OpenForBC Benchmark, the GPU benchmarking framework

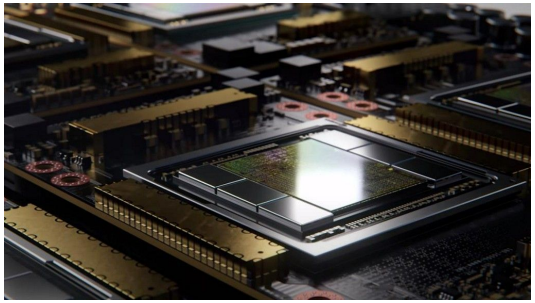
...

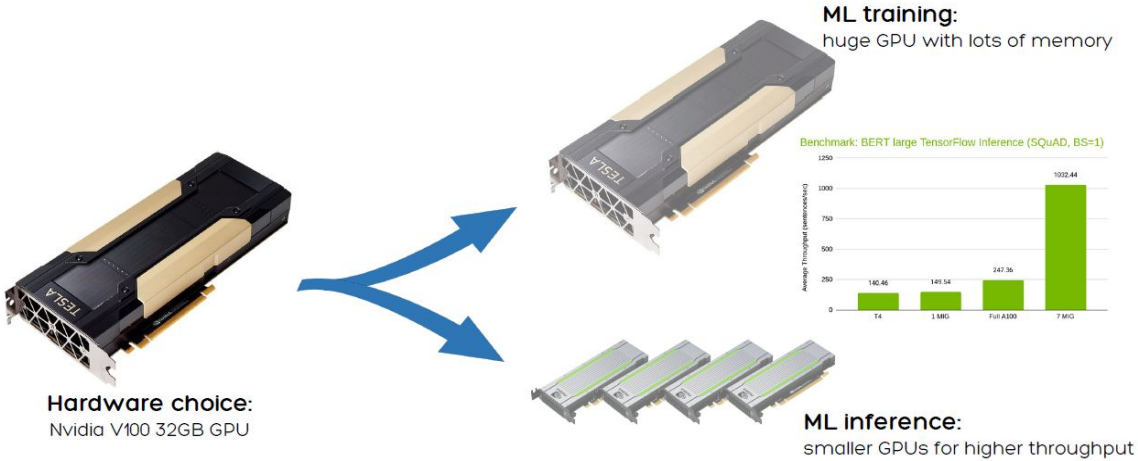
Alessio Borriero, Gabriele Gaetano Fronzé, Federica Legger, Daniele Monteleone

INFN
Istituto Nazionale di Fisica Nucleare

GPU partitioning

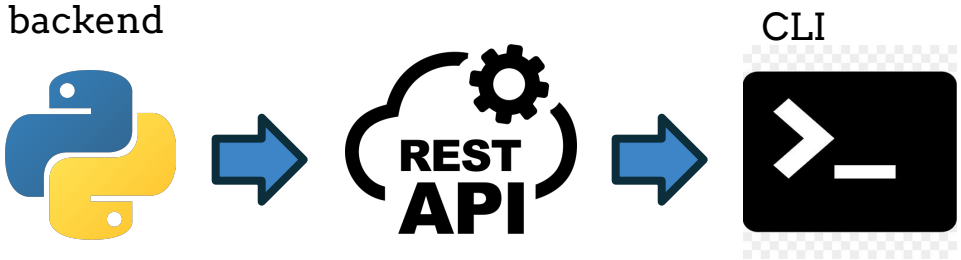
- High-end GPUs available in many computing centers
- Not easy to fully exploit a powerful GPU:
  - Applications need to be optimised
  - Some workflows simply do not need that many FLOPS and cannot saturate memory
- GPU partitioning: many vendors offer the possibility to partition the GPU and assign partitions to different users/processes





OpenForBC in a nutshell

- OpenForBC is an open source software framework that provides a common interface to create and manage partitioning of GPUs of different vendors in a virtualized Linux environment (KVM)



