Sorbonne université

Faculté Sciences et ingénierie

Master 2 Informatique, parcours DAC

Rapport RLD 2ème partie

Etudiants:

Bozhang HUANG 3872174

Haoran LI 28705184

Tuteur:

Sylvain Lamprier

Benjamin Piwowarski

Patrick gallinari

2022

Table des matières

1 TME9 : GAN

1.1 Introduction

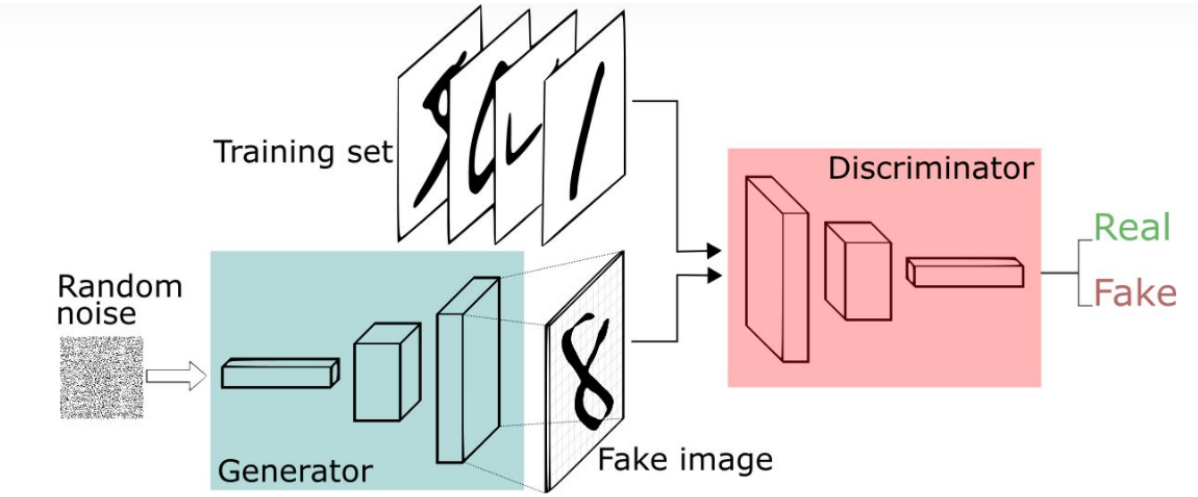

FIGURE 1 – GAN

Les réseaux antagonistes génératifs (Generative adversarial networks) sont un modèle génératif important dans le domaine de l'apprentissage en profondeur, c'est-à-dire que deux réseaux (générateur et discriminateur) sont entraînés en même temps et s'affrontent dans un algorithme minimax (minimax). Cette méthode contradictoire évite certaines difficultés dans l'application pratique de certains modèles génératifs traditionnels et approxime intelligemment certaines fonctions de perte insolubles grâce à un apprentissage contradictoire.

1.2 Résultats

Nous suivons le notebook et nous prenons comme hyperparamètres : 1^{-3} comme le learning rate pour le réseaux de générateur et 5^{-4} comme le learning rate pour lea reseaux de discriminateur.
Nous obtenons comme fake images générées :

