

Reconnaissance de caractères

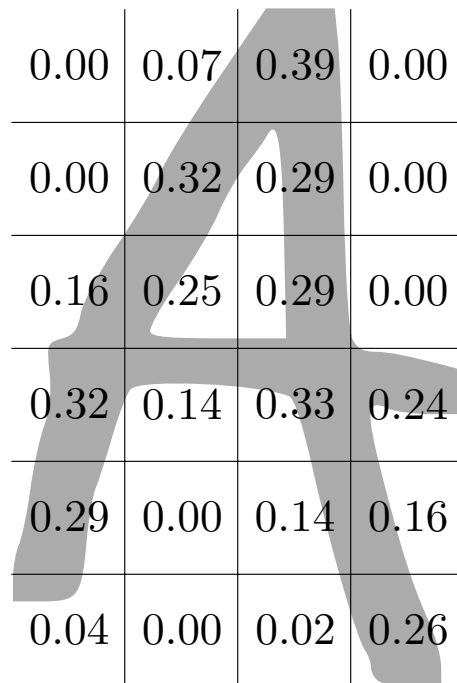
Principe général

- ▷ **Traitement d'images :**
 1. conversion en noir et blanc ;
 2. retrait des parasites éventuels ;
 3. séparation des caractères ;
 4. conversion en vecteur.
- ▷ **Reconnaissance des caractères :** utilisation d'un *réseau de neurones artificiels* et d'un algorithme d'apprentissage supervisé.

Reconnaissance de caractères

Conversion en vecteur

- ▷ On peut se contenter d'aplanir la matrice des pixels de l'image, mais vecteur trop grand.
- ▷ **Autre méthode** : placer une grille sur l'image, calculer la *proportion* de pixels noirs dans chaque zone, aplanir la nouvelle matrice obtenue.



0.00	0.07	0.39	0.00
0.00	0.32	0.29	0.00
0.16	0.25	0.29	0.00
0.32	0.14	0.33	0.24
0.29	0.00	0.14	0.16
0.04	0.00	0.02	0.26

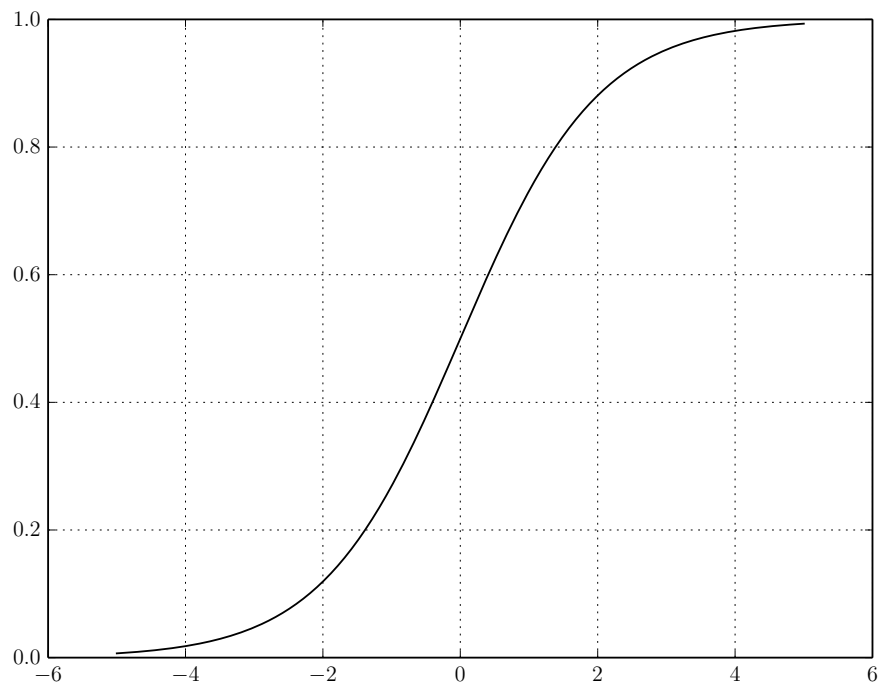
Reconnaissance de caractères

Réseaux et neurones

- ▷ **Neurone** : associe à un vecteur $x = (x_1, \dots, x_m)$ une sortie scalaire $\varphi(x)$, entrées pondérées par un *poids* p_i puis composition par une *fonction d'activation* f translatée par un *biais* b .

$$\varphi(x) = f\left(\sum_{i=1}^m p_i x_i - b\right)$$

- ▷ Fonction d'activation usuelle : $f(x) = \frac{1}{1+e^{-x}}$.

