Ecole d'Ingénieurs en Cognitique, 2ème année
UE Intelligence Artificielle

Projet I.A. partie II

Question 1 : classification avec Perceptron 1 couche (1 point)

On considère les mesures ci-dessous. 2 mesures ont été réalisées pour 6 animaux de l'espèce A et les mêmes mesures ont été réalisées pour 6 animaux de l'espèce B :

Espèce	Taille (cm)	Poids (kg)
A	36,1	17
A	41,1	16,5
A	47,9	18
A	47	18,5
A	49,8	22
A	48,5	21
B	47,8	11,7
B	55,5	16,4
B	62,0	13,0
B	64,1	22
B	64,3	18,5
B	65,3	21

Soit un perceptron 1 couche comportant 3 entrées : (x,y) correspondant à 2 mesures obtenues sur un animal (taille et poids) et une constante égale à 1 et une sortie (voir cours pour le calcul de la sortie du neurone, selon le principe d'une somme pondérée suivie d'un seuillage). Programmez l'apprentissage du perceptron avec les données fournies afin que la sortie du perceptron soit 1 si et seulement si le couple de données en entrée correspond à l'espèce A et 0 si elles correspondent à l'espèce B.

Rappel de l'algorithme :
Initialisation des poids W1, W2 et W3 de manière aléatoire.
Répéter
Initialisation à 0 du nombre d'erreurs de classification
Pour chacun des couples (E1,E2) de la base de données (ici x et y)
Initialiser les entrées du neurone
Calculer la sortie
Si la sortie vaut 0 alors que 1 était attendu, Wi ← Wi + Ei pour chaque Wi
Si la sortie vaut 1 alors que 0 était attendu Wi ← Wi – Ei pour chaque Wi
Si la sortie n'est pas la bonne, augmenter de 1 le nombre d'erreurs
Augmenter de 1 le nombre d'itérations
Jusqu'à ce qu'il n'y ait plus d'erreur ou qu'on ait atteint un très grand nombre d'itérations
Afficher le nombre d'erreurs et les valeurs finales des poids
NB : alternez les exemples, prenez-en 1 de A puis 1 de B et ainsi de suite, sinon ça converge mal.

A rendre :

Dans le rapport :

- Les valeurs finales des poids.
- Un graphe montrant la position des points et la droite séparatrice définie par les poids. L'affichage de la droite peut être réalisé sur une feuille à part, sur Excel ou sur la fiche de votre programme, au choix.

Les sources de votre programme.

Question 2 : régression avec un perceptron multi-couches (3 points)

On considère le fichier de données « datasetregression.txt », qui contient 3000 triplets (x,y,z) numérotés de 1 à 3000. Le numéro du triplet précède les 3 valeurs. x et y sont des entiers correspondant aux coordonnées x,y d'une image et z un réel correspondant à une intensité de niveau de gris codée entre 0 et 1.
On souhaite utiliser un perceptron multi-couches pour apprendre automatiquement, grâce à la méthode de rétropropagation du gradient de l'erreur, l'intensité en fonction de (x,y).
Utilisez les fichiers Csharp fournis (notamment reseau.cs et neurone.cs) pour écrire votre propre programme réalisant cet apprentissage.
Discutez ensuite de la qualité du résultat et de la vitesse de convergence de l'apprentissage en fonction des paramètres suivants :

- Nombre de neurones sur la couche cachée.
- Coefficient d'apprentissage.
- Taux d'erreur résiduel (par exemple la moyenne des erreurs entre chaque sortie désirée et chaque sortie obtenue).

Pour illustrer vos propos, présentez les résultats de vos tests sous la forme de 2 images de taille 500x500 pixels :
Pour la 1ère image, après apprentissage, calculez le niveau de gris en sortie du perceptron pour tous les couples (x,y) de l'image et convertissez le en valeurs entières entre 0 et 255 afin de l'afficher (un niveau de gris est obtenu en affectant le rouge, le vert et le bleu à la même valeur avec la méthode Bitmap.SetPixel(x,y,color)).
Pour la 2ème image, remplissez d'abord avec du noir, puis reprenez tous les triplets (x,y,z) du fichier de données, calculez la sortie du perceptron après apprentissage et affichez pour chacun des couples (x,y) l'erreur en valeur absolue sous la forme d'un niveau de gris, de sorte que plus l'erreur est importante, plus le niveau de gris est élevé. Attention, il faut que ce soit bien visible, donc dès que l'erreur est supérieure à 1 ou 2, il faut que le niveau de gris soit déjà assez clair. A vous de proposer une échelle de gris appropriée.
Question subsidiaire : quelle est selon vous la fonction mathématique approximativement encodée par le perceptron ?

Question 3 : classification (6 points)

On considère les données du fichier datasetclassif.txt. Il s'agit de 3000 couples de mesure numérotées de 1 à 3000. La première valeur est le numéro du couple, suivi de 2 valeurs réelles correspondant aux 2 mesures.
L'objectif est de tester l'apprentissage supervisé et l'apprentissage non supervisé sur ces données. 1 seul programme Csharp doit être créé pour traiter les questions 3.1 et 3.2.