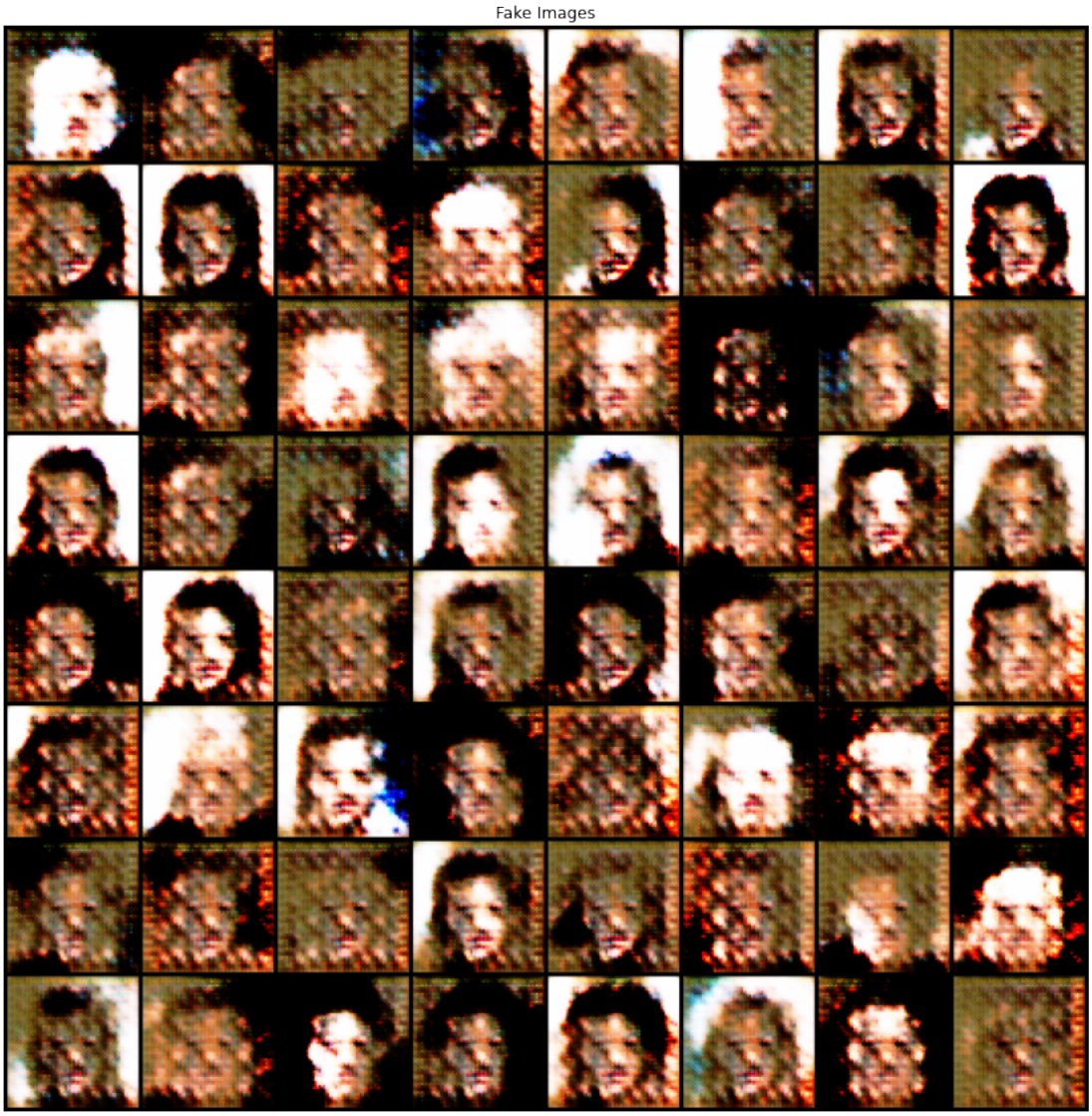

FIGURE 2 – Fake images générées

Nous voyons que les images générées resemblent bien à un portrait.

2 TME10 : VAE

2.1 Introduction

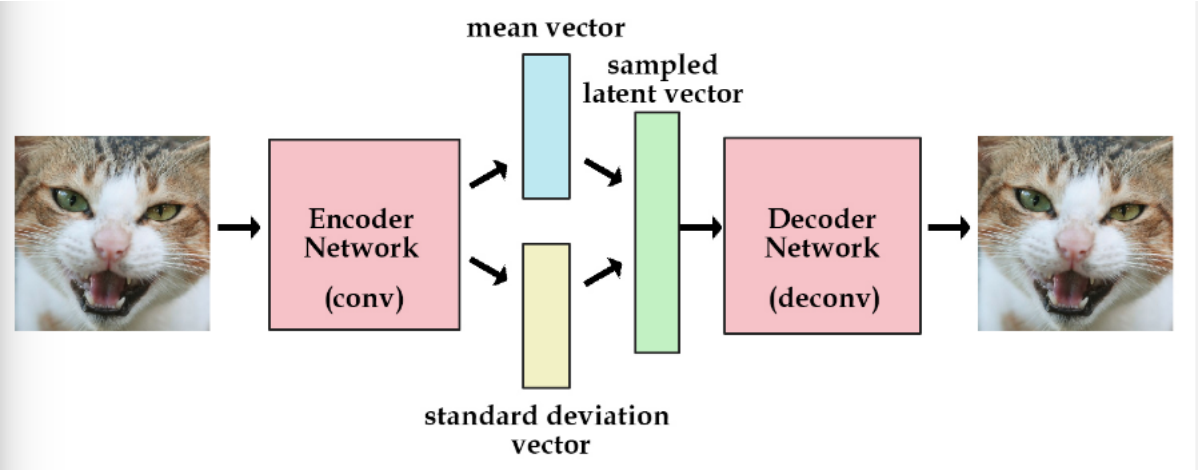

FIGURE 3 – VAE

Variational AutoEncoders (VAE) est un modèle probabiliste basé sur un réseau de neurones qui peut être formé sans supervision par descente de gradient.

2.2 Résultats

Nous suivons le notebook et nous prenons comme hyperparamètres : 512 comme batch size, 10 comme latent space size et 1^{-3} comme learning rate.
Nous générons les graphes par une moyenne et une variance aléatoire et obtenons :

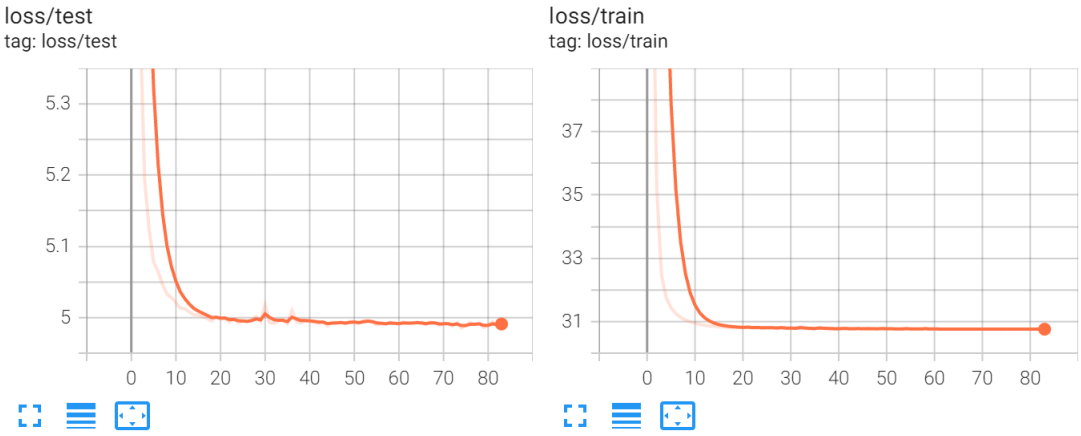

FIGURE 4 – Loss de VAE pendant entraînement

Nous pouvons voir les images générées resemblent bien à un chiffre au milieu.

