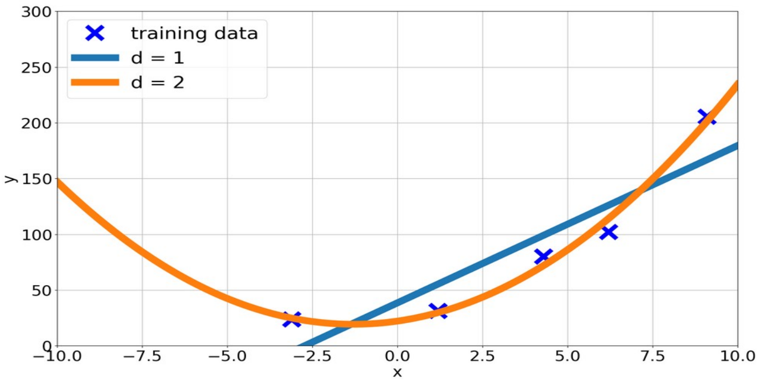
$\mathtt{X}_{\mathrm{train},1} = -3.1$ $\mathtt{X}_{\mathrm{train},2} = 1.2$ $\mathtt{X}_{\mathrm{train},3} = 4.3$ $\mathtt{X}_{\mathrm{train},4} = 6.2$ $\mathtt{X}_{\mathrm{train},5} = 9.1$

$\mathtt{Y}_{\mathrm{train},1} = 23.7$ $\mathtt{Y}_{\mathrm{train},2} = 31.3$ $\mathtt{Y}_{\mathrm{train},3} = 79.9$ $\mathtt{Y}_{\mathrm{train},4} = 101.9$ $\mathtt{Y}_{\mathrm{train},5} = 205.5$

Choix de la fonction $$f\left(x;\boldsymbol{\theta}\right) = \sum_{i=0}^{d} \boldsymbol{\theta}_i x^i = \boldsymbol{\theta}^\top \phi\left(x\right)$$

où $\phi\left(x\right) = \begin{bmatrix} 1 & x & \dots & x^d \end{bmatrix}^\top$

hyper-paramètre : d ▶ **degré du polynôme**



Apprentissage

Choix du coût $$l\left(y, o\right) = \left(y - o\right)^2$$

Optimisation $$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{5} \left(\mathtt{Y}_{\mathrm{train},i} - \boldsymbol{\theta}^\top \phi\left(\mathtt{X}_{\mathrm{train},i}\right)\right)^2$$ ▶ **Moindres carrés linéaires**

Inférence $$s_{\mathrm{test}} = f\left(x;\boldsymbol{\theta}^*\right) = \boldsymbol{\theta}^{*\top} \phi\left(x_{\mathrm{test}}\right)$$

III)

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

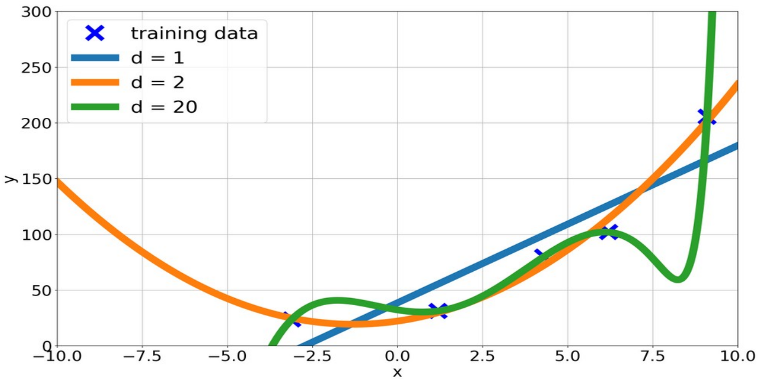
$\mathtt{X}_{\mathrm{train},1} = -3.1 \quad \mathtt{X}_{\mathrm{train},2} = 1.2 \quad \mathtt{X}_{\mathrm{train},3} = 4.3 \quad \mathtt{X}_{\mathrm{train},4} = 6.2 \quad \mathtt{X}_{\mathrm{train},5} = 9.1$

$\mathtt{Y}_{\mathrm{train},1} = 23.7 \quad \mathtt{Y}_{\mathrm{train},2} = 31.3 \quad \mathtt{Y}_{\mathrm{train},3} = 79.9 \quad \mathtt{Y}_{\mathrm{train},4} = 101.9 \quad \mathtt{Y}_{\mathrm{train},5} = 205.5$

Choix de la fonction $$f(x;\boldsymbol{\theta}) = \sum_{i=0}^{d} \boldsymbol{\theta}_i x^i = \boldsymbol{\theta}^\top \phi(x)$$

où $\phi(x) = \begin{bmatrix} 1 & x & \dots & x^d \end{bmatrix}^\top$

hyper-paramètre : d → **degré du polynôme**



Apprentissage

Choix du coût $$l(y,o) = (y-o)^2$$

Optimisation $$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{5} \left(\mathtt{Y}_{\mathrm{train},i} - \boldsymbol{\theta}^\top \phi(\mathtt{X}_{\mathrm{train},i}) \right)^2$$ → **Moindres carrés linéaires**

Inférence

$$s_{\mathrm{test}} = f(x;\boldsymbol{\theta}^*) = \boldsymbol{\theta}^{*\top} \phi(x_{\mathrm{test}})$$

III) Avantages et inconvénients d'une approche paramétrique

Avantages	**Inconvénients**

III)

Avantages et inconvénients d’une approche paramétrique

Avantages	**Inconvénients**
• Inférence efficace → **pas d’accès** à la base de données étiquetées	

III)

Avantages et inconvénients d'une approche paramétrique

Avantages

- Inférence efficace → **pas d'accès** à la base de données étiquetées

- Temps d'inférence constant → **ne dépend pas** de la taille de la base de données étiquetées

Inconvénients

III)

Avantages et inconvénients d'une approche paramétrique

Avantages

- Inférence efficace → **pas d'accès** à la base de données étiquetées
- Temps d'inférence constant → **ne dépend pas** de la taille de la base de données étiquetées

Inconvénients

- Choix de la fonction paramétrique et de ses hyper-paramètres

III)

Avantages et inconvénients d’une approche paramétrique

Avantages

- Inférence efficace → **pas d’accès** à la base de données étiquetées
- Temps d’inférence constant → **ne dépend pas** de la taille de la base de données étiquetées

Inconvénients

- Choix de la fonction paramétrique et de ses hyper-paramètres
- Étape d’apprentissage → souvent **longue** et **gourmande** en calculs

III)

Avantages et inconvénients d'une approche paramétrique

Avantages

- Inférence efficace → **pas d'accès** à la base de données étiquetées
- Temps d'inférence constant → **ne dépend pas** de la taille de la base de données étiquetées

Inconvénients

- Choix de la fonction paramétrique et de ses hyper-paramètres
- Étape d'apprentissage → souvent **longue** et **gourmande** en calculs
- Difficile de modifier la base de données étiquetées → nécessite de faire un **nouvel** apprentissage

IV) Réseaux de neurones

Choix de la fonction : Réseau de neurones

Réseau de neurones = Composition de fonctions paramétriques

Choix de la fonction : Réseau de neurones

Réseau de neurones = Composition de fonctions paramétriques

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

Choix de la fonction : Réseau de neurones

Réseau de neurones = Composition de fonctions paramétriques

Mathématique

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

Informatique (Python)

```python
def neuralNetwork_forward(x, theta, L):
    x1 = f1(x, theta[0])
    x2 = f2(x1, theta[1])
    ...
    s  = fL(..., theta[L-1])
    return s
```

Choix de la fonction : Réseau de neurones

Réseau de neurones = Composition de fonctions paramétriques

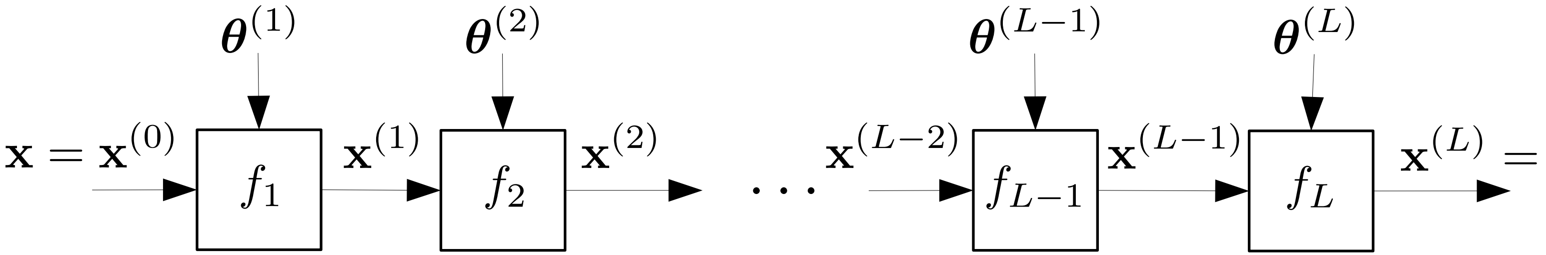
Mathématique

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

Informatique (Python)

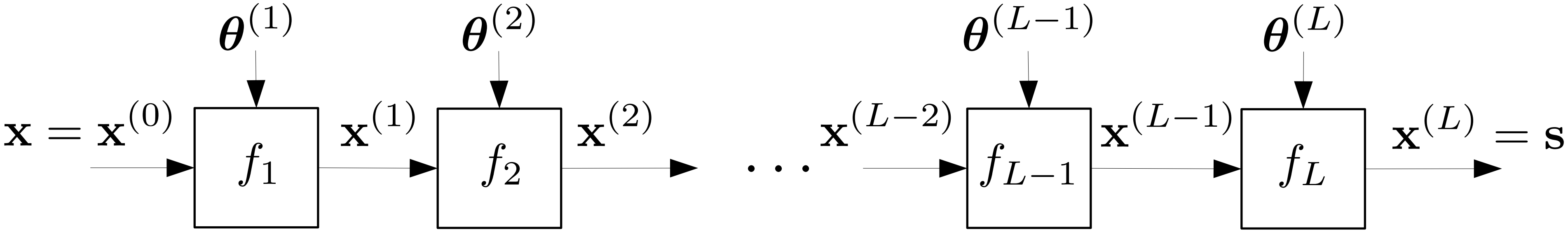
```
def neuralNetwork_forward(x, theta, L):
    x1 = f1(x, theta[0])
    x2 = f2(x1, theta[1])
    ...
    s  = fL(..., theta[L-1])
    return s
```

Graphique (Graphe de calcul)



IV)

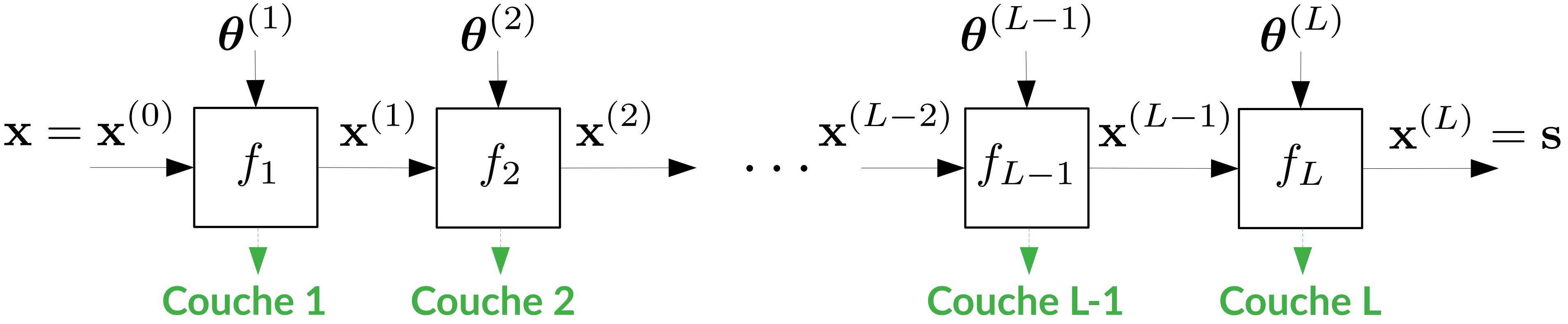
Réseau de neurones : Terminologie



Couche (« layer » en anglais)

IV)

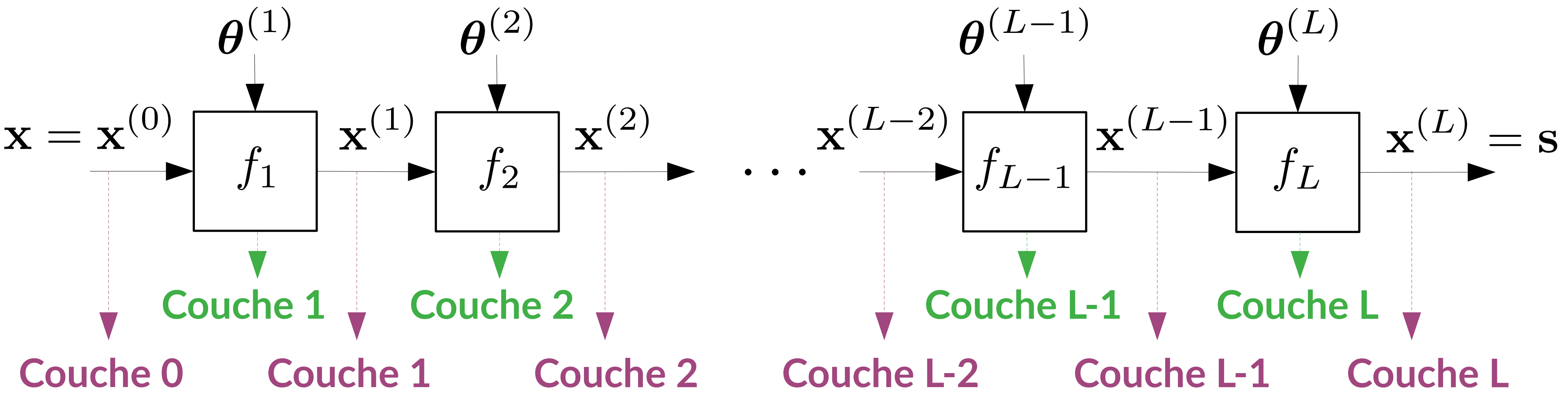
Réseau de neurones : Terminologie



Couche (« layer » en anglais) $\xrightarrow{\text{Sens 1}}$ **= une fonction paramétrique du réseau**

IV)

