

# Réseau associatif pour l'OCR

COGNITION ARTIFICIELLE - LICENCE DE SCIENCES COGNITIVES

*Didier Puzenat*

didier.puzenat@univ-lyon2.fr

## 1 Introduction

Le but de ce travail est de réaliser un programme capable de reconnaître des caractères dactylographiés. Ce problème est connu sous le nom d'OCR (*Optical Characters Recognition*). On considère les caractères discrétisés de 5 sur 5 points suivants :

```
###  ####  ####  ####  #####  #####  #####  #  #  #  ###  #  #  #  ## ##
#  #  #  #  #  #  #  #  #  #  #  #  #  #  #  #  #  #  #  #  #  #  #  #  #
#####  ####  #  #  #  ###  ###  #  #####  #  #  ###  #  #  #  #  #
#  #  #  #  #  #  #  #  #  #  #  #  #  #  #  #  #  #  #  #  #  #  #  #
#  #  ####  ####  ####  #####  #  #####  #  #  #  ###  #  #  #####  #  #

#  #  ###  ####  #####  #####  #####  #####  #  #  #  #  #  #  #  #  #  #  #  #####
##  #  #  #  #  #  #  #  #  #  #  #  #  #  #  #  #  #  #  #  #  #  #  #
#  #  #  #  #  ####  #  #  #####  ###  #  #  #  #  #  #  #  #  #  #  #
#  ##  #  #  #  #  #  ##  #  #  #  #  #  #  #  #  #  #  #  #  #  #
#  #  ###  #  #####  #  #  #####  #  ###  #  #  #  #  #  #  #  #####
```

Le programme sera fait en Python. Un fichier Python semi-vierge avec les caractères sera disponible. Et évidemment une capsule permettra de démarrer sereinement. À noter que bien que ce réseau ressemble à un réseau de Hopfield, nous allons utiliser notre propre algorithme d'apprentissage, imaginé spécifiquement pour ce TD-TP, pour démontrer qu'on n'est pas obligé de toujours suivre un modèle existant.

## 2 Réseau Associatif

### 2.1 Structure

Nous allons utiliser un réseau récurrent (cellules actives ou inactives) caractérisé par : une interconnexion complète, des poids et seuils réels, des connexions symétriques. Le réseau sera initialisé à 0,5 pour les poids et 0 pour les seuils.

## 2.2 Règle d'apprentissage

Lors de l'apprentissage, on modifie les poids des cellules si leur activation ne respecte pas la forme apprise (après propagation). C'est, par exemple, le cas si une cellule s'active alors que l'on désire la garder inactive pour la forme apprise. On calcule la différence entre la valeur obtenue et la valeur désirée. Cette *erreur* est distribuée sur les connections arrivant sur la cellule, ainsi que sur le seuil de la cellule.

### Remarques :

- attention au signe pour la modification du seuil,
- il faut veiller à garder la matrice symétrique lors de la modification des poids,
- l'ajout d'un  $\epsilon$  (par exemple 0.2) à l'erreur permet de s'assurer du résultat.

## 3 Algorithme d'apprentissage

### Pour toutes les formes :

1. Présentation d'une forme en activant les cellules
2. Activation des cellules en asynchrone
3. Modification des poids si nécessaire

On recommence jusqu'à ce que toutes les formes soient apprises, c'est à dire jusqu'à ce l'on puisse présenter toutes les formes sans avoir à faire de modification des poids.

## 4 Travail à faire

Le programme doit apprendre un maximum de lettre de l'alphabet, dans l'idéal les 26 lettres... Une fois les formes apprises, le réseau doit être capable de retrouver une lettre en partant d'une lettre bruitée. L'exemple ci-contre illustre le comportement désiré...

# # #	# #
## #	## #
# #	# # #
# ##	# ##
# #	# #

## 5 Travail à rendre

Le travail sera comme d'habitude à rendre sur Moodle.