> gpu list

> gpu types

> gpu partition create/destroy

> gpu partition list

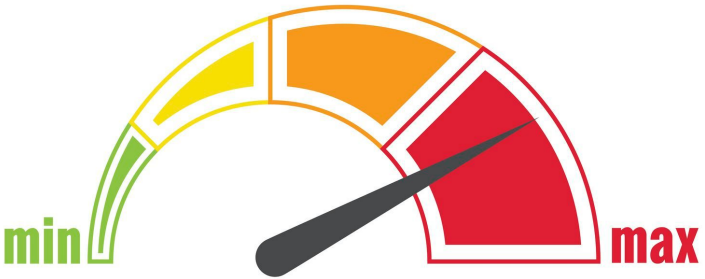
> gpu partition get

https://github.com/Open-ForBC/OpenForBC

Check out our talk at ISGC 2022

OpenForBC Benchmark

- No CLI-based and modular benchmark suite for GPUs on the market
- So we wrote our own: OpenForBC Benchmark
  - modular benchmark suite for GPUs
    - agnostic to GPU partitioning
    - benchmarks may also run on CPU
    - includes our own custom benchmarks
    - compatible with Phoronix benchmarks (WIP)
    - easily expandable with additional benchmark definitions
  - benchmarks run from CLI (support for various presets, suites of benchmarks, …)
  - easily parsable and customisable output



OpenForBC Benchmark

- Open source, available from github
- Python codebase

```
admin@d-h-34 /Users/admin/Work/OpenForBC > git clone --recursive https://githu
b.com/Open-ForBC/OpenForBC-Benchmark.git
Cloning into 'OpenForBC-Benchmark'...
remote: Enumerating objects: 2739, done.
remote: Counting objects: 100% (481/481), done.
remote: Compressing objects: 100% (189/189), done.
remote: Total 2739 (delta 290), reused 428 (delta 264), pack-reused 2258
Receiving objects: 100% (2739/2739), 678.68 KiB | 1.92 MiB/s, done.
Resolving deltas: 100% (1465/1465), done.
admin@d-h-34 /Users/admin/Work/OpenForBC > cd OpenForBC-Benchmark
admin@d-h-34 /Users/admin/Work/OpenForBC/OpenForBC-Benchmark > pip3 install .
```

GitHub

https://github.com/Open-ForBC/OpenForBC-Benchmark

Requirements

- OpenForBC-Benchmark is compatible with Windows, Linux, and macOS given the benchmark supports the tester operating system.
- **Python: >= 3.9, pip: >= 10.0**
- ML benchmarks depends on tensorflow, which requires NVIDIA CUDNN and some cuda libraries to be installed, specifically:
  - cuda-cudart
  - libcublas
  - Libcufft
  - Libcurand
  - Libcusolver
  - libcusparse

Benchmarks

- Currently, the following benchmarks are implemented:
  - **Dummy** -> HelloWorld
  - **Blender** -> Graphics benchmark (scene rendering)
  - **Matmul** -> Matrix multiplication
  - **Matmul C++**
  - **MNIST Real Time**
  - **CIFAR10 Real Time**
  - **Teacher-Student**
  - **TCGA topic modeling**

Tensorflow-based ML Benchmarks

\+ *Work in progress to include about 200 phoronix benchmarks*

Benchmark code structure

File hierarchy:

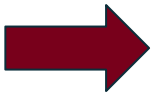
```
.
+-- benchmarks
      |-- Sample Benchmark A
      |  |-- benchmark.json                    *
      |  |-- presets                           *
      |  |  +-- preset<#>.json
      |  +-- <benchmark executables/scripts/docs>
```

* Essential for executing benchmark

Benchmark code structure

File hierarchy:

```
.
+-- benchmarks
      |-- Sample Benchmark A
      |  |-- benchmark.json
      |  |-- presets
      |  |  +-- preset<#>.json
      |  +-- <benchmark executables/
```



```
{
  "name": "Sample Benchmark A",
  "description": "A sample benchmark",
  "default_preset": "preset1",
  "test_preset": "preset1",
  "setup_command": "setup.sh --prepare",
  "run_command": {
    "command": "example_bench --gpu",
    "env": {
      "CPU": "0"
    }
  },
  "cleanup_command": "setup.sh --clean",
  "stats": {
    "mult_time_ms": {
      "regex": "Matrix multiplication time: (\\d+)ms"
    }
  },
  "virtualenv": true
}
```

Benchmark code structure

File hierarchy:

.
+-- benchmarks
|-- Sample Benchmark A
| |-- benchmark.json
| |-- presets
| | +-- **preset<#>.json**
| +-- <benchmark executables/scripts/docs>

```
{
        "args": "gpu training -n 20"
}
```

