HIGH PERFORMANCE RESEARCH COMPUTING

ACES: Using Graphcore Intelligence Processing Unit

03/05/2024

Zhenhua He

High Performance Research Computing
DIVISION OF RESEARCH

ACES
ACCELERATING COMPUTING FOR EMERGING SCIENCES

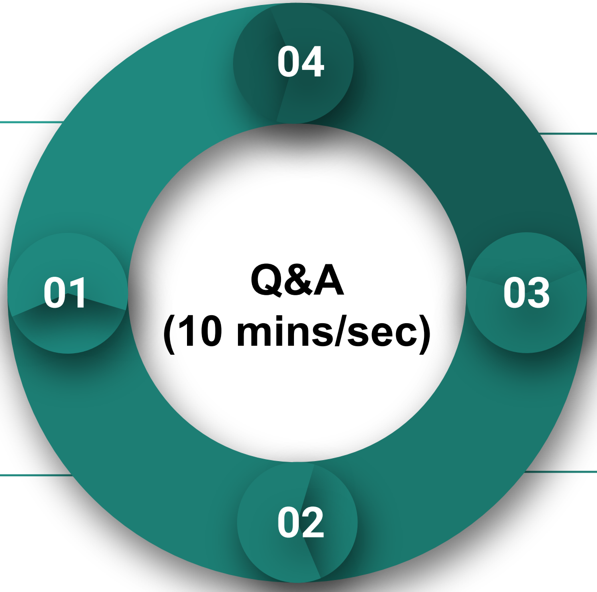
IPU Tutorial Outline

Section I. Intro to IPUs

We will introduce Graphcore IPU architecture, and the IPU systems on TAMU ACES platform.

Section II. Demo on ACES

We will demonstrate how to run models of different frameworks on ACES IPU system.



Section IV. Porting PyTorch code to IPU

We will learn to port a PyTorch Fashion-MNIST classification model to run on IPU

Section III Porting TensorFlow code to IPU

We will learn to port a Keras MNIST classification model to run on IPU

Section I. Overview

ACES

ACCELERATING COMPUTING
FOR EMERGING SCIENCES

NSF ACES

Accelerating Computing for Emerging Sciences

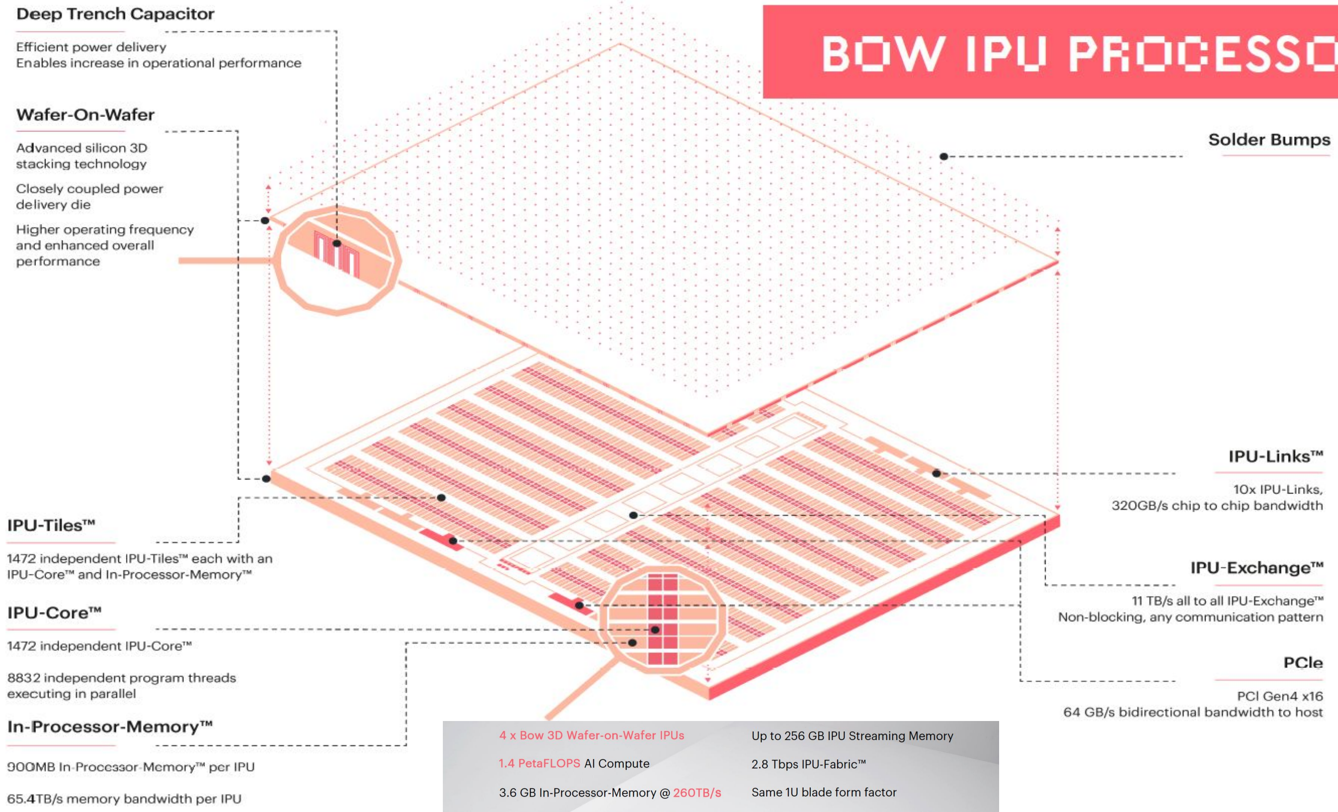
Our Mission:

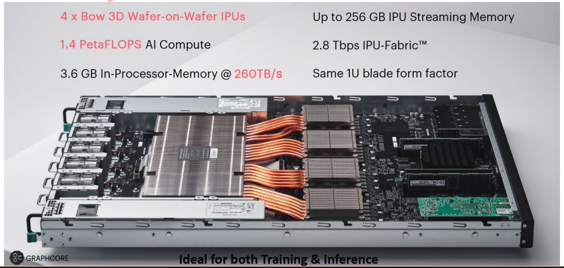
- NSF ACSS CI test-bed
- Offer an accelerator testbed for numerical simulations and **AI/ML workloads**
- Provide consulting, technical guidance, and training to researchers
- Collaborate on computational and data-enabled research.

