

HIGH PERFORMANCE RESEARCH COMPUTING

ACES: Using Graphcore Intelligence Processing Unit

10/01/2024

Zhenhua He



High Performance
Research Computing
DIVISION OF RESEARCH



IPU Tutorial Outline

Section I. Intro to IPU

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.

Q&A
(10 mins/sec)

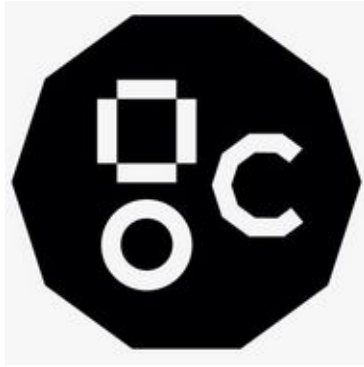
Section IV. Porting PyTorch code to IPU

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



NSF ACES

Accelerating Computing for Emerging Sciences

Our Mission:

- 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 Accelerators

Component	Quantity	Description
Graphcore IPU	32	16 Colossus GC200 IPU, 16 Bow IPU. Each IPU group hosted with a CPU server as a POD16 on a 100 GbE RoCE fabric
<i>FPGAs:</i>		
Intel PAC D5005	2	Accelerator with Intel Stratix 10 GX FPGA and 32 GB DDR4
BittWare IA-840F	3	Accelerator with Agilex AGF027 FPGA and 64 GB of DDR4
NextSilicon Coprocessor	2	Reconfigurable accelerator with an optimizer continuously evaluating application behavior.
NEC Vector Engine	8	Vector computing card (8 cores and HBM2 memory)
Intel Optane SSD	48	18 TB of SSDs addressable as memory w/ MemVerge Memory Machine.
<i>NVIDIA GPUs:</i>		
H100	30	For HPC, DL Training, AI Inference
A30	4	For AI Inference and Mainstream Compute
Intel PVC GPUs	120	Intel GPUs for HPC, DL Training, AI Inference

Refer to our Knowledge Base for more:

<https://hprc.tamu.edu/kb/User-Guides/ACES/Hardware/>

BOW IPU PROCESSOR

Deep Trench Capacitor

Efficient power delivery
Enables increase in operational performance

Wafer-On-Wafer

Advanced silicon 3D
stacking technology
Closely coupled power
delivery die
Higher operating frequency
and enhanced overall
performance

IPU-Tiles™

1472 independent IPU-Tiles™ each with an
IPU-Core™ and In-Processor-Memory™

IPU-Core™

1472 independent IPU-Core™
8832 independent program threads
executing in parallel

