TP
Restricted Boltzman Machines

Thomas Verrecchia, Sarah Abbana Bennani
22 Décembre 2023

Introduction

Ce document détail un ensemble de résultats expérimentaux sur l'utilisation de modèle d'IA génératives sur différents Dataset. Les différents algorithmes utilisés ont été étudiés en CM, nous omettrons donc d'expliquer leur fonctionnement et nous nous concentrerons sur leurs performances.

Nous allons étudier dans un premier temps les performances du modèle RBM sur le dataset Binary Alphadigits en fonction des hyperparamètres et du nombre de caractères à apprendre. Nous comparerons ensuite ces résultats avec ceux du RBM sur d'autres bases de données puis avec ceux d'autres modèles.

1 Influence des hyperparamètres sur le modèle RBM

Le modèle RBM se base sur l'utilisation d'hyperparamètres qui influence grandement les performances de l'algorithme :

- q le nombre d'unités cachées.
- nb_iter le nombre d'itérations lors de l'apprentissage.
- lr le learning rate

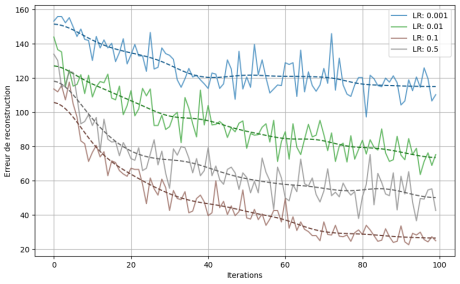
Pour pouvoir déterminer l'influence de ces hyperparamètres sur les performances de l'algorithme, nous allons fixer ces paramètres ($q = 50$, $nb\_iter = 100$, $lr = 0.1$) puis faire varier un paramètre à la fois. Nous nous concentrerons dans un premier temps sur l'entraînement puis la génération de la lettre A et nous allons considérer 2 critères de performances :

- Un critère visuel de qualité de l'image produite
- L'évolution de l'erreur de reconstruction lors de l'entrainement

1.1 Influence du learning rate

Dans un premier temps, nous avons tracé l'évolution de l'erreur de reconstruction lors de l'apprentissage pour différents learning rate choisi de manière arbitraire (voire figure 1). On peut remarquer que pour tous les learning rates considérés, l'erreur de reconstruction diminue au cours de l'apprentissage, ce qui semble indiqué que l'algorithme apprend à mieux générer les images. Cependant, la valeur de l'erreur de reconstruction à la fin de l'entraînement dépend grandement du learning rate considéré.

(a) Images générées pour différents learning rates (de gauche à droite : 0.001, 0.01, 0.1, 0.5)



(b) Évolution de l'erreur de reconstruction lors de l'apprentissage en fonction du learning rate

Figure 1: Influence du Learning rate sur les performance du RBM

Nous avons donc étudié plus en détail l'influence du learning rate sur l'erreur de reconstruction à la fin de l'entraînement (voir figure 2). On observe des performances optimales pour une learning rate compris entre 0.2 et 0.3.

1.2 Influence du nombre d'unités cachées

Nous allons maintenant fixer le learning rate et faire varier le nombre d'unités cachées (voir figure 3). On peut observer que l'augmentation du nombre d'unités cachées s'accompagne d'une

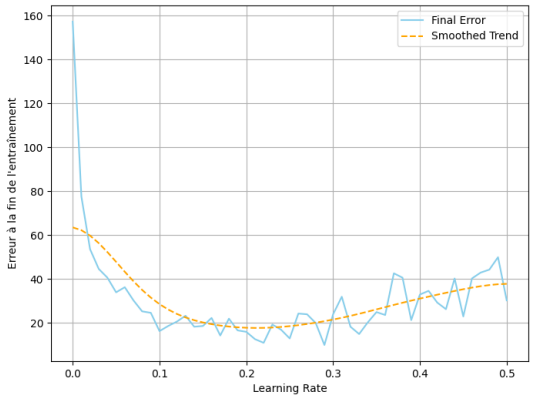


Figure 2: Evolution de l'erreur en fin d'apprentissage en fonction du learning rate

diminution de l'erreur de reconstruction en fin d'entraînement et donc d'une augmentation des performances de l'algorithme. Cependant, cette augmentation à un coup. Augmenter le nombre d'unités cachées augmente également selon une relation pseudo-linéaire le temps d'entraînement.