ACES

ACCELERATING COMPUTING
FOR EMERGING SCIENCES

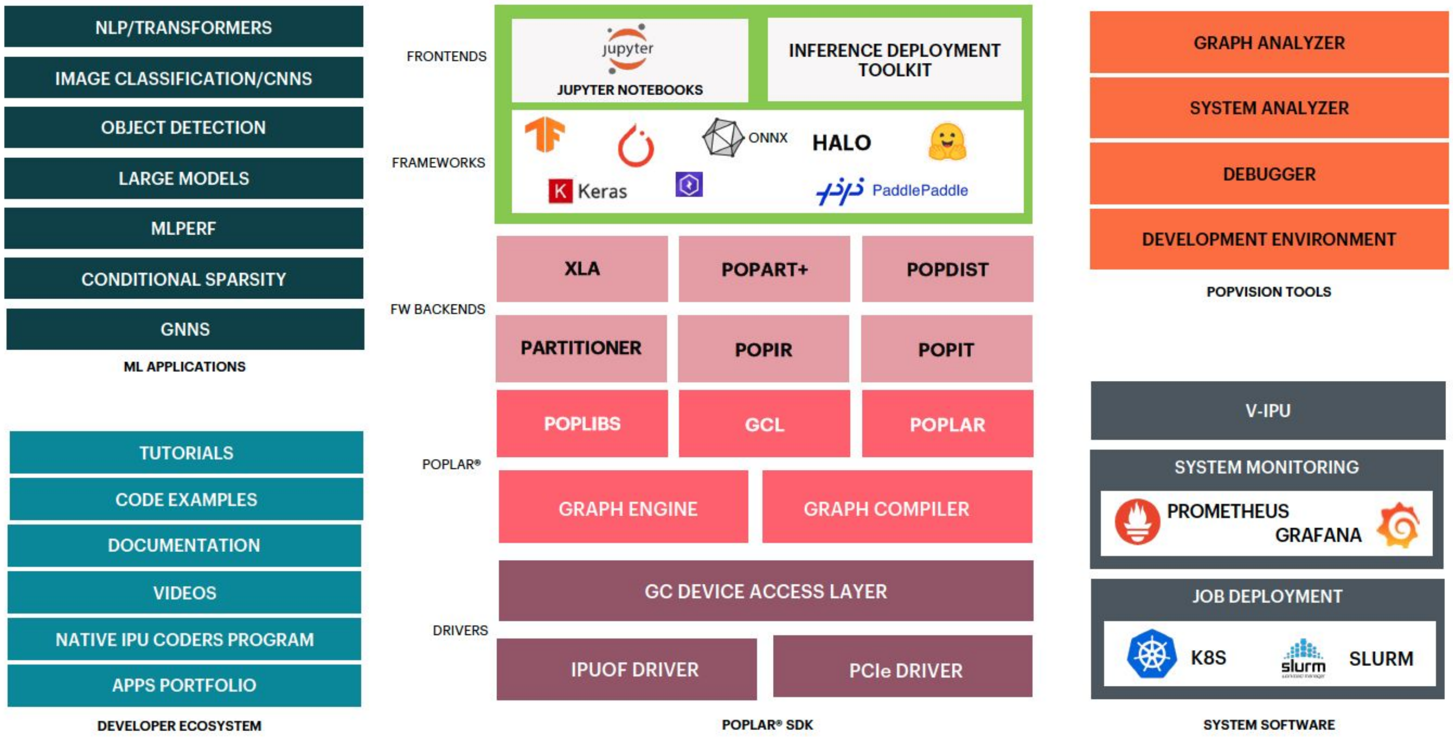
BOW IPU PROCESSOR





Source: Graphcore

Graphcore Software Stack



Source: Graphcore

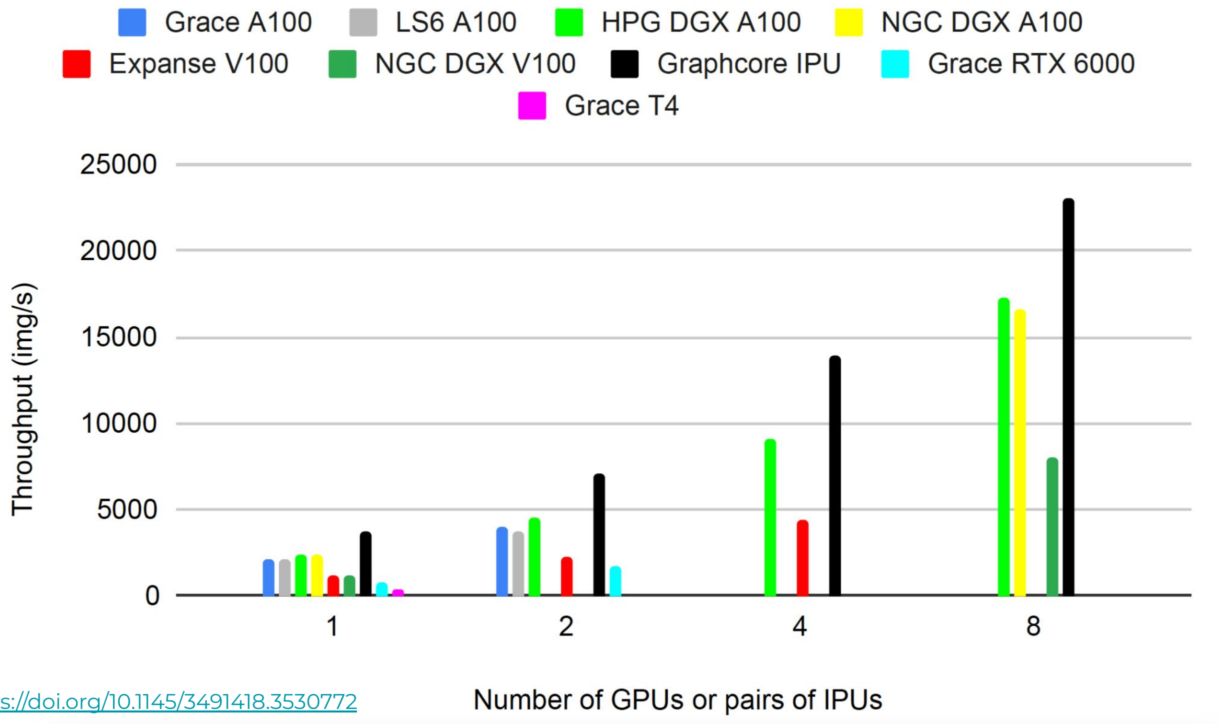
Models on Graphcore GitHub

Vision	ResNet50, EfficientNet, DINO, MAE, Neural Image Fields, SWIN ((Shifted Windows Vision Transformers)), U-Net, ViT (Vision Transformer), YOLOv4, etc.
NLP	BERT, BLOOM-176B (BigScience Large Open-science Open-access Multilingual), GPT-2, GPT-3 2.7B, GPT-3 175B, GPT-J, etc.
Speech	Conformer, FastPitch, etc.
GNN	Cluster-GCN, GIN (Graph Isomorphism Network), NBFnet (Neural Bellman-Ford networks), SchNet, Spektral, TGN (Temporal Graph Networks), etc.
Multi-modal	CLIP, Frozen in time, MAGMA (Multimodal Augmentation of Generative Models through Adapter-based Finetuning), Mini DALL-E, etc.