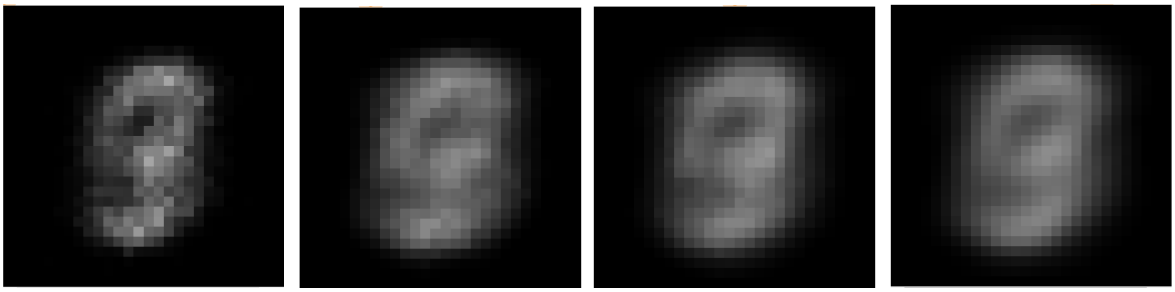

FIGURE 5 – Images générées

3 TME11 Multi-Agents RL

3.1 Introduction

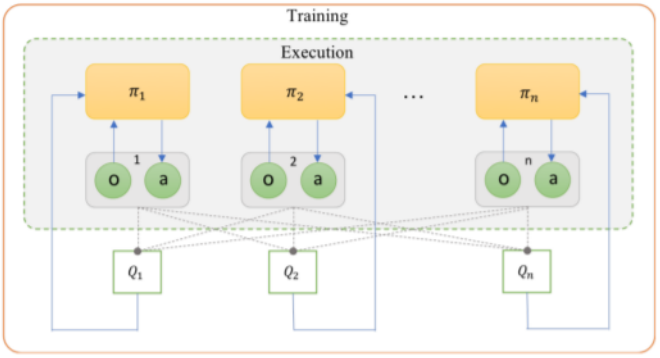

FIGURE 6 – MADDPG

L'algorithme MADDPG a les trois caractéristiques suivantes : 1. La stratégie optimale obtenue par apprentissage ne peut utiliser que des informations locales pour donner l'action optimale lors de l'application. 2. Pas besoin de connaître le modèle dynamique de l'environnement et les exigences particulières en matière de communication. 3. L'algorithme peut être utilisé non seulement dans un environnement coopératif mais également dans un environnement concurrentiel.

3.2 Résultats

nos paramètres sont ci-dessous :

```
n_episode = 100; max_steps = 100; batch_size = 16;
episodes_before_train = 10; GAMMA = 0.95; tau = 0.01;
loss = nn.MSELoss(); lr_C=0.001; lr_A=0.0001
```

simple spread

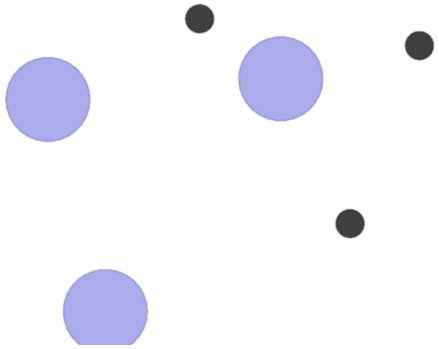

FIGURE 7 – simple spread

agents= [agent_0, agent_1, agent_2]