Réseau de neurones : Terminologie



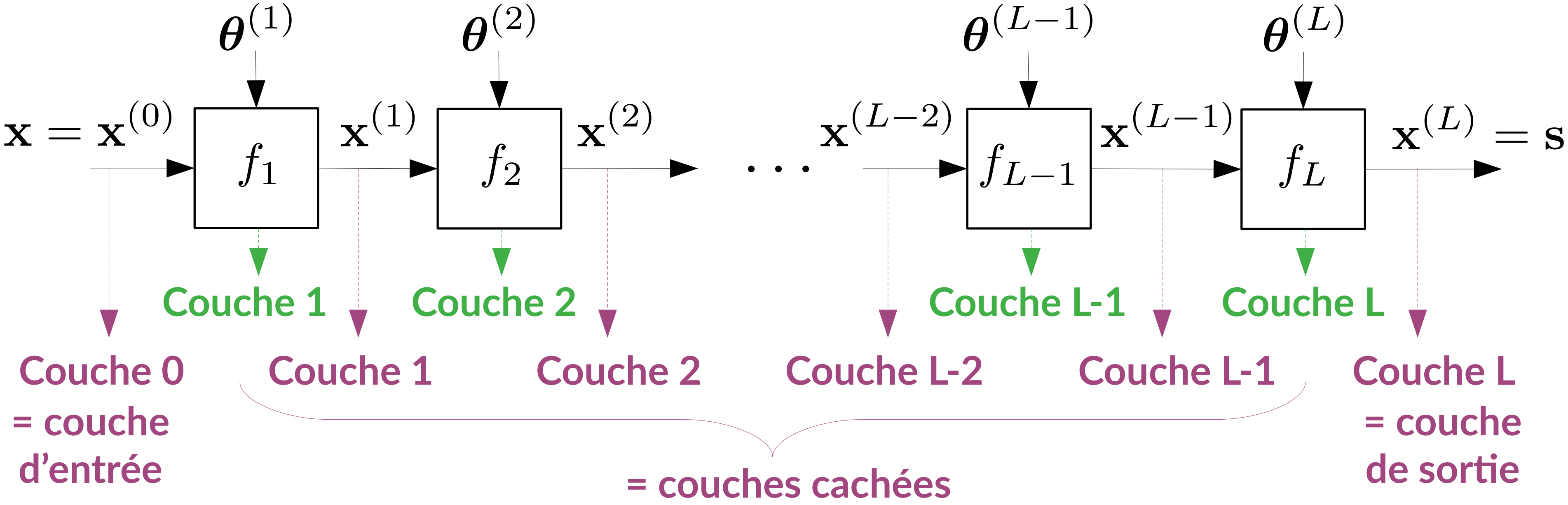
Couche (« layer » en anglais)

Sens 1 → = une fonction paramétrique du réseau

Sens 2 → = couche de « neurones » = un vecteur du réseau

IV)

Réseau de neurones : Terminologie

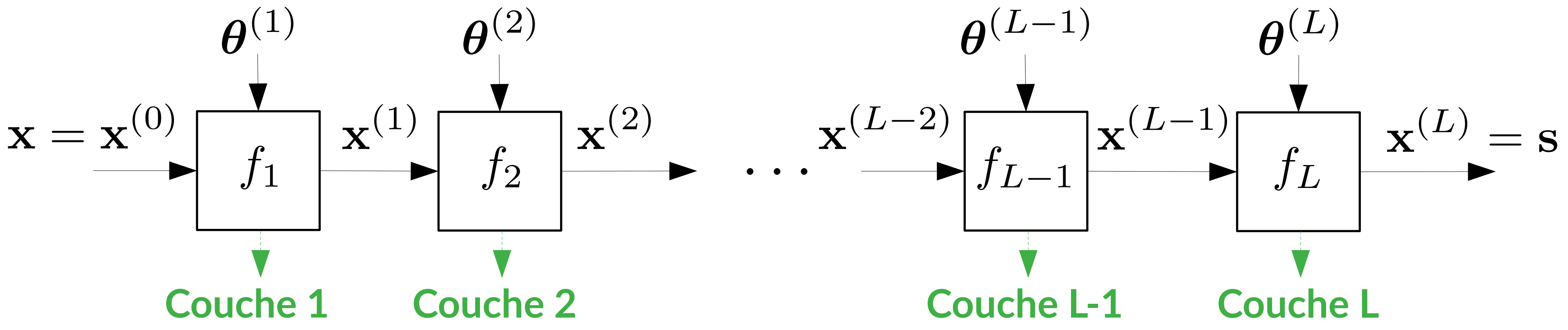


Couche (« layer » en anglais)

Sens 1 → = une fonction paramétrique du réseau

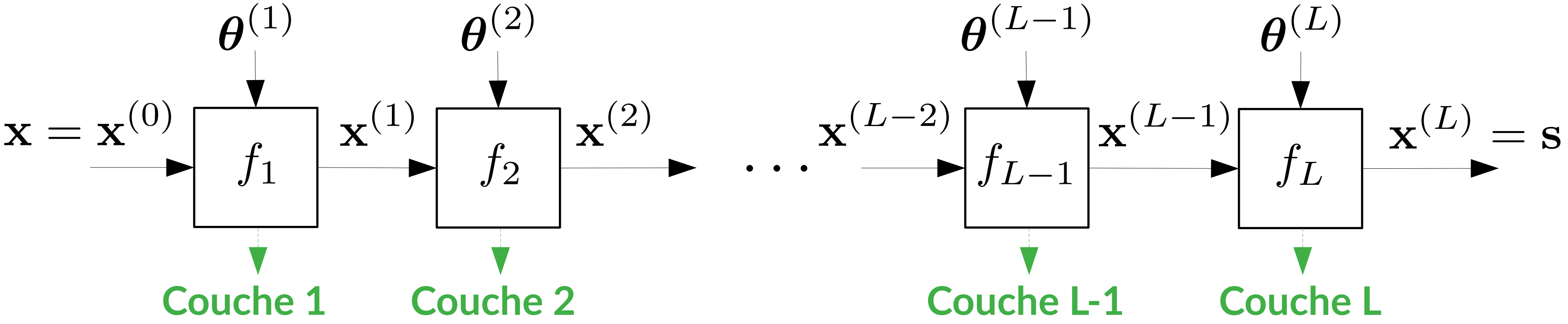
Sens 2 → = couche de « neurones » = un vecteur du réseau

Réseau de neurones : Terminologie (suite)



- **Profondeur du réseau de neurones = nombre de couches**

Réseau de neurones : Terminologie (suite)

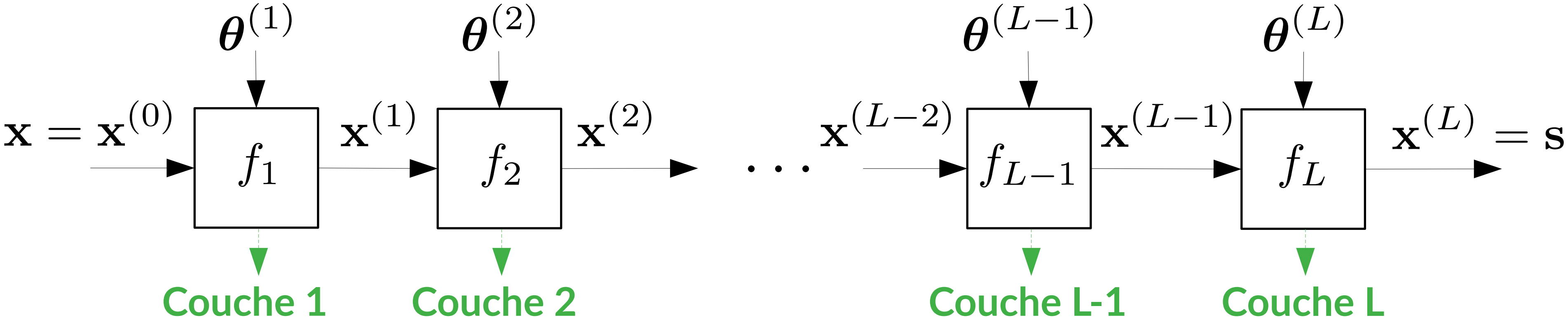


- **Profondeur** du réseau de neurones = nombre de couches

- « **Deep Neural Network**» = réseau de neurones profond

IV)

Réseau de neurones : Terminologie (suite)



- **Profondeur** du réseau de neurones = nombre de couches
- « **Deep Neural Network**» = réseau de neurones profond
- **Architecture** du réseau de neurones = choix du nombre de couches, du type de chaque couche et de ses hyperparamètres, etc.

ImageNet Large Scale Visual Recognition Challenge 2012

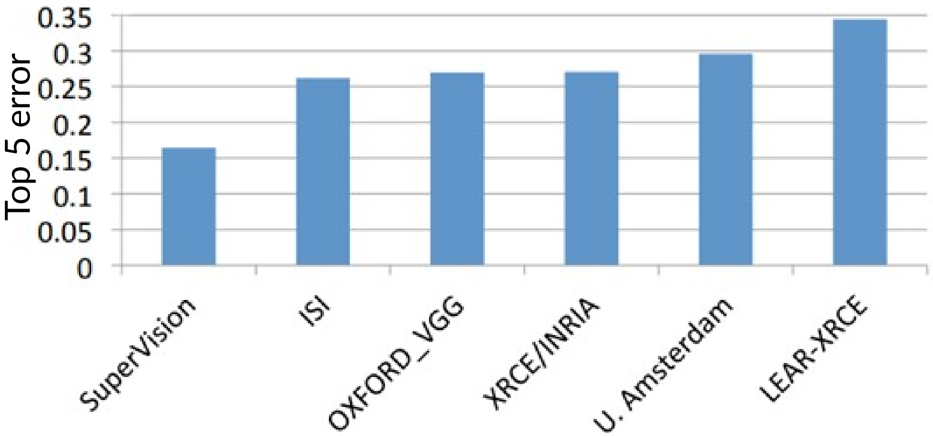
ImageNet : 1.2 millions d’images annotées, 1000 classes

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



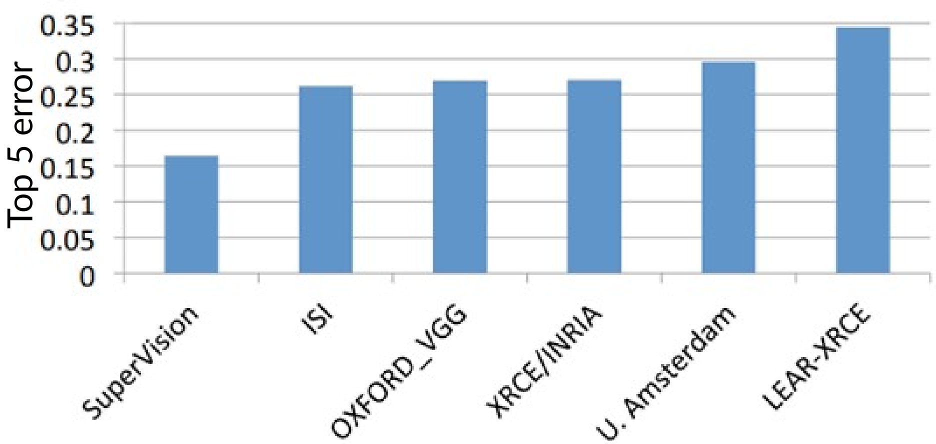
ImageNet : 1.2 millions d'images annotées, 1000 classes

SuperVision (*A. Krizhevsky*, *I. Sutskever*, *G. Hinton*, University of Toronto)

- Réseau de neurones à convolution (AlexNet)

IV)

ImageNet Large Scale Visual Recognition Challenge 2012



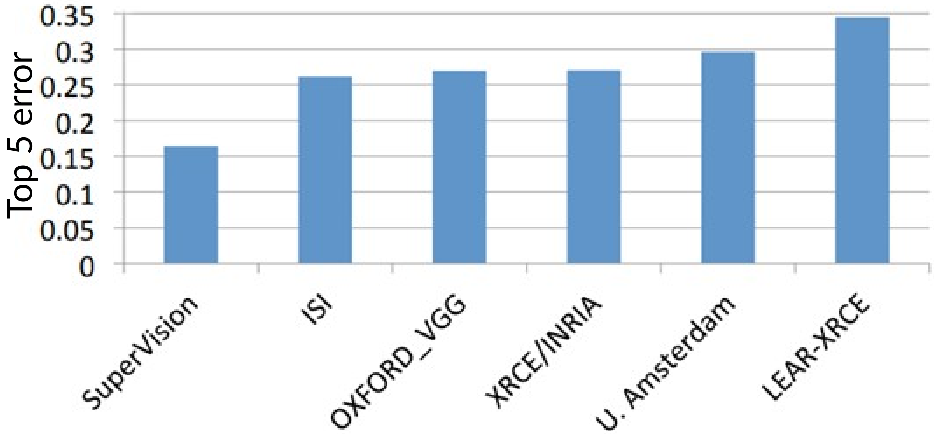
ImageNet : 1.2 millions d'images annotées, 1000 classes

SuperVision (*A. Krizhevsky*, *I. Sutskever*, *G. Hinton*, University of Toronto)

- Réseau de neurones à convolution (AlexNet)

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

ImageNet Large Scale Visual Recognition Challenge 2012

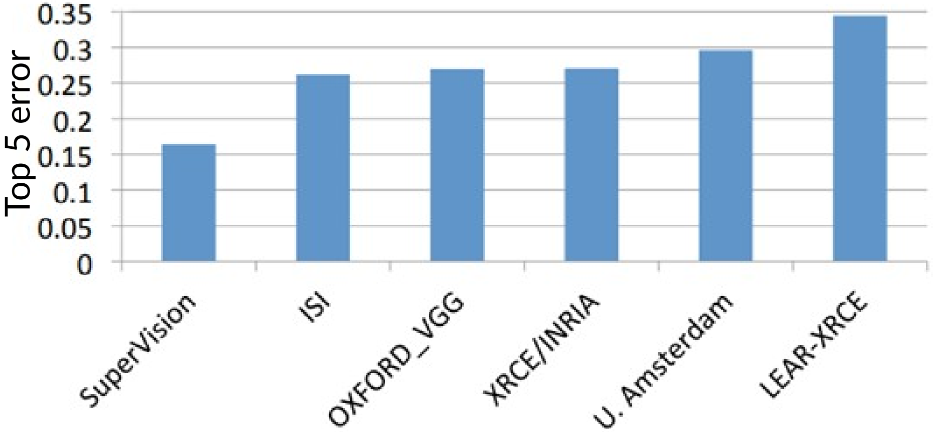


ImageNet : 1.2 millions d'images annotées, 1000 classes

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



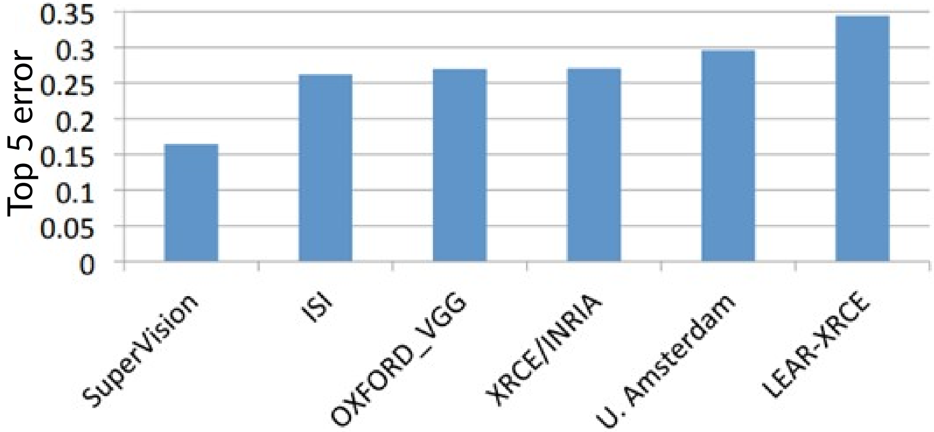
ImageNet : 1.2 millions d'images annotées, 1000 classes