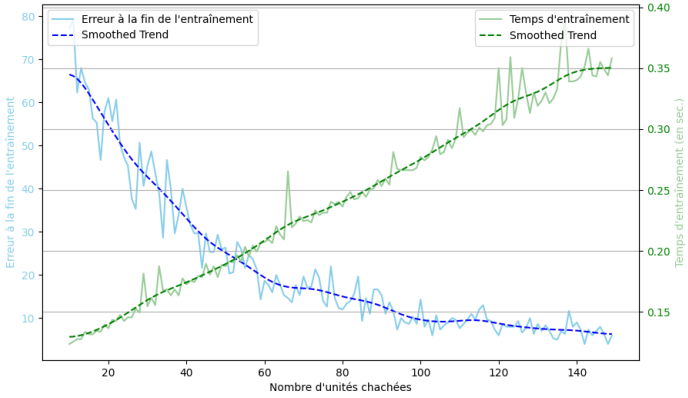


Figure 3: Evolution de l'erreur en fin d'apprentissage et du temps d'entraînement en fonction du nombre d'unités chachées

1.3 Influence du nombre d'itérations lors de l'apprentissage

Si on s'intéresse à l'influence du nombre d'itérations (voir 4) on observe des résultats similaires à la partie précédente. Augmenter le nombre d'itérations permet d'obtenir des meilleurs résultats au prix d'une augmentation du temps d'entraînement.

2 Nombre de caractères à apprendre

Dans la section précédente, nous avons considéré uniquement des données avec le même label de départ (A dans notre cas). Si nous demandons au RBM d'apprendre à partir de plusieurs types de données (A et B par exemple) on obtient le résultat visible sur la figure 5. Quand on donne

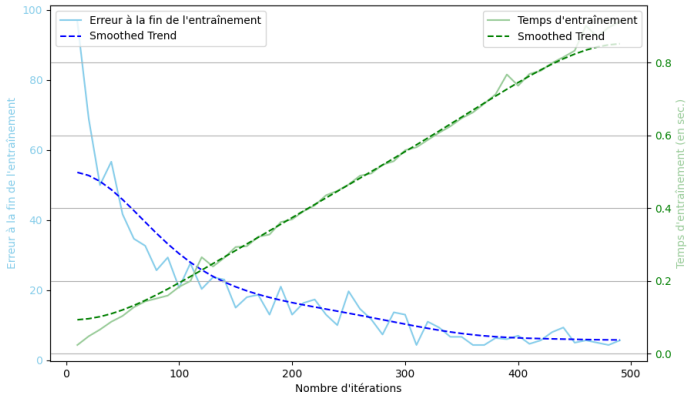


Figure 4: Evolution de l'erreur en fin d'apprentissage et du temps d'entraînement en fonction du nombre d'itérations

différents types de données en apprentissage au RBM il essaye de repérer les caractéristiques communes aux deux types de données pour générer une image qui synthétise ces caractéristiques.

Figure 5: Génération de l'algorithme après entraînement sur des images de A (Gauche), B (Milieu) puis A et B (Droite).

Si on augmente le type de données différentes lors de l'apprentissage, on observe des pixels blancs sur les zones les plus utilisés par les diverses lettres et noir sur les zones peu utilisés. Il pourrait être intéressant de comparer le taux d'utilisation des pixels par les lettres fournis en entrée et le taux constaté sur batch d'images générées.

3 Comparaison avec des bases de données alternatives

Nous allons maintenant appliquer la même méthode à d'autres bases de données pour tester sa polyvalence.

3.1 MNIST

Pour pouvoir utiliser le même process que précédemment, nous devons dans un premier temps passer d'un encodage 8-bit à un encodage binaire. Les résultats sur le dataset MNIST en utilisant les mêmes hyperparamètres que pour Binary Alphadigits dépendent fortement du chiffre considéré (voir figure 6).Pour le 1 les résultats sont peu convaincants. On peut remarquer que

les barres verticales et horizontales du 1 semblent être bien représentées. Pour le 3 en revanche les résultats sont beaucoup plus proches du training set

Figure 6: Génération de RBM après entraînement sur 5000 images de 1 (en haut) et de 3 (en bas) du dataset MNIST (a droite) et échantillon du training set (à gauche)

Les mauvais résultats pour le chiffre 1 peuvent être dû à plusieurs facteurs :

- Une résolution de l'image trop faible.
- Une trop grande diversité dans les écritures

L'hypothèse la plus probable reste une inconsistance dans les façons d'écrire certains chiffres qui reviendrait à entrainer le modèle sur différents types de données comme dans la partie précèdente.

4 Comparaison avec d'autres modèles

4.1 GAN

Pour finir ce TP nous avons utilisé un GAN implémenté dans la librairie deep learning de naokishibuya disponible sur le github suivant : https://github.com/naokishibuya/deep-learning/tree/master.

Les résultats de GAN sont meilleurs que RBM pour les chiffres avec différents types de données comme le 1. Cette meilleure performance peut également être due à la meilleure résolution des images.

Figure 7: Résultats de générations d'un GAN entraîné sur MNIST