UNIVERSITÉ PARIS SUD

Vie Artificielle

NOM : AMRANE, NAIT-LARBI

Prénom : Lydia, Takfarinas

Double-licence : Math-Informatique

Groupe : 01

Chargé cours et TD : Mr. DECELLE Aurélien

Sommaire

Compte rendu du projet

Perceptron et apprentissage supervisé

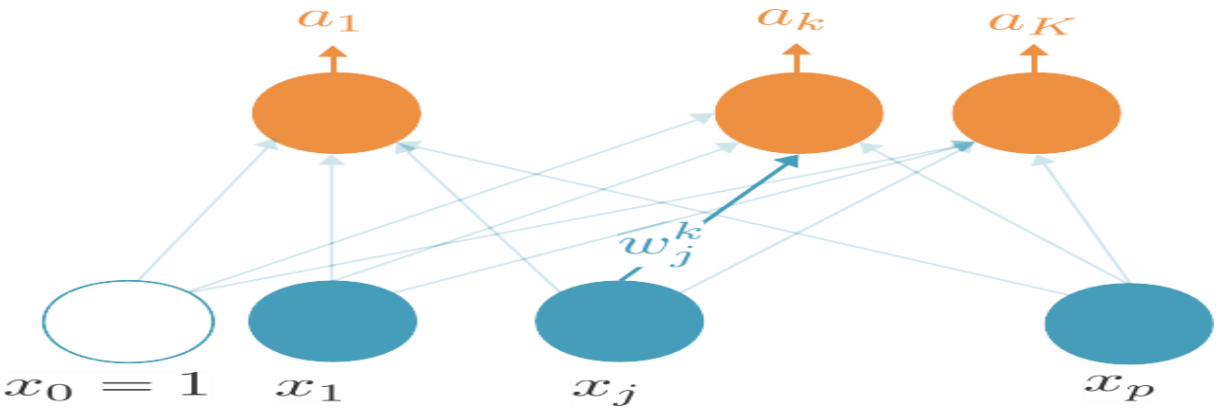
I. Introduction :

1. Le perceptron peut être vu comme le type de réseau de neurones le plus simple. C'est un classifier lineaire.il est formée d'une première **couche** d'**unités** qui permettent de « lire » les données : chaque unité correspond à une des variables d'entrée. On peut rajouter une **unité de biais** qui est toujours **activée** (elle transmet 1 quelles que soient les données). Ces unités sont reliées à une seule et unique **unité de sortie**, qui reçoit la somme des unités qui lui sont reliées, pondérée par des **poids de connexion**.

Il existe de nombreux types de réseaux neuronaux, on peut les diviser en deux grandes catégories selon la nature de leur algorithme d'apprentissage. En effet, les premiers sont dits *supervisés* car, lors de l'apprentissage, ils doivent disposer d'un professeur capable de leur indiquer ce qui devrait être produit en sortie pour chaqu'une des informations fournies en entrées (notre objective). Les seconds sont dits *non supervisés* car ils arrivent à s'auto-organiser

- Le perceptron (binaire)
- Le perceptron (multi classe)
- Le perceptron (réseau à couches)

Notre projet consiste à implémenter l'algorithme du perceptron multi-classe, c'est quoi le perceptron Multi Classe ?



2. **Classification multi-classe**, a pratiquement le même principe que le perceptron binaire mais leurs architecture diffèrent, Au lieu d'utiliser une seule unité de sortie, il va en utiliser autant que de classes. Chacune de ces unités sera connectée à toutes les unités d'entrée. On aura donc ainsi K (p+1) poids de connexion, où K est le nombre de classes.

II Structure du projet :

Notre projet consiste à implémenter le perceptron multi Classe et de l'appliquer sur deux bases de données différentes à savoir :

- **Caltech101** : c'est une base de données qui représente des images (28*28pixels) de différentes silhouettes (voitures, éléphant …etc.) elle contient 4100 images pour l'ensemble d'apprentissage et 2307 pour l''ensemble de test.
- **ImageChiffre** : c'est une base de données qui représente des images de format 28*28 de chiffre entre (0….9).

Le projet se décompose en plusieurs Classes, une Classe de l'algorithme et Code, deux Classe d'application et sept classe pour l'analyse et la discussion

i. La classe de l'Algorithme et Code :

Dans cette Classe on a implémenté les différentes fonctions qui permettent de mettre à jour les poids de chaque perceptron.

Cette classe se décompose en des fonctions secondaires à savoir :

- **Produit_Scal**
- **indiceMax**
- **SommeProba**
- **Tableau_proba**

La fonction principale de Ce code est la fonction Perceptron_Multi qui utilise toutes les fonctions secondaires, et s'occupe de mettre à jours les poids de chaque perceptron.

ii. Les classes d'application

1-Application_Image

Dans cette Classe on applique notre code sur la base de données ImageChiffre, on modifie le tableau d'étiquettes des données pour l'adapter au perceptron Multi_classes et on calcule le nombre d'erreur.