In-Processor-Memory™

900MB In-Processor-Memory™ per IPU
65.4TB/s memory bandwidth per IPU

Solder Bumps

IPU-Links™

10x IPU-Links,
320GB/s chip to chip bandwidth

IPU-Exchange™

11 TB/s all to all IPU-Exchange™
Non-blocking, any communication pattern

PCIe

PCI Gen4 x16
64 GB/s bidirectional bandwidth to host

4 x Bow 3D Wafer-on-Wafer IPUs

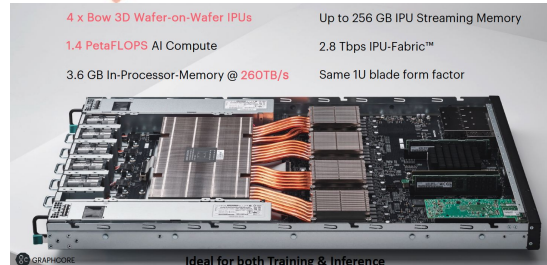
Up to 256 GB IPU Streaming Memory

1.4 PetaFLOPS AI Compute

2.8 Tbps IPU-Fabric™

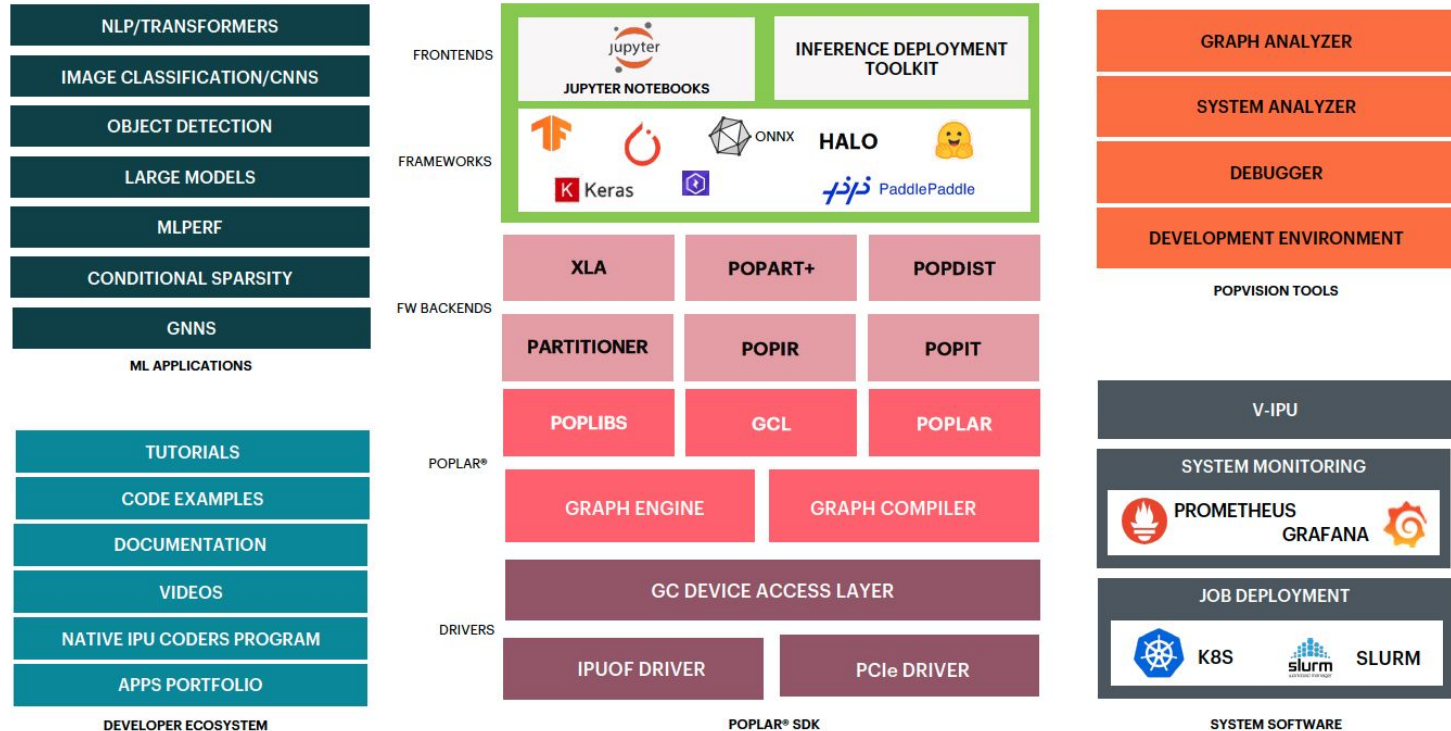