OpenForBC-Benchmark in action

```
admin@d-h-34 /Users/admin/Work/OpenForBC/OpenForBC-Benchmark > o4bc-bench benchmark list
CIFAR_realtime_benchmark
TeacherStudent_realtime_benchmark
MNIST_FCNeuralNetwork
matmulCpp_benchmark
dummy_benchmark
matmul_benchmark
dummy_py_benchmark
MNIST_realtime_benchmark
admin@d-h-34 /Users/admin/Work/OpenForBC/OpenForBC-Benchmark > o4bc-bench benchmark run matmulCpp_benchmark
Running "matmulCpp_benchmark" setup commands
$ python3 -m venv .venv
(venv) $ ./setup.sh
./matmulCpp_benchmark.cpp:38:3: warning: 'auto' type specifier is a C++11 extension [-Wc++11-extensions]
  auto **mul = new double *[dim1];
  ^
./matmulCpp_benchmark.cpp:85:3: warning: 'auto' type specifier is a C++11 extension [-Wc++11-extensions]
  auto **m1 = new double *[dim1];
  ^
./matmulCpp_benchmark.cpp:86:3: warning: 'auto' type specifier is a C++11 extension [-Wc++11-extensions]
  auto **m2 = new double *[dim2];
  ^
./matmulCpp_benchmark.cpp:90:3: warning: 'auto' type specifier is a C++11 extension [-Wc++11-extensions]
  auto time_start = high_resolution_clock::now();
  ^
./matmulCpp_benchmark.cpp:92:3: warning: 'auto' type specifier is a C++11 extension [-Wc++11-extensions]
  auto time_end = high_resolution_clock::now();
  ^
5 warnings generated.
(venv) $ chmod +x bin/matmulCppExe
Running "matmulCpp_benchmark" preset "matrix_20x30"
(venv) $ bin/matmulCppExe 20 30
Matrix multiplication time: 0.000055 s
Preset        Stat             Value

matrix_20x30  matmul_time_s  5.5e-05
admin@d-h-34 /Users/admin/Work/OpenForBC/OpenForBC-Benchmark >
```

Logs for each benchmark run can be found in **/logs** directory. Each run is saved by the following format:

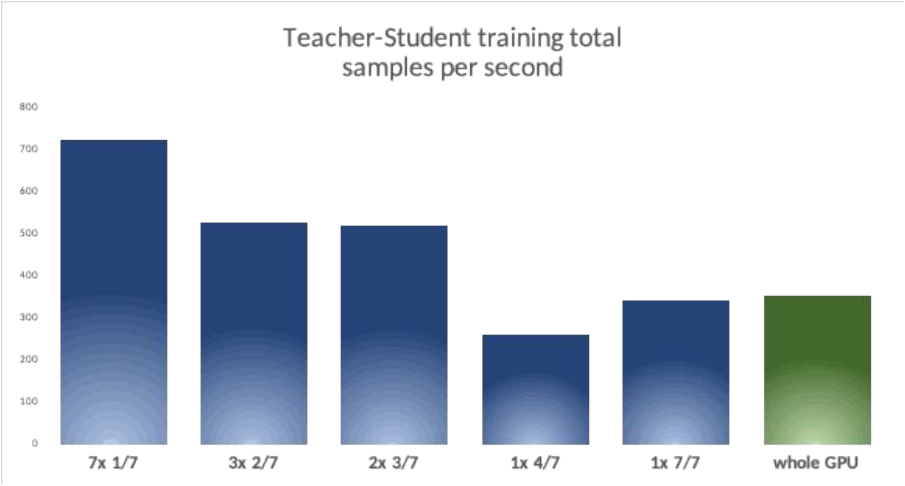
<benchmark_name>/<yyyymmdd_hhmmss>/<phase>_<preset>.<command_number>.<out/err>.log