Source: https://github.com/graphcore/examples

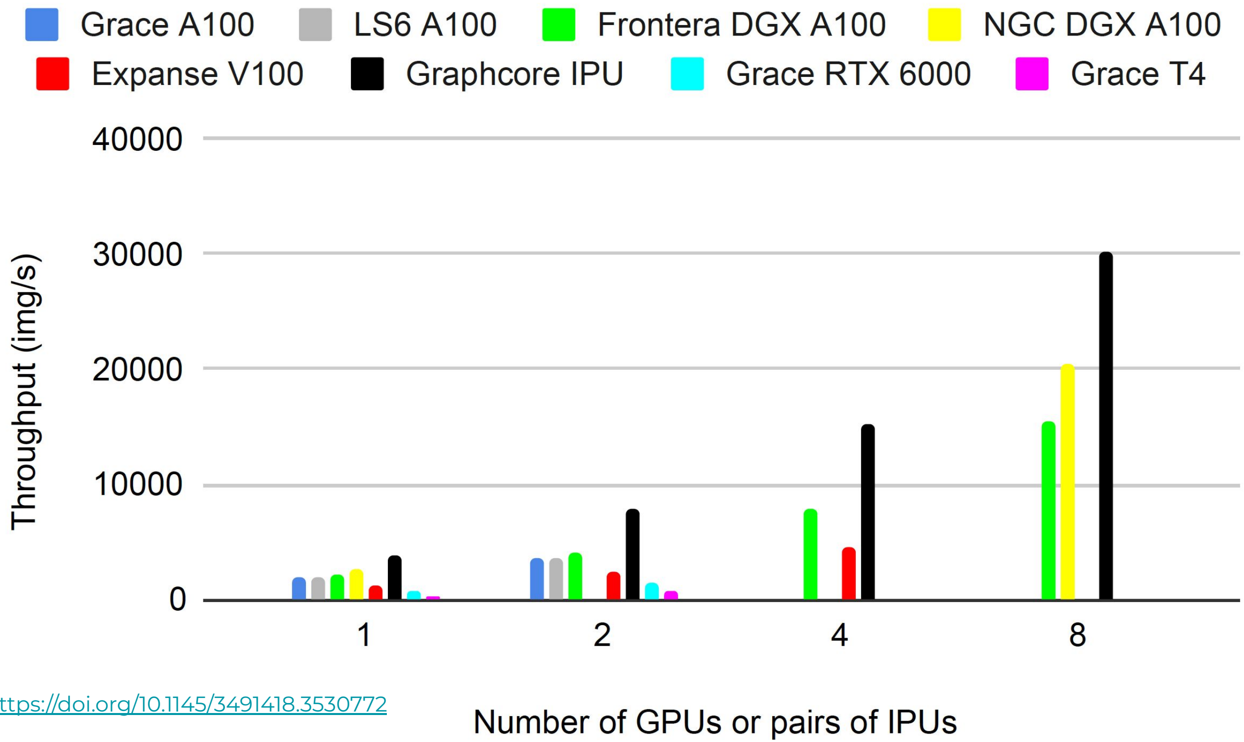
Choosing Between IPUs and GPUs

PyTorch ResNet-50 - GPU vs IPU



Abhinand S. Nasari, et al. https://doi.org/10.1145/3491418.3530772

TensorFlow ResNet-50 - GPU vs IPU



Abhinand S. Nasari, et al. https://doi.org/10.1145/3491418.3530772

Take-home message

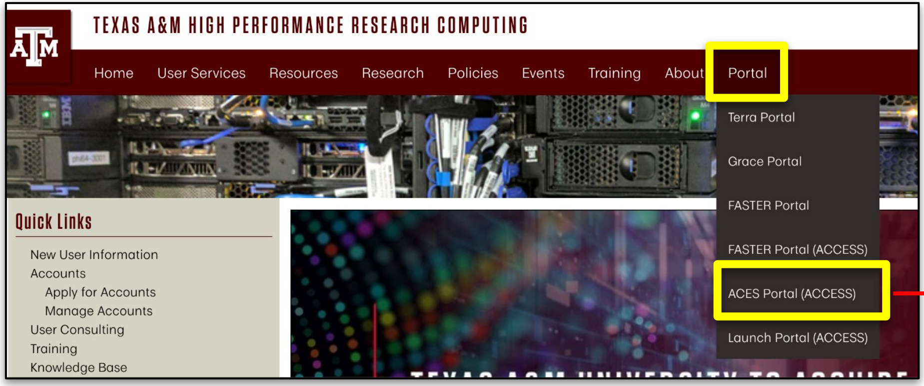
- The performance of ML workflows is highly sensitive to the distributed workflow configuration.
- Increasing the batch size increases the memory pressure and the amount of data that needs to be communicated, but most importantly it usually improves hardware utilization and thus computation.
