

# Verify your bindings with numa\_api

- It is useful to know if your distributed learning program is making use of the node resources in the most optimal manner possible
- The `numa\_api` library is used to test process ID (PID) bindings within a Python program.
  - Based on the cores, it then suggests the most optimal GPU based on the NUMA domains
- New documentation added to the PyTorch docs:  
[https://docs.olcf.ornl.gov/software/analytics/pytorch\\_frontier.html#torchrn-and-other-distributed-pytorch-launchers](https://docs.olcf.ornl.gov/software/analytics/pytorch_frontier.html#torchrn-and-other-distributed-pytorch-launchers)
- Intended for verification, don't pipe the output directly into something like `torch.cuda.set\_device`

```
core affinity for PID 1132004: 1 2 3 4 5 6 7
Suggested GPU for PID 1132004: 4

core affinity for PID 1132006: 17 18 19 20 21 22 23
Suggested GPU for PID 1132006: 2

core affinity for PID 1132008: 33 34 35 36 37 38 39
Suggested GPU for PID 1132008: 6

core affinity for PID 1132009: 41 42 43 44 45 46 47
Suggested GPU for PID 1132009: 7
```

Example srun usage (recommended)

```
core affinity for PID 884147: 1
Suggested GPU for PID 884147: 4

core affinity for PID 884145: 1
Suggested GPU for PID 884145: 4

core affinity for PID 884146: 1
Suggested GPU for PID 884146: 4

core affinity for PID 884141: 1
Suggested GPU for PID 884141: 4
```

Example output when using torchrn: misleading things, based on torchrn (incorrectly) launching all on the same core