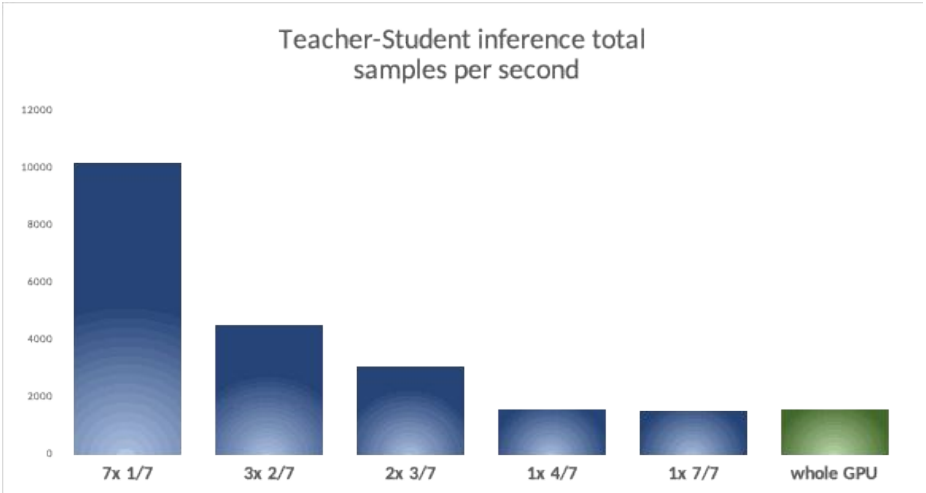
More screenshots

```
Running "CIFAR_realtime_benchmark" preset "training"
(venv) $ ./CIFAR_realtime.py gpu training -n 2
2022-06-08 14:34:44.402707: I tensorflow/core/platform/cpu_feature_guard.cc:193] This TensorFlow binary is optimized with oneAPI Deep Ne
ural Network Library (oneDNN) to use the following CPU instructions in performance-critical operations:  AVX2 FMA
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
2022-06-08 14:34:46.589700: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1532] Created device /job:localhost/replica:0/task:0/devi
ce:GPU:0 with 13795 MB memory:  -> device: 0, name: Tesla T4, pci bus id: 0000:41:00.0, compute capability: 7.5
Epoch 1/2
2022-06-08 14:34:53.172755: I tensorflow/stream_executor/cuda/cuda_dnn.cc:384] Loaded cuDNN version 8100
2022-06-08 14:34:53.891051: I tensorflow/core/platform/default/subprocess.cc:304] Start cannot spawn child process: No such file or dire
ctory
2022-06-08 14:34:53.895005: I tensorflow/core/platform/default/subprocess.cc:304] Start cannot spawn child process: No such file or dire
ctory
2022-06-08 14:34:53.895042: W tensorflow/stream_executor/gpu/asm_compiler.cc:80] Couldn't get ptxas version string: INTERNAL: Couldn't i
nvoke ptxas --version
2022-06-08 14:34:53.897143: I tensorflow/core/platform/default/subprocess.cc:304] Start cannot spawn child process: No such file or dire
ctory
2022-06-08 14:34:53.897227: W tensorflow/stream_executor/gpu/redzone_allocator.cc:314] INTERNAL: Failed to launch ptxas
50000/50000 [==============================] - 178s 3ms/step - loss: 1.6123 - accuracy: 0.4058
Epoch 2/2
50000/50000 [==============================] - 175s 4ms/step - loss: 1.2735 - accuracy: 0.5489
```

```
TRAINING COMPLETED!
total_time: 311.343798160553
Relying on driver to perform ptx compilation.
avg_time_per_sample: 0.0031134379816055296
Modify $PATH to customize ptxas location.
This message will be only logged once.
Preset    Stat                    Value
--------  ------------------  ------------
training  total_time          311.344
training  avg_time_per_sample   0.00311344
```

