GRAPHCORE

Poplar® SDK v2.0.0 Release notes

Version 2.0.0

Graphcore Ltd

Mar 22, 2021

CONTENTS

CHAPTER

ONE

LEGAL NOTICE

The information included in Release notes is only for use with or for Graphcore products.
Use of the information included in Release notes is at your own risk, and subject to the terms of the Graphcore end-user license agreement.
All Graphcore products including hardware and software, described in Release notes, are subject to continuous changes at any time, without notice.

Graphcore ® and Poplar ® are registered trademarks of Graphcore Ltd.

AI-Float™, Colossus™, Exchange Memory™, In-Processor-Memory™, IPU-Core™, IPU-Exchange™, IPU-Fabric™, IPU-Link™, IPU-M2000™, IPU-Machine™, IPU-POD™, IPU-Tile™, PopART™, PopLibs™, PopVision™, PopTorch™, Streaming Memory™ and Virtual-IPU™ are trademarks of Graphcore Ltd.
All other trademarks are the property of their respective owners.

CHAPTER

TWO

SCOPE OF THIS DOCUMENT

This document contains the Release notes for the Poplar SDK for Graphcore's IPU product family. The software deliverables covered by this document are the following:

Driver & Utilities Driver and associated utilities needed by the Graphcore IPU.

PopART The Poplar Advanced Run Time is a flexible ONNX-compatible runtime supporting both training & inference.

PopTorch The PopTorch library provides a set of extensions for PyTorch to enable it to run on the Graphcore IPU hardware.

Poplar A graph programming framework for the IPU.

PopDist/PopRun Poplar Distributed Configuration Library (PopDist) is a library for configuring and coordinating distributed execution of (large-scale) machine learning applications.

TensorFlow An implementation of the TensorFlow framework for the Graphcore IPU.

CHAPTER

THREE

PACKAGE CONTENTS

The downloaded unified Poplar SDK will contain the following packages:

3.1 Ubuntu 18.04

Package	Version
Driver & Utilities	1.0.50
PopART	2.0.0
PopTorch	2.0.0
Poplar	2.0.0
PopDist/PopRun	2.0.0
TensorFlow 1	Graphcore TensorFlow 2.0.0
TensorFlow 2	Graphcore TensorFlow 2.0.0

3.2 CentOS 7.6

Package	Version
Driver & Utilities	1.0.50
PopART	2.0.0
PopTorch	2.0.0
Poplar	2.0.0
PopDist/PopRun	2.0.0
TensorFlow 1	Graphcore TensorFlow 2.0.0
TensorFlow 2	Graphcore TensorFlow 2.0.0

Note: See *Appendix A* (page 27) for TensorFlow additional requirements.

CHAPTER

FOUR

PRODUCT SUPPORT AND COMPATIBILITY MATRIX

SUPPORTED

These products are actively worked on: they will receive new features, general updates and security updates.
Notice of deprecation will be sent in advance for supported products.

DEPRECATED

These products will only receive security updates.
These products are expected to work with the indicated products however correctness is not guaranteed.
It is advised **not** to upgrade to this software version, unless strictly necessary.
In the future, these products can move to a **Not Supported** state, without further notice.
Support level will reflect the deprecated status.

NOT SUPPORTED

These products are not expected to work with this release.
No support will be provided.

Important: Deprecated products can be moved to a **Not supported** status without further notice.

4.1 IPU-M2000 System Software compatibility matrix

IPUM Model	Version	Support level	Notes
IPU-M2000 300-0024	2.0.0	Supported	N/A

4.2 IPU PCIe Hardware Support level

Model	Revision	ICU Firmware version	Driver version	Support level	Notes
C2 300-0004	All revisions	1.4.14	1.0.50	Not supported	•

Note: Use Firmware revision in accordance with IPU revision.

Important: For Firmware revision, compatibility is only enforced for patch versions.

4.3 Driver Support level

OS	Support level	Supported Kernel Version	Notes
CentOS 7.4/7.5	Supported	3.10	CentOS LTS kernel.
CentOS 7.6	Supported	3.10	CentOS LTS kernel.
Microsoft Windows	Supported	Windows Server 2019	
Ubuntu 18.04	Supported	4.15	Ubuntu LTS kernel.