- IPU is more fine-grained parallelism compared to GPUs.

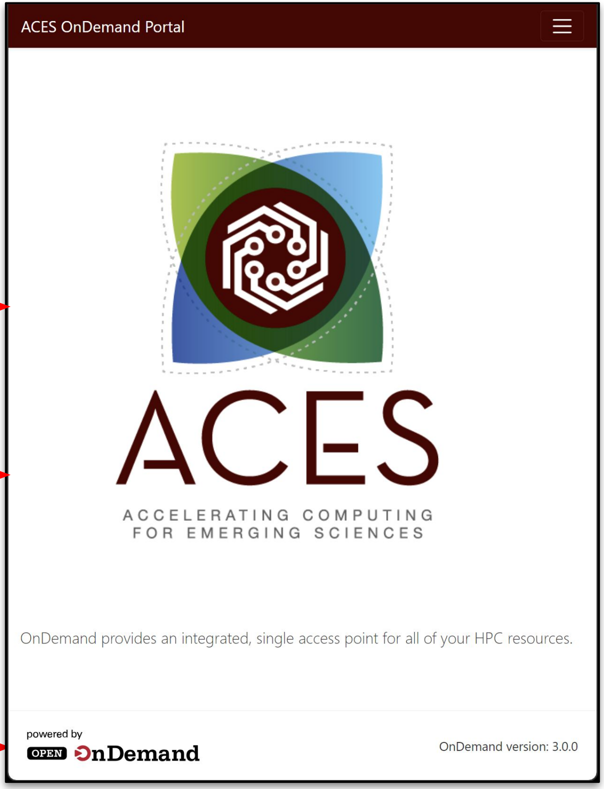
Section II. Demo on ACES

ACES

ACCELERATING COMPUTING FOR EMERGING SCIENCES

ACES Portal





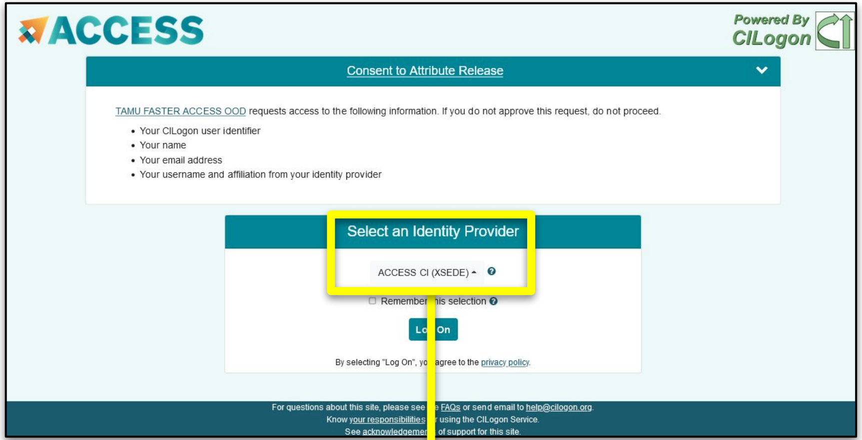
ACES Portal portal-aces.hprc.tamu.edu
is the web-based user interface for the ACES cluster

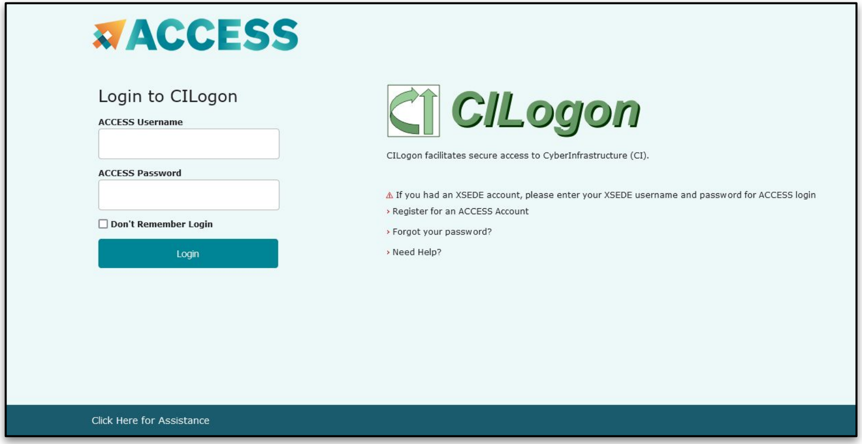
HPRC Portal YouTube tutorials

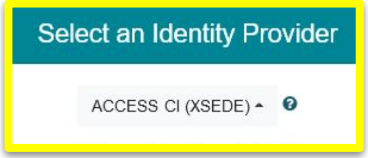
Open OnDemand (OOD) is an advanced web-based graphical interface framework for HPC users

Accessing ACES via ACCESS

Log-in using your ACCESS CI credentials.