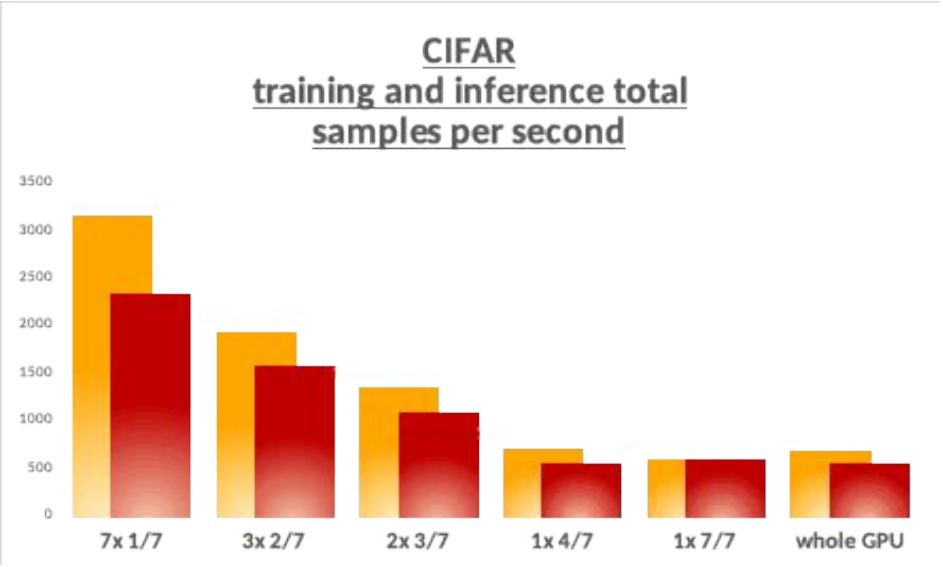
Teacher-Student ML Benchmark

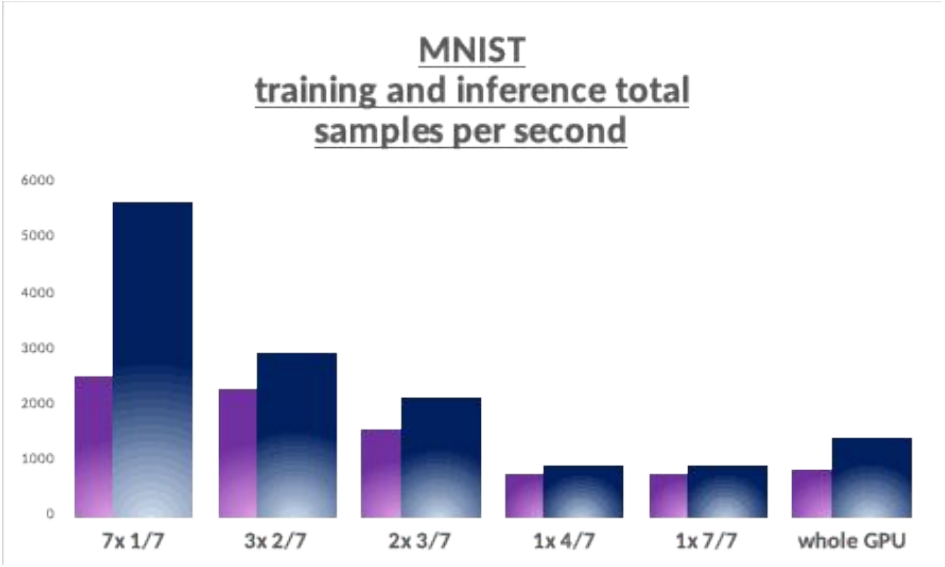




- peak throughput computed as the sum of the average throughput of all creatable partitions given a specific profile
- All creatable partitions have been allocated and loaded with computation

CIFAR and MNIST ML benchmarks





CNN for image recognition on CIFAR dataset

FFNN for hand-writing recognition on MNIST dataset

Outlook

- OpenForBC Benchmark is an **expandable modular** benchmark framework for GPUs
  - Open source python code base
  - Custom + blender + phoronix (WIP) benchmarks
  - Run single benchmarks or suites from CLI
  - Easy-to-parse output

```
(o4bc-3.9) jovyan@jupyter-leggerf /home/jovyan/OpenForBC-Benchmark > o4bc-bench benchmark
ID                                Name                Description                                Default preset
--------------------------------  ------------------  -----------------------------------------  ----------------------
dummy_benchmark                   Dummy Benchmark     Does nothing                               preset1
matmul_benchmark                  Matrix...           Some test of matrices operations           matrix_20x30_GPU
MNIST_FCNeuralNetwork             Fully Connected...  Training and inference with MNIST...       gpu_shallow_largebatch
dummy_py_benchmark                Dummy Python...     Does nothing (with virtualenv)             preset1
TCGA_topicmodeling                TCGA benchmark...   A benchmark on TCGA Breast cancer          inference
blender_benchmark                 Blender benchmark   Runs one/many blender scene(s) and...      classroom_cpu
matmulCpp_benchmark               Matrix...           Some test of matrices operations           matrix_20x30
MNIST_realtime_benchmark          MNIST realtime...   Training and inference with MNIST...       training
TeacherStudent_realtime_benchmark Realtime Teacher-... Training and inference using Teacher-... inference
CIFAR_realtime_benchmark          CIFAR realtime...   Training and inference with CIFAR...       training
```

OpenForBC: who?