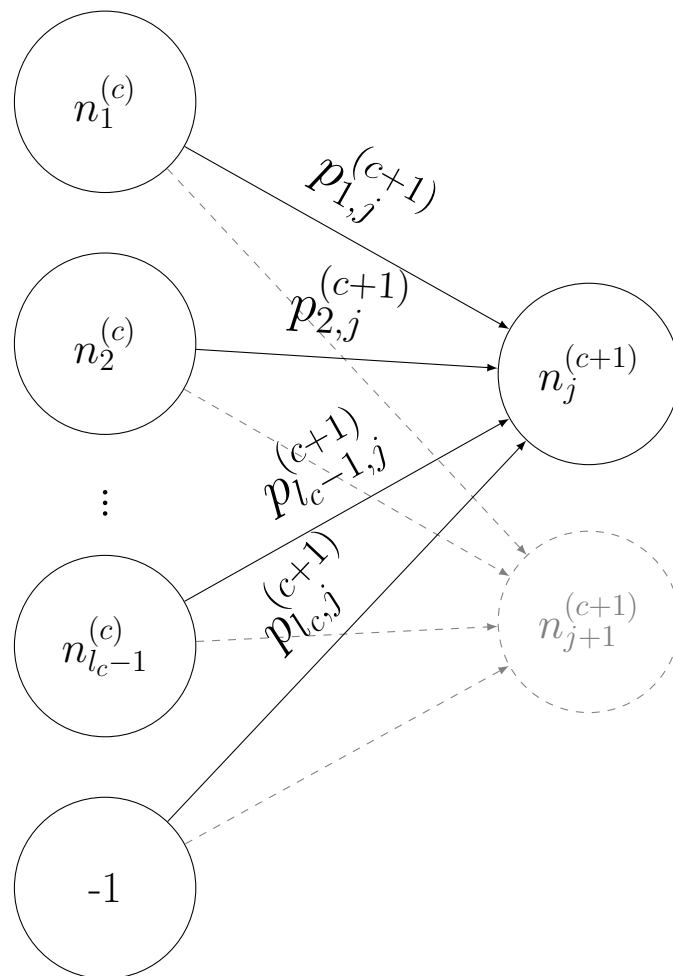


Reconnaissance de caractères

Réseaux et neurones

- ▷ **Réseau** : application de \mathbb{R}^p dans \mathbb{R}^q , modélisée par des couches de neurones.
- ▷ **Organisation en couches** : l'ensemble des sorties scalaires des neurones de la couche c notées $n_i^{(c)}$ sert d'entrée aux neurones de la couche $c + 1$:

$$n_j^{(c+1)} = f\left(\sum_{i=1}^{l_c} p_{i,j}^{(c+1)} n_i^{(c)}\right) = f(\sigma_j^{(c+1)}).$$



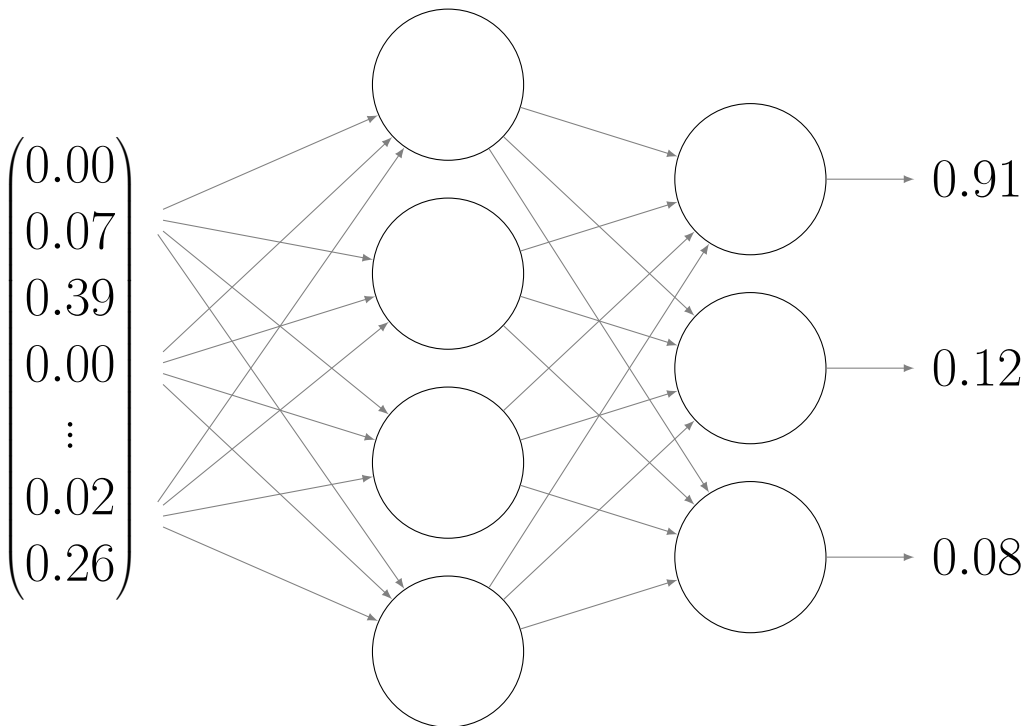
Reconnaissance de caractères

Reconnaissance

- ▷ On choisit une association entre vecteur de sortie dans \mathbb{R}^q et caractères à reconnaître :

$$\begin{pmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \mapsto \text{"A"} \qquad \begin{pmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \mapsto \text{"B"}.$$

- ▷ Si les poids sont adaptés, le principe de reconnaissance est simple.



Reconnaissance de caractères

Apprentissage supervisé

- ▷ Calcul de l'image y par le réseau de quelques échantillons x .



