Introduction

Le but de ce TP est de se familiariser et de tester des méthodes de génération de textures par réseaux de neurones. Pour des raisons de temps de calcul, les méthodes ne sont pas au top de leurs possibilités.

Mise en place informatique

Le TP utilise python (anaconda3) et le module de réseaux de neurones **tensorflow** et **pytorch**.

Pour gagner du temps et de l'espace disque, vous allez utiliser mon installation de **tensorflow, torch**[1]. Pour cela, si vous possédez déjà un répertoire ".conda" il faut changer son nom et créer un lien symbolique vers mon ".conda". Exécutez les commandes suivantes dans une nouvelle fenêtre de terminal (la première seulement si vous avez déjà un ".conda"[2]) :

```
mv .conda .conda_bak  (voir avertissement, (si vous avez déjà fait au TP1, ne pas faire).)
ln -s ~ladjal/.conda .conda
mkdir TP3_MVA_DELIRES
cd TP3_MVA_DELIRES
cp  ~ladjal/TP3_MVA_DELIRES/*.py .
ln -s ~ladjal/TP3_MVA_DELIRES/images images
```

Ce qui précède ne doit être fait qu'une seule fois. Appelez l'encadrant au moindre problème d'ordre informatique.

Ensuite, pour lancer l'environment de travail depuis un terminal

```
export PATH=/cal/softs/anaconda/anaconda3/bin:$PATH
cd TP3_MVA_DELIRES/
source activate tf_env
spyder
```

Ce TP est centré sur l'analyse de programmes déjà écrits. Vous travaillerez sous spyder essentiellement en sélectionnant une zone de programme et par clique-droit -> exécuter la cellule. Cela permet une souplesse d'utilisation et de comprendre le code à mesure que l'on progresse dans le TP (sauf l'exécution de gatys.py qui se fait en ligne de commande).

Comment rendre le TP

Idéalement vous aurez fini le TP dans les trois heures et vous me rendez le TP sous forme papier. Sinon, vous avez deux semaines pour rendre votre TP. Voici les instructions :

1. Les instructions sont valables si vous êtes sur des machines de la DSI de Télécom Paristech
2. Dans ce cas pour revenir à votre installation originale il faut faire : cd ; rm .conda ; mv .conda_bak .conda

```
- Fichier pdf uniquement
- Le sujet de mail doit contenir [MVA DELIRES TP3]
- Vous avez jusqu'au jeudi 05/03/2020
```

Pour ceux qui n'ont pas de compte Télécom les données sont là :

```
DONNEES POUR CEUX SANS COMPTE :  https://bit.ly/2GS7RNf Dans le même lien de retour
Il faudra aussi charger le fichier vgg19.npy qui se trouve au même endroit et adapter
les scripts en conséquence (juste les chemins) Les modeles Texturenet sont inclus
```

1 Génération de texture de type Gatys

Dans Gatys(2015) l'auteur propose d'utiliser pour caractéristiques de textures les covariances de features de réseau. Plus précisemment, si B est un bloc de neurones CNN qui a une taille spatiale $h \times w$ et c canaux. La caractéristique retenue pour la couche B est

$$g_{ij} = \frac{1}{wh} \sum_{l=1}^{h} \sum_{k=1}^{w} B(l,k,i)B(l,k,j), \quad \text{Pour } 1 \leq i,j \leq c$$

Ces caractéristiques sont calculées une fois pour une image exemple et pour plusieurs couches. Si B est la couche numéro l du réseau, on appelle G^l la matrice ainsi construite. Elle est de taille $c \times c$.

Lors de la synthèse on procède comme pour un entraînement de réseau sauf que la seule variable est l'image d'entrée (initialisée à un bruit blanc). Un détail important est que les auteurs préconisent d'utiliser, non pas la classique descente de gradient mais une méthode quasi-Newton nommée L-BFGS. La loss qui est minimisée est

$$\mathcal{L}(x) = \sum_i \|G^{l_i}(x) - G^{l_i}\|^2$$

où les l_i parcourent les couches choisies pour caractériser la texture et G^{l_i} est la matrice calculée à la couche l_i pour l'image exemple (et $G^{l_i}(x)$ son homologue pour l'image courante x). On lance le programme **gatys.py** depuis la ligne de commande par python3 gatys.py -h. Attention une execution prend quelques minutes, soyez parcimonieux pendant le TP.

1. Quelles couches ont été choisies pour caractériser les textures ? Générer une texture de votre choix avec le programme.
2. Essayer de changer l'ensemble des couches choisies et dire quel effet cela a sur le résultat.
3. Quel est l'algorithme d'optimisation utilisé ? (si nous avions utilisé L-BFGS sans GPU, chaque synthèse prendrait plusieurs heures)

2 Génération de texture par Texturenet

L'article TextureNets par Ulyanov et al (https ://arxiv.org/abs/1603.03417) donne une solution générative au problème de la synthèse de texture. Pour cela un réseau est entrainé à partir

d'un bruit blanc à reproduire une image dont les statistiques de Gatys (voir plus haut) sont proches de celles d'une image exemple. Dans cette technique c'est l'entrainement qui prend du temps (à peu près une heure sur une bonne carte graphique). C'est pourquoi nous avons préparé des modèles déjà appris qui sont stockés sur le disque dur. Vous avez à votre disposition deux programmes sample_Texturenet et train_Texturenet [3]. Durant le TP vous n'utilisez pas train mais vous pouvez le lire par curiosité. Le programme sample est à utiliser en mode interactif. Il utilise pytorch.

1. Donner un exemple d'overfit possible de l'approche choisie par cette technique.
2. Décrire succintement l'architecture du réseau génératif. Combien de paramètres (grosso-modo) a-t-il ?
3. Tester l'algorithme avec diverses textures et commenter les résultats.
4. Essayer de produire un film (un script est préécrit) qui consiste à tirer deux bruits et générer une séquence d'images avec des interpolées de ces bruits.
5. Observer attentivement les images du milieu du film. Comment expliquer ce que vous constater.
6. Que fait la fonction redresse ? Utilisez-là pour produire un film de meilleure qualité.
7. (Seulement s'il vous reste du temps) Que fait le script déplacement (fin du fichier sample) ? Est-ce que le résultat est satisfaisant (décrire les défauts) ? Proposer des pistes pour y remédier.

3. Crédit : Jorge Gutierrez basé sur https ://github.com/leongatys/PytorchNeuralStyleTransfer