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)
- 6 jours d'apprentissage sur 2 GPUs (GTX 580 3GB)

IV)

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)

Résumé des ingrédients du « Deep Learning »

1) Grande base de données étiquetées

2) « Bonne » architecture de réseau de neurones profond

→ « Perceptron » multicouche, Réseau de neurones à convolution, Transformer

3) Grande capacité de calculs en parallèle (GPU)

Résumé des ingrédients du « Deep Learning »

1) Grande base de données étiquetées

2) « Bonne » architecture de réseau de neurones profond

→ **« Perceptron » multicouche**, Réseau de neurones à convolution, Transformer

3) Grande capacité de calculs en parallèle (GPU)

Architecture : “Perceptron” multicouche (MLP)

Transformation affine = FC (“Fully Connected”)

$$\mathrm{FC}\left(\mathbf{x}; \boldsymbol{\theta} = \{\mathtt{W}, \mathbf{b}\}\right) = \mathtt{W}\mathbf{x} + \mathbf{b}$$

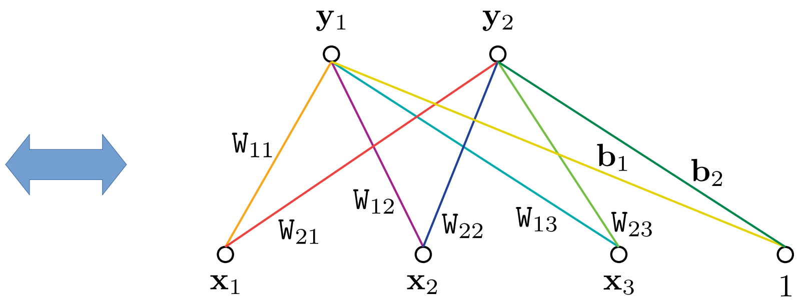
Architecture : “Perceptron” multicouche (MLP)

Transformation affine = FC (“Fully Connected”)

$$\mathrm{FC}\left(\mathbf{x};\boldsymbol{\theta}=\{\mathtt{W},\mathbf{b}\}\right)=\mathtt{W}\mathbf{x}+\mathbf{b}$$

Pourquoi « Fully Connected » ?

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



Architecture : “Perceptron” multicouche (MLP)

Transformation affine = FC (“Fully Connected”)

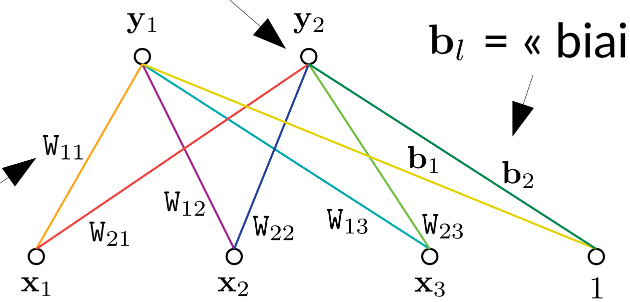
$$\mathrm{FC}\left(\mathbf{x};\boldsymbol{\theta}=\{\mathtt{W},\mathbf{b}\}\right)=\mathtt{W}\mathbf{x}+\mathbf{b}$$

Pourquoi « Fully Connected » ?

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

Élément d’un vecteur = « neurone »

$\mathbf{b}_l$ = « biais »



$\mathtt{W}_{ij}$ = « poids » d’une connexion entre deux « neurones »

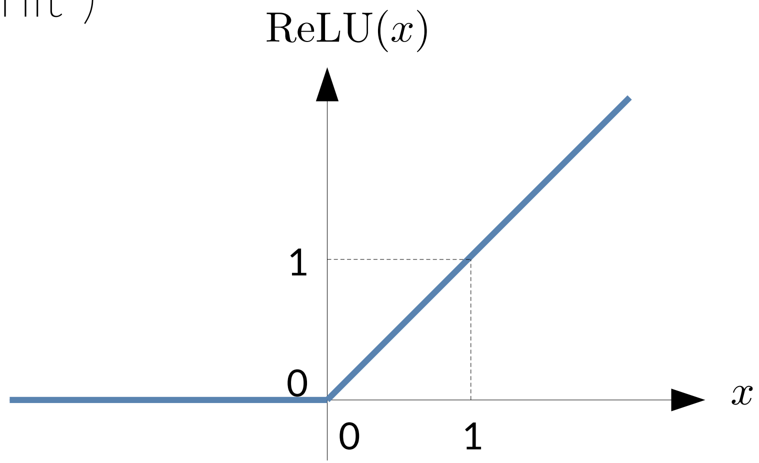
Architecture : “Perceptron” multicouche (MLP)

Transformation affine = FC (“Fully Connected”)

$$\mathrm{FC}\left(\mathbf{x};\boldsymbol{\theta}=\{\mathtt{W},\mathbf{b}\}\right)=\mathtt{W}\mathbf{x}+\mathbf{b}$$

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

$$\mathrm{ReLU}\left(x\right)=\max\left(0,x\right)$$



Architecture : “Perceptron” multicouche (MLP)

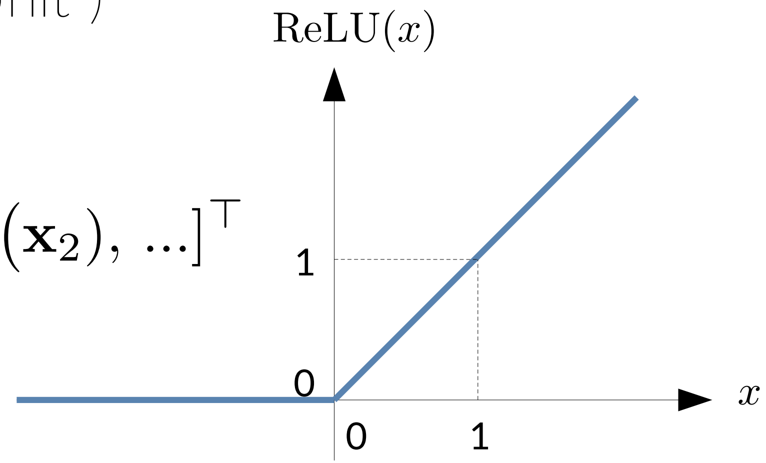
Transformation affine = FC (“Fully Connected”)

$$\mathrm{FC}\left(\mathbf{x};\boldsymbol{\theta}=\{\mathtt{W},\mathbf{b}\}\right)=\mathtt{W}\mathbf{x}+\mathbf{b}$$

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

$$\mathrm{ReLU}\left(x\right)=\max\left(0,x\right)$$

Abus de notation : $\mathrm{ReLU}\left(\mathbf{x}\right)=\left[\mathrm{ReLU}\left(\mathbf{x}_1\right),\ \mathrm{ReLU}\left(\mathbf{x}_2\right),\ \ldots\right]^\top$



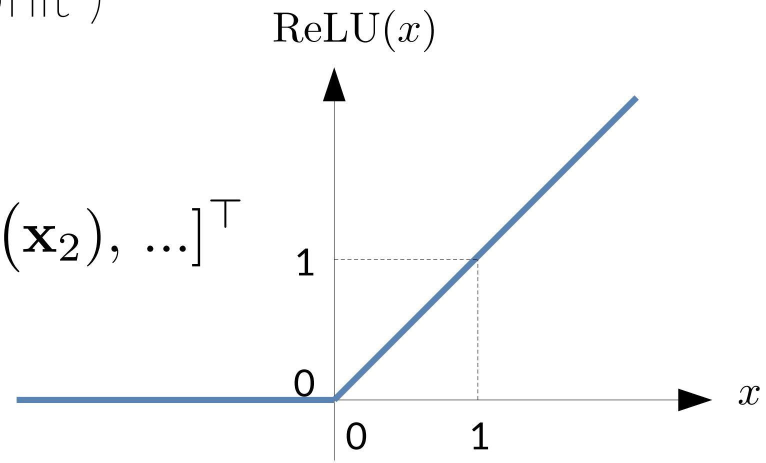
Architecture : “Perceptron” multicouche (MLP)

Transformation affine = FC (“Fully Connected”)

$$\mathrm{FC}\left(\mathbf{x};\boldsymbol{\theta}=\{\mathtt{W},\mathbf{b}\}\right)=\mathtt{W}\mathbf{x}+\mathbf{b}$$

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

$$\mathrm{ReLU}\left(x\right)=\max\left(0,x\right)$$



Abus de notation : $\mathrm{ReLU}\left(\mathbf{x}\right)=\left[\mathrm{ReLU}\left(\mathbf{x}_1\right),\ \mathrm{ReLU}\left(\mathbf{x}_2\right),\ \ldots\right]^\top$

Terminologie : fonction d’activation

Architecture : “Perceptron” multicouche (MLP) (suite)

Mathématique $$\mathbf{s} = \mathrm{MLP}\left(\mathbf{x}; \boldsymbol{\theta}\right) = \mathrm{FC}\left(\mathrm{ReLU}\left(\ldots\mathrm{ReLU}\left(\mathrm{FC}\left(\mathbf{x}; \boldsymbol{\theta}_1\right)\right)\ldots\right); \boldsymbol{\theta}_L\right)$$

Architecture : "Perceptron" multicouche (MLP) (suite)

Mathématique $$\mathbf{s} = \mathrm{MLP}\left(\mathbf{x};\boldsymbol{\theta}\right) = \mathrm{FC}\left(\mathrm{ReLU}\left(\ldots\mathrm{ReLU}\left(\mathrm{FC}\left(\mathbf{x};\boldsymbol{\theta}_1\right)\right)\ldots\right);\boldsymbol{\theta}_L\right)$$

Informatique (Python)

```
def MLP_forward(x, theta, L):
    x1 = FC(x, theta[0])
    x2 = ReLU(x1)
    x3 = FC(x2, theta[2])
    ...
    s  = FC(..., theta[L-1])
    return s
```

IV)

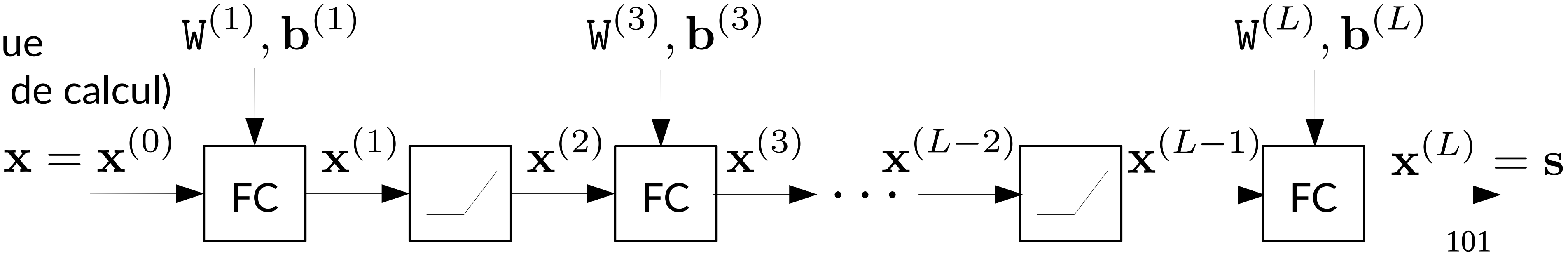
Architecture : “Perceptron” multicouche (MLP) (suite)

Mathématique $$\mathbf{s} = \mathrm{MLP}\left(\mathbf{x}; \boldsymbol{\theta}\right) = \mathrm{FC}\left(\mathrm{ReLU}\left(\ldots \mathrm{ReLU}\left(\mathrm{FC}\left(\mathbf{x}; \boldsymbol{\theta}_1\right)\right)\ldots\right); \boldsymbol{\theta}_L\right)$$

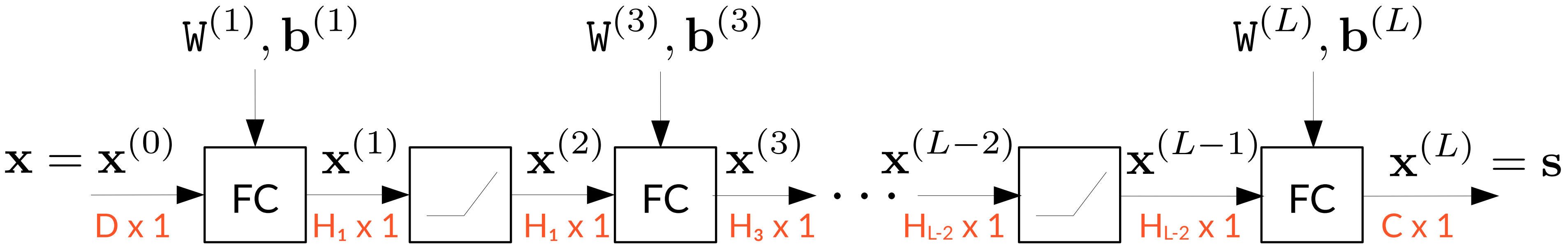
Informatique (Python)

```python
def MLP_forward(x, theta, L):
    x1 = FC(x, theta[0])
    x2 = ReLU(x1)
    x3 = FC(x2, theta[2])
    ...
    s  = FC(..., theta[L-1])
    return s
```

Graphique (Graphe de calcul)