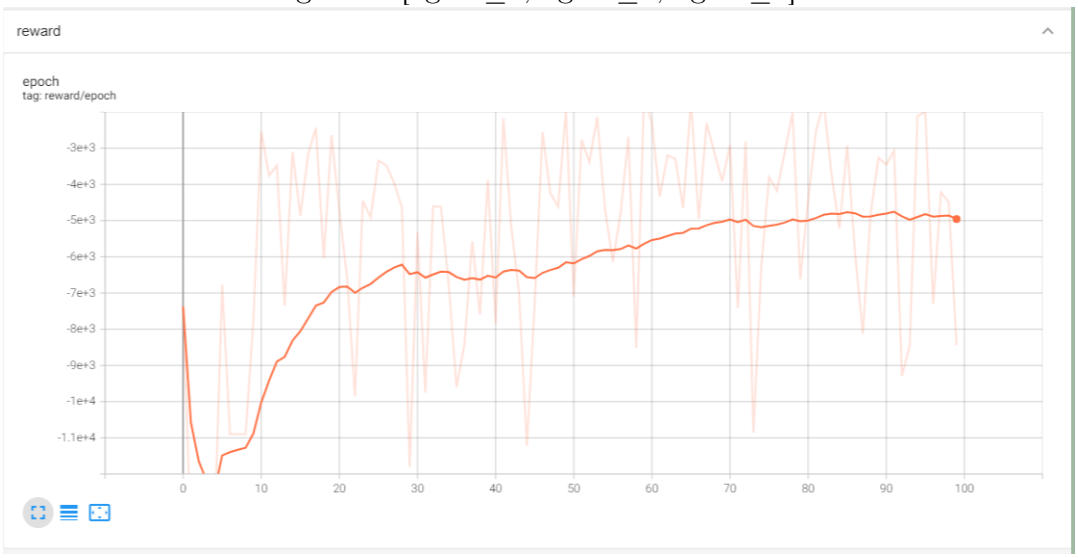

FIGURE 8 – REWARD simple spread

simple adversary

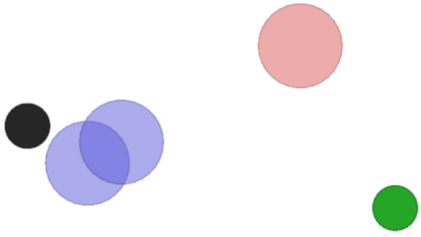

FIGURE 9 – simple adversary

agents= [adversary_0, agent_0,agent_1]

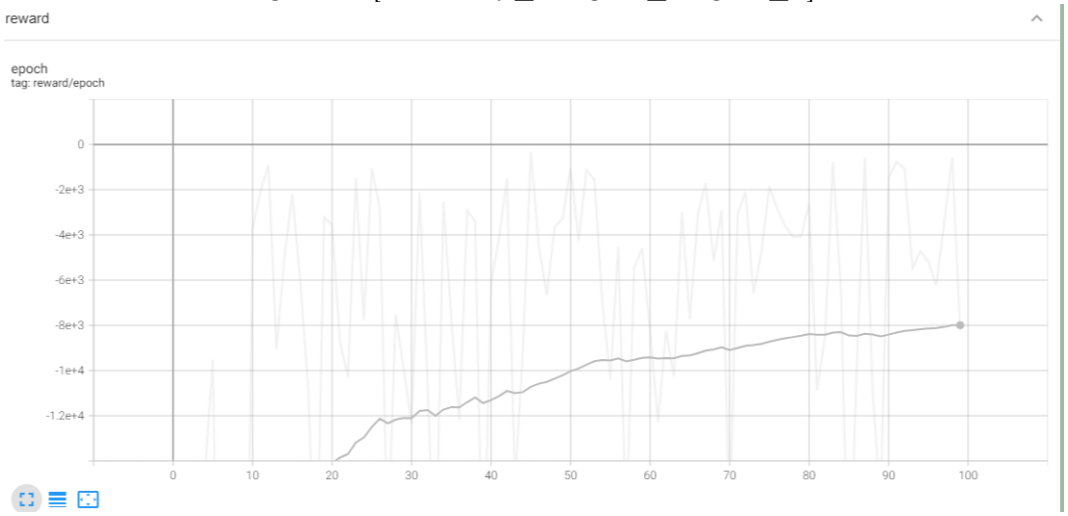

FIGURE 10 – REWARD simple adversary

simple tag

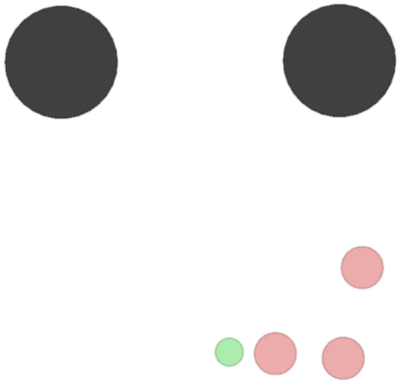

FIGURE 11 – simple tag

agents= [adversary_0, adversary_1, adversary_2, agent_0]

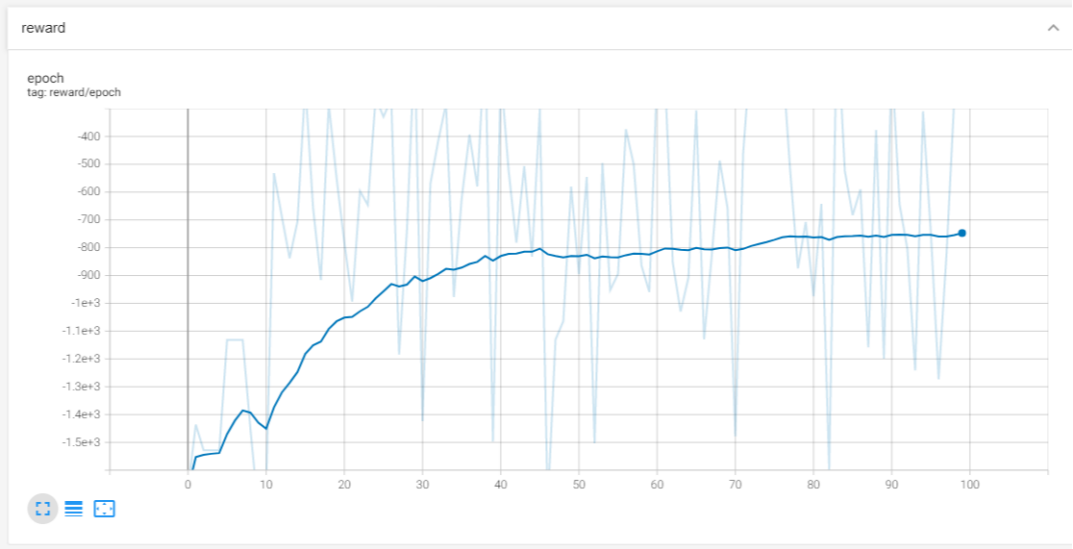

FIGURE 12 – REWARD simple tag

3.3 Conclusion

Pour construire nos réseaux, on utilise aussi Actor-Critic ici. Mais la dimension est nb_agent * dim. Les agents peuvent être contradictoires ou coopératives.
Nous ne pouvons pas voir des résultats parfaits en raison de limites comme max_step, episode. Mais nous observons que toutes les récompenses augmentent pour les 3 problèmes.

