PYTORCH - AMPERE® AI OPTIMIZER(AIO) DOCUMENTATION

v1.0.0

AMPERE COMPUTING

Table of Contents

OVERVIEW

Ampere® AIO inference acceleration engine is fully integrated with Pytorch framework. Pytorch models and software written with Pytorch API can run as-is, without any modifications.

PYTORCH FRAMEWORK

The Python is installed with AIO accelerated Pytorch package, together with all dependencies. No extra installation step is needed

Versions Compatibility

This release is based on Pytorch 1.9.0. Torchvision 0.10.0 is installed also which is compatible with Pytorch 1.9.0.

Python

This version of Pytorch is built for Python 3.8. For support of other Python versions, please contact your Ampere sales representative. If you are using the SW through a 3rd party, please contact their customer support team for help. Alternatively, you can also contact the AI team at ai-support@amperecomputing.com

AIO CONFIGURATIONS

AIO inference engine can be configured by a set of environment variables for performance and debugging purposes. They can be set in the command line when running Pytorch models (e.g. AIO_NUM_THREADS=16 python run.py -p fp32) or set in the shell initialization script.

AIO_PROCESS_MODE

This variable controls if AIO inference engine is used in running the Pytorch model.

- 0 : AIO is disabled
- 1 : AIO is enabled (Default)

AIO_CPU_BIND

Enables core binding. If enabled, each AIO thread will bind itself to single core.

- 0 : Core binding disabled

- 1 : Core binding enabled (Default)

AIO_MEM_BIND

Bind memory to NUMA (Non-uniform memory access) node 0. For optimal performance, numactl (https://linux.die.net/man/8/numactl) should be preferred. numactl bind will affect both Pytorch framework and AIO buffers, while AIO is unable to affect buffers allocated by Pytorch framework.

- 0: Mem bind disabled
- 1: Mem bind to node 0 (Default)

AIO_NUMA_CPUS

Select cores that AIO should bind to (if CPU_BIND is enabled).

- Not set: AIO will use the first N cores of the machine, excluding hyper-threaded machines (Default)
- Set: AIO will attempt to use N first cores from the list of cores for N threads. The list is in space separated, 0-based number format. For example, selecting cores 0 to 1: AIO_NUMA_CPUS="0 1"

AIO_NUM_THREADS

Specifies the number of threads that AIO should use.

- Not set: AIO will use one core (Default)
- "all": AIO will use all cores, as specified by AIO_NUMA_CPUS
- N: AIO will use N threads

AIO_DEBUG_MODE

Control verbosity of debug messages

- 0: No messages
- 1: Errors only
- 2: Basic information, warnings and errors (Default)
- 3: Most messages
- 4: All messages

QUICKSTART

Following instructions run on Altra / Altra Max Linux machines installed with Docker. To initialize AIO environment run:

Pull and run docker image

```
$ docker pull <your_unique_url>
$ docker run --privileged=true --rm --name pytorch-aio --network host -it ghcr.io/onspecta/aio-pytorch-1.9.0:1.0.0
```

You can try AIO by either running jupyter notebook examples or python scripts on CLI level.

To run jupyter notebook QuickStart examples please follow the instructions below:

Set AIO_NUM_THREADS to requested value first.

```
$ export AIO_NUM_THREADS=16
$ cd ~/aio-examples/
$ bash start_notebook.sh
```

If you run it on a cloud instance, make sure your machine has port 8080 open and on your local device run:

```
$ ssh -N -L 8080:localhost:8080 -I <ssh_key> your_user@xxx.xxx.xxx.xxx
```

Use a browser to point to the URL printed out by the Jupyter notebook launcher.
You will find Jupyter Notebook examples, examples.ipynb, under /classification folder.

The examples run through several inference models, visualize results they produce and present the performance numbers.

To use CLI-level scripts:

Set AIO_NUM_THREADS to requested value first.

```
$ export AIO_NUM_THREADS=16
$ cd ~/aio-examples/
```

Go to the directory of choice, eg.

```
$ cd classification/resnet_50_v1
```

Evaluate the model.

```
$ numactl --physcpubind=0-15 python3 run.py -p fp32
```

Ampere Computing® / 4655 Great America Parkway, Suite 601 / Santa Clara, CA 95054 / www.amperecomputing.com

AMP 2019-0039