- ▷ Comparaison de y et de la valeur vectorielle t que l'on souhaite associer aux exemples à l'aide d'une fonction d'erreur :

$$E = \frac{1}{2} \sum_{i=1}^q (t_i - y_i)^2.$$

- ▷ Correction des valeurs des poids, modulée par un *taux d'apprentissage* α que l'on fixe au départ :

$$\Delta p_{j,k}^{(c)} = -\alpha \frac{\partial E}{\partial p_{j,k}^{(c)}}.$$

- ▷ On répète ces étapes pour chaque caractère à reconnaître (associé à un unique vecteur t).

Reconnaissance de caractères

Rétropropagation

- ▷ Posons pour tout neurone indicé k d'une couche c

$$e_k^{(c)} = -\frac{\partial E}{\partial p_{j,k}^{(c)}} \frac{1}{n_j^{(c-1)}}.$$

- ▷ La correction à apporter à chacun des poids du réseau devient

$$\Delta p_{j,k}^{(c)} = \alpha e_k^{(c)} n_j^{(c-1)}.$$

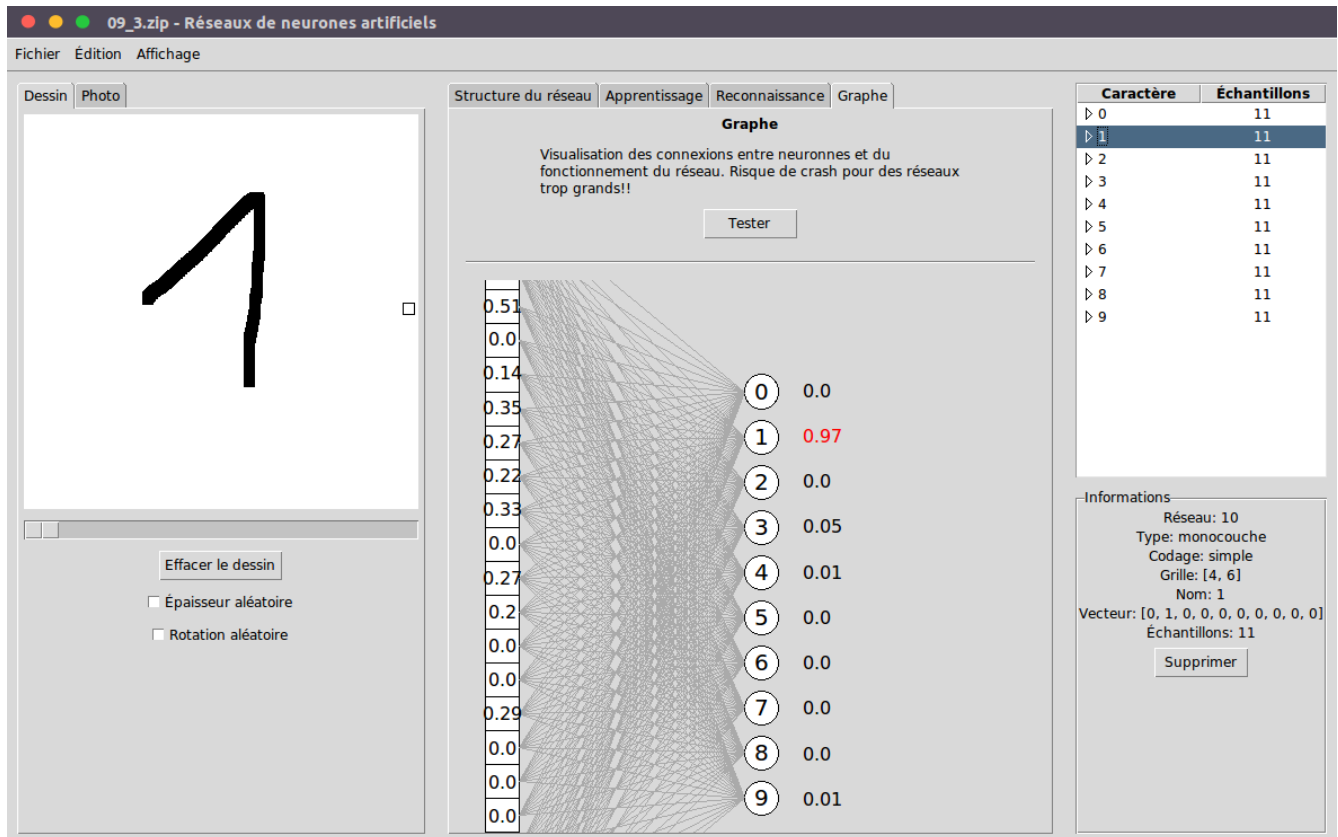
- ▷ On montre à l'aide de la formule de la chaîne les relations suivantes (réseau à n couches) :

$$e_k^{(n)} = f'(\sigma_k^{(n)})(t_k - y_k)$$

et $e_k^{(c)} = f'(\sigma_k^{(c)}) \sum_{i=1}^{l_{c+1}} p_{k,i}^{(c+1)} e_i^{(c+1)}.$