4 TME12 Imitation Learning

4.1 Introduction

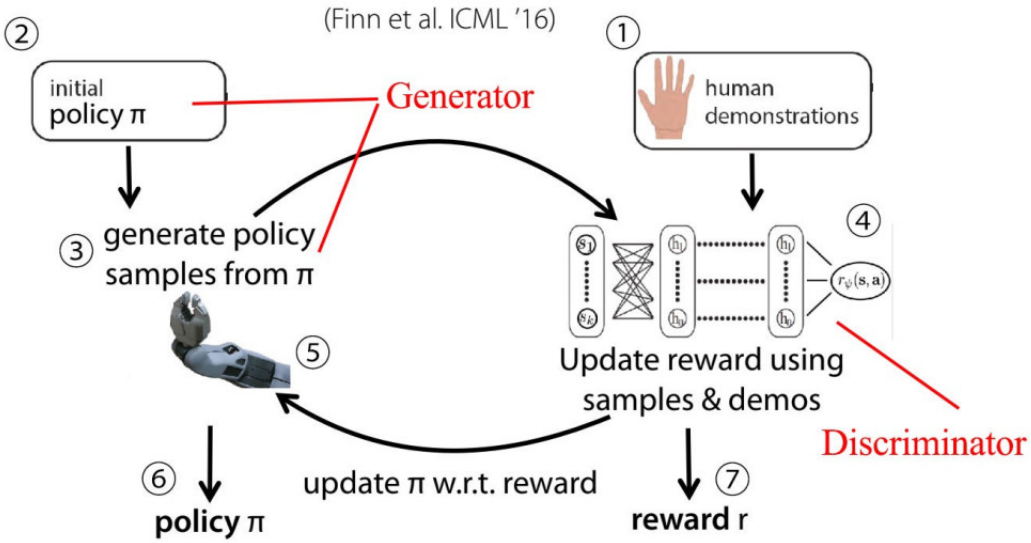

FIGURE 13 – GAIL

IRL apprend la fonction de coût à travers des données d'experts, puis apprend la politique à travers RL Prenant cela comme point de départ, une méthode pour apprendre la politique directement à travers des données d'experts est proposée - GAIL.
Il utilse Actor comme générateur et Critic comme discriminateur.

4.2 Résultats

nos paramètres sont ci-dessous :

```
n_episode = 1000; lr_a = 0.0002; lr_c = 0.0004
Capacity = 10000; num_epidose = 1000; Gamma = 0.98
Lambda = 0.99; Beta = 1; K = 3
KL_target = 0.01; eps_clip = 0.1
loss_function = 1 #0 : no clipping or penalty, 1 : KL penalty, 2 : clipping
```

On utilise ENV = 'LunarLander-v2'.

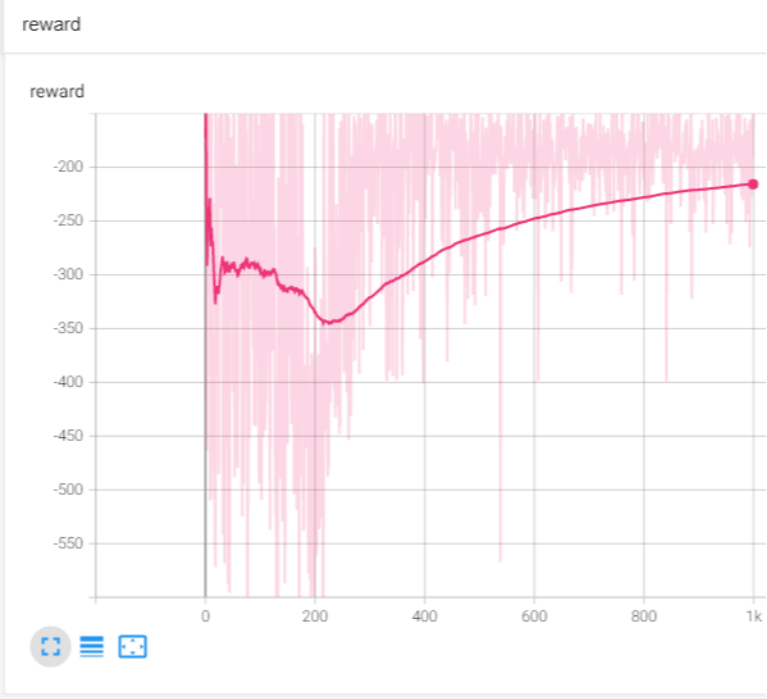

FIGURE 14 – Reward Courbe

On voit que Reward augmente. En ajustant les paramètres de récompense et en augmentant nb_épisode, je pense que nous pouvons améliorer notre performance.