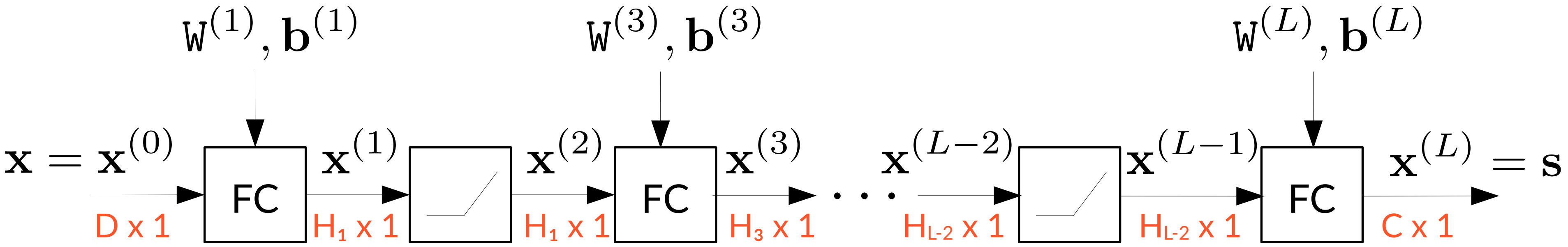
Architecture : "Perceptron" multicouche (MLP) (suite bis)



Paramètres : $\left\{\mathtt{W}^{(2j-1)}, \mathbf{b}^{(2j-1)}\right\}_{j=1,\ldots,\lceil L/2 \rceil}$

IV)

Architecture : “Perceptron” multicouche (MLP) (suite bis)



Paramètres : $\left\{ \mathtt{W}^{(2j-1)}, \mathbf{b}^{(2j-1)} \right\}_{j=1,\ldots,\lceil L/2 \rceil}$

Hyper-paramètres : L ,$\{H_{2j-1}\}_{j=1,\ldots,\lceil (L-2)/2 \rceil}$

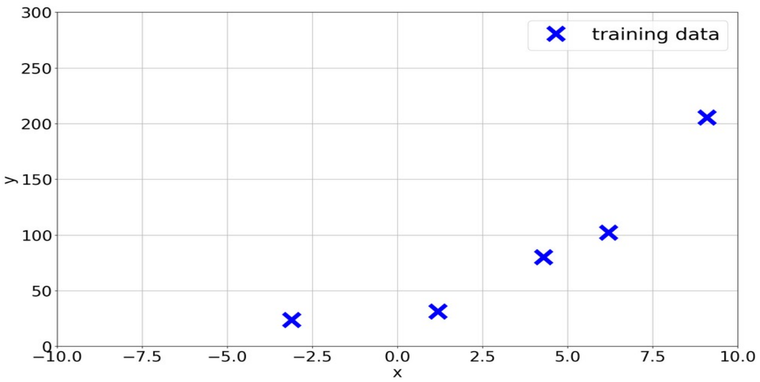
Nombre de couches

Dimension de chaque couche cachée

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

$\mathtt{X}_{\text{train},1} = -3.1$ $\mathtt{X}_{\text{train},2} = 1.2$ $\mathtt{X}_{\text{train},3} = 4.3$ $\mathtt{X}_{\text{train},4} = 6.2$ $\mathtt{X}_{\text{train},5} = 9.1$

$\mathtt{Y}_{\text{train},1} = 23.7$ $\mathtt{Y}_{\text{train},2} = 31.3$ $\mathtt{Y}_{\text{train},3} = 79.9$ $\mathtt{Y}_{\text{train},4} = 101.9$ $\mathtt{Y}_{\text{train},5} = 205.5$

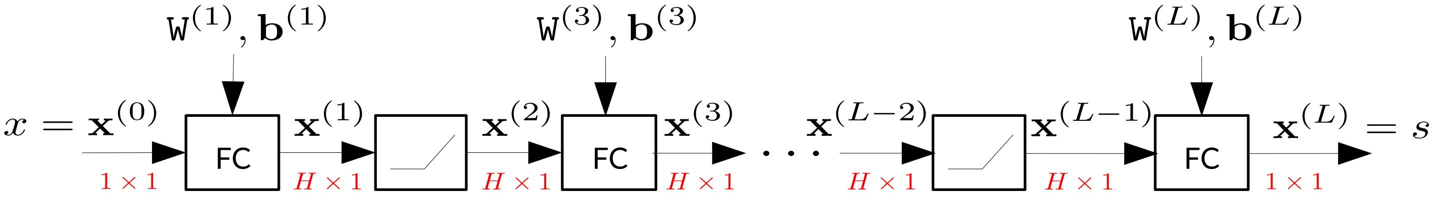


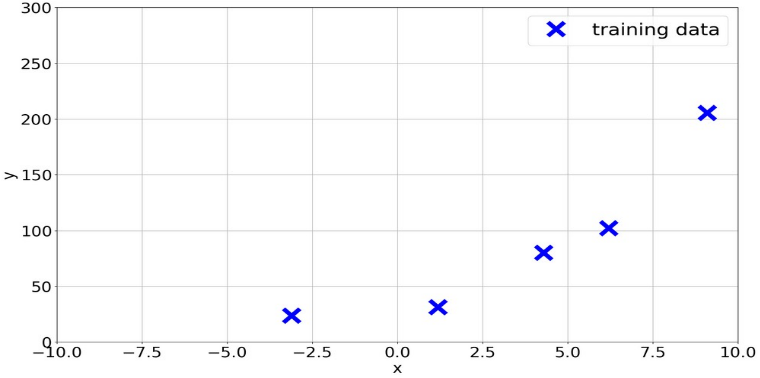
Exemple de régression : 5 données, $\mathtt{X} \in \mathbb{R}$ et $\mathtt{Y} \in \mathbb{R}$

$\mathtt{X}_{\text{train},1} = -3.1$ $\mathtt{X}_{\text{train},2} = 1.2$ $\mathtt{X}_{\text{train},3} = 4.3$ $\mathtt{X}_{\text{train},4} = 6.2$ $\mathtt{X}_{\text{train},5} = 9.1$

$\mathtt{Y}_{\text{train},1} = 23.7$ $\mathtt{Y}_{\text{train},2} = 31.3$ $\mathtt{Y}_{\text{train},3} = 79.9$ $\mathtt{Y}_{\text{train},4} = 101.9$ $\mathtt{Y}_{\text{train},5} = 205.5$

Choix de la fonction





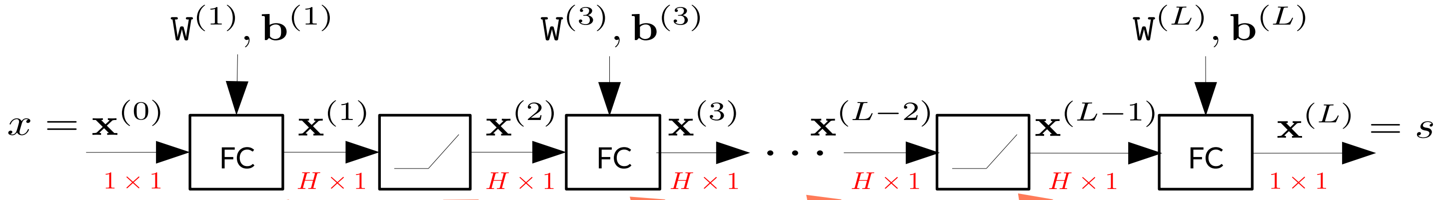
IV)

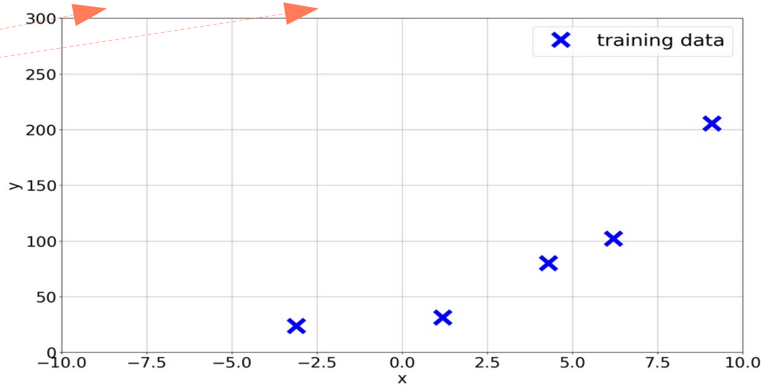
Exemple de régression : 5 données, $\mathtt{X} \in \mathbb{R}$ et $\mathtt{Y} \in \mathbb{R}$

$\mathtt{X}_{\mathrm{train},1} = -3.1 \quad \mathtt{X}_{\mathrm{train},2} = 1.2 \quad \mathtt{X}_{\mathrm{train},3} = 4.3 \quad \mathtt{X}_{\mathrm{train},4} = 6.2 \quad \mathtt{X}_{\mathrm{train},5} = 9.1$

$\mathtt{Y}_{\mathrm{train},1} = 23.7 \quad \mathtt{Y}_{\mathrm{train},2} = 31.3 \quad \mathtt{Y}_{\mathrm{train},3} = 79.9 \quad \mathtt{Y}_{\mathrm{train},4} = 101.9 \quad \mathtt{Y}_{\mathrm{train},5} = 205.5$

Choix de la fonction





Choix d'architecture : même dimension pour toutes les couches cachées.

C'est juste pour l'exemple, en pratique c'est un mauvais choix.

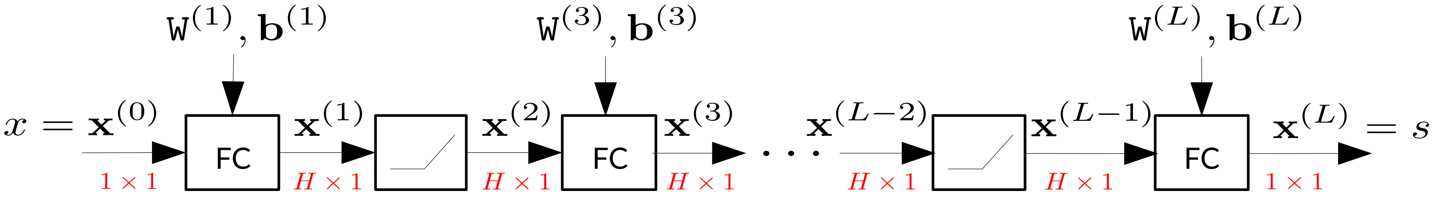
IV)

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

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

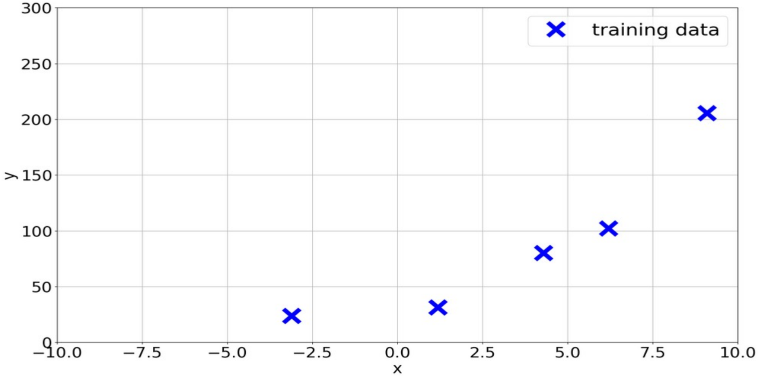
$$\mathtt{Y}_{\text{train},1} = 23.7 \quad \mathtt{Y}_{\text{train},2} = 31.3 \quad \mathtt{Y}_{\text{train},3} = 79.9 \quad \mathtt{Y}_{\text{train},4} = 101.9 \quad \mathtt{Y}_{\text{train},5} = 205.5$$

Choix de la fonction



$$f\,(x;\boldsymbol{\theta}) = \mathrm{MLP}\left(x;\boldsymbol{\theta} = \{\mathtt{W}^{(l)}, \mathbf{b}^{(l)}\}_l\right)$$

hyper-paramètres : L, H

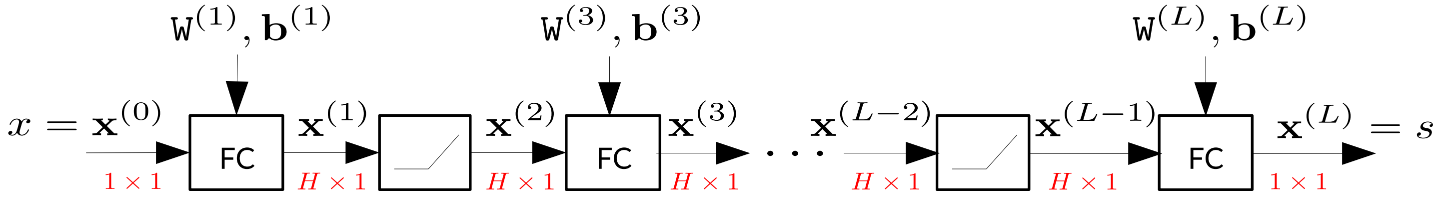


Exemple de régression : 5 données, $\mathtt{X} \in \mathbb{R}$ et $\mathtt{Y} \in \mathbb{R}$

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

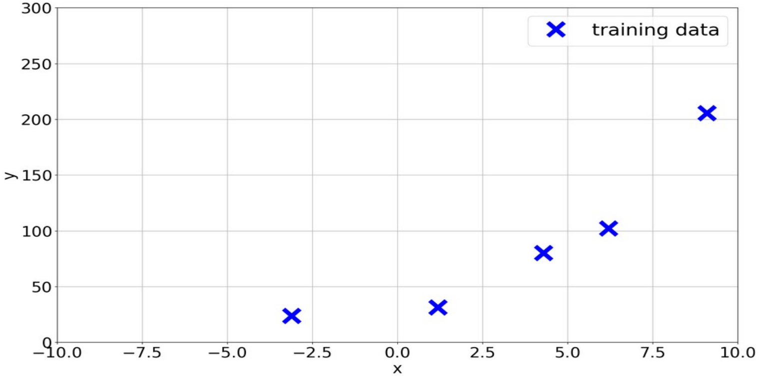
$$\mathtt{Y}_{\text{train},1} = 23.7 \quad \mathtt{Y}_{\text{train},2} = 31.3 \quad \mathtt{Y}_{\text{train},3} = 79.9 \quad \mathtt{Y}_{\text{train},4} = 101.9 \quad \mathtt{Y}_{\text{train},5} = 205.5$$

Choix de la fonction



$$f\,(x;\boldsymbol{\theta}) = \mathrm{MLP}\left(x;\boldsymbol{\theta} = \{\mathtt{W}^{(l)}, \mathbf{b}^{(l)}\}_l\right)$$

hyper-paramètres : L, H



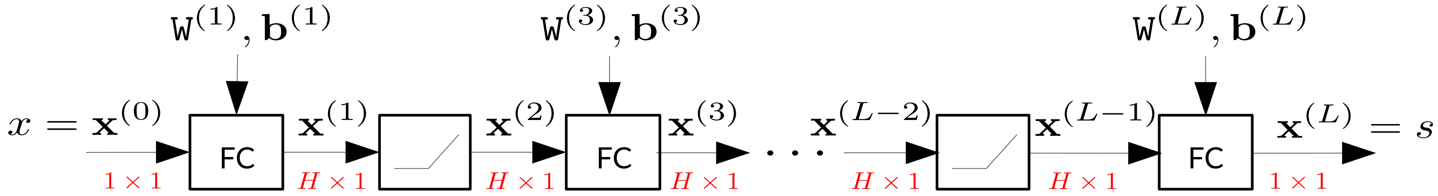
Quel est le nombre de paramètres du « modèle » ?