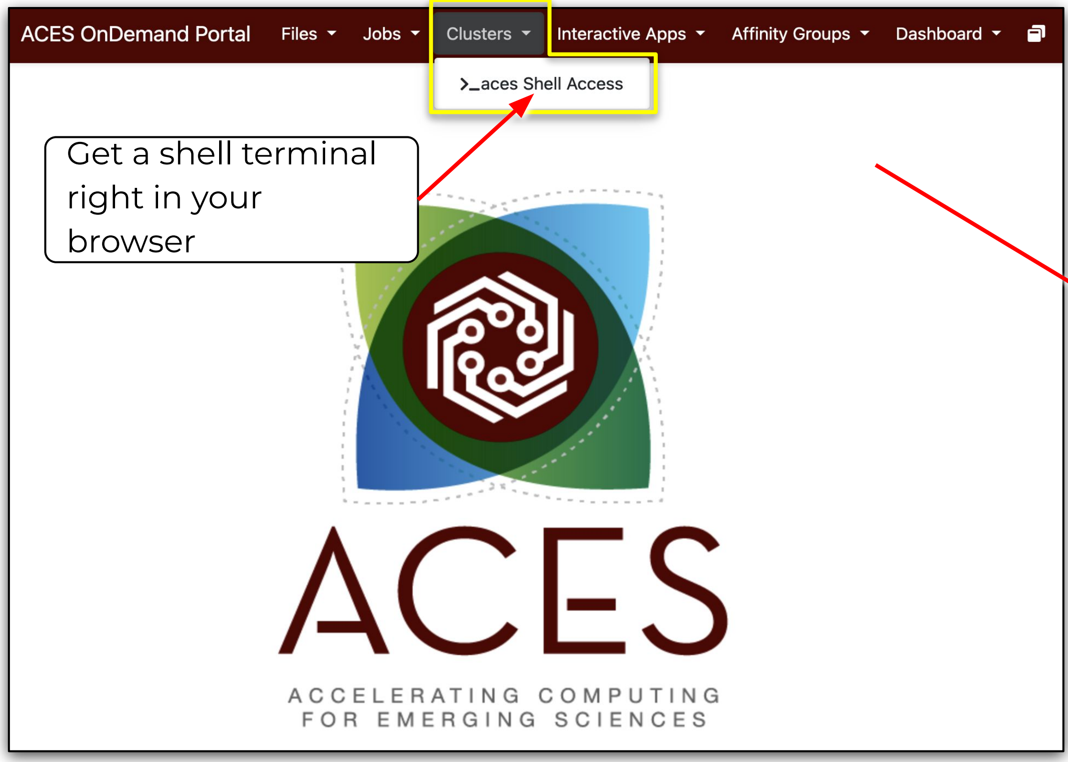




Select the Identity Provider appropriate for your account.

Shell Access via the Portal

Get a shell terminal right in your browser



Training Materials

From the ACES login node, ssh into the poplar2 (BOW Pod16) IPU system

```
ssh poplar2
```

Change to your scratch directory:

```
cd /localdata/$USER && mkdir ipu_labs && cd ipu_labs
```

Copy the example materials to your scratch directory:

```
git clone https://github.com/graphcore/examples.git
```

Copy the hands-on exercise materials to your scratch directory:

```
git clone https://github.com/happidence1/IPU-Training.git
```

Poplar SDK setup

```
source
/opt/gc/poplar/poplar_sdk-ubuntu_20_04-3.3.0+1403-208993bbb7/poplar-ub
untu_20_04-3.3.0+7857-b67b751185/enable.sh

source
/opt/gc/poplar/poplar_sdk-ubuntu_20_04-3.3.0+1403-208993bbb7/popart-ub
untu_20_04-3.3.0+7857-b67b751185/enable.sh

mkdir -p /localdata/$USER/tmp
export TF_POPLAR_FLAGS=--executable_cache_path=/localdata/$USER/tmp
export POPTORCH_CACHE_DIR=/localdata/$USER/tmp
```

Run a TensorFlow (TF) model on IPU

TF Virtual Environment Setup

```
virtualenv -p python3 venv_tf2

source venv_tf2/bin/activate

python -m pip install -U pip

python -m pip install
/opt/gc/poplar/poplar_sdk-ubuntu_20_04-3.3.0+1403-208993bbb7/
tensorflow-2.6.3+gc3.3.0+251582+08d96978c7f+intel_skylake512-
cp38-cp38-linux_x86_64.whl
```

Run a TensorFlow model on IPU

```
cd examples/tutorials/tutorials/tensorflow2/keras/completed_demos/

python completed_demo_ipu.py
```

- Deactivate the virtual environment after the model finishes running.

```
deactivate
```