5 TME13 Automatic Curriculum RL

5.1 Introduction

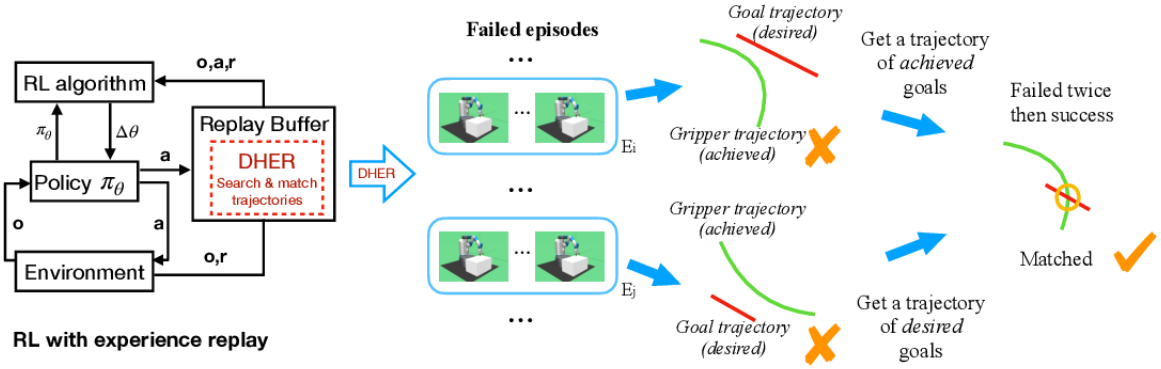

FIGURE 15 – HER

HER résout principalement le problème de la récompense rare, qui peut échantillonner efficacement des échantillons. Il peut apprendre les échantillons d'échec.

5.2 Résultats

Nous avons implémenté DQN avec buts avec HER (Highsight Experience Replay).
Pour les hyperparamètres nous prenons : 1^{-3} comme learning rate, 0.99 comme discount, 32 comme batch size et modèle but et modèle appris s'échange chaque 100 itérations. Nous définissons 100 comme le nombre d'actions maximal pour l'entraînement et 500 pour celui de test. Le test s'effectue au bout de chaque 10 itérations d'entraînement.
Nous obtenons les résultats ainsi :

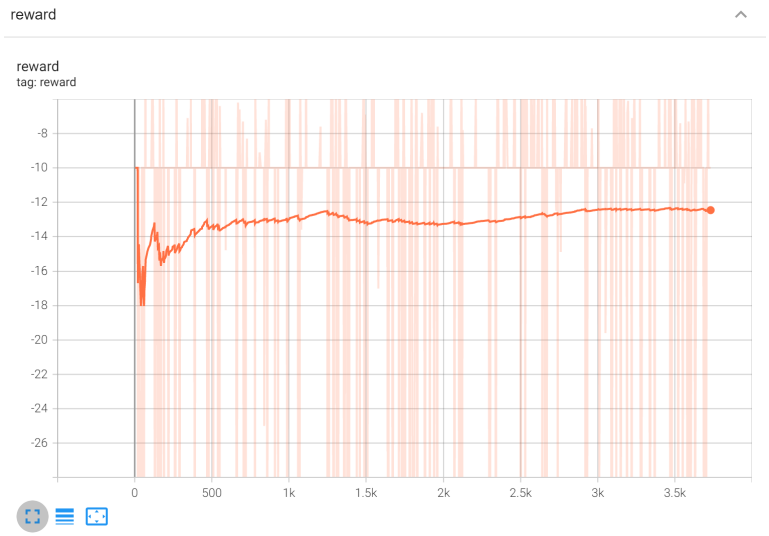

FIGURE 16 – Reward d'entraînement

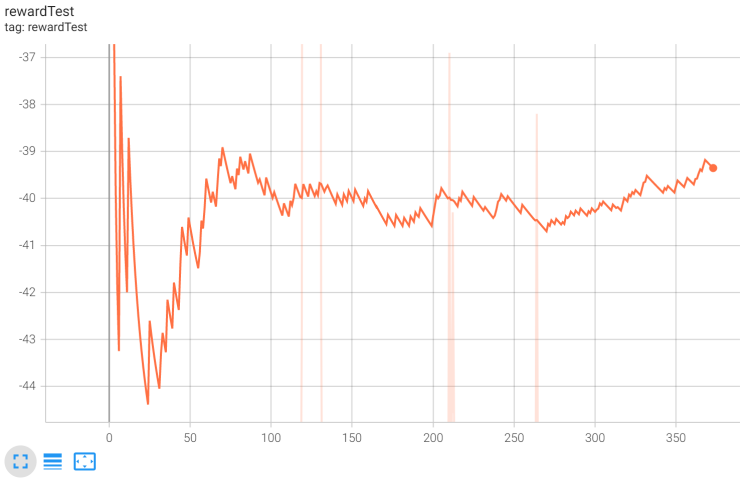

FIGURE 17 – Reward de test

Nous voyons que les rewards a tendance d'augmenter et comparé avec le DQN implémenté dans la première partie, augmentation de rewards est plus stable.

