Chapter 5.3: Running DNN Multi-GPU Training On ROCm
# Table of Contents

CHAPTER 5.3: RUNNING DNN MULTI-GPU TRAINING ON ROCM

- Preparation .................................................................................................................. 2
- Running the Example .................................................................................................... 3
Chapter 5.3: Running DNN Multi-GPU Training On ROCm

This hands-on tutorial shows how we can get started with the popular Keras framework and deploy a MNIST model that uses multiple GPUs on ROCm.

Preparation

1. We will be using the AMD provided Tensorflow docker container for this tutorial
   - Please ensure that ROCm is correctly installed and docker is also set up using the instructions in our installation guide

2. Obtain the TensorFlow docker image:
   - docker pull rocm/tensorflow

3. Start the docker container:
   - alias drun='sudo docker run --network=host --device=/dev/kfd --device=/dev/dri --ipc=host --shm-size 16G --group-add video --cap-add=SYS_PTRACE --security-opt seccomp=unconfined -v $HOME/dockerx:/dockerx'
     - drun rocm/tensorflow:latest

4. Set environment variable “HIP_VISIBLE_DEVICES”
   - export HIP_VISIBLE_DEVICES=0,1...N-1 where N is the number of multi-GPUs in your system.
   - For example, if we have two GPUs the variable is set as “export HIP_VISIBLE_DEVICES=0,1”
Running the Example

1. On the tutorial repo on your system run:
   - Clone the tutorial repo inside your docker container
     - `cd Chapter5/02_Keras_MultiGPU_ROCm/` in the tutorial repo
   - Based on the number of GPUs you have set in the environment variable `HIP_VISIBLE_DEVICES`, you will have to modify line number 36 accordingly.
   - For example; if you are using 2 GPUs it will look like this:
     ```python
     model = multi_gpu_model(model, gpus=2, cpu_merge=True, cpu_relocation=False)
     ```
     - Refer to the linked documentation for more info on this API call (https://www.tensorflow.org/api_docs/python/tf/keras/utils/multi_gpu_model)

2. Execute the script:
   - `python3 multi_gpu_mnist.py`

---

Figure 1: Output of the Keras Model for training on the MNIST dataset using a 2-GPU system