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

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

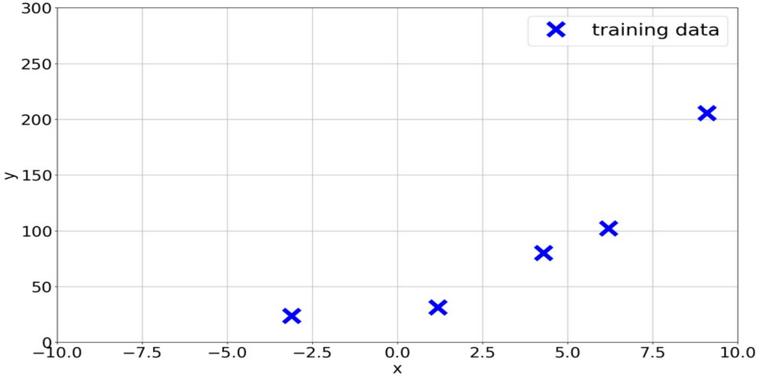
$\mathtt{Y}_{\text{train},1} = 23.7 \quad \mathtt{Y}_{\text{train},2} = 31.3 \quad \mathtt{Y}_{\text{train},3} = 79.9 \quad \mathtt{Y}_{\text{train},4} = 101.9 \quad \mathtt{Y}_{\text{train},5} = 205.5$

Choix de la fonction



$$f\left(x;\boldsymbol{\theta}\right) = \mathrm{MLP}\left(x;\boldsymbol{\theta} = \{\mathtt{W}^{(l)}, \mathbf{b}^{(l)}\}_l\right)$$

hyper-paramètres : L, H



Apprentissage

Choix du coût $\quad l\left(y, s\right) = \left(y - s\right)^2$

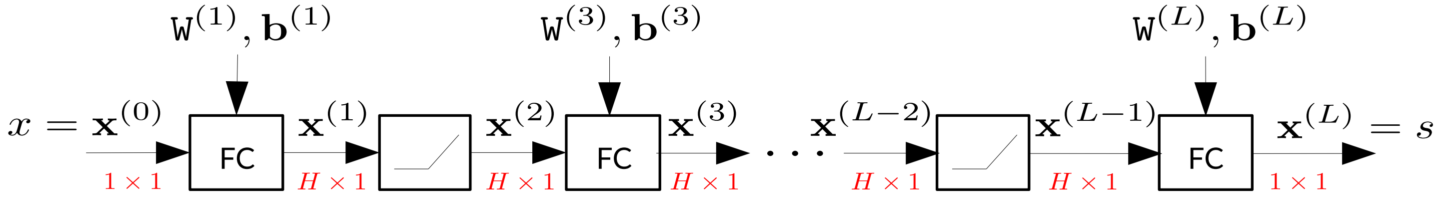
IV)

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

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

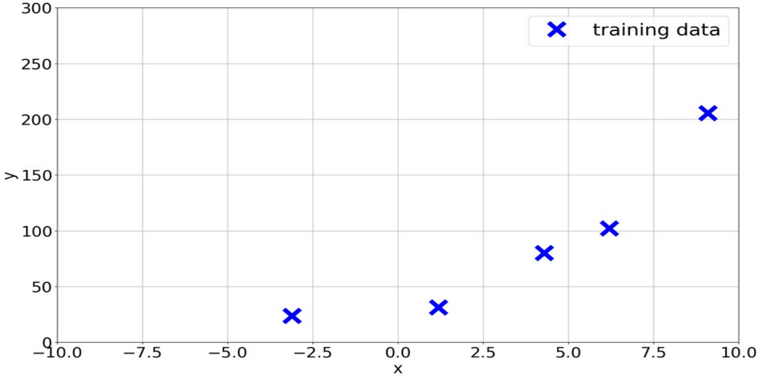
$\mathtt{Y}_{\text{train},1} = 23.7 \quad \mathtt{Y}_{\text{train},2} = 31.3 \quad \mathtt{Y}_{\text{train},3} = 79.9 \quad \mathtt{Y}_{\text{train},4} = 101.9 \quad \mathtt{Y}_{\text{train},5} = 205.5$

Choix de la fonction



$$f\left(x;\boldsymbol{\theta}\right) = \mathrm{MLP}\left(x;\boldsymbol{\theta} = \{\mathtt{W}^{(l)}, \mathbf{b}^{(l)}\}_l\right)$$

hyper-paramètres : L, H



Apprentissage

Choix du coût $\quad l\left(y, s\right) = \left(y - s\right)^2$

Optimisation $\quad \boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{5} \left(\mathtt{Y}_{\text{train},i} - \mathrm{MLP}\left(\mathtt{X}_{\text{train},i};\boldsymbol{\theta}\right)\right)^2$ → **Moindres carrés non linéaires**

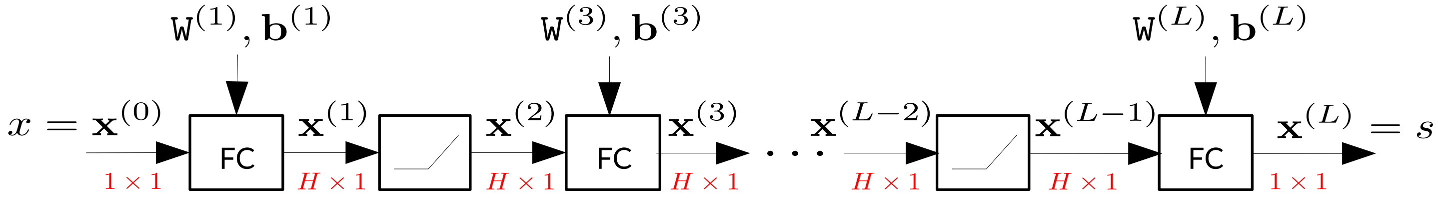
IV)

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

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

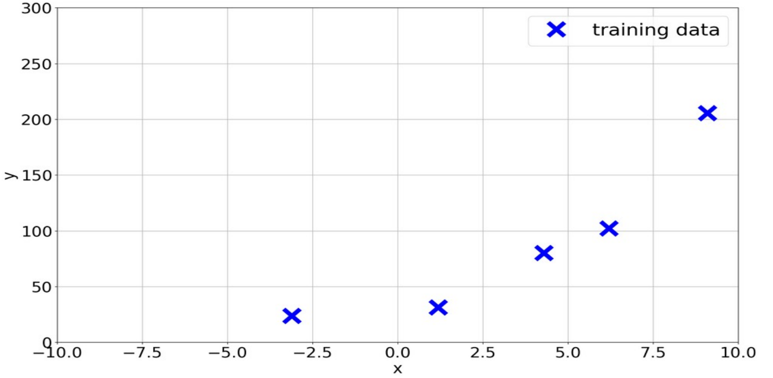
$\mathtt{Y}_{\text{train},1} = 23.7 \quad \mathtt{Y}_{\text{train},2} = 31.3 \quad \mathtt{Y}_{\text{train},3} = 79.9 \quad \mathtt{Y}_{\text{train},4} = 101.9 \quad \mathtt{Y}_{\text{train},5} = 205.5$

Choix de la fonction



$$f\left(x;\boldsymbol{\theta}\right) = \mathrm{MLP}\left(x;\boldsymbol{\theta} = \{\mathtt{W}^{(l)}, \mathbf{b}^{(l)}\}_l\right)$$

hyper-paramètres : L, H



Apprentissage

Choix du coût $\quad l\left(y, s\right) = \left(y - s\right)^2$

Optimisation $\quad \boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{5} \left(\mathtt{Y}_{\text{train},i} - \mathrm{MLP}\left(\mathtt{X}_{\text{train},i};\boldsymbol{\theta}\right)\right)^2$

Moindres carrés non linéaires

Inférence

$$s_{\text{test}} = \mathrm{MLP}\left(x_{\text{test}};\boldsymbol{\theta}^*\right)$$

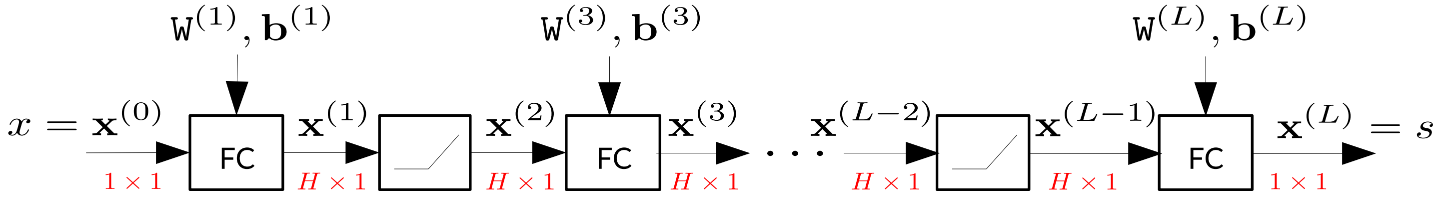
IV)

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

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

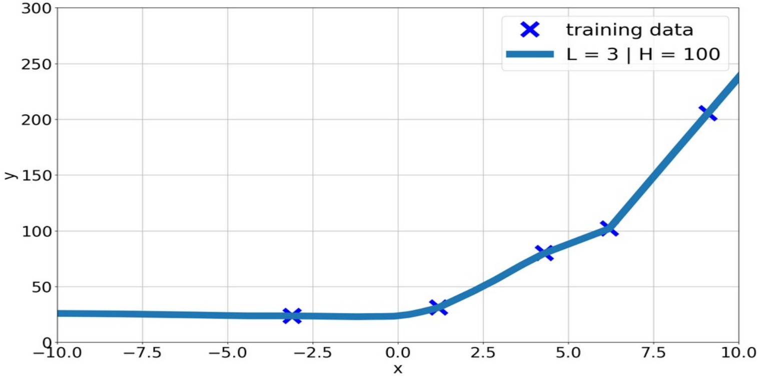
$$\mathtt{Y}_{\text{train},1} = 23.7 \quad \mathtt{Y}_{\text{train},2} = 31.3 \quad \mathtt{Y}_{\text{train},3} = 79.9 \quad \mathtt{Y}_{\text{train},4} = 101.9 \quad \mathtt{Y}_{\text{train},5} = 205.5$$

Choix de la fonction



$$f\left(x; \boldsymbol{\theta}\right) = \mathrm{MLP}\left(x; \boldsymbol{\theta} = \{\mathtt{W}^{(l)}, \mathbf{b}^{(l)}\}_l\right)$$

hyper-paramètres : L, H



Apprentissage

Choix du coût $\quad l\left(y, s\right) = \left(y - s\right)^2$

Optimisation $\quad \boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{5} \left(\mathtt{Y}_{\text{train},i} - \mathrm{MLP}\left(\mathtt{X}_{\text{train},i}; \boldsymbol{\theta}\right)\right)^2$ → **Moindres carrés non linéaires**

Inférence

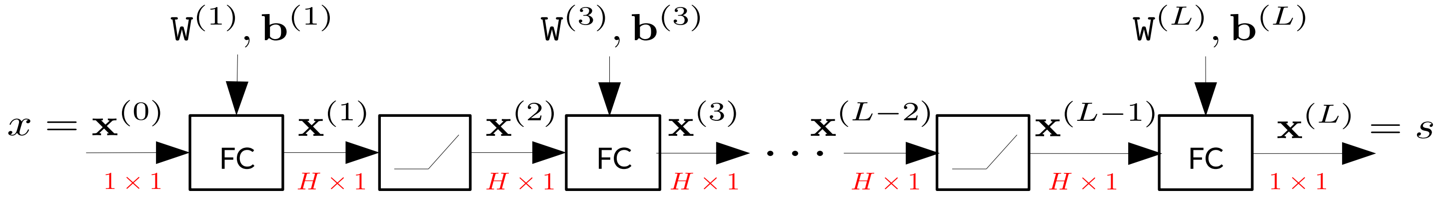
$$s_{\text{test}} = \mathrm{MLP}\left(x_{\text{test}}; \boldsymbol{\theta}^*\right)$$

IV)

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

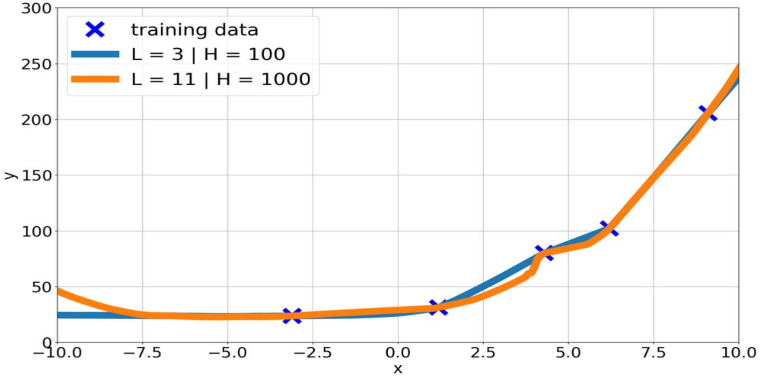
$$\begin{array}{lllll}
\mathtt{X}_{\text{train},1} = -3.1 & \mathtt{X}_{\text{train},2} = 1.2 & \mathtt{X}_{\text{train},3} = 4.3 & \mathtt{X}_{\text{train},4} = 6.2 & \mathtt{X}_{\text{train},5} = 9.1 \\
\mathtt{Y}_{\text{train},1} = 23.7 & \mathtt{Y}_{\text{train},2} = 31.3 & \mathtt{Y}_{\text{train},3}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

Choix de la fonction



$$f\left(x; \boldsymbol{\theta}\right) = \mathrm{MLP}\left(x; \boldsymbol{\theta} = \{\mathtt{W}^{(l)}, \mathbf{b}^{(l)}\}_l\right)$$

hyper-paramètres : L, H



Apprentissage

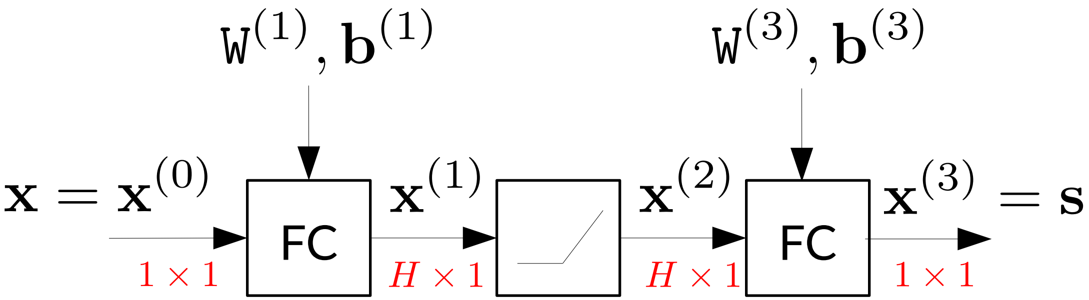
Choix du coût $\quad l\left(y, s\right) = \left(y - s\right)^2$