3.6 GB In-Processor-Memory @ 260TB/s

Same 1U blade form factor



Source: Graphcore

Graphcore Software Stack



Source:
Graphcore

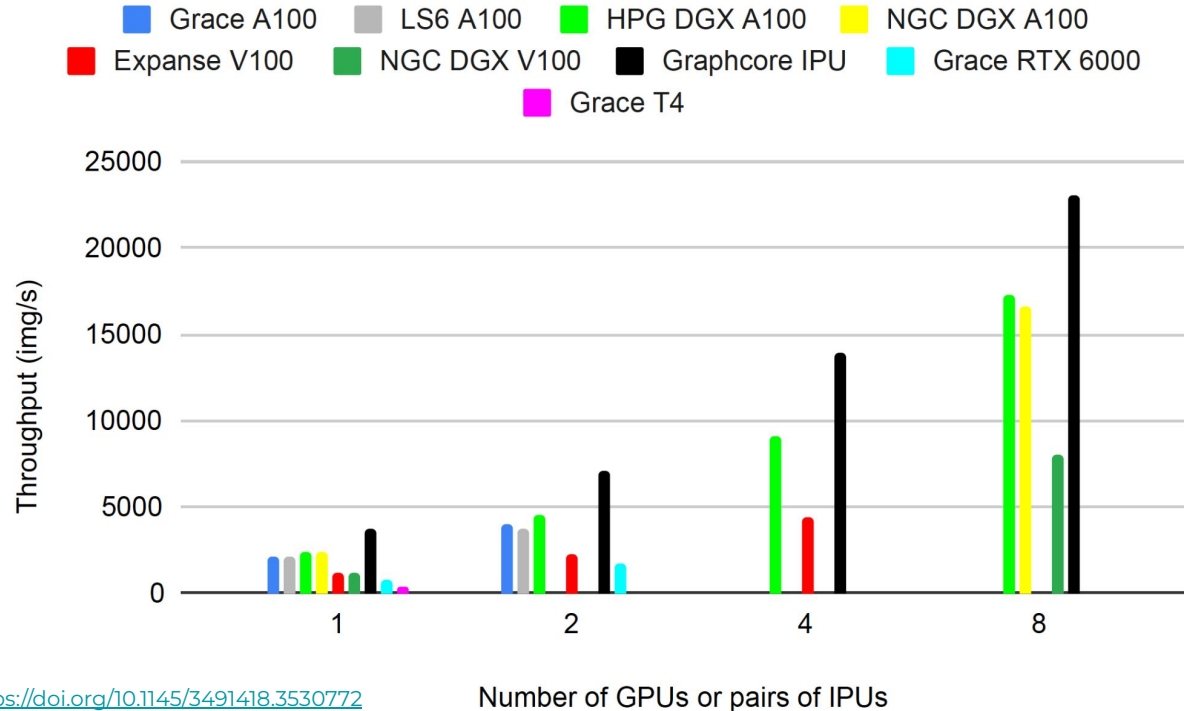
Models on Graphcore GitHub

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>

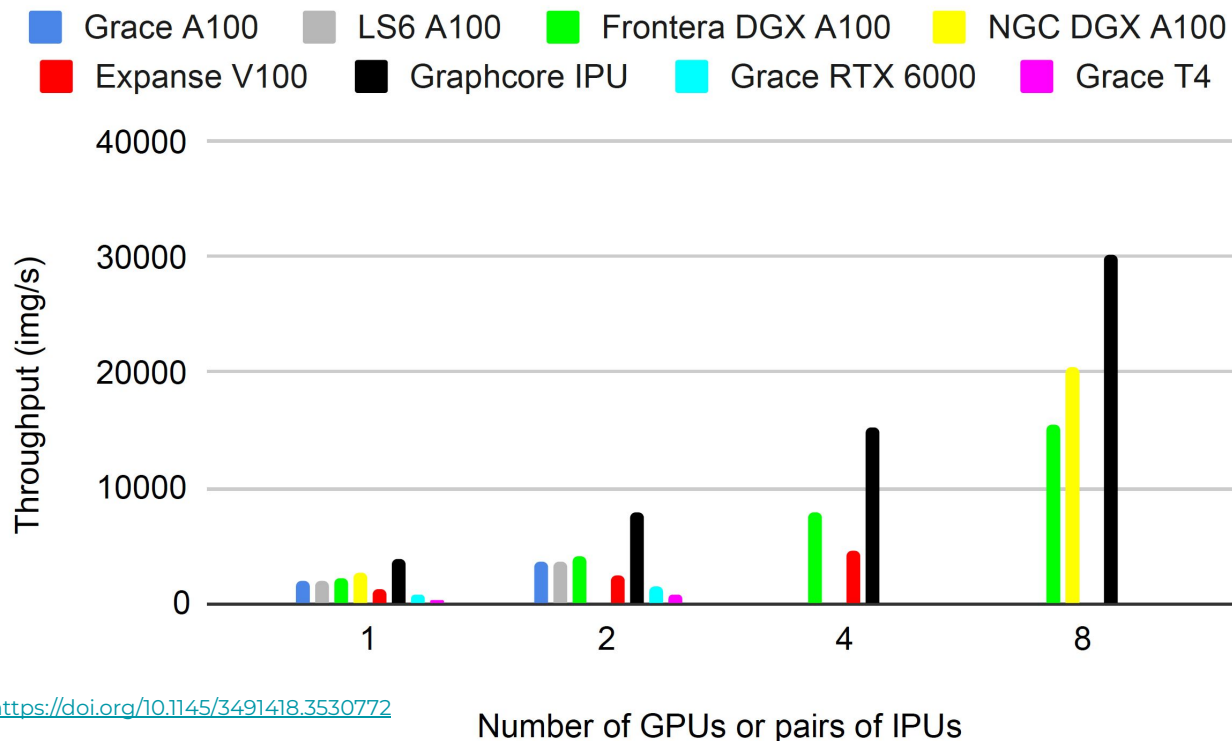
Choosing Between IPUs and GPUs

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>

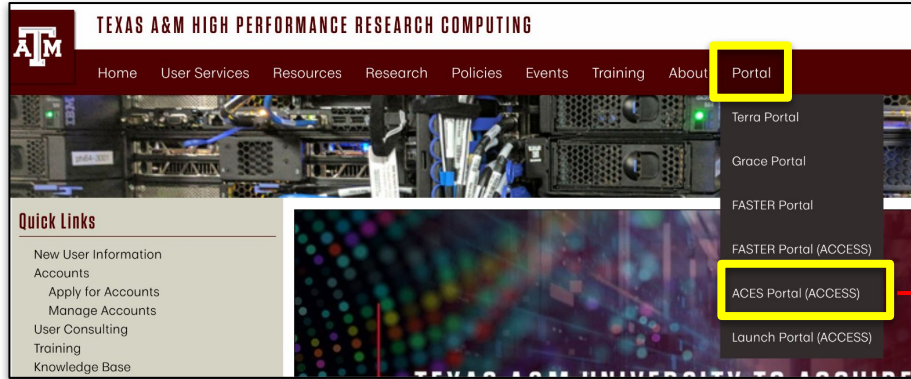
Take-home message

- 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



