

# Machine learning : Docker Desktop s'ouvre aux GPU sur WSL 2

Expérimenter la virtualisation GPU sur WSL 2 avec Docker Desktop ? C'est désormais [possible](#) sans inscription au [programme d'accès anticipé](#) (Developer Preview).

L'objectif n'est pas d'exécuter des applications graphiques. Mais d'exploiter la puissance de calcul des GPU NVIDIA pour réaliser des tâches d'apprentissage automatique en local, sans avoir à faire appel à un serveur Linux.

Dans l'absolu, c'était déjà [faisable](#). Mais la démarche impliquait d'installer et de paramétrer un OS.

Pour tester, on s'assurera de disposer de la dernière *build* publiée sur le canal Dev de Windows 10. Et d'installer les pilotes graphiques adéquats (en bêta ; à télécharger [ici](#)). Ces derniers fonctionnent avec les cartes GeForce et Quadro à partir de la génération Kepler. NVIDIA recommande toutefois d'utiliser des GPU plus récents, reposant sur les architectures Turing ou Ampere.

```

> docker run --rm -it --gpus=all nvcr.io/nvidia/k8s/cuda-sample:nbody
nbody -gpu -benchmark
Run "nbody -benchmark [-numbodies=<numBodies>]" to measure performance.
    -fullscreen          (run n-body simulation in fullscreen mode)
    -fp64                (use double precision floating point values for
simulation)
    -hostmem             (stores simulation data in host memory)
    -benchmark          (run benchmark to measure performance)
    -numbodies=<N>      (number of bodies (>= 1) to run in simulation)
    -device=<d>         (where d=0,1,2... for the CUDA device to use)
    -numdevices=<i>     (where i=(number of CUDA devices > 0) to use for
simulation)
    -compare            (compares simulation results running once on the
default GPU and once on the CPU)
    -cpu                (run n-body simulation on the CPU)
    -tipsy=<file.bin>   (load a tipsy model file for simulation)
NOTE: The CUDA Samples are not meant for performance measurements. Results
may vary when GPU Boost is enabled.
> Windowed mode
> Simulation data stored in video memory
> Single precision floating point simulation
> 1 Devices used for simulation
MapSMtoCores for SM 7.5 is undefined. Default to use 64 Cores/SM
GPU Device 0: "GeForce RTX 2060 with Max-Q Design" with compute capability
7.5
> Compute 7.5 CUDA device: [GeForce RTX 2060 with Max-Q Design]
30720 bodies, total time for 10 iterations: 69.280 ms
= 136.219 billion interactions per second
= 2724.379 single-precision GFLOP/s at 20 flops per interaction

```

Cela fait [quelques mois](#) que Windows 10 permet aux applications WSL 2 d'exploiter le GPU hôte. Outre CUDA, Microsoft s'appuie sur l'API DirectML, qui gère l'inférence sur les GPU compatibles DirectX 12. Il en a fait le *back-end* d'une variante de TensorFlow. AMD, Intel et NVIDIA proposent chacun des pilotes qui assurent son fonctionnement à la fois sur Windows et dans WSL.

Photo d'illustration © Jakub Jirsk – Fotolia