Monitor IPU Usage - *gc-monitor*

- 4 partitions
- 16 IPUs
- Processes

```
Every 2.0s: gc-monitor                                                poplar2: Thu Jul  6 10:33:31 2023

+---------------+--------------------------------------------------------------------------------+
|  gc-monitor   |              Partition: p17 [active] has 16 reconfigurable IPUs               |
+-------------+-------------------+--------+----------------+----------+-------+----+------+-------+
|    IPU-M    |       Serial      |IPU-M SW|Server version|  ICU FW  | Type  | ID | IPU# |Routing|
+-------------+-------------------+--------+----------------+----------+-------+----+------+-------+
|  10.5.5.1   | 0019.0002.8222521 | 2.6.0  |     1.11.0     |  2.5.9   | M2000 | 0  |  3   |  DNC  |
|  10.5.5.1   | 0019.0002.8222521 | 2.6.0  |     1.11.0     |  2.5.9   | M2000 | 1  |  2   |  DNC  |
|  10.5.5.1   | 0019.0001.8222521 | 2.6.0  |     1.11.0     |  2.5.9   | M2000 | 2  |  1   |  DNC  |
|  10.5.5.1   | 0019.0001.8222521 | 2.6.0  |     1.11.0     |  2.5.9   | M2000 | 3  |  0   |  DNC  |
+-------------+-------------------+--------+----------------+----------+-------+----+------+-------+
|  10.5.5.2   | 0021.0002.8222521 | 2.6.0  |     1.11.0     |  2.5.9   | M2000 | 4  |  3   |  DNC  |
|  10.5.5.2   | 0021.0002.8222521 | 2.6.0  |     1.11.0     |  2.5.9   | M2000 | 5  |  2   |  DNC  |
|  10.5.5.2   | 0021.0001.8222521 | 2.6.0  |     1.11.0     |  2.5.9   | M2000 | 6  |  1   |  DNC  |
|  10.5.5.2   | 0021.0001.8222521 | 2.6.0  |     1.11.0     |  2.5.9   | M2000 | 7  |  0   |  DNC  |
+-------------+-------------------+--------+----------------+----------+-------+----+------+-------+
|  10.5.5.3   | 0013.0002.8222521 | 2.6.0  |     1.11.0     |  2.5.9   | M2000 | 8  |  3   |  DNC  |
|  10.5.5.3   | 0013.0002.8222521 | 2.6.0  |     1.11.0     |  2.5.9   | M2000 | 9  |  2   |  DNC  |
|  10.5.5.3   | 0013.0001.8222521 | 2.6.0  |     1.11.0     |  2.5.9   | M2000 | 10 |  1   |  DNC  |
|  10.5.5.3   | 0013.0001.8222521 | 2.6.0  |     1.11.0     |  2.5.9   | M2000 | 11 |  0   |  DNC  |
+-------------+-------------------+--------+----------------+----------+-------+----+------+-------+
|  10.5.5.4   | 0016.0002.8222521 | 2.6.0  |     1.11.0     |  2.5.9   | M2000 | 12 |  3   |  DNC  |
|  10.5.5.4   | 0016.0002.8222521 | 2.6.0  |     1.11.0     |  2.5.9   | M2000 | 13 |  2   |  DNC  |
|  10.5.5.4   | 0016.0001.8222521 | 2.6.0  |     1.11.0     |  2.5.9   | M2000 | 14 |  1   |  DNC  |
|  10.5.5.4   | 0016.0001.8222521 | 2.6.0  |     1.11.0     |  2.5.9   | M2000 | 15 |  0   |  DNC  |
+-------------+-------------------+--------+----------------+----------+-------+----+------+-------+

+---------------------------------------------------------+----------------------+----------------+
|          Attached processes in partition p17            |         IPU          |     Board      |
+--------+----------------------+--------+--------------+----+---------+---------+--------+-------+
|  PID   |       Command        |  Time  |     User     | ID |  Clock  |  Temp   |  Temp  | Power |
+--------+----------------------+--------+--------------+----+---------+---------+--------+-------+
| 902631 |        python        |  50s   | u.zh108696   | 0  | 1500MHz | 23.5 C  | 21.9 C | 90.3 W|
+--------+----------------------+--------+--------------+----+---------+---------+--------+-------+
```

```
watch -n 2 gc-monitor
```

Run a PyTorch (PopTorch) model on IPU

PopTorch Virtual Environment Setup

```
cd /localdata/$USER/ipu_labs

virtualenv -p python3 poptorch_test

source poptorch_test/bin/activate

python -m pip install -U pip

python -m pip install
/opt/gc/poplar/poplar_sdk-ubuntu_20_04-3.3.0+1403-208993bbb7/
poptorch-3.3.0+113432_960e9c294b_ubuntu_20_04-cp38-cp38-linux
_x86_64.whl
```

Run a PopTorch model on IPU

```
cd examples/tutorials/simple_applications/pytorch/mnist/

pip install -r requirements.txt

python mnist_poptorch.py
```

- Deactivate the virtual environment after the model finishes running.

```
deactivate
```