Warning:

It is strongly recommended to update the kernel module of the driver to the version included with this release.
This is to avoid incompatibilities with the non-kernel components of this SDK

4.4 SDK Support level

OS	Support level	Notes
Microsoft Windows	Not Supported	
CentOS 7.6	Supported	In some specific instances we encountered a less then optimal model compilation time. Investigations are ongoing to address the problem.
Ubuntu 18.04	Supported	

4.5 Supported toolchain

4.5.1 Ubuntu 18.04

Tool	Support level	Version
GCC/G++	Supported	7.2.0
libstdc++	Supported	6.0.24
libc	Supported	2.27
binutils	Supported	2.30

4.5.2 CentOS 7.6

Tool	Support level	Version
GCC/G++	Supported	7.3.1
libstdc++	Supported	6.0.24
libc	Supported	2.17
binutils	Supported	2.28

4.6 Supported tools

Tool	Support level	Version	
Python	Supported	3.6	
Boost library	Deprecated	1.70	Older versions of boost are no longer supported.

CHAPTER

FIVE

LIST OF CHANGES

The following sections will list changes in version , as well as older releases, for all products contained in the Poplar SDK.

There are three main sections, divided by argument:

Changelogs *Changelogs* (page 8) section lists important bug fixes and relevant functionality that has been added. Minor fixes or features will not be listed.

Known issues *Known Issues* (page 21) section will list all important issues known to date. This section will list issues that will impact Poplar functionality.

Compatibility changes *Compatibilities changes* (page 24) section will capture any change that needs to apply existing code, to remain compatible with this version of the SDK.

CHAPTER

SIX

CHANGELOGS

Product	Changelog
Driver & Utilities	*Changelog Driver & Utilities* (page 8)
PopART	*Changelog PopART* (page 11)
PopTorch	*Changelog PopTorch* (page 13)
Poplar	*Changelog Poplar* (page 14)
Poplar Libraries	*Changelog Poplar Libraries* (page 16)
PopDist/PopRun	*Changelog PopRun/PopDist* (page 17)
Libpva Library	*Changelog Libpva Library* (page 18)
TensorFlow	*Changelog TensorFlow* (page 18)

6.1 Driver & Utilities Changelog

6.1.1 Kernel Module

1.0.50

- T34412: Do not fail in `driver_load.sh` script when the driver is automatically loaded.
- T28718: Added on chip memory recording. Visible via `gc-monitor`.
- T29903: Implemented reset of IPU-M service tables.
- T34199: Fix PCIe driver mailbox compatibility with Poplar SDK 1.4.
- T33255: Driver now logs message when IPU clock has been throttled.
- T32751: Added external memory info when available via IPUoF.
- T32193: Improve resilience for ICU comms during ICU event notifications.
- T31153: Fixed ICU async message handling.
- T32313: Use Python 3 to generate PCIe driver defines.
- T32162: Fix race in starting and stopping IPU sync utilisation `kthread`.
- T31152: Added mechanism to clear ipu driver `nlc_total_errcnt` sysfs attribute.
- T28692: Added IPU sync utilisation in PCIe driver.
- T29859: Fixed access to multiple contiguous buffers in the PCIe driver.
- T29750: Fix PCIe driver build for CentOS 8.2.

- T28727: Add support to store total tile memory usages and export them as attributes over IPUoF.
- T28231: Added support for reading AER error status from the PCIe driver.
- T26628: Accumulate NLC correctible error count in sysfs.

