
OCR Word Search Solver

Alexandre Joaquim Lima Salgueiro Hugo Guyennet
alexandre-joaquim.lima-salgueiro hugo.guyennet

Léa-Angéline Kolmerschlag Tom Huynh
lea-angelina.kolmerschlag tom.huynh

4 novembre 2025

Sommaire

| | | |
|----------|---|----------|
| 1 | Réseau de neurone pour la reconnaissance de caractères | 1 |
| 1.1 | Représentation des données en tenseur | 1 |
| 1.1.1 | Représentation en Mémoire et Performance | 2 |
| 2 | Architecture du Réseau de Neurones | 2 |
| 2.1 | Couche linéaire | 3 |
| 2.1.1 | Passe Avant (Forward Pass) | 3 |
| 2.1.2 | Passe Arrière (Backward Pass) | 3 |
| 2.2 | Optimisation de la Couche Linéaire | 3 |
| 2.2.1 | Optimisations Actuelles | 4 |
| 2.2.2 | Pistes d'Améliorations | 4 |

1 Réseau de neurone pour la reconnaissance de caractères

Pour la reconnaissance de caractères, nous utiliserons un réseau de neurones convolutif (CNN).

Un CNN est un type de réseau de neurones spécialement conçu pour le traitement d'images. Sa structure s'inspire du cortex visuel animal. Il utilise des couches de convolution pour extraire des caractéristiques hiérarchiques des images, comme les contours, les formes, puis des objets plus complexes. Ces couches sont suivies de couches de pooling, qui réduisent la taille des données pour en conserver les informations essentielles. Finalement, des couches entièrement connectées, similaires à celles d'un réseau de neurones classique, effectuent la classification finale pour identifier le caractère. Cette architecture rend les CNN particulièrement performants pour la reconnaissance de motifs dans les images.

Note:

Pour accélérer les calculs intensifs de notre réseau de neurones, nous exploitons les instructions SIMD (Single Instruction, Multiple Data). Le principe du SIMD est d'effectuer une seule opération sur plusieurs données simultanément. Les processeurs modernes disposent de registres spéciaux pouvant contenir des vecteurs de données (par exemple, 8 nombres flottants). Une seule instruction SIMD peut alors additionner ou multiplier tous ces nombres en un seul cycle d'horloge. L'utilisation de ces instructions, notamment via les intrinsèques AVX2, permet de paralléliser les calculs au plus bas niveau et d'obtenir des gains de performance considérables.

1.1 Représentation des données en tenseur

Au cœur de notre réseau de neurones se trouve le **tenseur**, une structure de données fondamentale. On peut le voir comme une généralisation des vecteurs (1 dimension) et des matrices (2 dimensions) à un nombre arbitraire de dimensions. Dans notre projet, toutes les données — qu'il s'agisse des images d'entrée, des poids des couches ou des résultats intermédiaires — sont unifiées sous cette forme.

1.1.1 Représentation en Mémoire et Performance

L'efficacité de la manipulation des tenseurs repose sur une meilleure représentation en mémoire. Plutôt que d'utiliser des structures de données imbriquées, un tenseur stocke toutes ses valeurs dans un **bloc de mémoire contigu**, c'est-à-dire un simple tableau unidimensionnel.

La structure multidimensionnelle devient alors une abstraction. Elle est définie par des méta-informations, dont la plus importante est sa **forme**. La forme décrit la taille de chaque dimension et permet d'interpréter le tableau de données brutes comme une grille multidimensionnelle. Par exemple, un tenseur de forme (28, 28) contient 784 éléments stockés séquentiellement, que sa forme nous permet de reconstituer en une grille de 28x28.

Cette organisation est cruciale pour les performances. Elle garantit une localité des données qui optimise l'utilisation de la mémoire cache du processeur et permet une meilleure utilisation des instructions **SIMD**.

2 Architecture du Réseau de Neurones

Le modèle de réseau de neurones convolutif (CNN) que nous avons conçu est structuré en plusieurs blocs, chacun ayant un rôle spécifique dans l'extraction des caractéristiques des images de lettres manuscrites. L'architecture est la suivante, et chaque composant sera détaillé dans les sections à venir :

- **Bloc convolutif 1**

- Une couche de convolution 2D avec 32 filtres de taille 3x3, une stride de 1 et un padding de 1. L'entrée est une image monocanal (1 canal).
- Une fonction d'activation ReLU.
- Une couche de convolution 2D avec 64 filtres de taille 1x1, une stride de 1 et aucun padding.
- Une fonction d'activation ReLU.
- Une couche de Max Pooling 2D de taille 2x2.
- Une couche de Dropout 2D avec un taux de 0.20.

- **Bloc convolutif 2**

- Une couche de convolution 2D avec 64 filtres de taille 3x3, une stride de 1 et un padding de 1.
- Une fonction d'activation ReLU.
- Une couche de convolution 2D avec 128 filtres de taille 1x1, une stride de 1 et aucun padding.
- Une fonction d'activation ReLU.
- Une couche de Max Pooling 2D de taille 2x2.
- Une couche de Dropout 2D avec un taux de 0.20.

- **Bloc convolutif 3**

- Une couche de convolution 2D avec 128 filtres de taille 3x3, une stride de 1 et un padding de 1.
- Une fonction d'activation ReLU.
- Une couche de convolution 2D avec 256 filtres de taille 1x1, une stride de 1 et aucun padding.
- Une fonction d'activation ReLU.
- Une couche de Max Pooling 2D de taille 2x2.
- Une couche de Dropout 2D avec un taux de 0.20.

- **Couches de classification**

- Aplatissement (Flattening) des cartes de caractéristiques. Pour une image d'entrée de 28x28, la taille après les 3 blocs est de $256 \times 3 \times 3 = 2304$.
- Une couche entièrement connectée (Dense) avec 128 neurones.
- Une fonction d'activation ReLU.
- Une couche de Dropout avec un taux de 0.45.
- Une couche de sortie entièrement connectée avec 26 neurones, correspondant aux 26 lettres de l'alphabet.

La fonction de perte utilisée est l'entropie croisée (Cross-Entropy Loss), adaptée pour les problèmes de classification multiclasse.

2.1 Couche linéaire