Techniques d'Intelligence Artificielle
Réseaux de neurones

Simon Verdu

Juin 2020

3 Étude "théorique" de cas simples

3.1 Influence de η

On notera l'entrée courante X et le le vecteur de poids courant du neurone gagnant W^*.
Le déplacement du vecteur référent d'un neurone j en fonction d'une entrée X est est obtenue à partir de la formule suivante :

$$w_j(t+1) = w_j(t) + \eta e^{-\frac{||j^*-j||_c^2}{2\sigma^2}}(X(t) - w_j(t)) \tag{1}$$

- Dans le cas où $\eta = 0$ quelle sera la prochaine valeur des poids du neurone gagnant ?
  - D'après (1) nous dédisons que la nouvelle position du neurone vainqueur peut être calculée à partir de la formule suivante :
  $$W^*(t+1) = W^*(t) + \eta e^{-\frac{||j^*-j^*||_c^2}{2\sigma^2}}(X(t) - W^*(t)) \tag{2}$$
  Dans notre cas $\eta = 0 \Rightarrow W^*(t+1) = W^*(t)$.
  Autrement dit, Le vecteur référent du neurone vainqueur ne se déplacera pas.
- Même question dans le cas où $\eta = 1$.
  - D'après (2), pour $\eta = 1$ nous avons :
  $$W^*(t+1) = W^*(t) + e^{-\frac{||j^*-j^*||_c^2}{2\sigma^2}}(X(t) - W^*(t))$$
  De plus $||j^* - j^*||^2 = 0$, donc $e^{-\frac{||j^*-j^*||_c^2}{2\sigma^2}} = 1$, nous obtenons donc :
  $$W^*(t+1) = W^*(t) + X(t) - W^*(t) \Leftrightarrow W^*(t+1) = X(t)$$
  Autrement dit, Le vecteur référent du neurone vainqueur prendra la valeur de l'entrée.
- Dans le cas où $\eta \in]0,1[$ (paramétrisation "normale") où se situera le nouveau poids par rapport à W^* et X en fonction de η (formule mathématique simple ou explication géométrique) ?
  - On sait que $||j^* - j^*||^2 = 0$, et d'après (2) avec $\eta \in]0,1[$ nous obtenons donc :
  $$W^*(t+1) = W^*(t) + \eta(X(t) - W^*(t)) \tag{3}$$
  Ici, le vecteur de poids du neurone gagnant se déplacera de η fois la distance entre sa position courante et celle de X.

3.2 Influence de σ

- Si σ augmente, les neurones proches du neurone gagnant (dans la carte), vont-ils plus ou moins apprendre l'entrée courante ?
  - Pour un neurone j donné, la valeur de $||j^* - j^*||_c^2$ ne change pas si σ varie. Cependant plus la valeur de σ augmente plus $-\frac{||j-j^*||_c^2}{2\sigma^2}$ se rapproche de 0. Ce qui fait croître la valeur de $e^{-\frac{||j-j^*||_c^2}{2\sigma^2}}$ et donc la distance de laquelle le neurone j se rapproche de X (cf. (1)).
- À convergence, si σ est (plus) grand, l'auto-organisation obtenue sera-t-elle donc plus "resserrée" ou plus "lâche" ?
  - D'après la réponse apportée à la question précédente, on en déduit que l'auto-organisation obtenue sera plus "resserrée".
- Quelle mesure pourriez-vous proposer pour quantifier ce phénomène et donc mesurer l'influence de σ sur le comportement de l'algorithme ?
  - On pourrait mesurer l'évolution de l'aire de l'enveloppe convexe du réseau de neurone en fonction des valeurs de σ.(Dans notre cas le nuage de points correspond à l'ensemble des positions des vecteurs de poids des neurones).

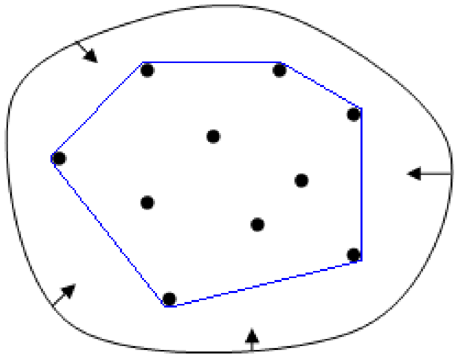

FIGURE 1 – L'enveloppe convexe est la région délimitée par la ligne bleue.

