GRAPHCORE

Poplar Quick Start

Version latest

Graphcore Ltd

Feb 27, 2024

CONTENTS

This Quick Start guide describes how to run a simple application on the IPU (from the Graphcore examples repository) that has been written using the Poplar Graph Programming Framework.

CHAPTER

ONE

OVERVIEW

If you are already familiar with using IPUs and Poplar then Section 2, Quick start for experts provides a summary of the steps, focussing on the essential commands. If you need more detail, this can be found in Section 3, Quick start for beginners.

We suggest watching the Fundamentals of the IPU and Poplar video, which introduces the IPU architecture and programming model.

You can also read the IPU Programmer's Guide and Switching from GPUs to IPUs for Machine Learning Models for more details on these topics.

You can expect the setup steps to take no more than 15 minutes.

This document is applicable if you are accessing IPUs on the systems listed in Section 1.1, IPU systems.

Note: There are three ways you can run Poplar applications:

1. Directly on the system using the setup described in this document.
2. In a Poplar Docker container that has already been setup. Refer to Using IPUs from Docker for more information.

1.1 IPU systems

Before following the instructions in this Quick Start guide, you must be able to log into a system with access to IPUs. Details are given in the getting started guide for your system:

- Gcore Cloud: Getting Started with IPUs
- Getting Started with Graphcloud
- Getting Started with Bow Pod and IPU-POD Systems

CHAPTER

TWO

QUICK START FOR EXPERTS

Complete any necessary setup to use your IPU system (see Section 1.1, IPU systems) before the following steps.

2.1 Enable the Poplar SDK

```
$ source [path-to-sdk]/enable
$ popc --version
```

where [path-to-sdk] is the path to the Poplar SDK.

2.2 Clone the Graphcore examples

You may need to clone the Graphcore examples GitHub repository in order to run the example application. For convenience we set an environment variable to the tutorials directory.

To clone the examples repository for the latest version of the Poplar SDK:

```
$ cd ~/[base dir]
$ git clone https://github.com/graphcore/examples.git
$ cd examples/tutorials
$ export POPLAR_TUTORIALS_DIR=$(pwd)
```

where [base_dir] is a location of your choice.

Note: If you are using a version of the Poplar SDK prior to version 3.2, then refer to Section 6, Install examples and tutorials for older Poplar SDK versions for how to install examples and tutorials.

2.3 Run the application

```
$ cd ${POPLAR_TUTORIALS_DIR?}/simple_applications/poplar/mnist/
$ ./get_data.sh
$ make
$ ./regression-demo -IPU 10 50
```

Refer to Section 3.4, Run the application for details of the command line options for the application.

The examples repo contains other tutorials and applications you can try. See Section 4, Next steps for more information.

CHAPTER

THREE

QUICK START FOR BEGINNERS

This section provides more detail on the steps described in the *Quick start for experts* section.

Complete any necessary setup to use your IPU system (see Section 1.1, IPU systems) before the following steps.

3.1 Enable the Poplar SDK

Note: It is best if you use the latest version of the Poplar SDK.

On some systems you must explicitly enable the Poplar SDK before you can use PyTorch or TensorFlow for the IPU, or the Poplar Graph Programming Framework. On other systems, the SDK is enabled as part of the login process.

Table 3.1 defines whether you have to explicitly enable the SDK and where to find it.

Table 3.1: Systems that need the Poplar SDK to be enabled and the SDK location

System	Enable SDK?	SDK location
Pod system	Yes	The SDK is in the directory where you extracted the SDK tarball.
Graphcloud	Yes	/opt/gc/poplar_sdk-ubuntu_18_04-[poplar_ver]+[build] where [poplar_ver] is the software version number of the Poplar SDK and [build] is the build information.
Gcore Cloud	No	The SDK has been enabled as part of the login process.

To enable the Poplar SDK:

SDK Versions 2.6 and later

SDK Versions earlier than 2.6

For SDK versions 2.6 and later, there is a single enable script that determines whether you are using Bash or Zsh and runs the appropriate scripts to enable both Poplar and PopTorch/PopART.

Source the single script as follows:

```
$ source [path_to_SDK]/enable
```

where [path_to_SDK] is the location of the Poplar SDK on your system.

For SDK versions earlier than 2.6, there are only Bash scripts available and you have to source the Poplar and PopART scripts separately.

Note: You only have to source the PopART enable script if you are using PopTorch or PopART.

Source the scripts as follows:

```
$ source [path_to_SDK]/poplar-ubuntu_[os_ver]-[poplar_ver]+[build]/enable.sh
$ source [path_to_SDK]/popart-ubuntu_[os_ver]-[poplar_ver]+[build]/enable.sh
```

where [path_to_SDK] is the location of the Poplar SDK on your system. [os_ver] is the version of Ubuntu on your system, [poplar_ver] is the software version number of the Poplar SDK and [build] is the build information.

Note: You must source the Poplar enable script for each new shell. You can add this source command to your .bashrc (or .zshrc for SDK versions later than 2.6) to do this on a more permanent basis.