Reconnaissance de caractères

Réalisation Python

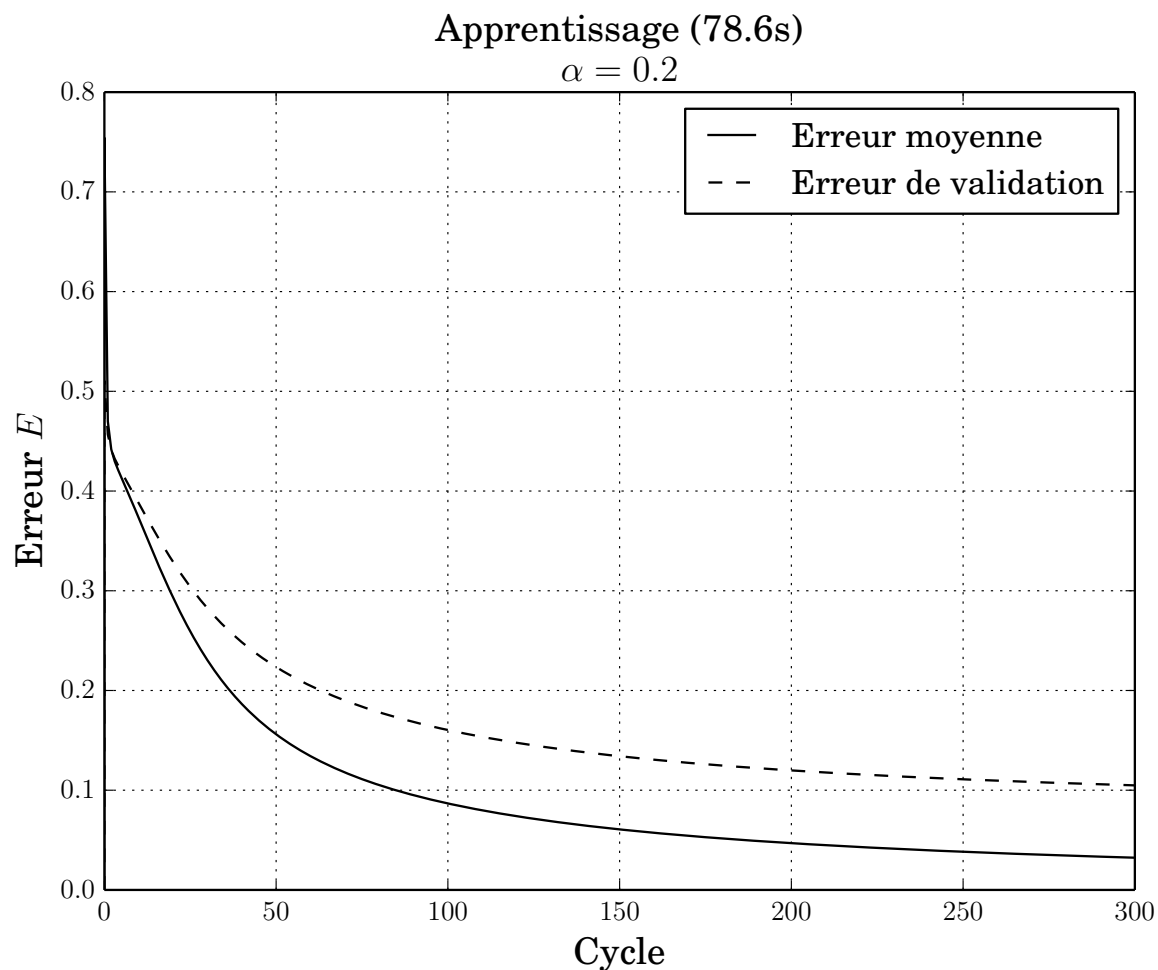


- ▷ Création d'une interface en Python permettant de créer un réseau et de l'entraîner à partir d'échantillons entrés à la souris.