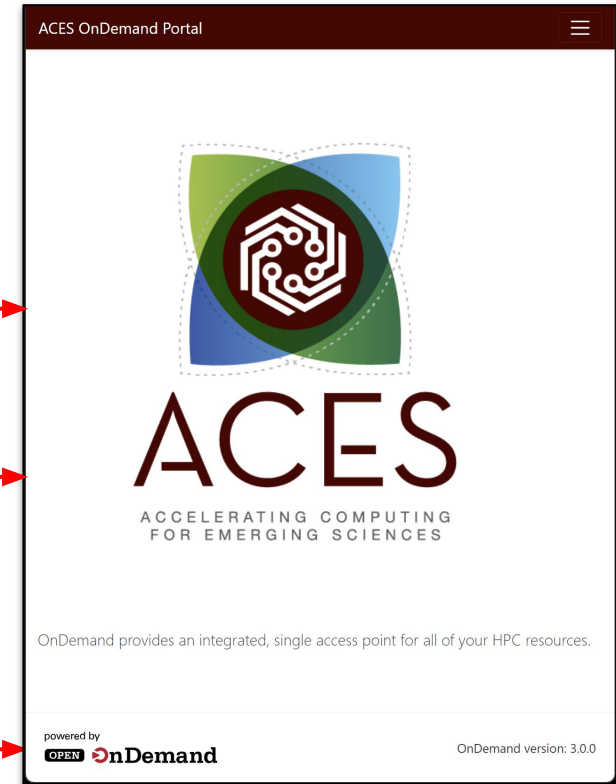
ACES Portal



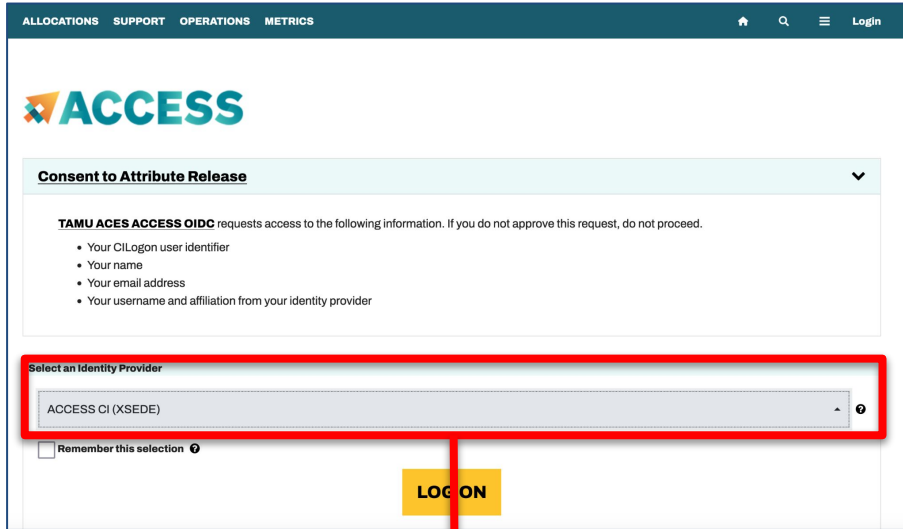
ACES Portal portal-aces.hprc.tamu.edu
is the web-based user interface for the ACES cluster

[HPRC Portal YouTube tutorials](#)

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

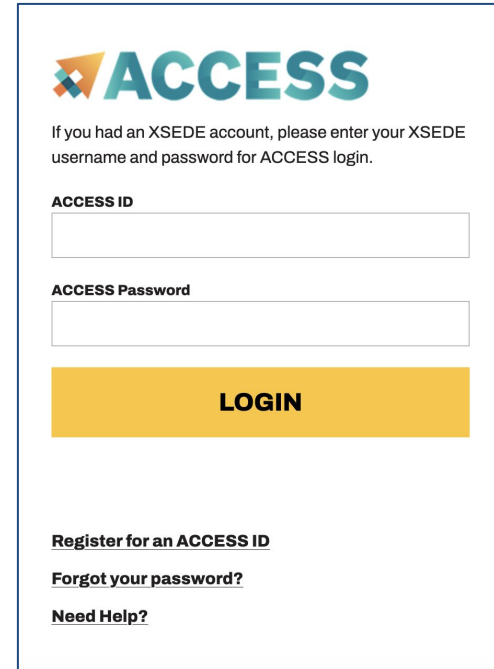


Accessing ACES via the Portal (ACCESS)



The screenshot shows the ACCESS portal interface. At the top is a navigation bar with links: ALLOCATIONS, SUPPORT, OPERATIONS, METRICS, and a Login link. Below the navigation bar is the ACCESS logo. A section titled "Consent to Attribute Release" is expanded, showing a list of information requested by TAMU ACES ACCESS OIDC: CILogon user identifier, name, email address, and username/affiliation. Below this is a "Select an Identity Provider" dropdown menu with "ACCESS CI (XSEDE)" selected. A red box highlights this dropdown. Below the dropdown is a "Remember this selection" checkbox. A yellow "LOG ON" button is at the bottom right of the form.