If you attempt to run any Poplar software without having first sourced this script, you will get an error from the C++ compiler similar to the following (the exact message will depend on your code):

```
fatal error: 'poplar/Engine.hpp' file not found
```

If you try to source the script after it has already been sourced, then you will get an error similar to:

```
ERROR: A Poplar SDK has already been enabled.
Path of enabled Poplar SDK: /opt/gc/poplar_sdk-ubuntu_20_04-3.2.0-7cd8ade3cd/poplar-ubuntu_20_04-3.2.0-7cd8ade3cd
If this is not wanted then please start a new shell.
```

You can verify that Poplar has been successfully set up by running:

```
$ popc --version
```

This will display the version of the installed software.

3.2 Clone the Graphcore examples

You may need to clone the Graphcore examples repository on some systems as detailed in Table 3.2.

If you don't need to clone the examples repository, then go straight to Section 3.3, Define environment variable.

Table 3.2: Systems that need the Graphcore tutorials and examples repositories to be cloned

System	Clone repos?	Comment
Pod system	Yes	You can clone the tutorials and examples repos in any location.
Graphcloud	Yes	You can clone the tutorials and examples repos in any location.
Gcore Cloud	No	The tutorials and examples have already been cloned in ~/graphcore/tutorials and ~/graphcore/examples respectively.

You can clone the examples repository into a location of your choice.

To clone the examples repository for the latest version of the Poplar SDK:

```
$ cd ~/[base_dir]
$ git clone https://github.com/graphcore/examples.git
```

where [base_dir] is a location of your choice. This will install the contents of the examples repository under ~/[base_dir]/examples. The tutorials are in ~/[base_dir]/examples/tutorials.

Note: If you are using a version of the Poplar SDK prior to version 3.2, then refer to Section 6, Install examples and tutorials for older Poplar SDK versions for how to install examples and tutorials.

3.3 Define environment variable

In order to simplify running the tutorials, we define the environment variable POPLAR_TUTORIALS_DIR that points to the location of the cloned tutorials.

Pod systems

Graphcloud

Gcore Cloud

```
$ export POPLAR_TUTORIALS_DIR=~/[base_dir]/examples/tutorials
```

[base_dir] is the location where you installed the Graphcore tutorials.

```
$ export POPLAR_TUTORIALS_DIR=~/[base_dir]/examples/tutorials
```

[base_dir] is the location where you installed the Graphcore tutorials.

```
$ export POPLAR_TUTORIALS_DIR=~/graphcore/tutorials
```

3.4 Run the application

This section describes how to run a simple application from the Graphcore tutorials repository, the MNIST example, written in Poplar.

1. Download MNIST data

   ```
   $ cd $POPLAR_TUTORIALS_DIR/simple_applications/poplar/mnist/
   $ ./get_data.sh
   ```

2. Build the code with the Makefile provided:

   ```
   $ make
   ```

3. Train and test the model

   You run the application with the command:

   ```
   $ ./regression-demo [-IPU] [number of epochs] [proportion of images to use]
   ```

   where:

   - -IPU indicates that an IPU must be used, otherwise the IPU Model is used. The IPU Model is a simulation of the behaviour of the IPU hardware. It does not completely implement every aspect of a real IPU.
   - number of epochs indicates the number of epochs for training.
   - proportion of images to use indicates what percentage of the images must be used to run the model.

   The command to run the model to use an IPU with 10 epochs and 50% of the image dataset is:

```
$ ./regression-demo -IPU 10 50
```

4. If the code has run successfully, you should see an output similar to that in Listing 3.1.

Listing 3.1: Example of output for Poplar application (not the complete output).

```
Using the IPU
Trying to attach to IPU
Attached to IPU 0
Target:
  Number of IPUs:         1
  Tiles per IPU:          1,472
  Total Tiles:            1,472
  Memory Per-Tile:        624.0 kB
  Total Memory:           897.0 MB
  Clock Speed (approx):   1,330.0 MHz
  Number of Replicas:     1
  IPUs per Replica:       1
  Tiles per Replica:      1,472
  Memory per Replica:     897.0 MB

Graph:
  Number of vertices:           6,262
  Number of edges:              21,207
  Number of variables:          47,402
  Number of compute sets:           22

Memory Usage:
  Total for all IPUs:
    Including Gaps:         43,789,756 B
    Excluding Gaps:
      By Memory Region:
        Non-interleaved:     3,222,384 B
        Interleaved:                 0 B
        Overflowed:                  0 B
      Total:                3,222,384 B
      By Data Type:
        Not Overlapped
            Variables:                          62,924 B
            Program and Sync IDs:                  16 B
            ...
          Total:                             3,148,696 B
        Overlapped
            Variables:                          94,860 B
            Program and Sync IDs:                5,244 B
            ...
          Total:                               181,672 B
          Total After Overlapping:              73,688 B
      Vertex Data (192,478 B):
        By Category:
          Internal vertex state:        92,478 B
          Edge pointers:                83,400 B
          Copy pointers:                 3,160 B
          Padding:                           0 B
          Descriptors:                  13,440 B
        By Type:
          poplin::OuterProduct<float>                                   109,760 B
          ...
      Vertex Code (1,580,336 B):
        By Type:
          poplin::OuterProduct<float> (asm)                                 455,880 B
          ...

  By Tile (Excluding Gaps):
    Range (KB) Histogram (Excluding Gaps)             Count (tiles)
      0 - 1 *********                                  161
      1 - 2 ****************************************   751
      2 - 3 ****                                       69
      3 - 4 ********************                       362
      4 - 5 *******                                    119
```

