Université de Cergy-Pontoise,
UFR Sciences & Techniques,
Département des sciences informatiques,
Master 1 I&ISC

Intelligence Artificielle

TD 3 : Auto-encodeurs

Louis Annabi

27 avril 2021

Le but de ce TP est d'implémenter et d'entraîner un auto-encodeur sur la base de données MNIST : http://yann.lecun.com/exdb/mnist/

1 Rappel du cours

1.1 Rétro-propagation

La rétro-propagation (backpropagation ou backprop en anglais) est l'algorithme d'optimisation dominant en deep learning. Il permet d'optimiser l'ensemble des paramètres d'un réseau de neurones en propageant le gradient par rapport à une erreur, en général calculée au niveau d'une couche de sortie (même si on peut ajouter à cette erreur des quantité dépendant des activations des couches intermédiaires ou directement des paramètres).

Dans ce TP, il n'est pas nécessaire d'implémenter l'algorithme de backpropagation. Nous utiliserons une bibliothèque (`pytorch`) s'occupant de ces calculs, nous avons juste à définir le réseau de neurones à entraîner.

1.2 Auto-encodeur

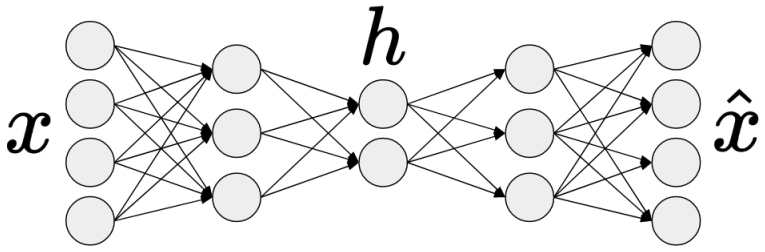

FIGURE 1 – Schéma d'un auto-encodeur.

Un auto-encodeur (figure 1) est un réseau de neurones entraîné par apprentissage auto-supervisé. Il est composé d'un encodeur et d'un décodeur. L'encodeur transforme l'entrée x en une représentation h, en général de plus petite dimension (compression). Le décodeur calcule une reconstruction $\hat{x}$ de l'entrée x à partir de la représentation h.

2 Travail demandé

Pour ce TP, vous aurez besoin d'installer la bibliothèque `pytorch` sur python : https://pytorch.org/get-started/locally/.

2.1 Implémentations

— Implémentez un modèle héritant de `torch.nn.Module` d'auto-encodeur utilisant un perceptron pour l'encodeur et un perceptron pour le décodeur. La dimension d de la représentation h est un paramètre du modèle.
— Implémentez la boucle d'apprentissage de votre modèle sur la base de données MNIST.
— Implémentez une fonction permettant d'ajouter un bruit blanc sur toutes les valeurs des pixels d'un batch d'images.

2.2 Analyses

— Entraînez le modèle en variant la dimension de la représentation : $d = 16$, $d = 32$, $d = 64$ et comparez vos résultats. Quelle dimension permet la meilleure reconstruction ?
— Affichez les poids synaptiques du décodeur sous forme d'images 28x28. Est-ce que certains de ces motifs ressemblent à des chiffres ?
— L'auto-encodeur peut aussi servir à retirer le bruit d'une image d'entrée. Après apprentissage, donner à votre auto-encodeur des images bruitées grâce à la fonction que vous aurez implémentée. Vous pouvez mesurer la qualité du débruitage en calculant l'erreur entre l'image d'entrée avant application du bruit et l'image reconstruite par l'auto-encodeur. Analysez l'impact de l'amplitude du bruit sur la qualité du débruitage.

2.3 Questions

— Quels sont les intérêts d'apprendre une représentation h de nos entrées x ?
— Quel est le nombre de paramètres dans le réseau en fonction de la dimension d de la représentation ?
— Supposons que nous souhaitions utiliser notre auto-encodeur pour compresser et envoyer des images. En supposant que le destinataire de l'image ait à sa disposition le décodeur, on peut se contenter de lui envoyer la représentation h correspondant à chaque image. Quelle sera la taille relative de notre fichier par rapport à l'image complète de taille 28x28 ?

2.4 Pour aller plus loin

Il est possible d'ajouter des contraintes sur la représentation apprise. Par exemple, il peut être intéressant de construire une représentation *sparse*, c'est à dire une représentation dans laquelle la plupart des valeurs sont à 0.

Pour ce faire, on peut ajouter à la loss de reconstruction deux nouveaux termes :
— Un terme pénalisant la norme 1 de la représentation $\|h\|_1$. Ce terme est très souvent utilisé pour encourager une activation *sparse* dans une couche d'un réseau de neurones.
— Un terme pénalisant la norme 2 des poids du décodeur. Ce terme est appelé régularisation L2, sert à empêcher les poids de prendre des valeurs trop grandes. La régularisation L2 peut être utilisée notamment pour contrer l'*overfitting* en apprentissage supervisé.

Les trois termes composant la loss finale doivent être pondérés judicieusement. Il pourrait notamment être intéressant de partir d'un poids faible pour le terme $\|h\|_1$ puis de l'augmenter progressivement au cours de l'apprentissage.