Reconnaissance de caractères

Résultats expérimentaux

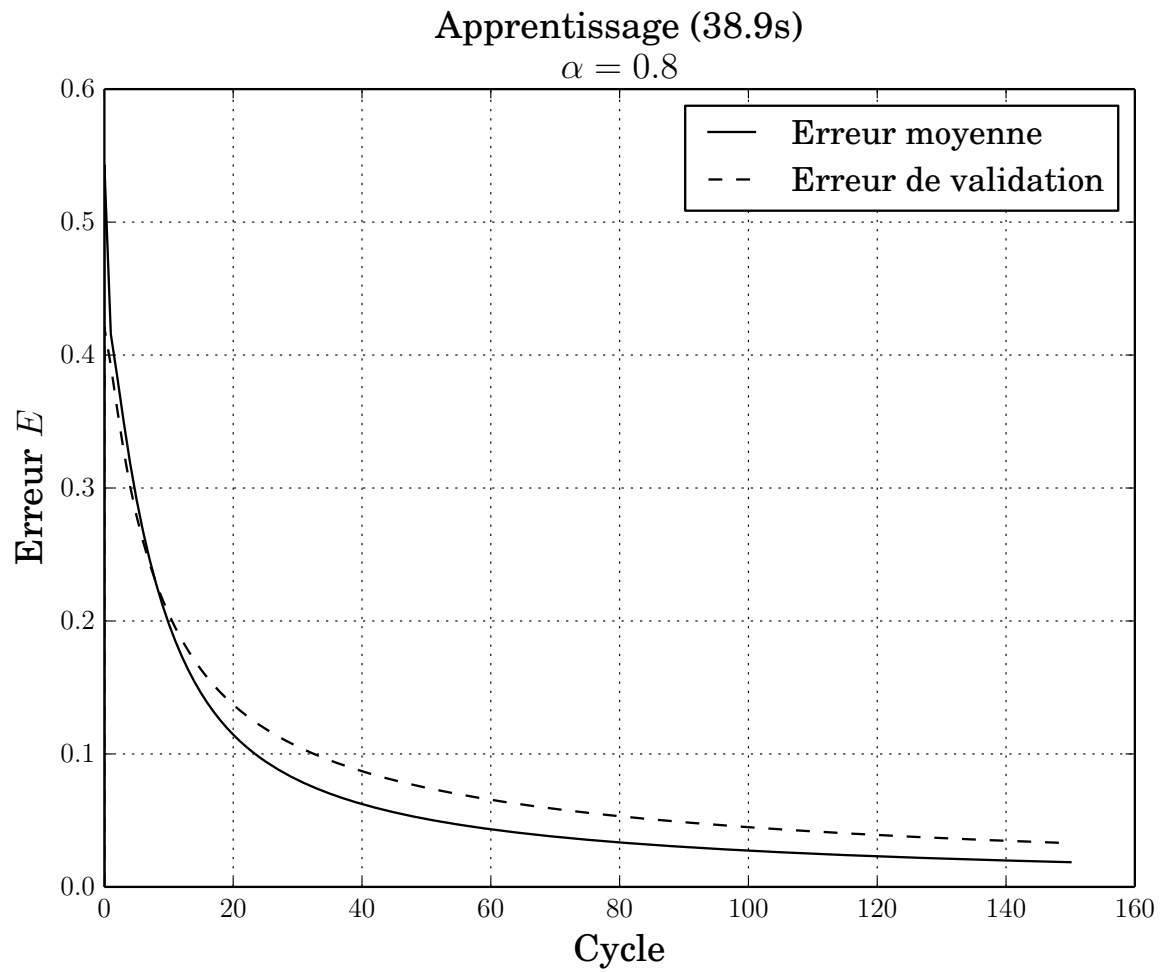
- ▷ **Entrée** : 3×5 .
- ▷ **Sortie** : chiffres de 0 à 9.
- ▷ **Échantillons** : 10 par classe pour l'apprentissage, 1 par classe pour la validation.