3.3 Influence de la distribution d'entrée

- Prenons le cas très simple d'une carte à 1 neurone qui reçoit deux entrées X 1 et X 2 .
  - Si X_1 et X_2 sont présentés autant de fois, vers quelle valeur convergera le vecteur du poids du neurone (en supposant un η faible pour ne pas avoir à tenir compte de l'ordre de présentation des entrées et suffisamment de présentations pour négliger l'influence de l'initialisation des poids) ?
    - Plus le nombre d'itérations sera grand plus le vecteur de poids du neurone se rapprochera avec précision du centre de la droite ayant pour extrémité X_1 et X_2 puisque l'influence des valeurs d'entrée X_1 et X_2 seront quasi identiques sur le déplacement du vecteur de poids du neurone j. En effet le déplacement du vecteur de poids du neurone j en fonction de l'entrée X_1 correspond à :

$$W^*(t+1) = W^*(t) + \eta(X_1(t) - W^*(t))$$

Et le déplacement du vecteur de poids du neurone j en fonction de l'entrée X_2 correspond à :

$$W^*(t+1) = W^*(t) + \eta(X_2(t) - W^*(t))$$

La position du vecteur de poids du neurone j sera donc à l'équilibre lorsque :

$$W^*(t) + \eta(X_1(t) - W^*(t)) = W^*(t) + \eta(X_2(t) - W^*(t))$$

  - Même question si X_1 est présenté n fois plus que X_2 .
    - Ici la position du vecteur de poids du neurone j sera à l'équilibre lorsque :

$$n(W^*(t) + \eta(X_1(t) - W^*(t))) = W^*(t) + \eta(X_2(t) - W^*(t))$$

Ansi, la position du vecteur de poids du neurone j sera sur la droite ayant pour extrémité X_1 et X_2 à une distance n fois plus proche de X_1 que de X_2.

4 Étude pratique

4.1 Analyse de l'algorithme

- Pour réaliser l'étude pratique du comportement de l'algorithme (à mettre en regard de votre étude théorique faite dans la section 3), étudiez l'influence des éléments suivants sur le fonctionnement de l'algorithme de Kohonen (qualitativement et quantitativement avec la mesure d'erreur de quantification vectorielle fourni plus la mesure d'auto-organisation que vous avez proposé à la section 3.2) :

  Dans cette partie l'ensemble des mesures ont été effectuées sur le premier jeu de donné mis à part pour la dernière question.
  - taux d'apprentissage η

    Dans cette partie σ (largeur du voisinage) = 1.2

    Dans cette partie N (Nombre de pas de temps d'apprentissage) = 30000

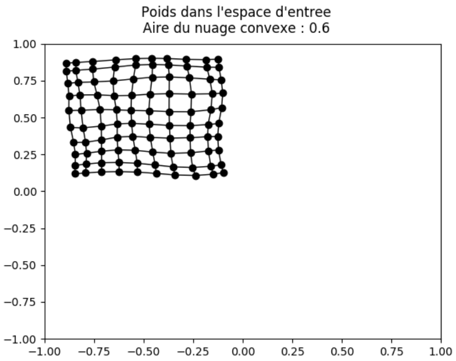

Figure 2 – $\eta = 0.05$

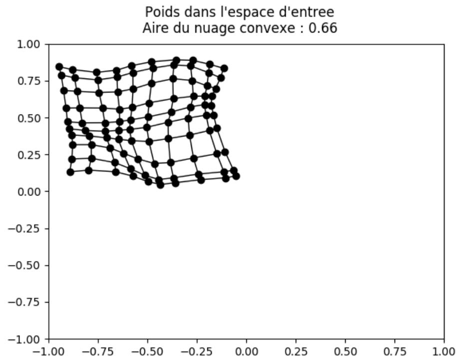

Figure 3 – $\eta = 0.4$

— On peut constater que plus eta augmente plus la disposition des vecteurs de poids des neurones perds en pertinance vis à vis des données d'entrée. En effet plus le taux d'apprentissage est élevé m ...

