université PARIS-SACLAY

FACULTÉ DES SCIENCES D'ORSAY

Apprentissage par renforcement
La règle de Hebb

La règle de Hebb

C'est une technique d'apprentissage pour les réseaux de neurones permettant l'apprentissage par renforcement. Elle permet de modifier les poids des connexions entre les neurones. Cette technique s'appuie sur le postulat suivant : Si un réseau effectue une action correcte, alors on conforte cette action. Cela est décliné de la manière suivante :

- Si la sortie de deux neurones connectés est similaire, les poids de leur connexion doit augmenter.
- Si la sortie de deux neurones connectés est différente, les poids de leur connexion doit décroître.

La variation introduite dans les poids des connexions entre le neurone a et le neurone b est formalisée par l'équation :

$$\Delta w_{ab} = \alpha \cdot y_a x_b, \tag{1}$$

où w_{ab} représente le poids reliant les neurones a et b, Δw_{ab} est la variation de w_{ab}, y_a est la sortie du neurone a, x_b représente l'entrée du neurone b et α est une constante modélisant la vitesse d'apprentissage.

L'application de la règle de Hebb est possible en assumant deux hypothèses :

- Le robot explore son environnement et rencontre des situations pour lesquelles il obtient des entrées qui le permettent à son réseau de neurones d'estimer un ensemble de sorties.
- Le robot reçoit des informations lui indiquant si son comportement est bon ou non.

L'évaluation de la qualité du comportement du robot sera effectuée par un observateur humain. Il indiquera au robot manuellement un retour concernant son comportement. Il est important de différentier le fait que l'évaluation ne porte pas sur l'état du robot (ex. proche/loin d'un obstacle) mais sur le comportement du robot (ex. il s'approche ou évite un obstacle). Ainsi, les connexions du réseau peuvent générer un comportement sur la base de l'état mesuré par les capteurs.

Apprendre à éviter un obstacle

Nous souhaitons faire apprendre à un robot à éviter un obstacle. Pour cela, on pourrait permettre au robot de se déplacer librement dans un environnement et on récompensera le robot quand il évite correctement un obstacle par l'appuie d'une touche. Alternativement, on pourra faire l'appui d'une autre touche s'il collisionne avec l'obstacle. Cette méthode peut demander un temps considérable pour converger.

De manière alternative, nous présenterons au robot multiples situations et on indiquera le comportement attendu, par exemple :

- Détecter un obstacle à gauche et tourner à droite est un bon comportement ;
- Détecter un obstacle à droite et tourner à gauche est un bon comportement ;
- Détecter un obstacle devant et avancer est un mauvais comportement.

Cette stratégie renforcera les poids du réseau associés à un bon comportement grâce à un retour d'apprentissage binaire. En contraste, un apprentissage supervisé quantifie l'erreur entre la sortie désirée et la sortie actuelle pour appliquer une correction des poids et obtenir ainsi la sortie exacte souhaitée.

L'algorithme pour l'apprentissage d'évitement d'obstacles

Exo 1. Le modèle

Afin de valider la règle de Hebb à l'expérimentale et de l'appliquer à l'apprentissage de l'évitement d'obstacle, il est nécessaire de déployer le réseau illustré dans la Fig. 1.