6.1.2 Low level libraries and tools

2.0.0

- T33422: MultiIPU bootloader support.
- T32608: Use monotonic clock for duration calculations in GCDA and `hw_testing`.
- T33121: Add `--ipum-loopback` mode to `gc-iputraffictest`, for testing IPU-M loopback links.
- T33281: Extended `gc-iputraffictest` timeout when using the `--all-links --amp` options.
- T32498: `gc-iputraffictest`: display per-board power details.
- T31578: Add `--pulseamp` option to `gc-iputraffictest`.
- T29899: Add environment variable `GCDA_EXT_LINKS_TX_EQ` to override default TX EQ setting for external links of M2000.
- T31149: Add `--amp` option to `gc-iputraffictest`.
- T28484: Don't allow `gc-hosttraffictest` to run on a multi-IPU device.
- T31247: PVTI documentation updates.
- T24109: `libpvti` traces saved in sqlite format by default.
- T30878: Added process ID into the PVTI log file name.
- T29035: PVTI documentation updates.
- T30033: Added PVTI python decorators.
- T34775: Make IPU binary loading thread safe.
- T28718: Added on chip memory recording. Visible via `gc-monitor`.
- T29903: Implemented reset of IPU-M service tables.
- T34199: Fix pcie driver mailbox compatibility with Poplar SDK 1.4.
- T32751: Added external memory info when available via IPUoF.
- T32911: Added GCDA functionality to enable IPU GW-Links.
- T32193: Improve resilience for ICU comms during ICU event notifications.
- T31153: Fixed ICU async message handling.
- T32219: Fix to allow allocation of >16GB of streaming memory.
- T23659: More control over which architectures are supported during software builds.
- T14918: Display full path to executable in `gc-monitor`.
- T29973: Fixed memory leak in the PCIe driver userspace library.
- T28721: Update `gc-monitor` to add new `-f/--field` option to show `tile-mem/ext- mem/sync` utilisation at runtime, if available.
- T34860: BTNC routing must be used with loop back cables.
- T34711: Improved ICU timeout error handling.
- T30788: Added POD sw version info into `gc-monitor`.
- T32626: Fix GSD sync configuration.

- T34365: Improved error message when failing to open a required target support library.
- T31942: Fixed symbol name clash in the ICU source code vs Windows.
- T28727: Add support to store total tile memory usages and export them as attributes over IPUoF.
- T33779: Improved ICU mailbox failure recovery.
- T33275: Clear the ICU mailbox on error so that it has a better chance of recovery.
- T23314: Fixed ICU access regression in Windows.
- T31440: Added `--tile-overview` to `gc-info`.
- T31486: Add GCDA Board API and record information from all board power/temperature sensors.
- T31912: `gc-docker` discovers Fabric devices.
- T30500: Improvements and simplification of `reset`.
- T32266: Added required double read of 'volatile' SERDES registers.
- T31151: Correctable error count is displayed for Fabric devices.
- T22666: Default to `--ipc=host` in `gc-docker`.
- T28724: Add XML and CSV output formats to gc-monitor.
- T30710: Fixed `gc-info --phy-dump` for M2000.
- T31592: Improved debug logs during `reset`.
- T31255: Ensure IPU reset is done before Newmanry reset.
- T29684: Updated test tools to separate configure and attach via GCDA.
- T30814: Fix clearing of `.bss` in tile memory.
- T28716: Make `gc-monitor` more POD specific.
- T29171: Device reset stability improvements.
- T28805: Added support for `gc-powertest` over IPUoF.
- T29033: Added support for configuring extra sync zones via GCDA.
- T29444: Parameterise response time of tile exception notifications in GCDA RPC.
- T30152: Removed unused ICU mailbox mutex.
- T30090: Prevent GCDA users from being able to configure links when the IPU may be transmitting packets.
- T30143: Guarantee a SoC and IPU reset happen before a parity reset.
- T29036: Add an iterator in IPU arch info to access `<name, base-address>` of all instances of a hardware block.
- T29931: Fixed disassembler output.
- T14152: Extended info available via IPU arch info.
- T34724: Fix `rdma_disconnect` to use the correct CM ID.
- T33533: Improved IPUoF server shutdown.
- T34144: Improved IPUoF error reporting on attach and detach failure.
- T34223: Fix IPUoF server crash if application tries to release contiguous memory buffer before memory clearing has finished.
- T33937: IPUoF HSP polling performance improvement.
- T33406: Increased the maximum allowed V-IPU server timeout value.
- T33424: Improved full SQ efficiency. Released in IPUoF server version 1.4.0 and later.

- T31458: Avoid send queue overflow in Fabric client and server in non-polling mode.
- T33067: Improved IPUoF server and client log messages.
- T32493: set the `max_send_sge` correctly on the loopback QP.
- T31710: Improved mirror_host_buffer/RDMA-write latency.
- T30737: Ensure all user buffers are detached when a user process detaches.
- T30998: Added an environment variable for setting V-IPU timeout.
- T29870: Implemented PL-DDR clear at startup.