— largeur du voisinage σ
Dans cette partie η (Taux d'apprentissage) = 0.05
Dans cette partie N (Nombre de pas de temps d'apprentissage) = 30000

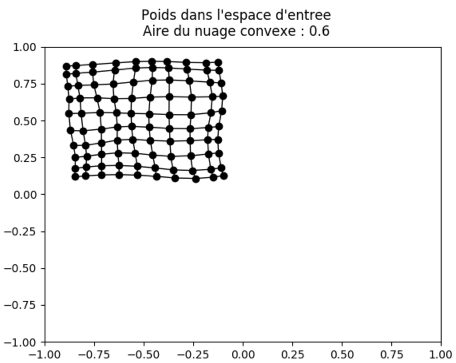

Figure 4 – $\sigma = 1.2$

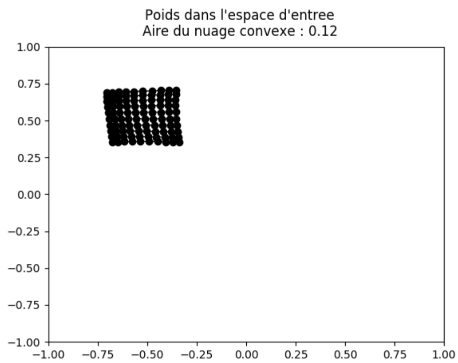

Figure 5 – $\sigma = 4.8$

On observe ici que plus sigma est grand plus les vecteurs de poids des neurones sont resserrés.La valeur de l'air du nuage convexe permet de quantifier ce phénomène.

— nombre de pas de temps d'apprentissage N
Dans cette partie η (Taux d'apprentissage) = 0.05
Dans cette partie σ (largeur du voisinage) = 1.2

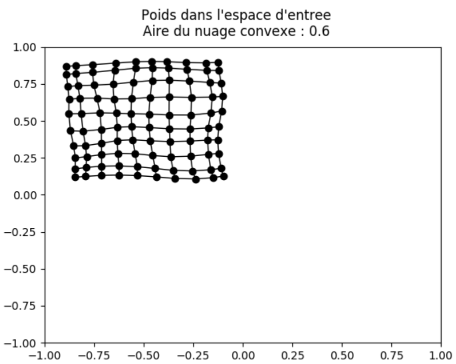

Figure 6 – N = 30000

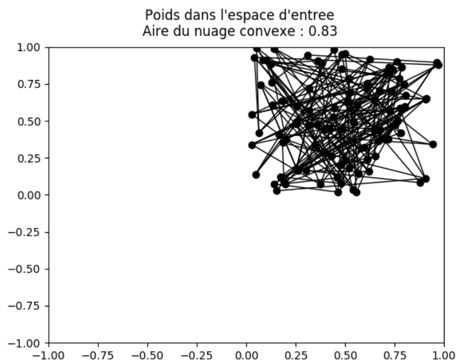

Figure 7 – N = 300

On peut constater ici que plus le nombre de pas d'apprentissage est important plus les vecteurs de poids des neurones prendront des valeurs en adéquation avec celles des données.

— taille et forme de la carte (vous pouvez tester facilement des formes 'lignes', 'carrées' et 'rectangles')
Dans cette partie η (Taux d'apprentissage) = 0.05
Dans cette partie σ (largeur du voisinage) = 1.2
Dans cette partie N (Nombre de pas de temps d'apprentissage) = 30000

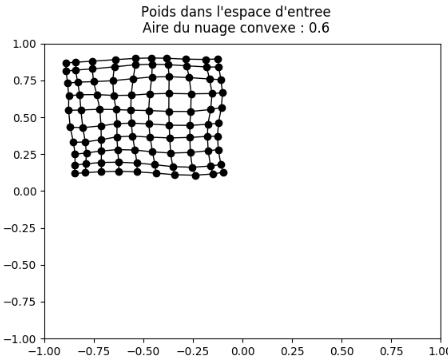

FIGURE 8 – Carte carré

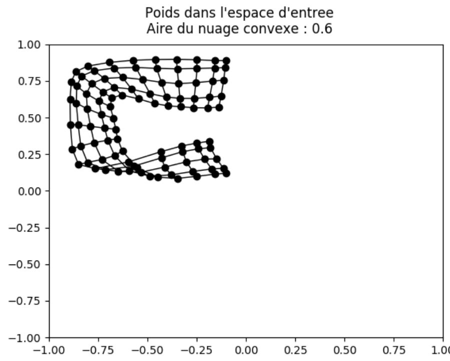

FIGURE 9 – Carte rectangulaire

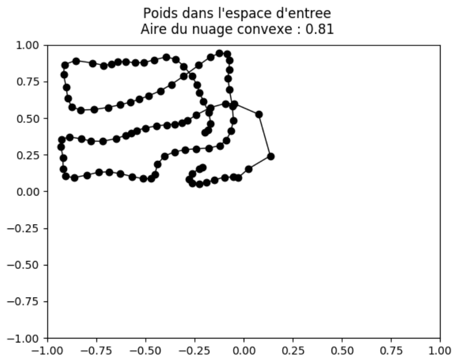

FIGURE 10 – Carte linéaire

Ici on remarque que la forme de la carte est un facteur qui influence la forme finale de la carte.