Monocouche

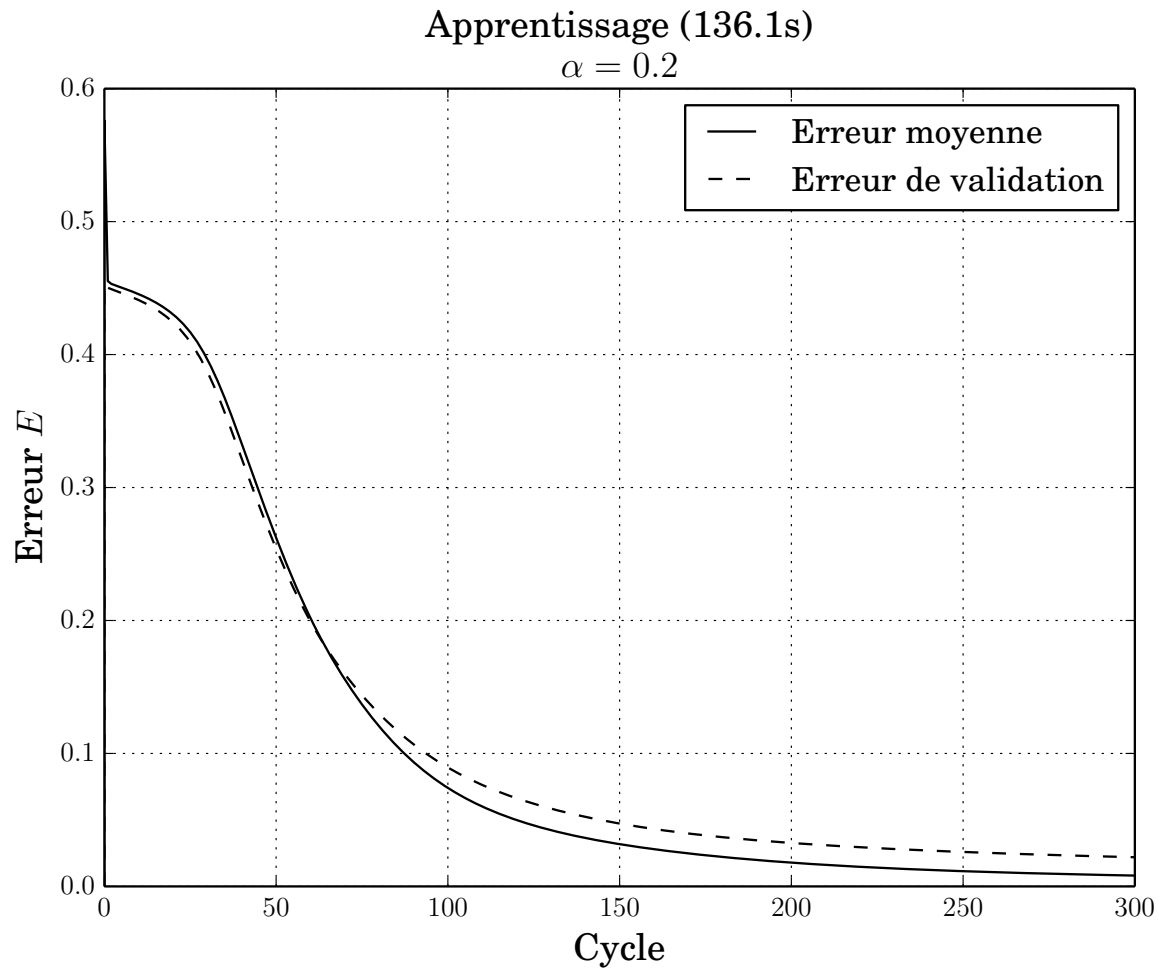
Reconnaissance de caractères

Résultats expérimentaux



Reconnaissance de caractères

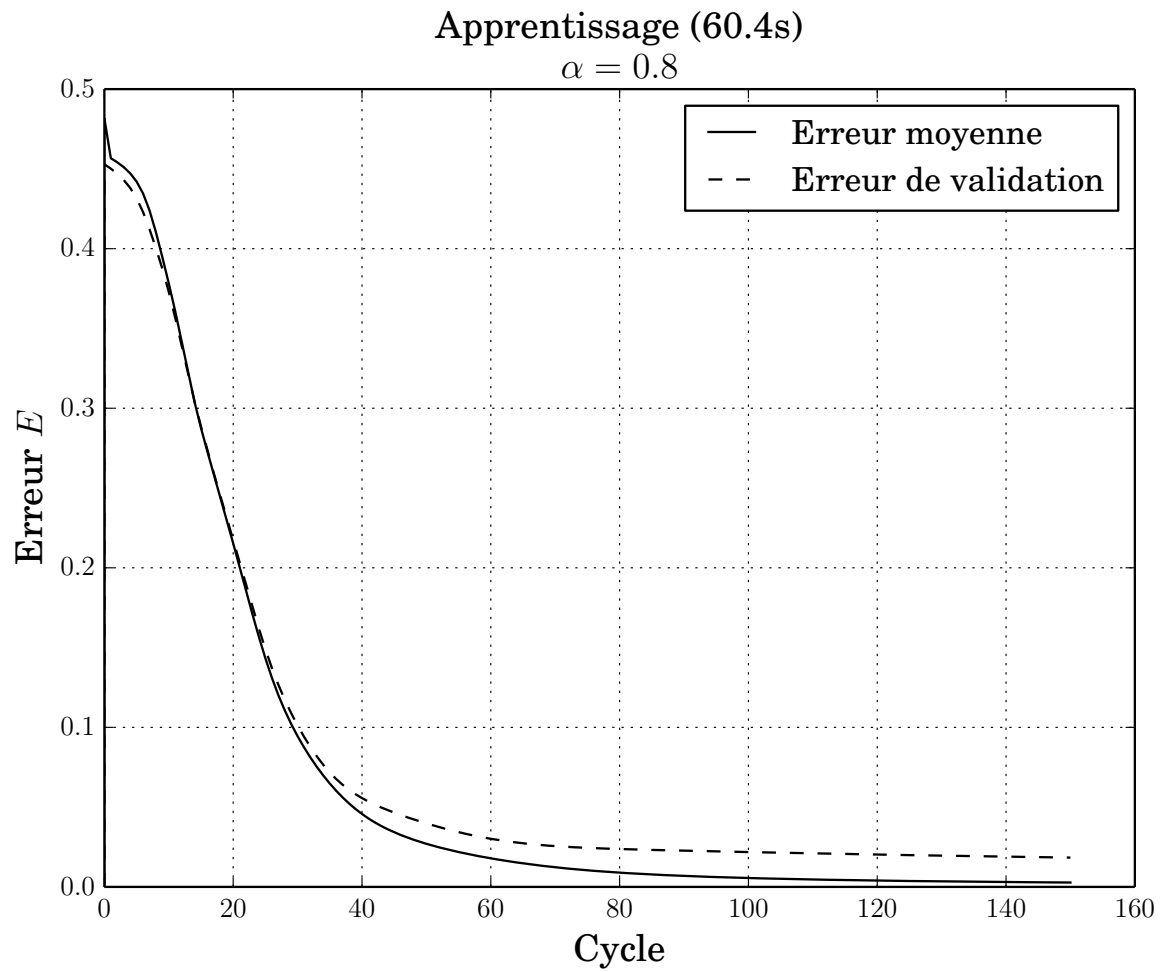
Résultats expérimentaux



Multicouche (16, 10)