6.2 PopART Changelog

6.2.1 2.0.0

New features

- Change optimizer to use globalReplicationFactor when distributed
- Add error when trying replicated tensor sharding and global replication
- Copy partialType in serializematmuls
- Allow Offline device with distributed replication
- Ensure enableEngineCaching is set before loading cache and handle loading errors
- Allow Session to load binaries from serialized files
- Update compileAndExport to use same file format as the executable cache
- Fuse cache files and support multiple cache entries
- Add ability to PopArt to build in a standalone Conda environment
- Add DepthToSpace as a custom op
- Add backwards op for atan2
- Re-enable `synthetic_data_test` that requires HW
- Add enhanced debug information to popart
- Make multiconv op attributes optional
- Add support for biased convolutions in popart MultiConv
- Add numerous doxygen comments and comment improvements
- Add SessionOption `accumulationAndReplicationReductionType`, which controls the reduction type of accumulated gradients and reduction of replicas
- Stop using reduction option of loss operations to determine how loss gradients are reduced if `accumulationReductionType` is used
- Add reverse operation in the aiGraphcore domain
- Support input/output/anchor tensors of type int8 (including synthetic data)
- Support stashing/restoring/IPUCopying an int8 input tensor when pipelining
- Infer tensor to update in VarUpdateOps from input Tensors
- Check early for CMake versions in the found dependencies
- Improve model compile time by planning convolutions and matmuls in parallel
- Use a more poplar tensor-expression efficient implementation of ResizeOp on non-neg integers

- Add Connectionist Temporal Classification (CTC) loss
- Add support for setting Poplar options for LSTM operations
- Improve the debug information that is recorded in PopART
- Improve the PopART user guide
- Add ability to copy inputs/outputs to subgraphs in a just-in-time manner
- Improve error message when unknown tensors are used in the builder
- Add Sequence slice op working with packed sequences
- Add custom op shape inference that doesn't require onnx
- Add environment variable, `POPART_TRACE_TENSORS`, as an alternative to `PrintTensorOp`
- Support user added CallOps being used with pipelining
- Improve error messages in the case of optimizer compatibility errors
- Change PriTaskDependency to allow delaying picking tensor creators and express more complex forms of dependencies
- Add LAMB optimizer off-chip test with mean reduction
- Add support for ONNX ScanOp (forward-only)
- Add global batch size test with batch serialisation
- Implement operator support for AllReduce & ReduceScatter
- Add transformation to unroll and decompose LoopOps
- Add dead code elimination support as post-IR optimisation
- Add range-based indexing for remote buffers when used in LoopOps
- Add transform to merge adjacent LoopOps together
- Allow pipelining to resolve correct IPU from CallOps
- Add new warnings and errors for incompatible user settings
- Improve IR transform order to run critical transforms before outlining
- Improve replicated tensor sharding (RTS) optimizers for pipelining
- Add reshape to correct output shape for binary/mul gradoppattern outputs
- Avoid double scoping TensorIds when creating new outputs for operations within subgraphs
- Improve pre/post loss and RemoteLoad/RemoteStore scheduling
- Improve catching all types of (indirectly or directly) unmodifiable tensors for inplacing
- Ensure input and output data type of replicated allgather remains identical
- Remove extra outline attributes from ops that do not need them
- Add custom reshape with one input and one attribute
- Migrate DepthToSpace and SpaceToDepth to onnxpasses
- Add C code comments to python docs for cases where C and python functionality are the same.
- Add support for negative axes in reduce ops
- Move graph compiled logging to info level
- Add logging feature: Compile time breakdown into main components
- Add ONNX $\rightarrow$ ONNX transformations.
- Add Mod and Remainder Operators

- Run deserialized executable on a different device
- Make loadEngineAndConnectStreams public
- Add isAttached function to DeviceInfo
- Allow resetting device associated with session
- Enable loading OfflineIpu executables into Ipu device
- Add assert operation to popart using poplar's Abort and AbortOnCondition program
- Added hashing for all session option values (improved serialization)
- Defer cached engine loading to `Session::prepareDevice`