6 TME 14 Modèles de flux (Normalising Flow)

6.1 introduction

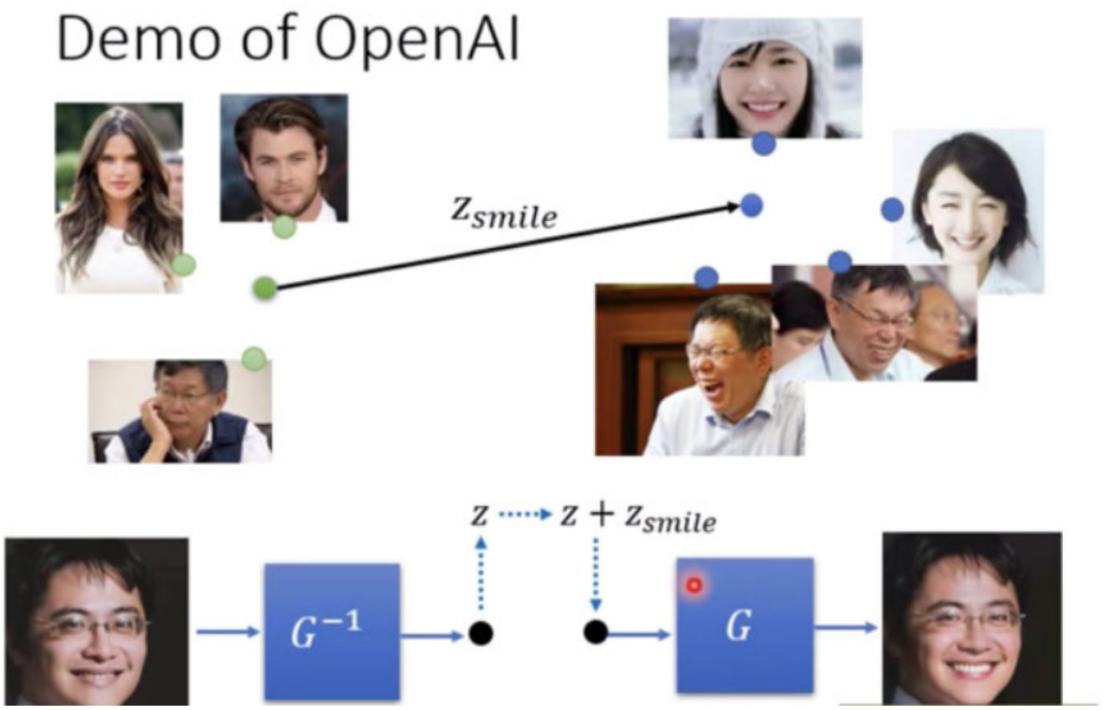

FIGURE 18 – Normalising Flow

Normalizing Flow peut convertir une distribution de probabilité simple en une distribution de probabilité extrêmement complexe, vice versa. Il peut être utilisée dans les modèles génératifs, l'apprentissage par renforcement, l'inférence variationnelle et d'autres domaines.

Les outils nécessaires pour le construire sont : Déterminant, matrice jacobienne (Jacobi), substitution de variable theorem (Change of Variable Theorem).

6.2 Check

Dans ce cas-là, on utilise **Transformation affine flow** comme base FlowModule. Car on ne trouve pas un réel basemodel, on laisse comme ça. Ça marche bien.
Comprendre la hiérarchie des fonctions :
FlowModule -> FlowModules(*FlowModule) -> FlowModel(*FlowModules)

```
check(x, mm):
    _, y, det\_y = mm.f(x)
    _, x1, det\_x = mm.invf(y)
    error = abs(x1-x).sum()
    if error < 1e-3 and abs(det_y + det_x) < 1e-3:
        print(True)
    else:
        print(False)
```

Pour voir que modèle **mm** respecte bien les transformation NF. On fait comme ci-dessus. Simplement dire, les transformations doivent êtres inversibles.

6.3 Paramètres

ActNorm

$$f(\mathbf{y}; \mathbf{s}, \mathbf{t}) = \mathbf{y} \odot \exp(\mathbf{s}) + \mathbf{t}$$

On utilise le premier batch pour initialiser s et t.

Affine Coupling Layer

$$\begin{aligned} \mathbf{s} &= \mathrm{MLP}_{\mathrm{scale}}(\mathbf{x}_{1:l}) \\ \mathbf{t} &= \mathrm{MLP}_{\mathrm{shift}}(\mathbf{x}_{1:l}) \\ \mathbf{y}_{1:l} &= \mathbf{x}_{1:l} \\ \mathbf{y}_{l+1:2l} &= \mathbf{x}_{l+1:2l} \odot \exp(\mathbf{s}) + \mathbf{t} \end{aligned}$$

Convolution 1x1

$$Y = XW$$

$$W = PLU$$
$$W = P(L' + Id)(U' + diag(s))$$

6.4 GLOW

On construit un modèle Glow, avec 10*ActNorm, Convolution 1x1, et Affine Coupling Layer comme ci-dessous :

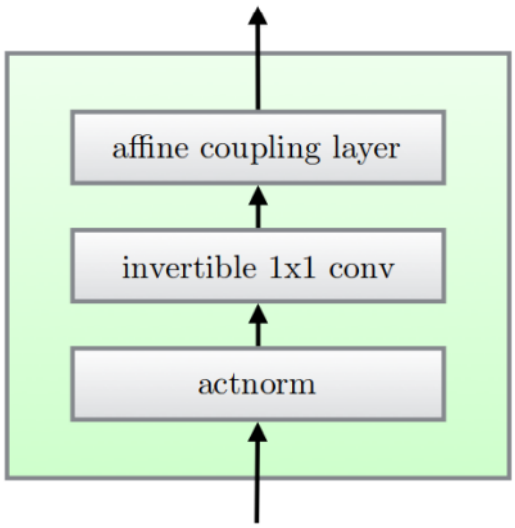

FIGURE 19 – Glow

On peut modifier le nombre de couches des différents modules comme on veut.

