COMPTE RENDU TD°2

Intelligence Artificielle & Optimisation

DUNAND Tom - 5A ILIA

GAN Lab - Hyperparamètres, observations et commentaires

Dimension et répartition du bruit :
Il est possible de choisir de représenter le bruit en 1D ou 2D et de choisir une distribution uniforme ou Gaussienne des points dans l'espace des paramètres, c'est sur cela que va se baser le générateur pour créer les 'fake samples'. Il conditionne donc l'ensemble de l'entraînement. En 1D, l'ensemble des fakes samples sont regroupés aux mêmes coordonnées à chaque époque, de ce fait le générateur à du mal à couvrir l'ensemble de la distribution des real samples. Aussi, l'entraînement du générateur à l'air de converger plus rapidement avec une répartition du bruit Gaussienne en 2D.

Nombre de couches cachées & nombre de neurones :
Pour le **générateur**, on remarque que le nombre de couches cachées peut rendre les prédictions des fakes samples plus précises et leur convergence plus rapide mais ça rend aussi l'entraînement plus difficile, d'autant plus couplé à un nombre élevé de neurones. On remarque aussi qu'à un certain stade, il y a trop de couches cachées et tous les points se retrouvent aux mêmes coordonnées en une époque. Aussi, si on réduit le nombre initial de neurones, dans certains cas tous les points sont groupés et généralement les faux échantillons sont moins précis.

Pour le **discriminateur**, un nombre de couches cachées et de neurones plus élevé le rend plus "puissant" pour déterminer les vrais des faux samples et rend la convergence des sorties du générateur nettement plus rapide. A l'inverse, trop peu de neurones et de couches cachées et le réseau distingue très mal les vrai des faux samples donc la convergence des sorties du générateur est beaucoup plus lente voir impossible.

Nombre de mises à jour par époques des réseaux :
Cela nous permet d'ajuster l'apprentissage de chaque réseau par rapport à l'autre, on peut décider si on veut que chaque réseau de neurones apprenne une fois en même temps à chaque époques ou par exemple si on veut que le discriminateur apprenne plus que le générateur car il est moins performant on peut dire que celui-ci apprendra deux fois tandis que le générateur n'apprendra qu'une fois. On remarque empiriquement que c'est mieux que le discriminateur apprenne plus que le générateur sinon l'entraînement va stagner rapidement. Mais pas trop car sinon les points convergeront vers une même zone de l'espace. Tout se trouve dans le bon équilibre.

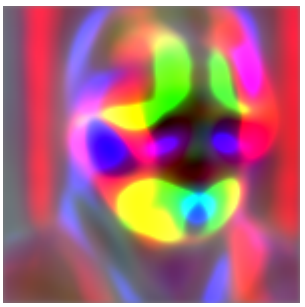
Fonction de perte :
Elle conditionne l'optimisation des poids car c'est grâce à cela qu'on calcule la différence entre les prédictions et ce qu'on attend. On remarque ici surtout que la loss Least Square permet de lisser la convergence et d'obtenir des "essais" plus cohérents du générateur à chaque époque sans des écarts de coordonnées complètement opposés à chaque mise à jour des poids.

Learning rate :
Le learning rate de chaque réseau conditionne fortement la rapidité de convergence de celui-ci lors de l'entraînement. Aussi, on remarque qu'un learning rate trop élevé ralentit voir bloque complètement l'entraînement (exemple : lr=1, le réseau ne progresse plus), le réseau n'apprend pas suffisamment. A contraire, un learning rate trop bas conduit souvent à un apprentissage rapide initialement puis à un moment le réseau stagne. Il faut surtout bien vérifier que les deux learning rates soient cohérents entre eux pour que le réseau progresse au mieux mais ça ne veut pas dire qu'ils doivent être les même, au contraire, il est mieux que le learning rate du discriminateur soit plus grand que celui du générateur.

Optimiseur :
Deux choix s'offrent à nous ici, SGD ou Adam, ceux-ci gèrent l'actualisation des poids synaptiques de chaque réseau, empiriquement, on remarque qu'utiliser un optimiseur différent pour chaque réseau n'est pas optimisé pour l'entraînement, la fonction de perte explose complètement et rend l'entraînement imprécis. Tous les points se retrouvent aux mêmes coordonnées au bout d'un certain nombre d'époques. On remarque aussi qu'en utilisant Adam plutôt que SGD dans la configuration de base, la convergence est grandement ralentie, contrairement à ce que beaucoup d'articles disent.

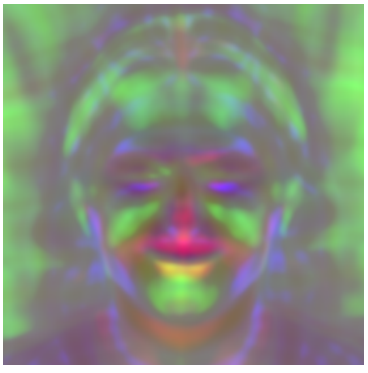
StyleGAN3 - Amorçage du générateur aléatoire et image générée

Paramètres : seed = 0 ; stylemix = 1000



Paramètres : seed = 50 : stylemix = 145

Amorçage du générateur



Étape intermédiaire

Résultat et paramètres

Paramètres : seed=50 ; stylemix=1000