— jeu de données. En particulier créez vos propres jeux de données avec des données non uniformément distribuées pour étudier la répartition des poids des neurones.

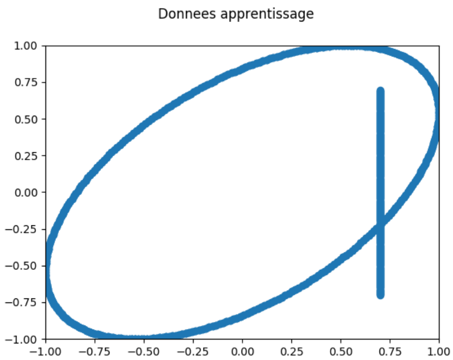

FIGURE 11 – Ensemble de données personnel 1

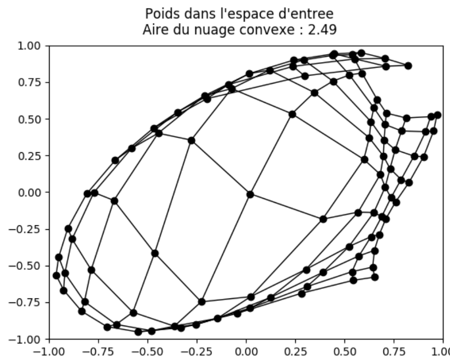

FIGURE 12 – Carte de Kohonen Associée

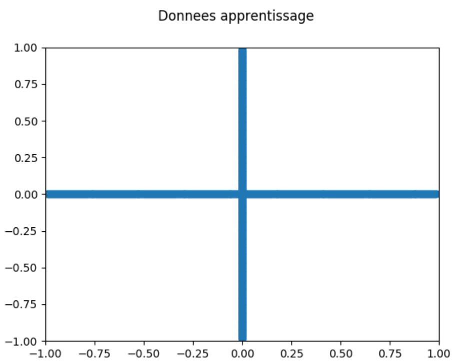

FIGURE 13 – Ensemble de données 2

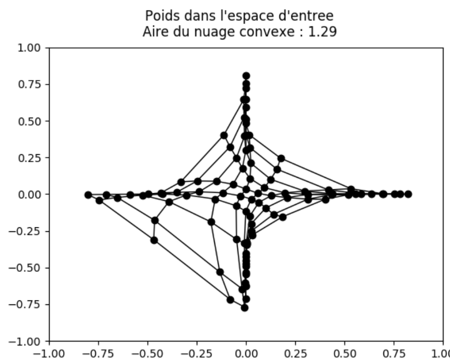

FIGURE 14 – Carte de Kohonen Associée

On observe ici que les vecteurs de poids des neurones s'adaptent bien en fonction des valeurs de l'espace d'entrée des données.De plus on remarque que la forme de carte implique que certains neurones aient des des vecteurs de poids dont la valeur n'est pas dans l'espace des données.

4.2 Bras robotique

Dans cette partie (hors partie bonus) on ne demande pas d'implémentassions mais d'expliquer précisément com ment vous procéderiez.

- Une fois la carte apprise, comment faire pour prédire la position qu'aura le bras étant donné une position motrice ? Comment prédire la position motrice étant donné une position spatiale que l'on souhaite atteindre ?
  - On propose au neurone un couples de valeurs correspondant soit a la position de l'extrémité du bras soit a l'angle de chaque articulation. Le vecteur de poids ayant la valeur la plus proche retournera alors le quadruplet correspondant.
- De quel autre modèle vu en cours se rapproche cette méthode (apprentissage sur le quadruplet pour ensuite en retrouver une sous partie étant donnée l'autre) ? Quels auraient été les avantages et les inconvénients d'utiliser cette autre méthode ?
  - J'aurai choisis le modèle modèle de Hopfield qui par l'utilisation d'une forme de mémoire associative permet justement de retrouver une valeur en fonction d'une entré bruitée. Le principal avantange est que si l'entrée correspond à une des entrée sur laquelle le reseau de neurones a deja été entrainé la valeur de sortie sera exacte. De plus on peux facilement modifier les valeurs que prend le reseeau de neurone en entrée. L'inconvénient est que lorsque les valeurs sont trop éloignées des vecteurs de poids de ses neurones il peut proposer des résultats abrents.

Références

[1] https ://fr.wikipedia.org/wiki/Enveloppe_convexe

[2] https ://docs.scipy.org/doc/scipy/reference/generated/scipy.spatial.ConvexHull.html