Select the Identity Provider appropriate for your account.



The screenshot shows the ACCESS portal login form. At the top is the ACCESS logo. Below it is a message: "If you had an XSEDE account, please enter your XSEDE username and password for ACCESS login." There are two input fields: "ACCESS ID" and "ACCESS Password". Below these fields is a large yellow "LOGIN" button. At the bottom of the form are three links: "Register for an ACCESS ID", "Forgot your password?", and "Need Help?".

Log-in using your ACCESS or institutional credentials.

Shell Access via the Portal



```
Host: login.aces Themes: Default
Warning: Permanently added 'login.aces,10.71.1.13' (ECDSA) to the list of known hosts.
*****
This computer system and the data herein are available only for authorized
purposes by authorized users. Use for any other purpose is prohibited and may
result in disciplinary actions or criminal prosecution against the user. Usage
may be subject to security testing and monitoring. There is no expectation of
privacy on this system except as otherwise provided by applicable privacy laws.
Refer to University SAP 29.01.02, MO.02 Acceptable Use for more information.
*****

Last login: Mon Feb 12 13:11:13 2024 from 10.71.1.6

=====
Texas A&M University High Performance Research Computing
=====

Website: https://hprc.tamu.edu
Consulting: help@hprc.tamu.edu (preferred) or (979) 845-0219
ACES Documentation: https://hprc.tamu.edu/kb/User-Guides/ACES
FASTER Documentation: https://hprc.tamu.edu/kb/User-Guides/FASTER
Grace Documentation: https://hprc.tamu.edu/kb/User-Guides/Grace
Terra Documentation: https://hprc.tamu.edu/kb/User-Guides/Terra
YouTube Channel: https://www.youtube.com/texasamhprc

***** IMPORTANT POLICY INFORMATION *****
* - Unauthorized use of HPRC resources is prohibited and subject to
* criminal prosecution.
* - Use of HPRC resources in violation of United States export control
* laws and regulations is prohibited. Current HPRC staff members are
* US citizens and legal residents.
* - Sharing HPRC account and password information is in violation of
* Texas State Law. Any shared accounts will be DISABLED.
* - Authorized users must also adhere to ALL policies at:
* https://hprc.tamu.edu/policies/
*****

**** ACES Partial Availability, February 12 ****

We are still troubleshooting issues for various compute nodes that were
reconfigured for PCIe fabric connectivity to the H100 and PVCs.

!! WARNING: THERE ARE ONLY NIGHTLY BACKUPS OF USER HOME DIRECTORIES. !!

Please restrict usage to 8 CORES across ALL login nodes.
Users found in violation of this policy will be SUSPENDED.

To see these messages again, run the moitd command.
Your current disk quotas are:
Disk Usage Limit File Usage Limit
/home/u.jw123527 169M 10.0G 499 10000
/scratch/user/u.jw123527 28.1G 1.0T 102472 250000
Type 'showquota' to view these quotas again.
[u.jw123527@aces-login3 ~]$
```

Training Materials

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

```
ssh poplar2
```

Change to your locadata directory:

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

Copy the example materials to your ipu_labs directory:

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

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

```
git clone https://github.com/happidencel/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/$USER/tmp  
export TORCH_HOME=/localdata/$USER/tmp/
```

Run a TensorFlow (TF) model on IPU



TensorFlow



Keras

TF Virtual Environment Setup

```
virtualenv -p python3 venv_tf2  
  
source venv_tf2/bin/activate  
  
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

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

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	

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
/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 IPUUs
- 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 IPUUs									
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 **#Todos** 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

- PyTorch: `torch.utils.data.DataLoader` class
- PopTorch extension: `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 IPU

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
```

Section V. Model Replication (bonus)