Federica Legger
Technologist INFN

Gabriele Gaetano Fronzé
UniTo Post-doc grant

Alessio Borriero
INFN Student grant

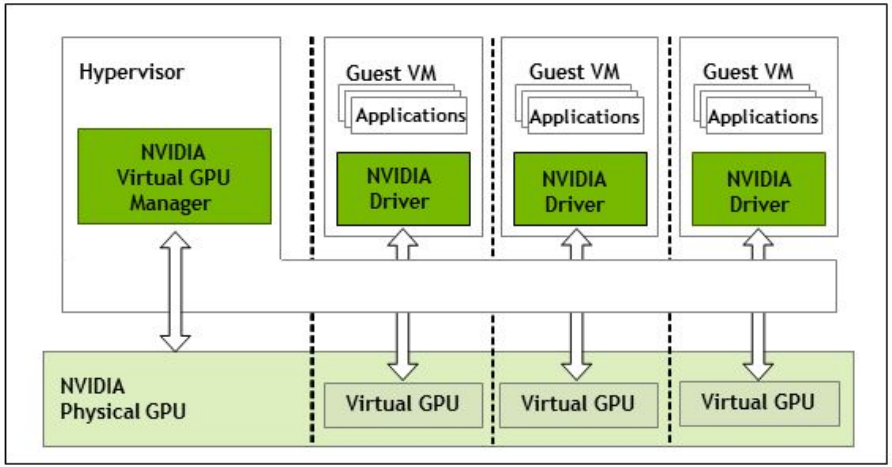
Daniele Monteleone
INFN Student grant

Sponsors

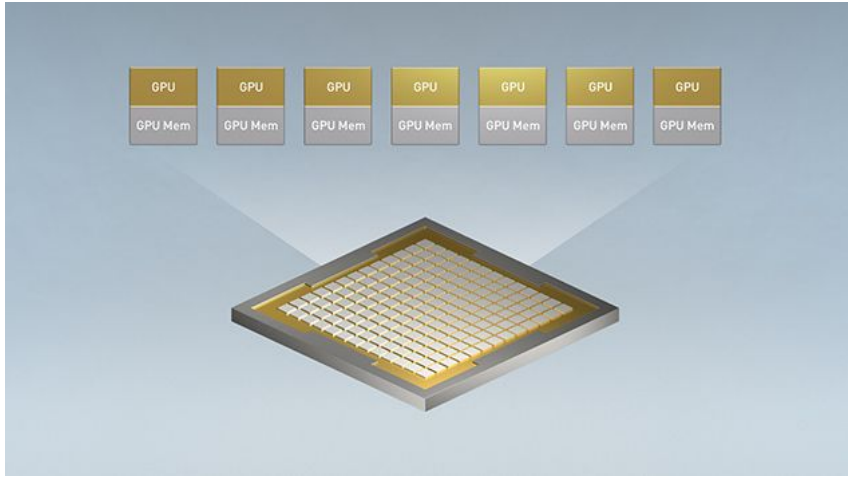
Backup

Nvidia GPU partitioning

- Temporal partitioning: **vGPU**
  - On NVIDIA A100 (40 GB) up to 10 vGPUs with 4 GB memory allocated per VM
- Spatial partitioning: **MIG**
  - Up to 7 fully isolated instances with 5 GB memory each on an A100

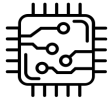


vGPU



MIG

Easy?

	Nvidia VGPU	Nvidia MIG	AMD MxGPU	PCIe SR-IOV
 **Full API support across profiles** complete set of API for compute and graphics	⚠️	⚠️	⚠️	N/A
 **P2P communications between partitions** connects multiple virtual partitions for computing	✅	⛔	N/A	N/A
 **Free and easy licensing model** license included or requires additional costs/procedures	⛔	⚠️	✅	✅
 **Trivial compatibility matrix** delegated to OS with no limitations wrt an equivalent physical GPU	⚠️	✅	✅	✅
 **Certified on any compatible host system** Compatible with any physically and electrically supporting hardware	⛔	⛔	✅	✅