3.1 Apprentissage supervisé
Dans le cadre de cet apprentissage, on connait à l'avance la classification recherchée : les 1500 premières mesures (1500 couples) appartiennent à la classe A et les 1500 suivantes à la classe B. L'objectif, après apprentissage, est que le perceptron apprenne un prototype de A et de B et soit en mesure de classer n'importe quel couple (x,y) pour des valeurs de x et y quelconques dans l'intervalle [0 ; 800].
Nous allons utiliser un perceptron multi-couches pour réaliser cet apprentissage (voir fichiers Csharp fournis). Réalisez un programme qui permet d'effectuer un apprentissage supervisé et totalement automatisé de la classification de ces points. Le résultat doit être obtenu en 1 seul clic (définissez le bon nombre d'itérations a priori, de même que le coefficient d'apprentissage).
Pour visualiser le résultat de la classification après apprentissage, adoptez la méthode suivante : faites passer 1 par 1 tous les couples (x,y) correspondant aux pixels d'une image de taille 800x800 pixels en entrée du perceptron, puis lancez le calcul de la sortie et coloriez les pixels de cette image en bleu ou en jaune selon la sortie/classe obtenue. Ensuite, par-dessus, affichez les points correspondant aux données du fichier, ceux de la classe A en blanc et ceux de la classe B en noir. Si le résultat est bon, vous devez avoir tous les points blancs dans une région bleue et tous les points noirs dans une région jaune.

L'objectif est d'obtenir le meilleur résultat possible en un minimum de temps. Vous chercherez donc à optimiser votre technique d'apprentissage par toute adaptation que vous jugerez utile.
Discutez ensuite de la qualité des résultats et de la vitesse de convergence en fonction des paramètres suivants :

- Nombre de neurones sur la couche cachée.
- Coefficient d'apprentissage.
- Taux d'erreur résiduel (par exemple la moyenne des erreurs entre chaque sortie désirée et chaque sortie obtenue).
- Traitement complémentaire ajouté ou pas.

NB : pour l'apprentissage, présentez de manière alternée un exemple de A et un exemple de B, sinon vous risquez d'observer un problème de convergence.

3.2 Apprentissage non supervisé

Pour l'apprentissage non supervisé, on utilisera une carte autoorganisatrice, l'algorithme de Kohonen et une technique de regroupement permettant d'aboutir, in fine, à un nombre de classes fixé à l'avance (voir programme fourni). Pour réaliser un apprentissage, il faut :

a) Recueillir l'ensemble E des points de la base de données.
b) Créer une carte auto-organisatrice, procéder à un apprentissage non supervisé selon la technique de Kohonen, puis effectuer un regroupement pour obtenir 6 classes. Notez que, puisqu'il s'agit d'un apprentissage non supervisé, l'algorithme ne peut pas exploiter l'information d'appartenance de chaque point comme dans l'exercice 3.1.
c) On teste ensuite la classification en colorant les points selon la classification obtenue (comme dans le 1., et on affiche également le pourcentage de bien classés et de mal classés.

Comme pour l'apprentissage supervisé, l'objectif est d'obtenir le meilleur résultat possible. Vous chercherez donc à optimiser votre technique d'apprentissage et à procéder à des tests pertinents. Discutez ensuite des résultats obtenus en fonction, notamment :

- Du nombre de neurones de la carte.
- De la vitesse de convergence.
- Du nombre de classes demandées pour le regroupement.
- D'éventuels ajouts ou modifications algorithmiques.

Présentez vos résultats comme dans l'exercice 3.1. à l'aide d'une image 800x800 où chaque point (x,y) est classé grâce à un code couleur, avec superposition des données initiales en utilisant 1 seule couleur.

A rendre pour le vendredi 14 avril 2017 minuit :

- Un rapport comportant les éléments suivants :

Un tableau de synthèse résumant la participation de chaque élève à chaque question. NB : En cas de trop grande disparité, une note différente sera attribuée à chaque élève du trinôme.
Pour la partie I :

- Pur chaque question 1, 2 et 3, l'explication de vos choix, notamment la définition de l'état, et l'heuristique utilisée.
- Un ensemble de tests pertinents illustrant le bon fonctionnement de votre programme et les différences de performance obtenues avec ou sans heuristique, notamment en indiquant le nombre de nœuds des ensembles Ouvert et Fermés (comme dans le programme taquin).

Pour la partie II :
Question 1 :

- Les valeurs finales des poids.
- Un graphe montrant la position des points et la droite séparatrice définie par les poids. L'affichage de la droite peut être réalisé sur une feuille à part, sur Excel ou sur la fiche de votre programme, au choix.

Question 2 :

- Discussion sur les performances de la classification en fonction des différents paramètres.
- Images illustrant les tests effectués et commentaires associés (voir question 2).

Question 3.1 et 3.2

- Discussion sur les performances de la classification en fonction des différents paramètres.

- Images illustrant les tests effectués et commentaires associés (voir questions 3.1 et 3.2).

Vous fournirez également un dossier incluant les fichiers de vos 4 projets Csharp (un pour la partie IA classique, un pour le perceptron 1 couche, 1 pour la régression, et 1 pour la classification).