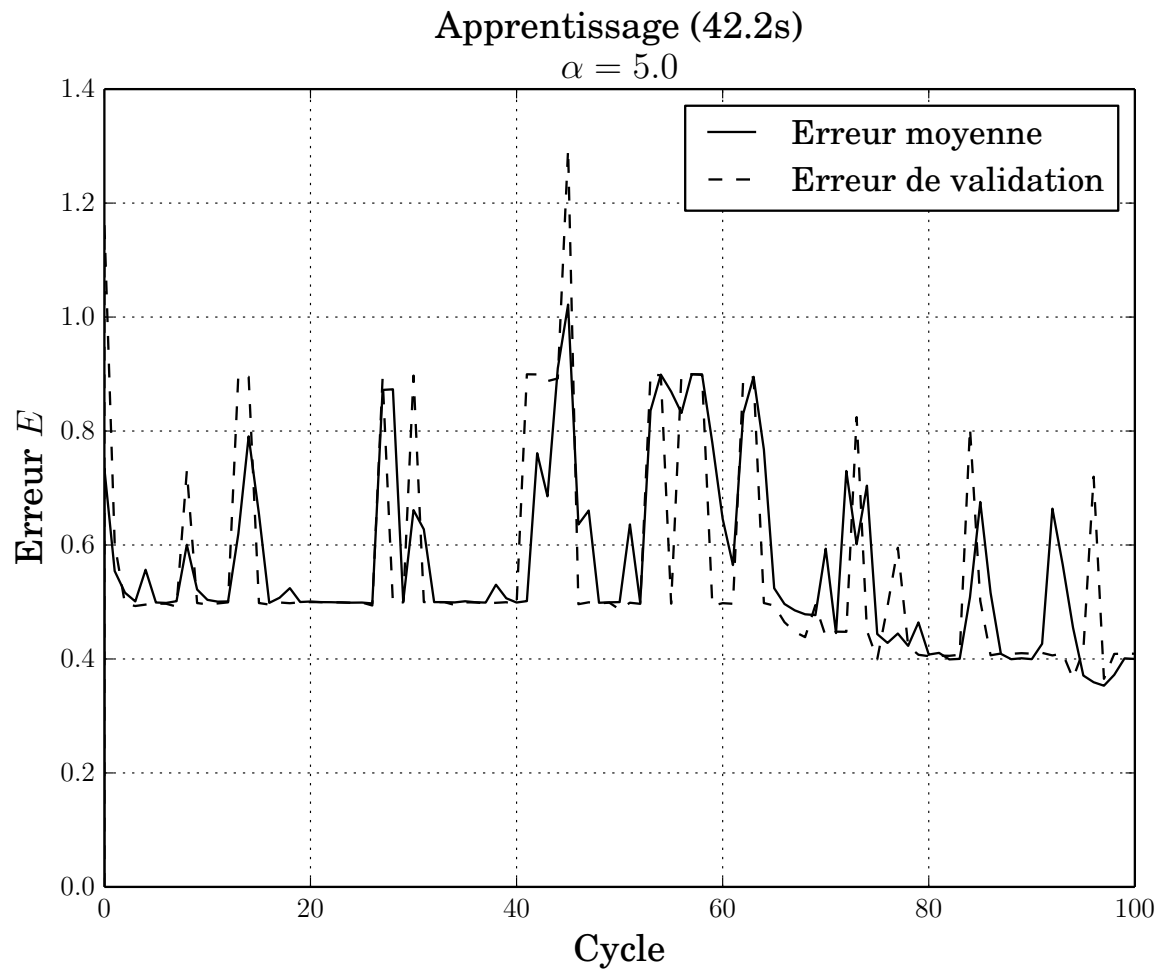
Reconnaissance de caractères

Résultats expérimentaux



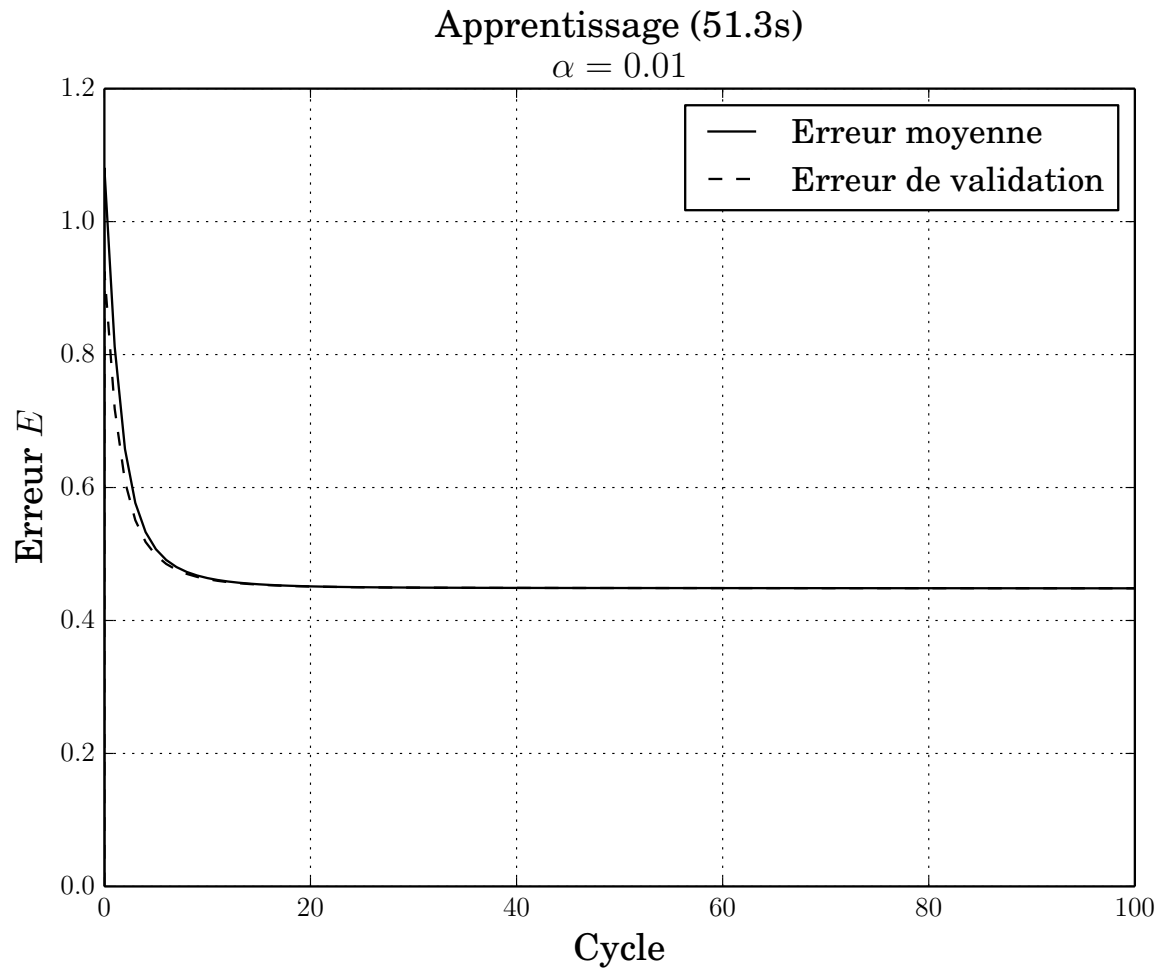
Reconnaissance de caractères

Résultats expérimentaux



Reconnaissance de caractères

Résultats expérimentaux



Multicouche (16, 12, 10)

Reconnaissance de caractères

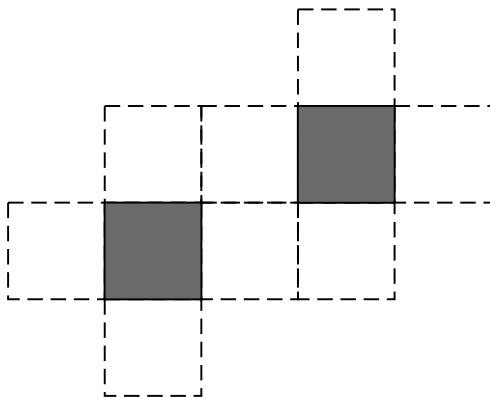
Méthode des k plus proches voisins

- ▷ Alternative très simple aux réseaux de neurones.
- ▷ On convertit en vecteur l'image du caractère à reconnaître.
- ▷ L'entier k est fixé. On sélectionne dans la base d'échantillons les k plus proches vecteurs du vecteur image pour la norme euclidienne.
- ▷ On identifie la classe la plus représentée parmi ces k vecteurs.
- ▷ **Avantage** : pas de phase d'apprentissage, uniquement besoin d'une base d'échantillons.
