Aergo : Un modèle multifonction pour des réseaux profonds plus accessibles

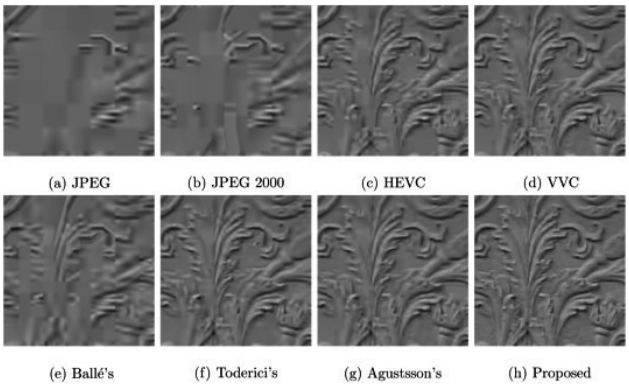
Sommaire

I. Contexte, explication et définition du projet

Nous désirons créer Aergo, une intelligence artificielle à multiples fonctions, afin de faciliter l'utilisation d'un auto-encodeur ou d'un réseau neuronal profond. Aergo utilisera la descente du gradient par rétropropagation en tant que méthode d'apprentissage principale. Nous désirons rendre le projet le plus simple et accessible possible, ainsi que réduire les temps d'entraînement par l'optimisation du code.

L'objectif d'un auto-encodeur est d'apprendre à construire de toutes pièces une façon de coder et décoder des informations (comme des fichiers), et ce de manière à perdre un minimum de qualité à la sortie. Ainsi, le programme devra produire un protocole de compression optimisé.

c.f. une comparaison de compression d'image entre JPEG *(a. et b.)* et un auto-encodeur profond proposé par Sen Xiang en 2020 *(h.)* :



Le projet ne contiendra aucune bibliothèque étrangère, et utilisera à la place une librairie de calcul matricielle développée pour le projet.

Nous devons alors imaginer, confectionner et programmer une structure de réseau neuronal à l'aide du langage C#. Cette structure aura de multiples fonctions principales :

- *L'apprentissage automatique et non supervisé de caractéristiques discriminantes,*

- *L'apprentissage de la représentation d'un ensemble de données dans le but de réduire la dimension de cet ensemble,*

- *La convergence de manière itérative vers une configuration optimisée des poids synaptiques.*

Nous programmerons la structure avec un objectif d'ergonomie et de compatibilité. La structure devra également faire preuve de flexibilité et pouvoir être utilisée dans un auto-encodeur de taille et de profondeur variable.

L'entraînement par GPU (Carte graphique) n'est pas envisageable d'un point de vue architectural, ainsi l'entraînement devra être optimisé et exécuté sur le CPU (Processeur). Nous disposons d'un serveur de calcul indépendant et dédié à cette fin.

Aergo en tant qu'auto-encodeur sera constitué d'un réseau de 1 à n couches de neurones de quantité variable non récurrents propagés vers l'avant, et séparé en deux parties : l'encodeur et le décodeur. Chaque partie sera encapsulée et pourra être utilisée indépendamment après entraînement.

Les échantillons présentés à d'Aergo seront constitués de :

- *Images carrées*

- *Fichiers texte*

- *Données brutes*

II. Objectifs du projet

L'objectif final du projet est d'obtenir une structure d'apprentissage et d'encodage simple, flexible et ergonomique.
Le cadre devra être en capacité de générer une table d'encodage complexe sans intervention et, plus généralement, d'apprendre par le procédé de rétropropagation du gradient.

Les objectifs intermédiaires consistent en l'élaboration de l'encapsulation et la modélisation du calcul de descente du gradient, ainsi que l'optimisation pour atteindre des temps moindres d'entraînement.

Un objectif supplémentaire est également le développement et la programmation d'une bibliothèque de calcul matriciel afin de ne pas avoir de dépendance extérieure.

Les résultats envisagés pour l'auto-encodeur est une chaîne de données définissant une table d'encodage pouvant réduire la taille et dimension de l'échantillon donné, ainsi que la version encodée de cet échantillon et la version décodée en tant que sortie du réseau.

III. Périmètre du projet

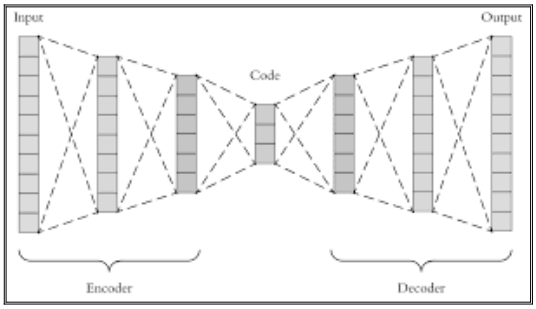
Afin de réaliser ce projet, nous disposons du langage C#.

IV. Description fonctionnelle du projet

1. Architecture

La structure aura l'architecture d'un réseau de neurones connectés. Le nombre de neurones d'entrées, tout comme le nombre de neurones de sorties, sera variable. Chaque neurone (Également appelé perceptron) sera composé de ses entrées, de ses poids, de sa fonction d'activation (Ici, la fonction sigmoïde), et de sa sortie (Variant entre 0 et 1). Plusieurs variantes d'architecture existent, dont l'auto-encodeur.

c.f. L'auto-encodeur possède autant de sorties que d'entrées, et n'a qu'un nombre réduit de neurones au milieu, comme voici :



Les neurones au centre de l'architecture représentent le « code », autrement dit la version codée du fichier d'entrée, par la première partie alors appelée encodeur. La seconde partie, le décodeur, a pour fonction de reproduire le fichier à partir du code. Cette disposition force le réseau de neurones à n'utiliser que certains neurones pour garder le code. Nous pouvons ajouter autant de couches que nous le désirons à l'encodeur comme au décodeur.

2. Neurones

Les neurones composant le réseau suivent la fonction d'activation sigmoïde pour produire leurs sorties.
Soit I l'ensemble des entrées du perceptron, W l'ensemble de ses poids, et O sa sortie :

$$O = S \,.\, \sum^{0 \rightarrow n}(W_n \,.\, I_n)$$

S étant la fonction d'activation.
Le processus s'étend à l'ensemble des neurones de la structure, et la sortie d'une couche devient alors l'une des entrées de la couche suivante.

3. Apprentissage

Pour apprendre, Aergo utilisera la méthode de rétropropagation du gradient. Nous calculons les erreurs de chaque neurone de la fin jusqu'au début du réseau, puis nous ajustons leurs poids grâce à la formule suivante :
Soit W l'ensemble des poids du neurone, I l'ensemble de ses entrées, E l'erreur calculée, et λ le taux d'apprentissage (compris entre 0 et 1).

$$W_n = W_n - (\lambda \,.\, E) \,.\, I_n$$

4. Utilisation

Le réseau pourra être utilisé grâce à un script d'entrainement à lancer avec comme paramètre un dossier contenant les images à compresser. Après quelques temps, l'utilisateur pourra arrêter le processus et lancer le script de décompression qui s'occupera d'un fichier donné en paramètre.