Optimisation $\quad \boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\arg\min} \sum_{i=1}^{5} \left(\mathtt{Y}_{\text{train},i} - \mathrm{MLP}\left(\mathtt{X}_{\text{train},i}; \boldsymbol{\theta}\right)\right)^2$ $\rightarrow$ **Moindres carrés non linéaires**

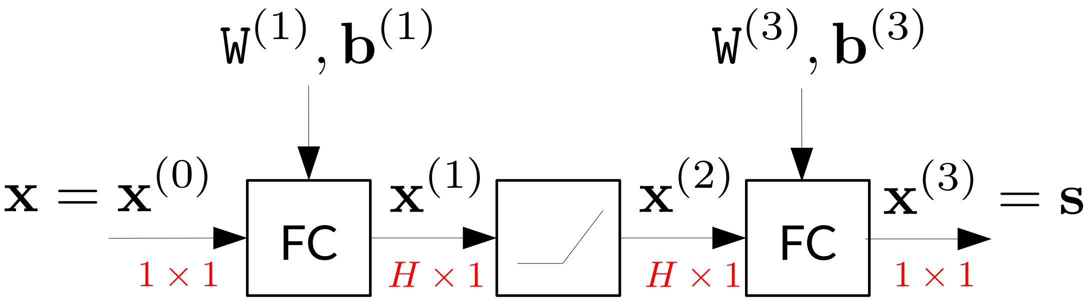
Inférence

$$s_{\text{test}} = \mathrm{MLP}\left(x_{\text{test}}; \boldsymbol{\theta}^*\right)$$

Étude du MLP à une « couche cachée » en 1D

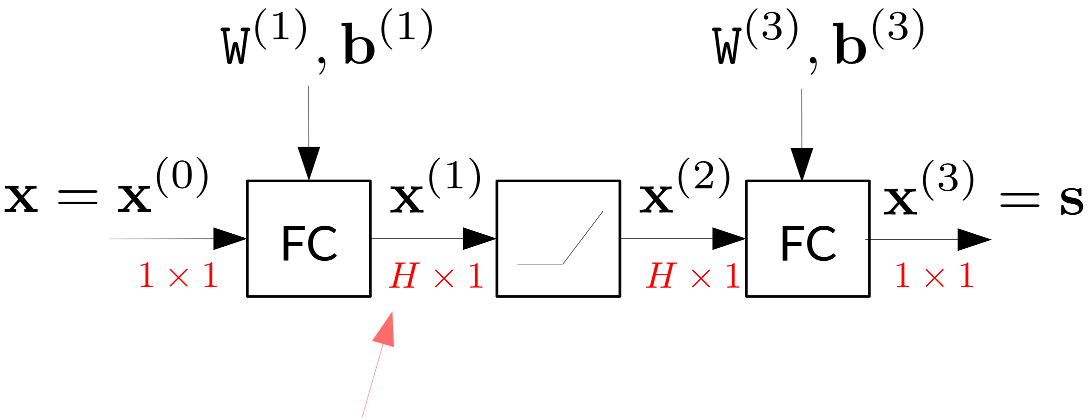


Étude du MLP à une « couche cachée » en 1D



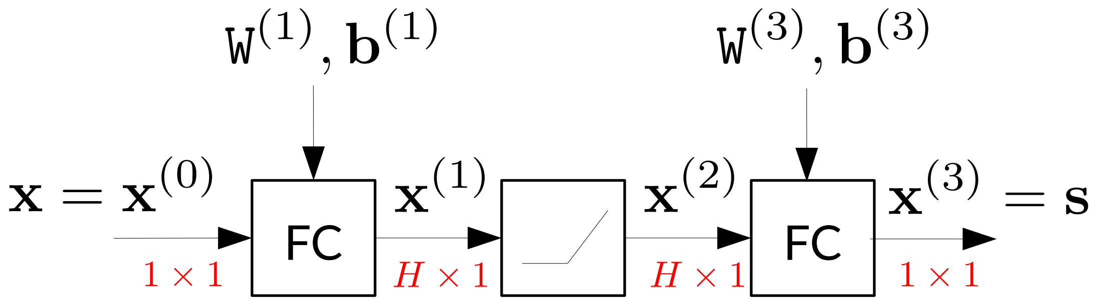
Remarque : si on ignore la fonction ReLU, qui n'est pas paramétrique, alors il n'y a qu'une seule couche cachée $\mathbf{x}^{(1)}$.

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



$\mathtt{W}^{(1)} \doteq \mathbf{w}_1$ Vecteur colonne Hx1

$\mathbf{b}^{(1)} \doteq \mathbf{b}_1$ Vecteur colonne Hx1

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

$\mathbf{b}^{(3)} \doteq b_3$ Scalaire 1x1

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

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

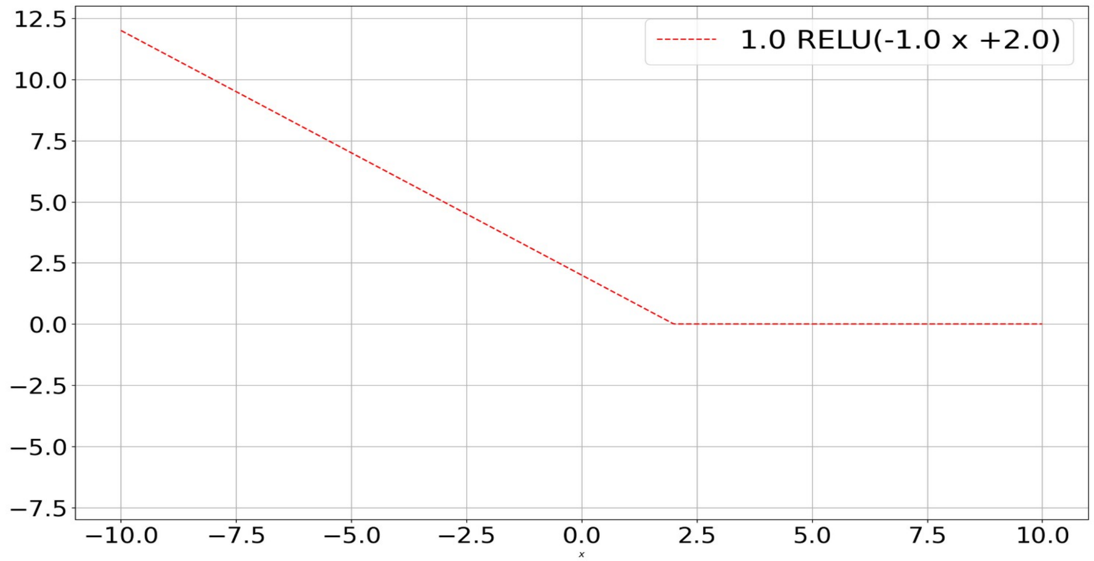
$$
\begin{aligned}
f(x) &= \mathbf{w}_3^\top \mathrm{ReLU}(\mathbf{w}_1 x + \mathbf{b}_1) + b_3 \\
&= \sum_{j=1}^{H} \mathbf{w}_{3,j} \mathrm{ReLU}(\mathbf{w}_{1,j} x + \mathbf{b}_{1,j}) + b_3
\end{aligned}
$$

Somme pondérée de fonctions ReLU !

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

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

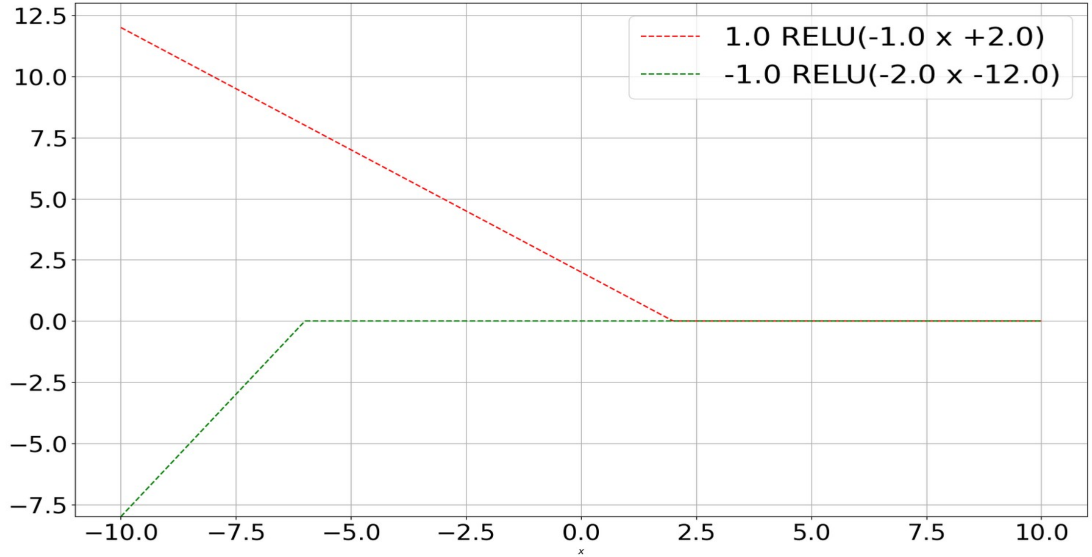
Exemple numérique avec H=3



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

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

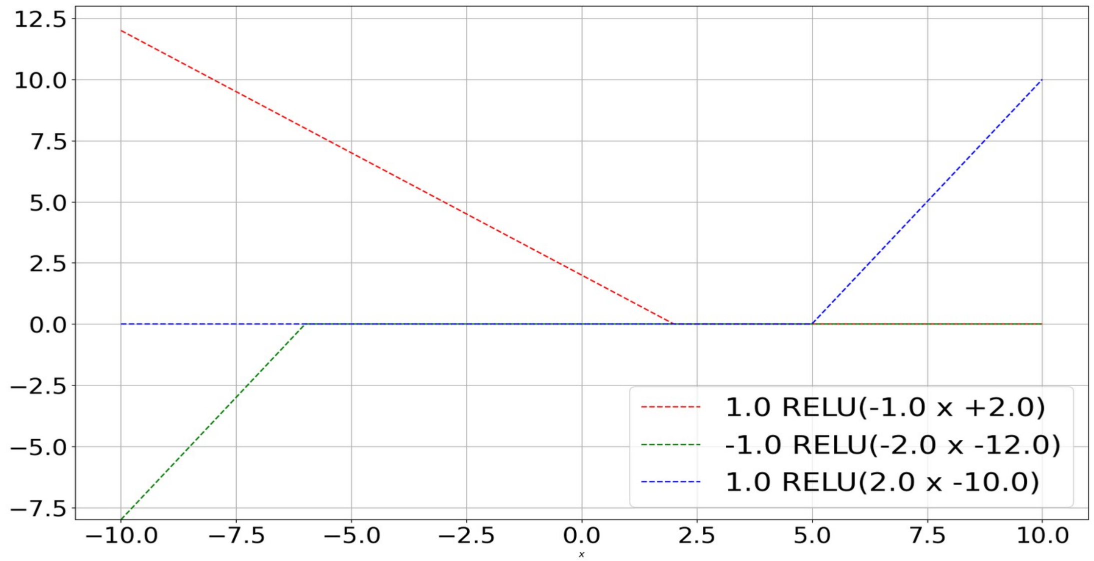
Exemple numérique avec H=3



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

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

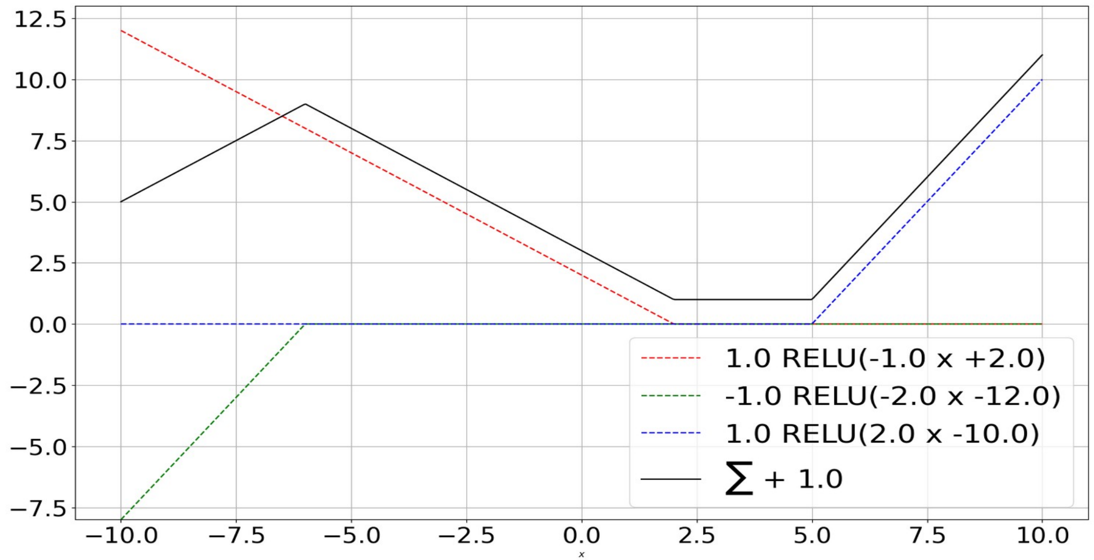
Exemple numérique avec H=3



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

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

Exemple numérique avec H=3

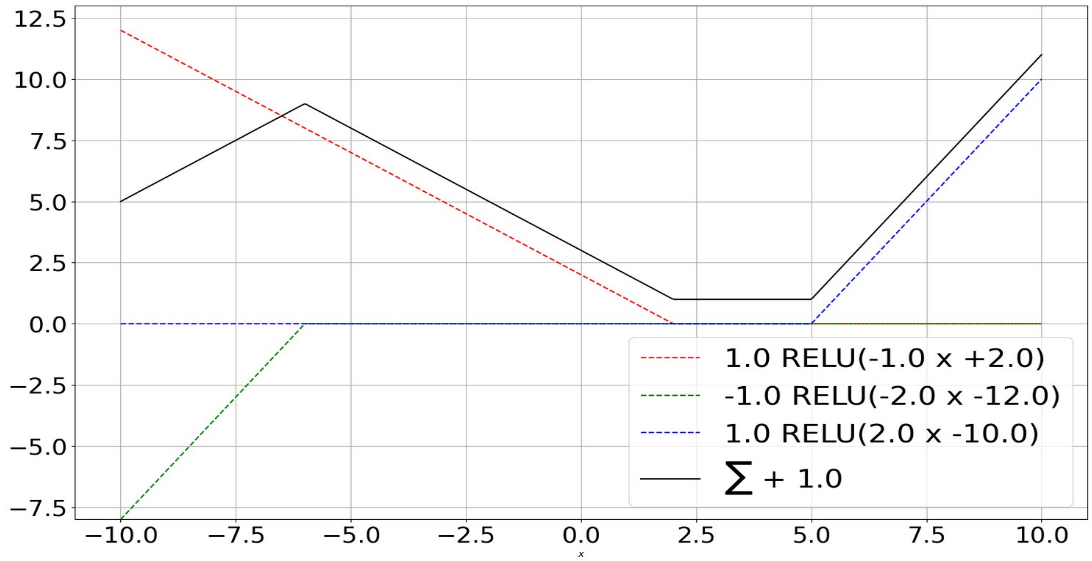


Étude du MLP à une « couche cachée » en 1D (suite)