Monitor IPU Usage - *gc-monitor*

- 4 partitions
- 16 IPUs
- Processes
- IPU used
- Temperature
- Power

```
watch -n 2 gc-monitor
```

```
Every 2.0s: gc-monitor                                             poplar2: Thu Jul  6 10:55:55 2023

+--------------+-------------------------------------------------------------------------------+
|  gc-monitor  |             Partition: p17 [active] has 16 reconfigurable IPUs               |
+-------------+-------------------+--------+--------------+----------+-------+----+-----+-------+
|   IPU-M     |      Serial       |IPU-M SW|Server version|  ICU FW  | Type  | ID | IPU# |Routing|
+-------------+-------------------+--------+--------------+----------+-------+----+-----+-------+
|  10.5.5.1   | 0019.0002.8222521 | 2.6.0  |    1.11.0    |  2.5.9   | M2000 | 0  |  3  |  DNC  |
|  10.5.5.1   | 0019.0002.8222521 | 2.6.0  |    1.11.0    |  2.5.9   | M2000 | 1  |  2  |  DNC  |
|  10.5.5.1   | 0019.0001.8222521 | 2.6.0  |    1.11.0    |  2.5.9   | M2000 | 2  |  1  |  DNC  |
|  10.5.5.1   | 0019.0001.8222521 | 2.6.0  |    1.11.0    |  2.5.9   | M2000 | 3  |  0  |  DNC  |
+-------------+-------------------+--------+--------------+----------+-------+----+-----+-------+
|  10.5.5.2   | 0021.0002.8222521 | 2.6.0  |    1.11.0    |  2.5.9   | M2000 | 4  |  3  |  DNC  |
|  10.5.5.2   | 0021.0002.8222521 | 2.6.0  |    1.11.0    |  2.5.9   | M2000 | 5  |  2  |  DNC  |
|  10.5.5.2   | 0021.0001.8222521 | 2.6.0  |    1.11.0    |  2.5.9   | M2000 | 6  |  1  |  DNC  |
|  10.5.5.2   | 0021.0001.8222521 | 2.6.0  |    1.11.0    |  2.5.9   | M2000 | 7  |  0  |  DNC  |
+-------------+-------------------+--------+--------------+----------+-------+----+-----+-------+
|  10.5.5.3   | 0013.0002.8222521 | 2.6.0  |    1.11.0    |  2.5.9   | M2000 | 8  |  3  |  DNC  |
|  10.5.5.3   | 0013.0002.8222521 | 2.6.0  |    1.11.0    |  2.5.9   | M2000 | 9  |  2  |  DNC  |
|  10.5.5.3   | 0013.0001.8222521 | 2.6.0  |    1.11.0    |  2.5.9   | M2000 | 10 |  1  |  DNC  |
|  10.5.5.3   | 0013.0001.8222521 | 2.6.0  |    1.11.0    |  2.5.9   | M2000 | 11 |  0  |  DNC  |
+-------------+-------------------+--------+--------------+----------+-------+----+-----+-------+
|  10.5.5.4   | 0016.0002.8222521 | 2.6.0  |    1.11.0    |  2.5.9   | M2000 | 12 |  3  |  DNC  |
|  10.5.5.4   | 0016.0002.8222521 | 2.6.0  |    1.11.0    |  2.5.9   | M2000 | 13 |  2  |  DNC  |
|  10.5.5.4   | 0016.0001.8222521 | 2.6.0  |    1.11.0    |  2.5.9   | M2000 | 14 |  1  |  DNC  |
|  10.5.5.4   | 0016.0001.8222521 | 2.6.0  |    1.11.0    |  2.5.9   | M2000 | 15 |  0  |  DNC  |
+-------------+-------------------+--------+--------------+----------+-------+----+-----+-------+

+------------------------------------------------------+-----------------------+----------------+
|         Attached processes in partition p17          |          IPU          |     Board      |
+--------+---------------------+--------+------------+----+---------+---------+--------+-------+
|  PID   |       Command       |  Time  |    User    | ID |  Clock  |  Temp   |  Temp  | Power |
+--------+---------------------+--------+------------+----+---------+---------+--------+-------+
| 907530 |       python        |  17s   | u.zh108696 | 0  | 1500MHz | 23.5 C  | 22.0 C | 90.8 W |
+--------+---------------------+--------+------------+----+---------+---------+--------+-------+
```

Hands-On Session 1

- Please access ACES and poplar2 now.
- Copy the tutorial materials to your scratch directory.
- Run the TensorFlow and PyTorch (PopTorch) example models on IPU

Section III. Porting TensorFlow Code to IPU

1. Import the TensorFlow IPU module

Add the following import statement to the beginning of your script:

```
from tensorflow.python import ipu
```

2. Preparing the dataset

- Make sure the sizes of the datasets are divisible by the batch size

```
def make_divisible(number, divisor):
    return number - number % divisor
```

- Adjust dataset lengths

```
(x_train, y_train), (x_test, y_test) = load_data()
train_data_len = x_train.shape[0]
train_data_len = make_divisible(train_data_len, batch_size)
x_train, y_train = x_train[:train_data_len], y_train[:train_data_len]
test_data_len = x_test.shape[0]
test_data_len = make_divisible(test_data_len, batch_size)
x_test, y_test = x_test[:test_data_len], y_test[:test_data_len]
```

3. Add IPU configuration

To use the IPU, you must create an IPU session configuration:

```
ipu_config = ipu.config.IPUConfig()
ipu_config.auto_select_ipus = 1
ipu_config.configure_ipu_system()
```

A full list of configuration options is available in the API documentation.

4. Specify IPU strategy

```
strategy = ipu.ipu_strategy.IPUStrategy()
```

The `tf.distribute.Strategy` is an API to distribute training and inference across multiple devices. `IPUStrategy` is a subclass which targets a system with one or more IPUs attached.

5. Wrap the model within the IPU strategy scope

- Creating variables and Keras models within the scope of the `IPUStrategy` object will ensure that they are placed on the IPU.
- To do this, we create a `strategy.scope()` context manager and move all the model code inside it.

Hands-on Session 2

- Activate the TF virtual environment

```
cd /localdata/$USER/ipu_labs
source venv_tf2/bin/activate
```

- Change directory to Keras

```
cd IPU-Training/Keras
```

- Complete the **#Todo**s in the mnist-ipu-todo.py file.
- Run it in the **venv_tf2** virtual environment.

```
python mnist-ipu-todo.py
```

- After finishing the job, you can deactivate the virtual environment

```
deactivate
```

Section IV. Porting PyTorch Code to IPU

PopTorch

- PopTorch is a set of extensions for PyTorch released by Graphcore to enable PyTorch models to run on Graphcore's IPU hardware.

