1 Matériel de TP

Sur le Jetson, récupérez l'archive et décompressez-la :

```
$ cd ~
$ wget http://bit.do/fo4uq -O jetson-ai.tar.gz
$ tar xvzf jetson-ai.tar.gz
$ rm jetson-ai.tar.gz
```

2 Performances

Avec l'icône Nvidia en haut à droite de l'écran, mettez le *Power mode* sur `MAXN`.
Ouvrez un terminal, et lancez `tegrastats` :

```
$ tegrastats
RAM 2924/7860MB (lfb 457x4MB) SWAP 0/3930MB (cached 0MB)
CPU [13%@345,39%@806,32%@806,3%@345,0%@345,0%@345]
EMC_FREQ 0% GR3D_FREQ 0% PLL@33C MCPU@33C PMIC@100C Tboard@29C
GPU@32.5C BCPU@33C thermal@32.8C Tdiode@31.5C VDD_SYS_GPU 153/153
VDD_SYS_SOC 535/535 VDD_4V0_WIFI 0/0 VDD_IN 2451/2451
VDD_SYS_CPU 536/536 VDD_SYS_DDR 540/540
```

L'outil vous rapporte diverses mesures sur la carte (températures, utilisation CPU, tensions, ...). Le taux d'utilisation du GPU intégré est indiqué via `GR3D_FREQ`.
Tout en laissant `tegrastats` fonctionner, lancez dans un autre terminal :

```
$ glxgears
```

Quel taux d'utilisation du GPU constatez-vous ? Y a-t-il un warning affiché dans le terminal ? Quel est le taux d'images par seconde rapporté ?

Essayez maintenant avec :

```
$ __GL_SYNC_TO_VBLANK=0 glxgears
```

Quel taux d'utilisation du GPU constatez-vous ? Quel est le taux d'images par seconde rapporté ?
Maintenant, essayez avec :

```
$ __GL_SYNC_TO_VBLANK=0 glmark2
```

Quel taux d'utilisation du GPU mesurez-vous avec `glmark2` ?
Les informations récoltées par `tegrastats` peuvent être affichées de façon plus conviviale, par exemple avec `jtop` :

```
$ sudo jtop
```

3 OpenCV et la caméra intégrée du Jetson

3.1 Caméra

Dans le répertoire `opencv`, lancez le code `tegra-cam.py` avec `python3` et vérifiez que la caméra fonctionne bien :

```
$ python3 tegra-cam.py
```

Ouvrez le fichier `tegra-cam.py` et étudiez l'exemple.
Quel est le taux d'utilisation du GPU pendant le fonctionnement de cet exemple ?

3.2 Traitement OpenCV

Toujours dans le répertoire `opencv`, lancez avec `python3` le code `CannyDetection.py`.
Ouvrez le fichier et étudiez le code.
Expérimentez avec les différents modes d'affichage.
Quel est le taux d'utilisation du GPU pendant le fonctionnement de cet exemple ?
Le traitement Canny est-il effectué sur le CPU ou sur le GPU ?

4 Classification d'images via un CNN pré-entraîné

4.1 Console

Dans le répertoire `imagenet`, étudiez le fichier `imagenet-console.py` : ce script lit un fichier image jpg et tente de déterminer l'object représenté dans l'image. Lancez le script comme suit :

```
$ python3 imagenet-console.py images/polar_bear.jpg
$ python3 imagenet-console.py images/brown_bear.jpg
$ python3 imagenet-console.py images/black_bear.jpg
```

Les résultats de la reconnaissance s'affichent dans le terminal. Y a-t-il eu des erreurs ?
Testez avec des images supplémentaires, par exemple celles se trouvant dans le sous-répertoire `more-images`.
L'inférence de la classe de l'objet dans l'image est-elle effectuée par le CPU ou le GPU ?

4.2 Caméra live

Le script `imagenet-camera.py` permet de réaliser le même travail que précédemment sur le flux d'images de la caméra intégrée du Jetson. Faites des essais. Passez divers objets devant la caméra pour tester.
Quel est le nombre d'images par seconde traité ? Le traitement total est-il effectué sur le CPU ou sur le GPU ? Quel est le taux d'utilisation du GPU ?

Étudiez le code de cet exemple afin de comprendre comment utiliser la bibliothèque `jetson-inference` pour charger un modèle pré-entraîné, effectuer le traitement temps réel sur le flux vidéo, etc.

5 Reconnaissance de chiffres manuscrits par DNN et CNN

Le jeu de données MNIST, contenant des images carrées noir et blanc normalisées centrées de 28 pixels de côté de chiffres manuscrits, regroupe 60000 images d'apprentissage et 10000 images de test, correctement étiquetées. Ce jeu de données est déjà disponible dans `~/.keras/datasets` .

Allez dans le répertoire `mnist`.

5.1 MNIST sur un DNN

Ouvrez le fichier `mnist_dnn.py` : ce script entraîne un DNN (Deep Neural Network) en utilisant *Keras* et *TensorFlow*. Lisez attentivement le code pour essayer de comprendre comment on implémente un DNN.

Vérifiez que *TensorFlow* peut effectivement utiliser le GPU du Jetson en lui demandant d'énumérer les *devices* disponibles :

```
$ python3 -c "from tensorflow.python.client import device_lib; device_lib.list_local_devices()"
```

Exécutez le script `mnist_dnn.py` et observez les étapes de l'entraînement.
Changez le nombre de couches cachées et le nombre de neurones pour voir si vous pouvez améliorer le taux de classification [1].
Les calculs de l'entraînement sont-ils majoritairement effectués par le CPU ou le GPU ? Vérifiez.

5.2 MNIST sur un CNN

5.2.1 Entraînement

Ouvrez le fichier `mnist_cnn.py` : ce script entraîne et teste un CNN (Convolutional Neural Network). Lisez attentivement le code pour essayer de comprendre comment on implémente un CNN.
À la fin de l'entraînement, ce script sauvegarde le CNN au format `.h5` dans le sous-répertoire `models`.
Exécutez le script et laissez-le finir pour obtenir votre CNN entraîné au format `.h5`.
Les calculs de l'entraînement sont-ils majoritairement effectués par le CPU ou le GPU ? Vérifiez.
Changez le nombre de couches et le nombre de neurones pour essayer de comprendre l'impact sur les performances. Changez la fonction d'activation pour voir si vous obtenez de meilleures performances [2].

5.2.2 Inférence : test console

Pendant l'entraînement, ouvrez le fichier `mnist_console.py` et étudiez comment utiliser un CNN pré-entraîné pour la reconnaissance de chiffres manuscrits dans des fichiers .jpg.
Une fois le CNN entraîné, testez-le avec :

```
$ python3 mnist_console.py images/test/0/1001.jpg
$ python3 mnist_console.py images/test/2/5921.jpg
```

5.2.3 Inférence : test caméra live

Ouvrez le fichier `mnist_camera.py`.
Utilisez les précédents fichiers `opencv/tegra-cam.py` et `mnist/mnist_console.py` pour implémenter un classifieur temps réel de chiffres manuscrits avec la caméra du Jetson.

Références : `https://github.com/dusty-nv/jetson-inference#hello-ai-world`

1. L'entraînement d'un DNN sur le Jetson est lent. Ne testez pas trop de combinaisons.
2. L'entraînement d'un CNN sur le Jetson est encore plus lent. Si, si. Ne testez pas trop de combinaisons.