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

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

Exemple numérique avec H=3

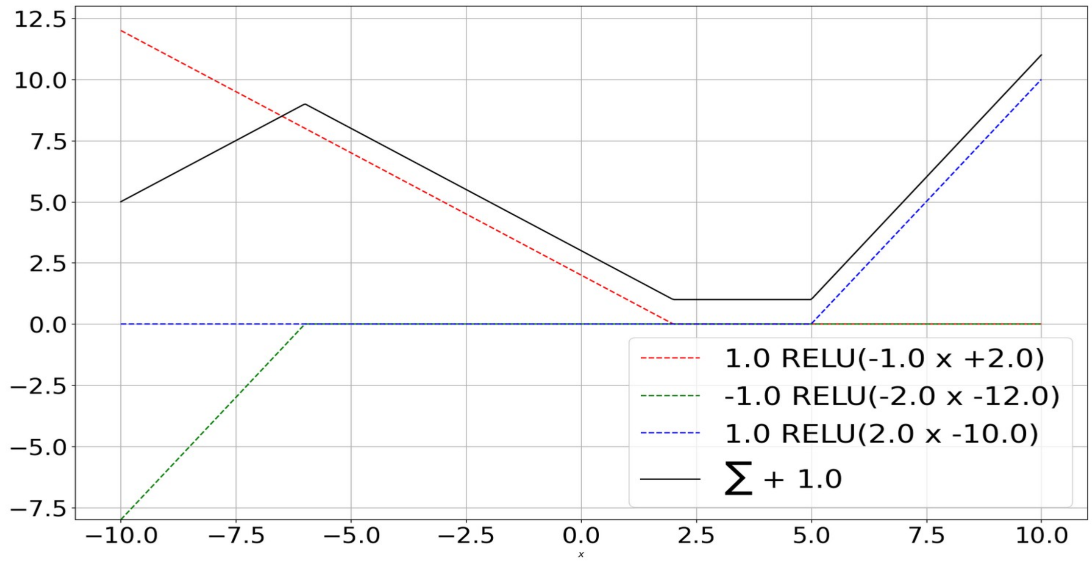


Fonction continue et affine par morceaux

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

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

Exemple numérique avec H=3

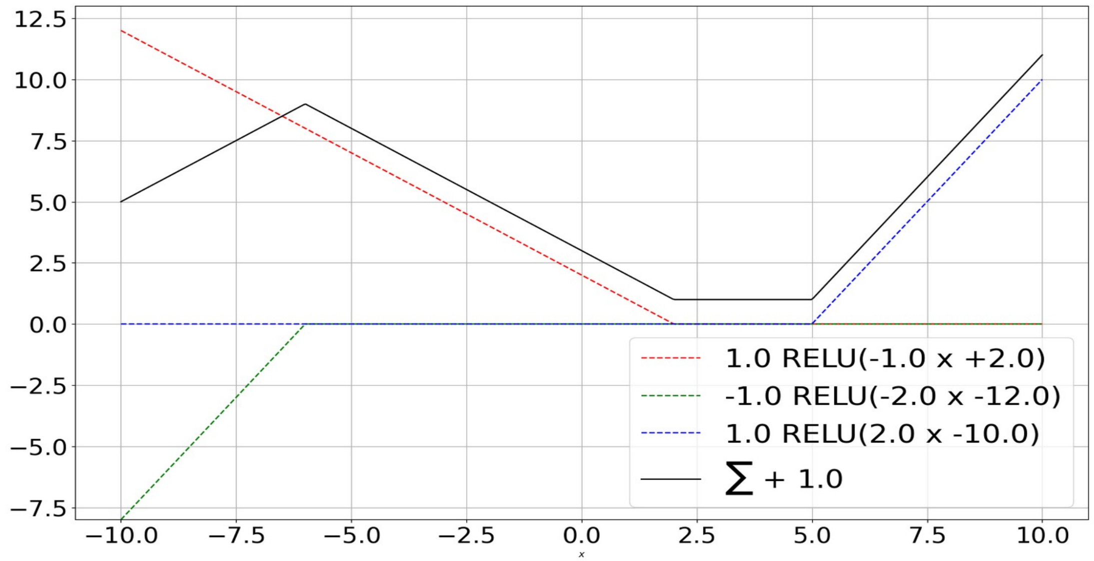


H changements de pente (un par ReLU) = H+1 morceaux

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

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

Exemple numérique avec H=3



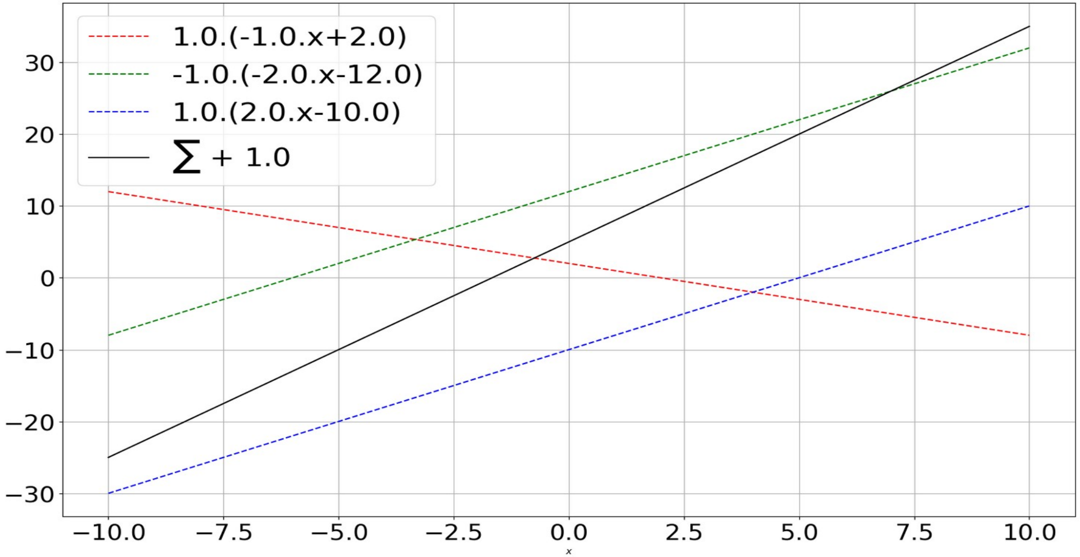
Augmenter H accroît la **capacité** du réseau.

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

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

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

Et sans ReLU ?



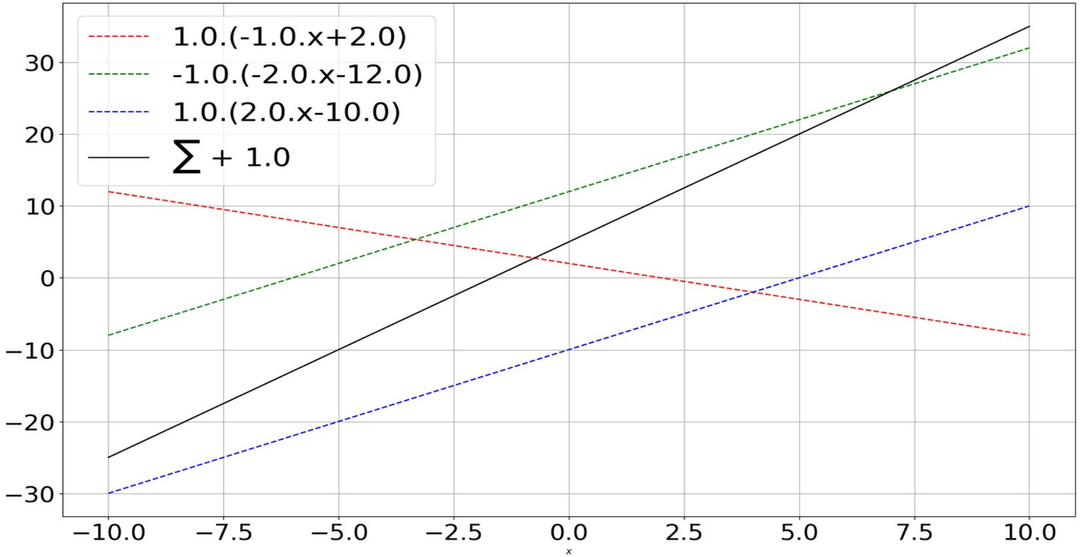
Une somme de fonctions linéaires est une fonction linéaire…

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

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

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

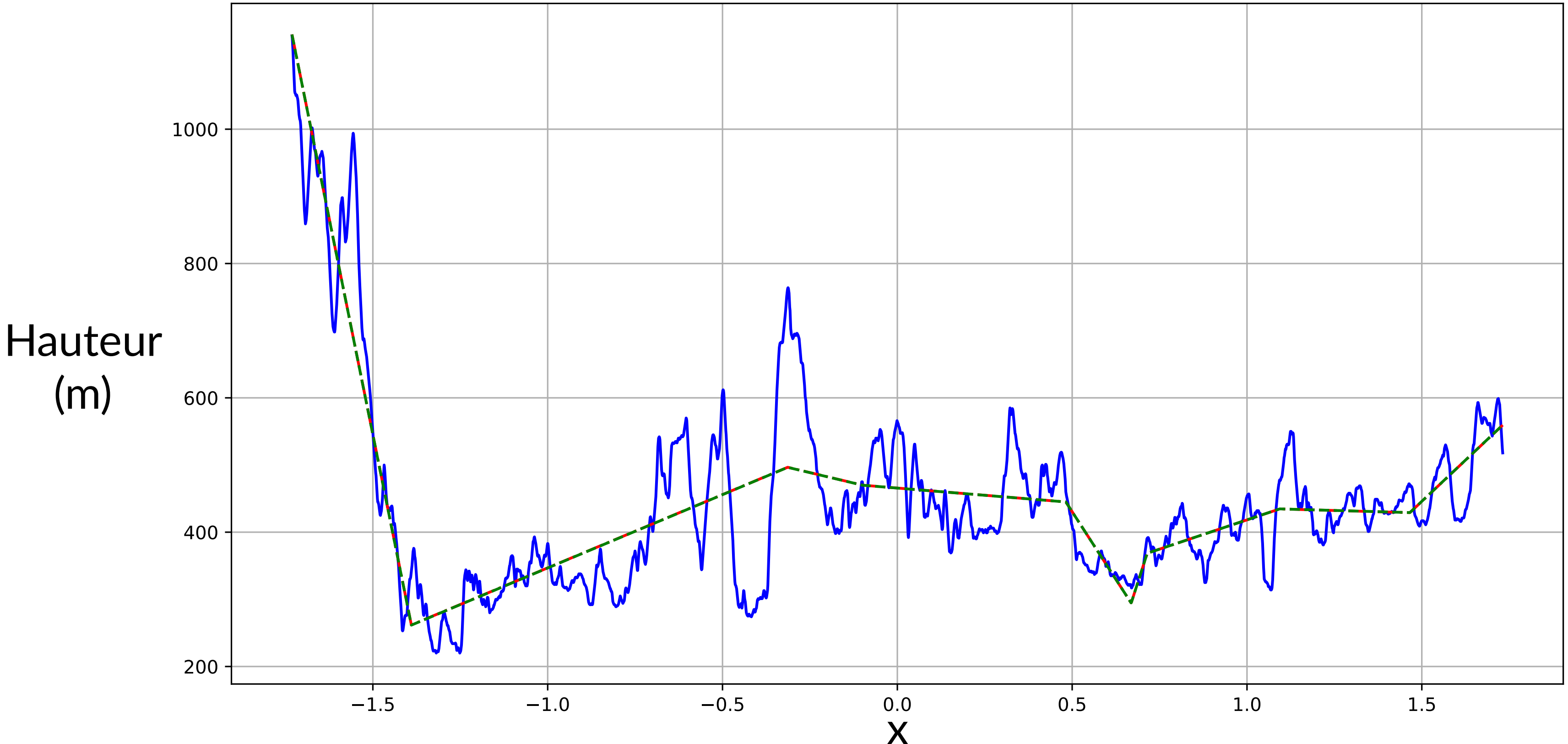
Et sans ReLU ?



Augmenter H n'accroît pas la **capacité** du réseau.