(continues on next page)

(continued from previous page)

```
      5 - 6                                   0
      6 - 7 *                                 9
      7 - 8                                   0
      8 - 9 *                                 1

   Maximum (Including Gaps): 49,232 (48.1 K) on tile 11
   Maximum (Excluding Gaps): 8,872 (8.7 K) on tile 0
   0 tile(s) out of memory

Epoch 1 (6%), accuracy 9%
Epoch 1 (12%), accuracy 14%
Epoch 1 (18%), accuracy 20%
Epoch 1 (24%), accuracy 28%
Epoch 1 (30%), accuracy 31%
Epoch 1 (36%), accuracy 37%
```

You have run an application that demonstrates how to use the IPU to train and test a simple model on the MNIST dataset using the Poplar Graph Programming Framework.

3.5 Try out other applications

The examples repo contains other tutorials and applications you can try. See Section 4, Next steps for more information.

CHAPTER

FOUR

NEXT STEPS

This page lists sources of information to help you learn about programming for, and running code on, the IPU.

4.1 Documentation

All documentation is on the Graphcore documentation portal. There you can find user guides for hardware and software, API references and technical notes.

The IPU Programmer's Guide provides an introduction to the IPU architecture, programming model and tools available.

Switching from GPUs to IPUs for Machine Learning Models provides a high-level overview of the programming changes required when switching from GPUs to IPUs.

Technical notes cover a range of topics about implementing and optimising models for the IPU.

4.2 Running applications in Docker

You can run Poplar applications in Docker on a Linux machine using one or more physical IPU devices.

Refer to Using IPUs from Docker for more information.

4.3 Tutorials, examples and applications

You can browse the Graphcore Model Garden to find applications that run on IPUs. You can filter by category, model type and framework.

There are more applications in the examples repository on GitHub.

The latest tutorials and simple code examples are also in the examples repository:

- Tutorials
- Simple applications
- Feature examples

Note: If you are using a version of the Poplar SDK prior to version 3.2, then refer to Section 6, Install examples and tutorials for older Poplar SDK versions for how to install examples and tutorials.

4.4 Other support

- When looking for answers or asking questions on StackOverflow, use the tag "ipu".
- You can request support on the Graphcore Support portal by clicking on the **Submit a ticket** link.
- For general help, discussions and announcements, please join our Graphcore Slack Community.

CHAPTER

FIVE

TRADEMARKS & COPYRIGHT

CHAPTER

SIX

INSTALL EXAMPLES AND TUTORIALS FOR OLDER POPLAR SDK VERSIONS

This section describes how to install the Graphcore examples and tutorials if you are using a version of the Poplar SDK prior to version 3.2.

6.1 Clone the Graphcore tutorials

If you are using a version of the Poplar SDK prior to version 3.2, you will need to use the old GitHub tutorials repository. You will also have to checkout a branch of the tutorials repository corresponding to the version of the Poplar SDK you are using.

You can clone the tutorials repository into a location of your choice.

```
$ cd ~/[base_dir]
$ git clone https://github.com/graphcore/tutorials.git
$ cd tutorials
$ git checkout sdk-release-[poplar-ver]
```

where [base_dir] is a location of your choice and [poplar-ver] is the version of the Poplar SDK that you are using. This will install the contents of the tutorials under ~/[base_dir]/tutorials. The tutorials are in ~/[base_dir]/tutorials.

For example to checkout the tutorials repo for Poplar SDK version 3.1:

```
$ cd ~/[base_dir]
$ git clone https://github.com/graphcore/tutorials.git
$ cd tutorials
$ git checkout sdk-release-3.1
```

6.2 Clone the Graphcore examples

If you are using a version of the Poplar SDK prior to version 3.2, you will need to checkout a specific tagged version of the examples repository corresponding to the version of the Poplar SDK you are using.

You can clone the examples repository into a location of your choice.

```
$ cd ~/[base_dir]
$ git clone https://github.com/graphcore/examples.git
$ cd examples
$ git checkout tags/[tag_name]
```

where [base_dir] is a location of your choice and [tag_name] is the name of the tagged commit corresponding to the version of the Poplar SDK that you are using. The tags are of the form v3.2.0 for the code compatible with each SDK release.

This will install the contents of the examples repository under ~/[base_dir]/examples.

There is a list of tagged versions.

For example, to checkout the tagged version corresponding to Poplar SDK 3.1:

```
$ cd ~/[base_dir]
$ git clone https://github.com/graphcore/examples.git
$ cd examples
$ git checkout tags/v3.1.0
```