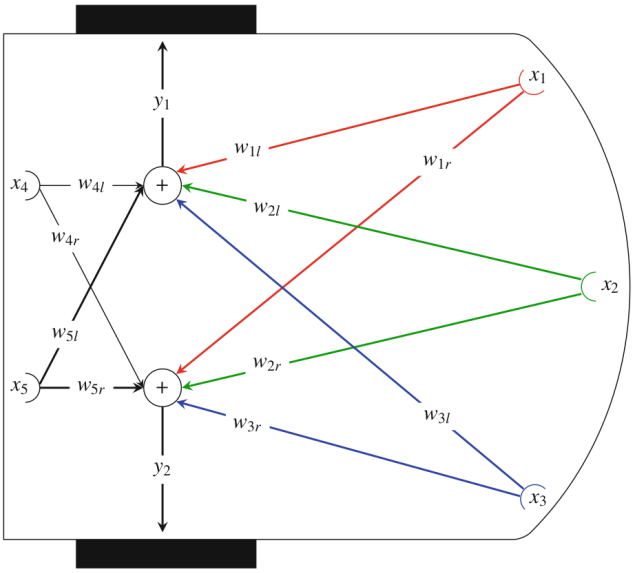

Fig. 1 : Modèle du réseau pour la validation de la règle de Hebb

Pour ce faire, la structure du logiciel définie par l'algorithme 1 préconise un échantillonnage du réseau de 100ms où les sorties y_1 et y_2 sont mises à jour en appliquant l'équation 4. Ces sorties sont appliquées aux moteurs gauche et droit.

Algorithm 1 Algorithme du réseau de la Fig. 1

```
1 : float période=0.1 // Période d'échantillonage (s)
2 : if période du timer est expirée then
3 :     x ← Valeurs mesurées par le capteurs de proximité
4 :     y ←   W · x
5 :     Vitesse roue gauche ← y[1]
6 :     Vitesse roue droite ← y[2]
7 : end if
```

Les capteurs qui interviennent en tant qu'entrées du réseau de la Fig. 1 sont cinq, identifiées ci-dessous :

$$\begin{aligned} x_1 &\leftarrow \text{Capteur frontal gauche} \\ x_2 &\leftarrow \text{Capteur central} \\ x_3 &\leftarrow \text{Capteur frontal droit} \\ x_4 &\leftarrow \text{Capteur arrière gauche} \\ x_5 &\leftarrow \text{Capteur arrière droit} \end{aligned} \tag{2}$$

Nous assumons que les valeurs issues de capteurs sont normalisées et définies dans l'intervalle continu $[0, 100]$ (0 représente l'absence d'obstacle et 100 la présence très proche d'un obstacle). Les sorties du réseau sont définies entre $[-100, 100]$, toute valeur en dehors de l'intervalle sera saturée à la valeur 100 si $y > 100$ et -100 si $y < -100$.

L'algorithme 1 sous la notation vectorielle considère alors le vecteur d'entrées :

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} \tag{3}$$

alors, la sortie du réseau est définie comme suit :

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} w_{1l} & w_{2l} & w_{3l} & w_{4l} & w_{5l} \\ w_{1r} & w_{2r} & w_{3r} & w_{4r} & w_{5r} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \mathbf{Wx} \tag{4}$$

Exo 2. La récompense

Afin d'introduire la notion d'apprentissage dans l'algorithme, il est nécessaire de modifier les poids du modèle, **W**. Nous utiliserons alors 4 boutons pour indiquer au robot le comportement souhaité face à une situation. Par exemple, si le capteur frontal gauche détecte un obstacle, alors le robot devra tourner à droite. L'implantation de ce mécanisme devra associer les actions à chaque bouton. L'algorithme 2 synthétise cela comme suit :

Algorithm 2 Algorithme de récompense

```
if le bouton avancer est appuyé then
    y1 ← 100
    y2 ← 100
end if
if le bouton reculer est appuyé then
    y1 ← −100
    y2 ← −100
end if
if le bouton tourner à gauche est appuyé then
    y1 ← −100
    y2 ← 100
end if
if le bouton tourner à droite est appuyé then
    y1 ← 100
    y2 ← −100
end if
```

Exo 3. La règle de Hebb

La dernière phase de l'implantation de l'apprentissage concerne la mise à jour des poids suivant la règle de Hebb. L'algorithme 3 décrit cette dernière phase :

Algorithm 3 Algorithme de récompense

```
x ← valeurs issues des capteurs
for j dans {1,2,3,4,5} do
    wjl ← wjl + αy1xj
    wjr ← wjr + αy2xj
end for
```

Exo 4. L'expérimentation

- Faites-apprendre à votre robot à éviter les obstacles. Rapportez une copie du script, une vidéo et les poids du modèle.
- Modifiez le programme afin qu'il puisse apprendre à avancer en absence d'obstacle. Rapportez une copie du script, une vidéo et les poids du modèle.