6.5 Résultats

Circles

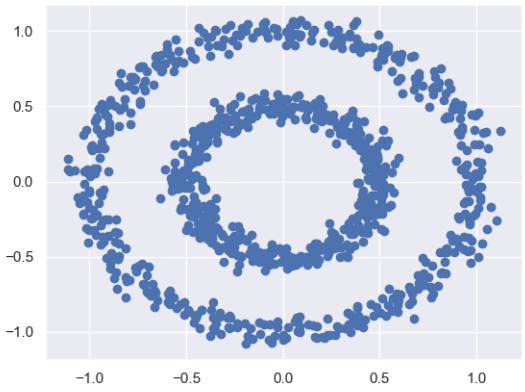

FIGURE 20 – Origine X

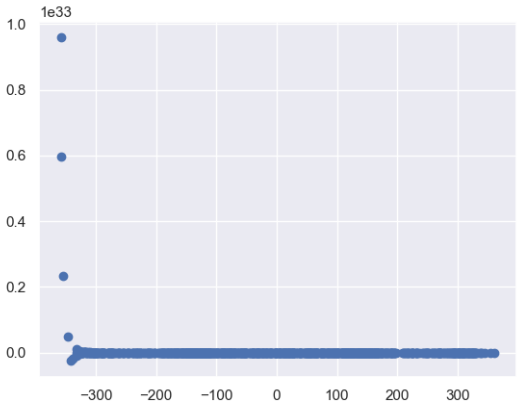

FIGURE 21 – G(X) = Z

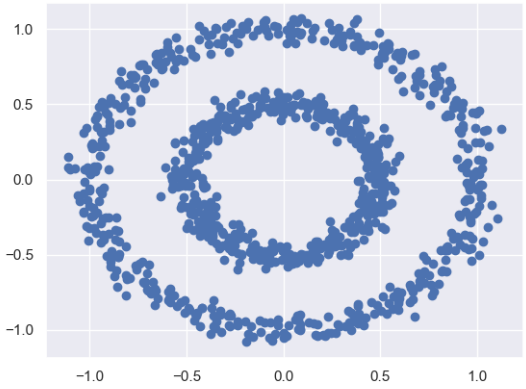

FIGURE 22 – $G^{-1}(Z) = X$

Ici, on regarde 1er graphe et 3er graphe ont l'aire pareils. Et distribution de 2er graphe est plus simple. CAD notre modèle a bien réussi.

Moons

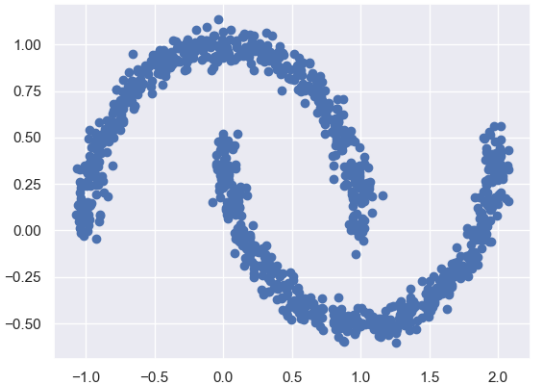

FIGURE 23 – Origine X

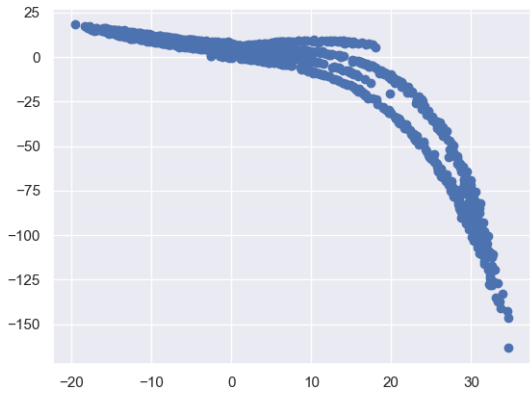

FIGURE 24 – G(X) = Z

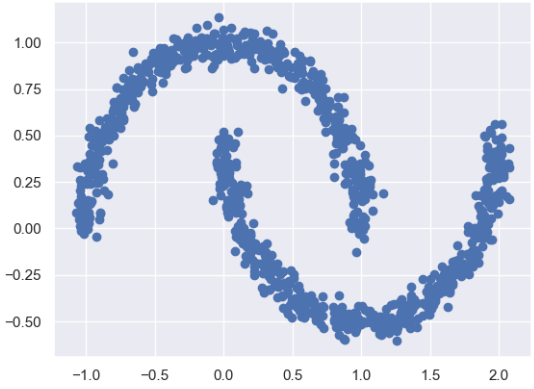

FIGURE 25 – $G^{-1}(Z) = X$

Ici, on regarde 1er graphe et 3er graphe ont l'aire pareils. Et distribution de 2er graphe est plus simple. Ici, ce n'est pas idéal comme Circles. Mais on peut modifier les paramètres de Glow pour améliorer, par exemple, le nombre de couches des différents modules.

6.6 Conclusion

NF est une technologie relativement nouvelle. L'idée de base est la réversibilité. On teste les simples transformations ici. Il reste plusieurs transformations plus complexes à améliorer.

7 Résumé final

On a étudié les modèles génératifs(GAN, VAE, NF), et également des méthodes d'apprentissage par renforcement plus avancées dans cette partie.
Enfin, grâce à des informations sur l'internet et les aides de profs, nous pouvons finalement finir les travaux.