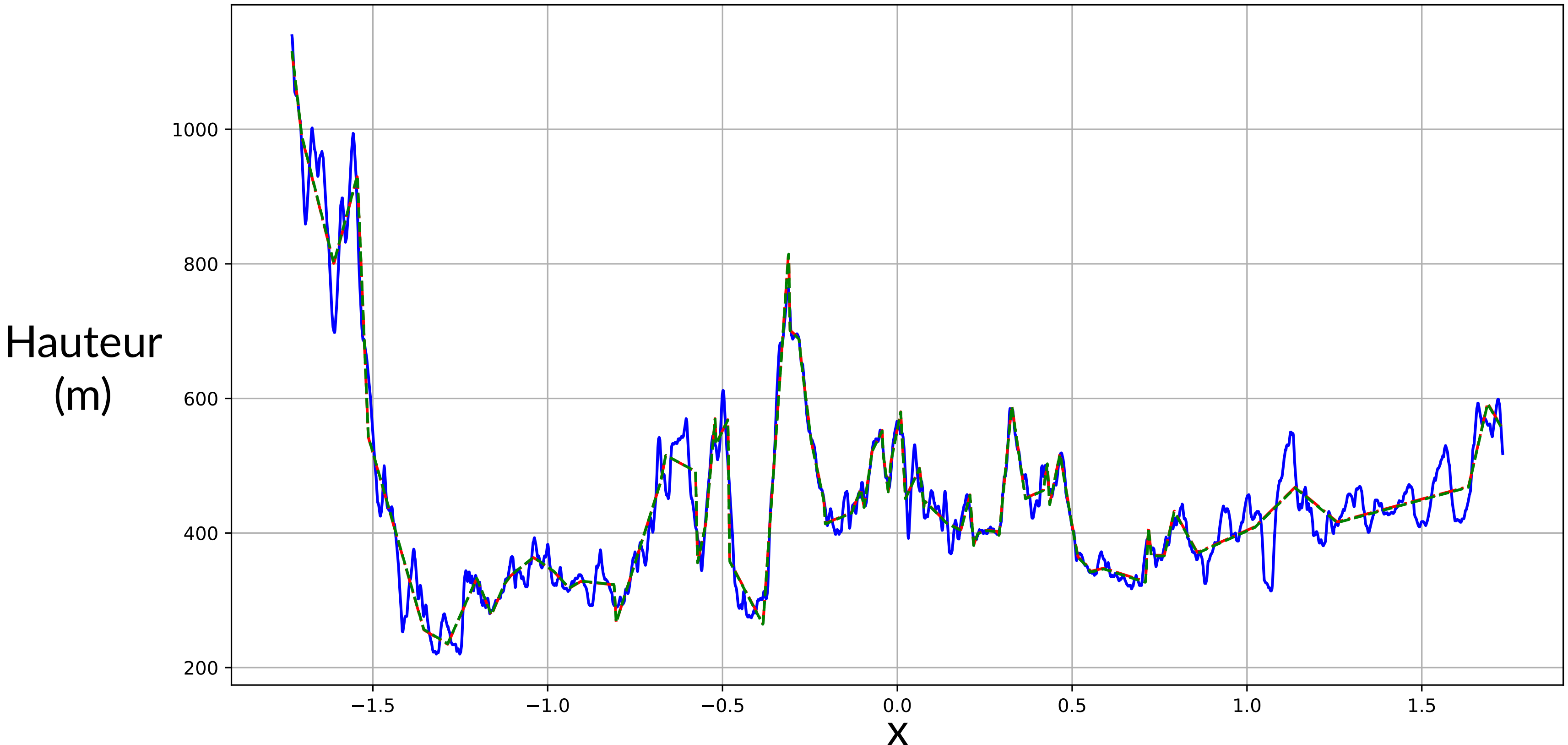
Étude du MLP à une « couche cachée » en 1D (suite bis)

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



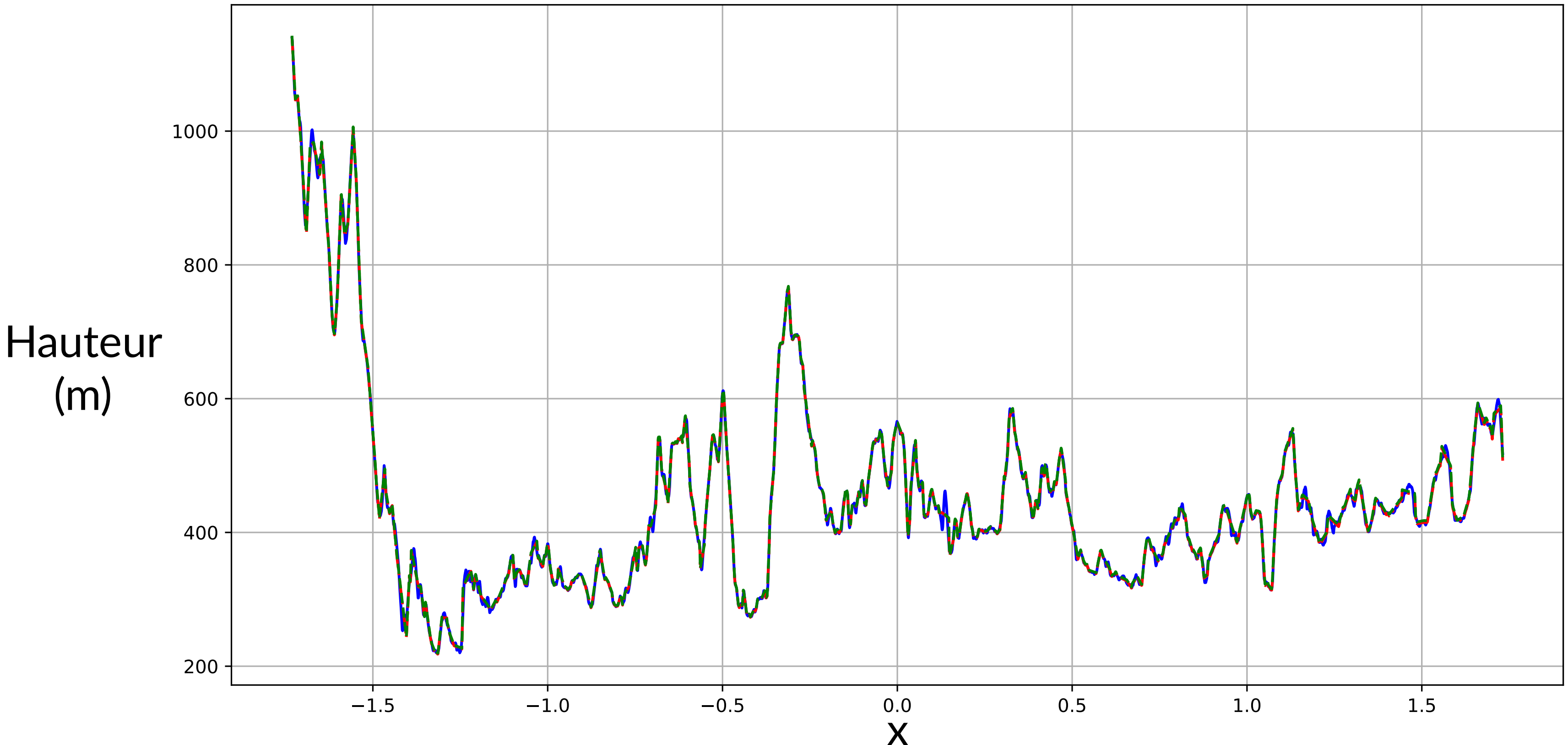
Étude du MLP à une « couche cachée » en 1D (suite bis)

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



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

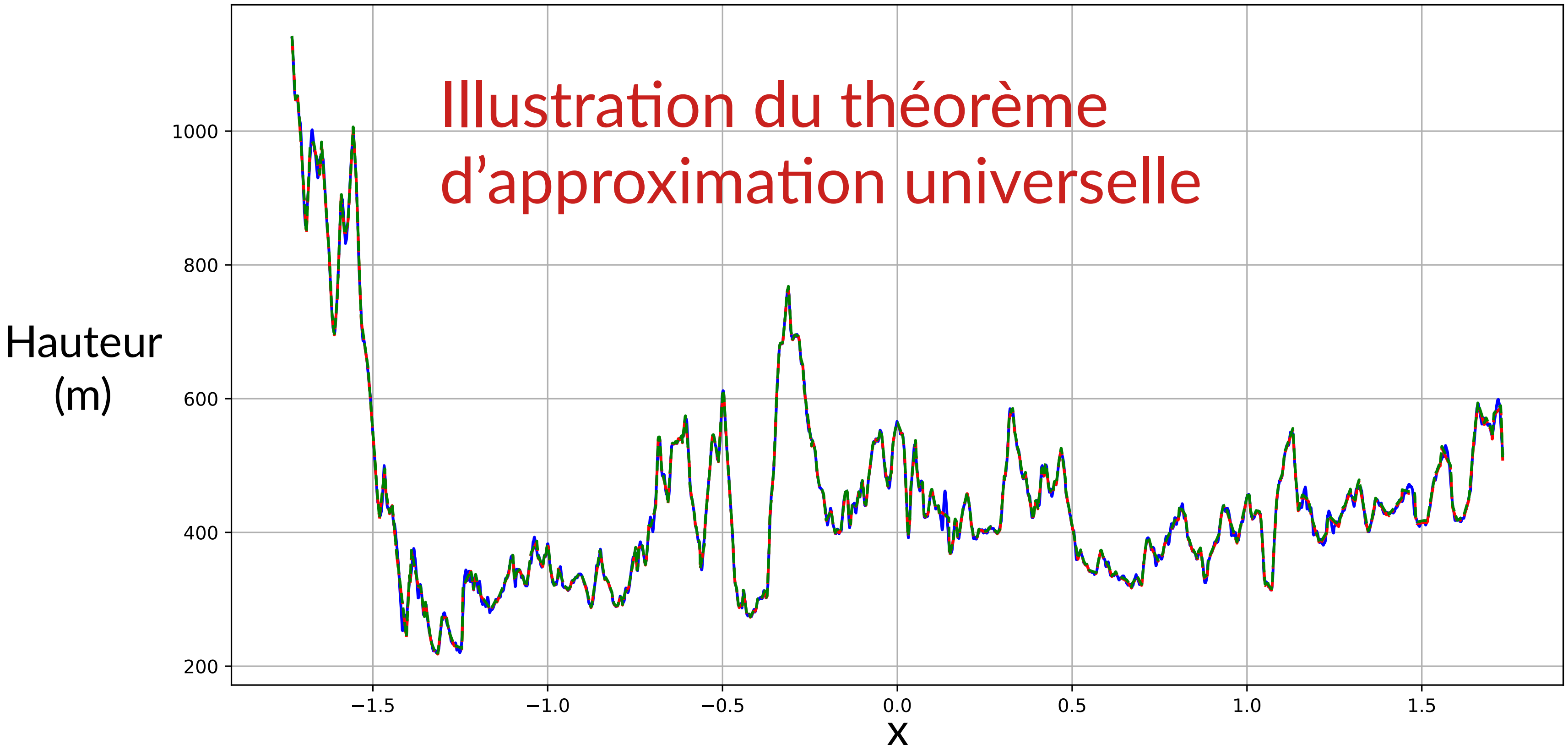
Optimisation d’un MLP sur un profil de terrain : H=1000



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 ter)

Mais alors pourquoi faire des réseaux de neurones profonds ?

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

Mais alors pourquoi faire des réseaux de neurones profonds ?

Théorème d'approximation universelle = MLP à une couche cachée peut **apprendre par cœur**

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

Mais alors pourquoi faire des réseaux de neurones profonds ?

Théorème d'approximation universelle = MLP à une couche cachée peut **apprendre par cœur**

Mais en général ce qui nous intéresse c'est :

Généralise bien ?

Efficace en calculs ?

Efficace en mémoire ?

Facile à optimiser ?

V) Enjeux

V)

IA et vie privée

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

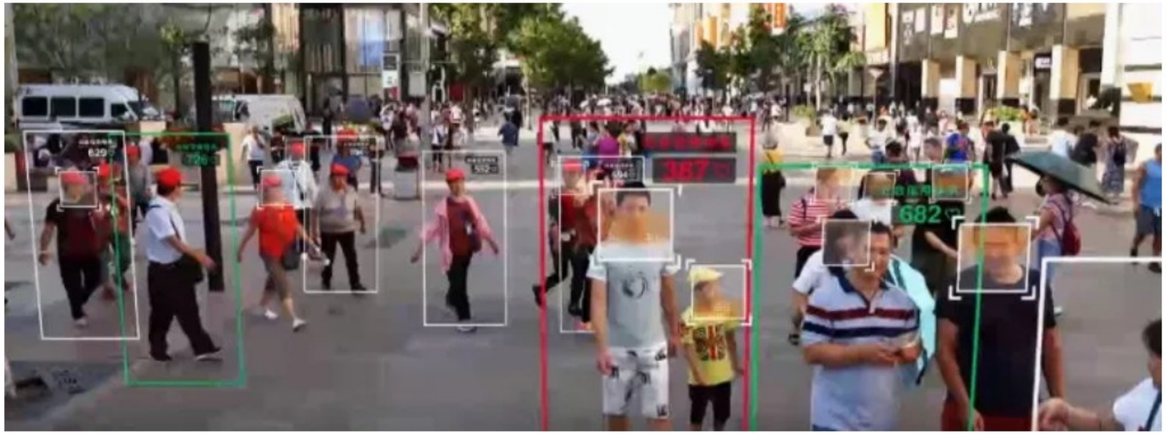
Dylan Curran

- 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

V)

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.

FLORIANE LECLERC | PUBLIÉ LE 25 OCTOBRE 2018 À 14H10

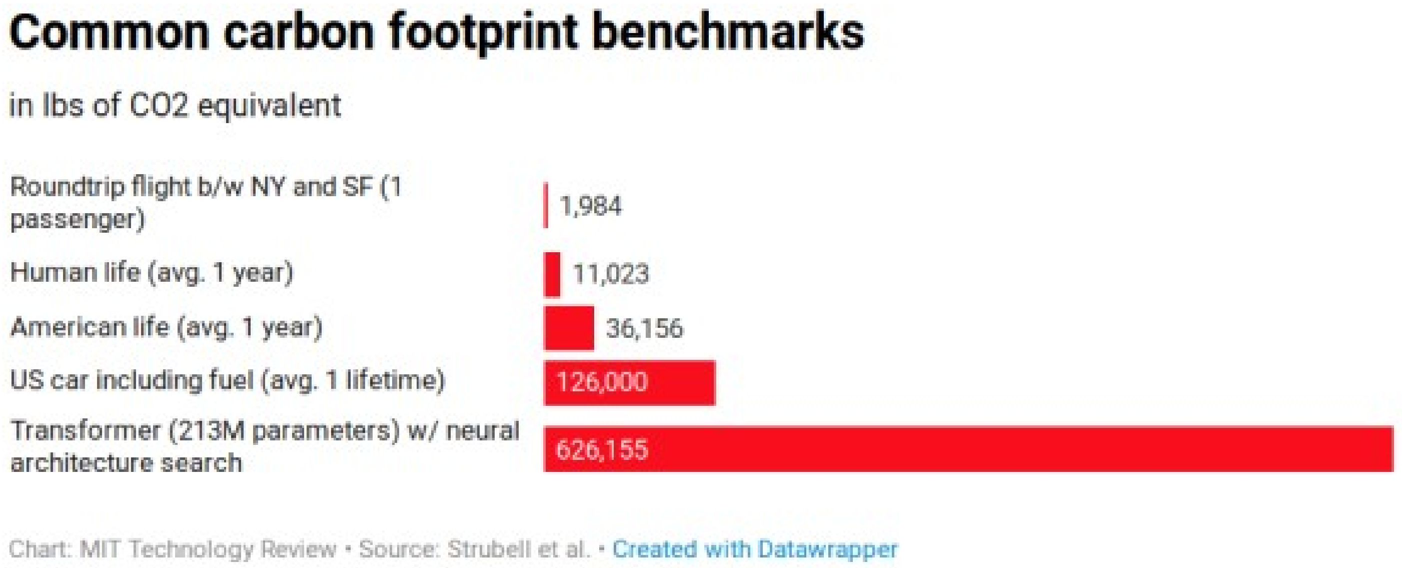
V)

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*

V)

IA et consommation énergétique



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

V)

IA et transhumanisme

NEURALINK

Calico