- ▷ **Inconvénient** : il faut avoir une large base d'échantillons accessibles lors de la reconnaissance.

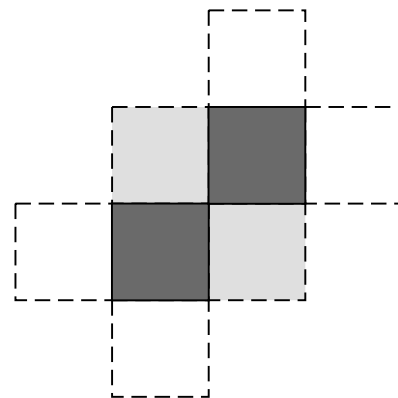
Reconnaissance de caractères

Un algorithme de séparation

- ▷ Sélection de chaque pixel et de ses pixels adjacents et stockage des groupes temporaires ainsi formés dans une liste T .
- ▷ Initialisation d'une liste (vide) de caractères C (groupes de pixels).
- ▷ Pour chaque groupe temporaire $a \in T$, et pour chaque caractère $b \in C$, a prend la valeur $a \cup b$ si $a \cap b \neq \emptyset$ et on enlève b de C .
- ▷ À la fin de la boucle qui parcourt T , on ajoute a à C , même s'il n'a pas été modifié.



pas d'intersection
caractères différents



intersection
même caractère