Bug Fixes

- Fix bug where it could fail to retrieve subgraph virtual graph IDs
- Improve unwinding and fix cases of circular tensor dependencies
- Fix batch axis pickup on weights for batch serialisation
- Fixes for engine caching of optimizer state data
- Make instancenorm, convolutions, LSTM/GRU re-setupable
- Fix aliasing (fwd/bwd region mapping) for elementwise broadcasting
- Fix replicated tensor sharding (RTS) for elementwise broadcasting
- Fix batch serialization in the case where graph replication factor is greater than zero
- Fix incorrect computation in the 'atan2' and 'fmod' gradient operators
- Fix the scaling of the output of the IdentityLossGradOp when the 'reduction' option is set to popart.ReductionType.Mean
- Fix an outlining bug related to pruning

6.3 PopTorch Changelog

6.3.1 2.0.0

- Added support for the following activation functions:
  - torch.nn.acosh
  - torch.nn.asinh
  - torch.nn.atanh
  - torch.nn.Hardshrink
  - torch.nn.SiLU
  - torch.nn.Softplus
  - torch.nn.Softshrink
  - torch.nn.Threshold
- Add support for the following random sampling operations:
  - torch.bernoulli
  - torch.distributions.Bernoulli
- Add experimental support for torch.nn.CTCLoss

- Added Adam optimizer
- Added support for `torch.nn.AdaptiveAvgPool1d`, `torch.nn.AdaptiveAvgPool3d`
- Migrated to PyTorch version 1.7.1
- Add support for `aten::index`, `aten::index_put_`
- Add support for `torch.zeros_like`, `torch.ones_like`
- Allow the user to specify which Optimizer attributes are constant or not.
- Allow the user to specify `mode=poptorch.DataLoaderMode.Async` in `poptorch.DataLoader` constructor instead of explicitly creating an AsynchronousDataAccessor
- Add support for `torch.nn.EmbeddingBag`
- Added support for `torch.clamp_max` and `torch.clamp_min`
- Add support for `torch.min(tensor, dim=., keepdim=.)` and `torch.max(tensor, dim=., keepdim=.)` overloads.
- Add support for `poptorch.isRunningOnIpu`. This function returns True when executing on IPU and False when executing the model outside IPU scope.
- Add support for `torch.amax` and `torch.amin`
- Add support for attributes in custom ops.
- Add support for precompilation and reloading exported executables (`poptorch.PoplarExecutor.compileAndExport` and `poptorch.load`)
- Add support for slices with variable start index (slice size must be constant).
- Add `ipuHardwareVersion` function to read the version of the IPU hardware present on the system.
- Changed default targetd Ipu version for the model and offline compilation to 2.
- Changed `accumulationReductionType(reduction)` option to now apply to replication reduction as well
- Add environment variable `POPTORCH_CACHE_DIR`
- Deprecated `Options.Popart`, `Options._Popart` may be used experimentally.

6.4 Poplar Changelog

6.4.1 2.0.0

New features

- A new implementation of the host IO runtime that doesn't block until needed
- Compile time improvements, especially related to complicated tensor expressions
- Switched to V3 of the profiler format by default

Bugs fixes

- Fixed a host memory explosion due to certain tensor expressions
- Fixed a bug when resetting replica subset
- Fixed a compile time issue when using non-default cell ordering in a GRU
- Fixed a hang caused by not respecting the sync group during control flow
- Fixed a crash when storing a `Target` in a hash table
- Fixed an overflow in the profiler during long executions
- Prevented the ProfilerAnalyseMemory stage from being called twice during lower
- Added error checking when creating a tensor with too many elements
- Fixed a bug caused by assuming an incorrect master tile in gateway mode
- Fixed a memory spike that only occurs while using gateway mode
- Don't output RTTI information (which wasn't supported anyway) when using `popc`
- IPU device NUMA information is now recomputed when the device changes
- Fixed a corruption in data streams when using serialised executables
- Profiling now only tracks the first replica
- Avoid database external reads during profiling before completion
- Do not reset ring counts completely when reconnecting a stream callback
- Fixed a tile exception caused by exceeding the 16-bit delay value in exchange lowering
- Fixed wrong link in Poplar User Guide
- Fixed problem with image filename in Poplar User Guide