2-Application_Caltech101 : on a fait la même procédure que la base de données ImageChiffre

iii. Les classes d'analyse et de discussion

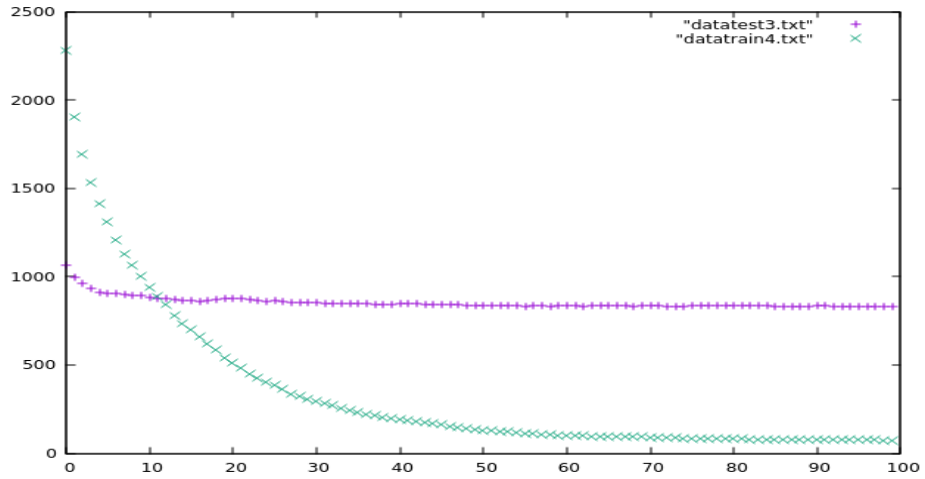
Cette partie est constituée de plusieurs classes :

1. **Allure_PoidsCaltech** : elle sert à montrer l'allure des poids de chaque neurone correspondant à la base de données caltech

2. **Allure_PoidsChiffre :** elle sert à montrer l'allure des poids de chaque neurone correspondant à la base de données ImageChiffre
3. **ImageMoyenne :** elle sert à montrer l'allure de l'image moyenne obtenue par classe
4. **Graphe01-graphe02-graphe3 :** c'est des Classes qui servent de générer les fichiers .TXT afin de dessiner les différents graphes.

III. Analyse et discussion :

1) Comment varie le taux d'erreur sur les ensembles d'entrainement et de test?

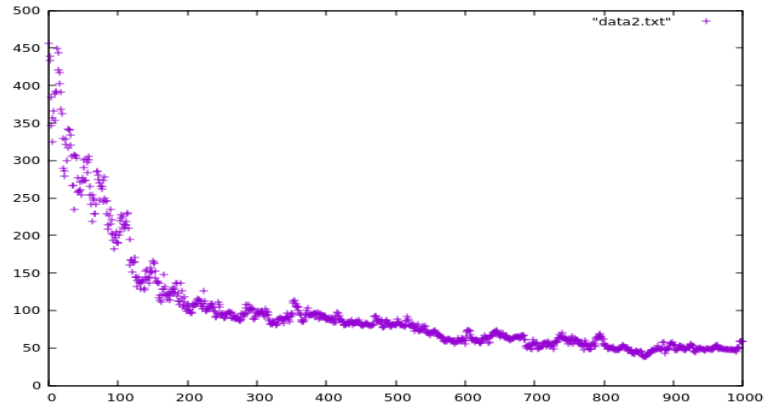


-Ensemble d'apprentissage (datatrain4.txt) : on remarque que le taux d'erreur sur cet ensemble diminue en fonction de nombre d'époque jusqu'à atteindre zéro.

-Ensemble de test (datatest3.txt) : on remarque que le taux d'erreur sur cet ensemble diminue mais à partir d'un certain nombre d'époque ce taux d'erreur reste stable, c'est le sur apprentissage

D'après le graphe on peut déduire le seuil de sur-apprentissage, dans l'exemple de Caltech, ce seuil est environ 10 époques

2) Comment varie le taux d'erreur en fonction de l'ensemble d'entrainement ?



On remarque que le taux d'erreur décroit en fonction de la taille de l'ensemble d'entrainement mais à partir d'une certaine taille le taux d'erreur reste stable (figure

3) Montrer l'allure des poids de chaque neurone

En compilant les deux classes **Allure_PoidsCaltech , Allure_PoidsChiffre** on obtient les allures des poids de chaque neurone .Et on remarque que ces allures sont presque semblables aux silhouettes des objet représentés par chaque Classe .

4) Montrer l'allure de l'image moyenne obtenue par classe

En compilant la classe **ImageMoyenne** on remarque que l'allure de l'image moyenne obtenue par classe est semblable aux silhouettes des objets représentés par chaque Classe.

IV. Conclusion :

Pour conclure, ce projet fut d'une grande aide car il nous a donné une meilleure vision et une application sur le perceptron multi classe.

De plus, il nous a permis d'appliquer nos connaissances en informatique à un domaine pratique, son objectif est de créer des systèmes artificiels s'inspirant des systèmes vivants,