1. Set the number of IPUUs and replicas

```
num_ipus = num_replicas = 2
```

2. Set the steps per execution and adjust data length

```
(x_train, y_train), (x_test, y_test) = load_data()

train_data_len = x_train.shape[0]
train_steps_per_execution = train_data_len // (batch_size * num_replicas)
train_data_len = make_divisible(train_data_len, batch_size *
num_replicas)
x_train, y_train = x_train[:train_data_len], y_train[:train_data_len]

test_data_len = x_test.shape[0]
test_steps_per_execution = test_data_len // (batch_size * num_replicas)
test_data_len = make_divisible(test_data_len, batch_size * num_replicas)
x_test, y_test = x_test[:test_data_len], y_test[:test_data_len]
```

3. Update the configuration step

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

```
ipu_config = ipu.config.IPUConfig()
ipu_config.device_connection.type =
ipu.config.DeviceConnectionType.ON_DEMAND
ipu_config.auto_select_ipus = num_ipus
ipu_config.configure_ipu_system()
```

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

4. Add steps per execution to model compile

```
model.compile(  
    "sgd",  
    "categorical_crossentropy",  
    metrics=["accuracy"],  
    steps_per_execution=train_steps_per_execution,  
)
```

```
model.compile(  
    "sgd",  
    "categorical_crossentropy",  
    metrics=["accuracy"],  
    steps_per_execution=test_steps_per_execution,  
)
```

Hands-on Session

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

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

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

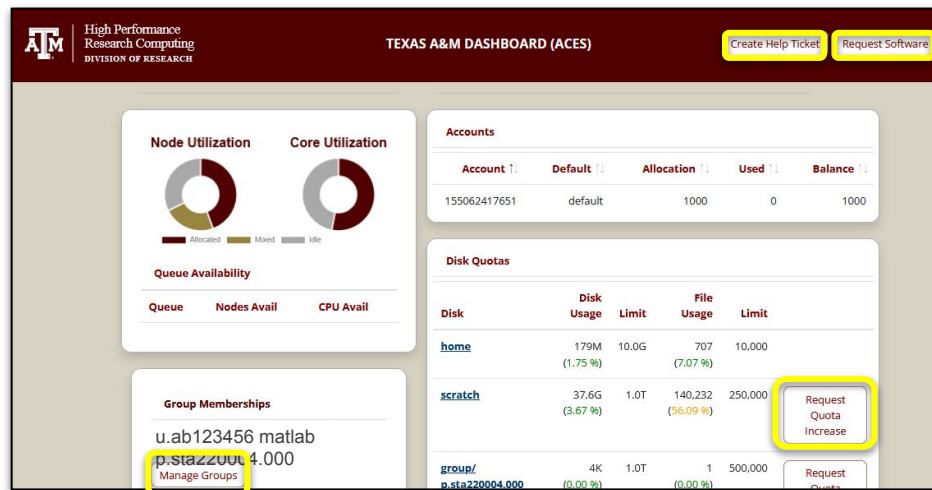
```
deactivate
```

Need Help?

First check the FAQ: <https://hprc.tamu.edu/kb/FAQ/Accounts>

- ACES user Guide: <https://hprc.tamu.edu/kb/User-Guides/ACES>
- Email your questions to help@hprc.tamu.edu

Remember the
Dashboard!



Need Help?

Help us help you -- tell us:

- Which cluster
- Username
- Job id(s) if any
- Location of your jobfile, input/output files
- Application used if any
- Module(s) loaded if any
- Error messages
- Steps you have taken, so we can reproduce the problem

Acknowledgements

This work was supported by

- the National Science Foundation (NSF), award numbers:
 - 2112356 - ACES - Accelerating Computing for Emerging Sciences
 - 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.



High Performance
Research Computing
DIVISION OF RESEARCH

Thank you
Questions?

Give us feedback on the class with this survey:
https://u.tamu.edu/hprc_shortcourse_survey



HPRC Survey