Other improvements

- Optimised the performance when using the `syncReplicasIndependently` option
- Use fast RDMA transfers by default
- Better error message when trying to attach to unsupported number of IPUs
- Make `poplar::DeviceManager` and `poplar::Engine` thread-safe
- Code gen optimisation for `poplar::program::Switch`
- Support POD64 sync configuration where master is in the middle
- Export a CMake config version file for Poplar
- Added a new `poplar::program::Abort` program
- Added metadata about the memory usage to the final ELF files
- Remove limitation that vertex state must go in the lower 256KBs
- Better logging related to Poplar's temporary files
- Add Poplar functions to get/modify/restore CSR
- Optimised the interleave tensor expression to make it dimension agnostic
- Added support for counting number of FLOPs as well as cycles for a model
- Report cycles for all executions of the same StreamCopy rather than just the last one
- Improved binary load times by using a bootloader on the device

- Added a method to help printing tensor shapes
- Better support for DebugContext among Poplar's native programs
- New method for performing upsample by repetition on a tensor
- Improvements to the profiler to distuingish between host and remote buffer IO
- Support for GS2 sync in Poplar
- Added an engine option for better logging of copy lowering
- `popc` now sets -Wdouble-precision by default
- Better error checking when setting a print stream
- Improved documentation around the fact that random numbers when using the IPUModel are not reliable
- Removed references to an internal macro in the vertex/assembly documentation

6.5 Poplar Libraries Changelog

6.5.1 2.0.0

New features

- Added support for CTC loss in training graphs
- Added support for Faster Transformer using dynamic sequence padding
- Reimplemented TopK and sort using a bitonic sort algorithm
- Extended LSTM and GRU support to better handle layers with large sequences
- Added support for Cholesky decomposition
- Early Access POD128 (2xPOD64) and 2xPOD16 support for replica size 4 IPUs as demonstrated by RN50 and Bert L. Enabled by 3 phase allreduce where phase 2 crosses the gateway links.
- Early Access Grouped collectives API enabling collective operations on a sub-group of all replicas constituting the application e.g. all replicas within a rack (over IPU links) or across racks over gateway links only.

Bug fixes

- Fixed a bug where where doing `abs` on an int treating it as a float
- Fixed a bug in the embeddings where the wrong vertex was chosen
- Reduced the probability of overflow because of Block Sparse masking
- Fixed some performance issues with the `sparse_fc_layer` tool
- Fix problems in block sparse matmul documentation
- Fixed an overflow in the transpose vertex that caused a tile exception
- Fixed the operators of SlicePlan when used in an associative container
- Fixed a bug where the wrong vertex was chosen for a fully connected layer that had different input and output type
- Instantiate histogram codelets correctly for absolute input values
- Remove internal API from the public convolution header
- Fixed a bug that caused `popops::scaledAddTo(a, X, b, Y)` to fail when X is integer type
- Fixed a compile time explosion when planning certain convolution

- Updated documentation to fix broken links
- Fixed discrepancies in the 1x1 convolution vertex cycle estimates
- Fix histogram graph construction to allow for multi-dimensional tensors
- Force convolution groups to be 1 when the outer product vertex is used
- Fixed reference to “Unnamed Group” in the Poplibs documentation
- Fixed prearranges using the triangular solver with better tile mappings
- Clarified unclear documentation of the `remapOutputTensor` convolution option
- Fixed an error with the API in the popsparse documentation

Other improvements

- Compile time improvements during graph construction
- Added support for casting between int8 and fp16/fp32
- Add a grain size to created output tensor for convolutions if it cannot be based off the input operand tensors
- Improve calculation of the division of workers in the elementwise operations
- Improved the triangular solver layer with a planner and allocation functions
- Update the planner to specialise the convolution partial types based on the candidate vertex type
- Optimise binary elementwise operations that return a boolean
- Added non-inplace variants for non-linearities
- Added enhanced debug information to poprand
- Moved poplibs collective operations to GCL (see known issues for API change)
- Updated collectives documentation to reflect this move
- Improved logging during graph constructions
- Improved logging of convolutions
- Added `popops::hasInfOrNan` and `popops::hasInf` operators
- Added mixed precision version of aXMinusbY codelet
- Added support for elementwise equality of 16-bit integer types
- Improved the code generation of the `popops::iota` op
- Added support for reduce `LogAdd` in the reduction library
- Improve documentation on what types are supported by `popops::fill`
- Added an introduction to poplibs to the README