- PopTorch will use PopART to parallelise the model over the given number of IPUs. Additional parallelism can be expressed via a replication factor, which enables you to data-parallelise the model over more IPUs.

Training a model on IPU

- Import the packages

```
import torch
import poptorch
import torchvision
import torch.nn as nn
import matplotlib.pyplot as plt
from tqdm import tqdm
from sklearn.metrics import accuracy_score
```

Load the data

PopTorch offers an extension of `torch.utils.data.DataLoader` class with its `poptorch.DataLoader` class, specialized for the way the underlying PopART framework handles batching of data.

Build the model

```
class ClassificationModel(nn.Module):
    def __init__(self):
        super().__init__()
        self.conv1 = nn.Conv2d(1, 5, 3)
        self.pool = nn.MaxPool2d(2, 2)
        self.conv2 = nn.Conv2d(5, 12, 5)
        self.norm = nn.GroupNorm(3, 12)
        self.fc1 = nn.Linear(972, 100)
        self.relu = nn.ReLU()
        self.fc2 = nn.Linear(100, 10)
        self.log_softmax = nn.LogSoftmax(dim=1)
        self.loss = nn.NLLLoss()
```

```
def forward(self, x, labels=None):
        x = self.pool(self.relu(self.conv1(x)))
        x = self.norm(self.relu(self.conv2(x)))
        x = torch.flatten(x, start_dim=1)
        x = self.relu(self.fc1(x))
        x = self.log_softmax(self.fc2(x))
        # The model is responsible for the calculation of the loss when using an IPU. We do it this way:
        if self.training:
            return x, self.loss(x, labels)
        return x


model = ClassificationModel()
model.train()
```

Prepare training for IPUs

The compilation and execution on the IPU can be controlled using `poptorch.Options`. These options are used by PopTorch's wrappers such as `poptorch.DataLoader` and `poptorch.trainingModel`.

```
opts = poptorch.Options()
train_dataloader = poptorch.DataLoader(
    opts, train_dataset, batch_size=16, shuffle=True, num_workers=20
)
```

Train the model

```
optimizer = poptorch.optim.SGD(model.parameters(), lr=0.001, momentum=0.9)

poptorch_model = poptorch.trainingModel(model, options=opts,
optimizer=optimizer)

epochs = 30
for epoch in tqdm(range(epochs), desc="epochs"):
    total_loss = 0.0
    for data, labels in tqdm(train_dataloader, desc="batches", leave=False):
        output, loss = poptorch_model(data, labels)
        total_loss += loss

poptorch_model.detachFromDevice()

torch.save(model.state_dict(), "classifier.pth")
```

Evaluate the model

```
model = model.eval()

poptorch_model_inf = poptorch.inferenceModel(model, options=opts)

test_dataloader = poptorch.DataLoader(opts, test_dataset, batch_size=32,
     num_workers=10)

predictions, labels = [], []
for data, label in test_dataloader:
    predictions += poptorch_model_inf(data).data.max(dim=1).indices
    labels += label

poptorch_model_inf.detachFromDevice()

print(f"Eval accuracy: {100 * accuracy_score(labels, predictions):.2f}%")
```

Hands-on Session 3

- Activate the Poptorch virtual environment

```
cd /localdata/$USER/ipu_labs
source poptorch_test/bin/activate
```

- Change directory to PyTorch

```
cd IPU-Training/PyTorch
```

- Complete the **#Todos** in the `fashion-mnist-pytorch-ipu-todo.py`file.
- Run it in the **poptorch_test** virtual environment.

```
pip install -r requirements.txt
python fashion-mnist-pytorch-ipu-todo.py
```

- After finishing the job, you can deactivate the virtual environment

```
deactivate
```

Acknowledgements

This work was supported by

- the National Science Foundation (NSF), award numbers:
  - 2112356 - ACES - Accelerating Computing for Emerging Sciences
  - 1925764 - SWEETER - SouthWest Expertise in Expanding, Training, Education and Research
  - 2019129 - FASTER - Fostering Accelerated Scientific Transformations, Education, and Research
- Staff and students at Texas A&M High-Performance Research Computing.
- ACCESS CCEP pilot program, Tier-II

References

- https://www.graphcore.ai
- https://github.com/graphcore/examples/tree/v3.2.0/tutorials/tutorials/tensorflow2/keras
- https://github.com/graphcore/examples/tree/v3.2.0/tutorials/tutorials/pytorch/basics
- https://hprc.tamu.edu/wiki/Main_Page
- Abhinand S. Nasari, Richard Lawrence, Zhenhua He, Hieu Le, Mario Michael Krell, Alex Tsyplikhin, Mahidhar Tateneni, Tim Cockerill, Lisa M. Perez, Dhruva K. Chakravorty, and Honggao Liu. (2022). Benchmarking the Performance of Accelerators on National Cyberinfrastructure Resources for Artificial Intelligence/Machine Learning Workloads. In Practice and Experience in Advanced Research Computing, pp. 1-9. 2022. https://dl.acm.org/doi/10.1145/3491418.3530772

High Performance
Research Computing
DIVISION OF RESEARCH

https://hprc.tamu.edu

HPRC Helpdesk:

help@hprc.tamu.edu
Phone: 979-845-0219

Help us help you. Please include details in your request for support, such as, Cluster (ACES), UserID (ACCESS ID), Job information (Job id(s), Location of your jobfile, input/output files, Application, Module(s) loaded, Error messages, etc), and Steps you have taken, so we can reproduce the problem.