6.6 PopDist Changelog

6.6.1 2.0.0

New features

- Added documentation
- PopTorch support
- Improved all user error messages

- ipus_per_replica is now optional when calling `getDeviceId`

6.7 PopRun Changelog

6.7.1 2.0.0

New features

- Added documentation
- POD native synchronisation support
- Improved input validation
- Offline mode support (running application without requiring IPUs)
- Support multi IPU-Link domain and multi-host in offline mode
- Newly created V-IPU partitions are not reset
- Ability to specify a timeout for V-IPU server requests
- Partitions created by PopRun will be automatically evicted
- PopRun will provide interactive progress status while running
- All available NUMA nodes may be used and pinned consecutively
- OpenMPI 4.0 is now bundled with the Poplar SDK, removing OpenMPI as an external dependency.
- Temporary executable caching to avoid redundant compilations on the same host
- Added verification of the number of replicas in existing partitions

6.8 Libpva Library Changelog

6.8.1 2.0.0

New features

- Preview version of the PopVision Analyser Library for programmatic access to the `profile.pop` report. API provide for C++ and Python applications.

Bug fixes

- None

6.9 TensorFlow Changelog

6.9.1 2.0.0

New features

- Add support for overlapping communication and computation when using `IPUInfeedQueue` or the IPU Keras API. See the `Efficient IPU I/O` documentation section for more details.

- Support 8-byte signed/unsigned integer datatypes for `IPUInfeedQueue`, `IPUOutfeedQueue` and datasets passed to any of the IPU Keras models.
- Improved support for Cholesky and Triangular Solver operations. See `ipu.utils.set_optimization_options` for controlling the block size of the operations.
- Improved performance and integration when using PopDist and PopRun for multi-instance programs.
- Add a pre-compilation tracing mode for compiling programs for IPUs on machines without IPU hardware. See the `Compiling and pre-compiling executables` section in the documentation for more details.
- Use OpenMPI bundled with the Poplar SDK for Horovod, removing external dependency on OpenMPI.
- Implemented IPU versions of `nce_loss` and `sampled_softmax_loss`. Note that these are not compatible with host embeddings or `compute_accidental_hits`.
- Implemented IPU versions of `ctc_loss`, `ctc_loss_with_logits` and Keras `ipu.keras.CTCLoss`.
- Implemented IPU version of Dynamic GRU layer (`PopnnDynamicGRU`) to support dynamic sequence lengths.
- Implemented IPU version of Attention Update GRU layer (`PopnnAUGRU`) with support for dynamic sequence lengths.
- Improvements for tensor allocations when using embeddings or one-hot operations.
- Take gradient accumulation into account for steps in IPUPipelineEstimator.
- Use temporary executable cache from PopDist as a fallback to avoid redundant compilations.
- Add `synthetic_data_categories` flag to `TF_POPLAR_FLAGS` for fine grain control of synthetic data.
- Add further optimizations when `enable_fast_math` is enabled.
- Generated profiling files for the PopVision Graph Analyser tool are now stored in directories in a `tf_report{ISO date}{Process ID}` format. Note that when using PopDist this format is `tf_report{ISO date}{Process ID}__instance_{Instance Number}` to allow profiling of each instance.
- Improved integration with the PopVision Graph Analyser.
- Improved documentation of the IPU Keras API.
- Add `unique_sharding` and `keep_input_layouts` to `ipu.outlined_function`. See the API documentation for more details.
- Improved integration with remote buffers when using `ipu.outlined_function` in custom TensorFlow optimizers.
- Support the `axis` parameter in `InstanceNormalization`.
- Improvements to the performance of the data feeding mechanism.
- Automatically display a compilation progress bar when compiling larger models. Can be disabled by setting `TF_POPLAR_FLAGS=--show_progress_bar=false`.
- Improved error messages and error reporting.

Bug fixes

- Make average the default operation for Horovod allreduce.
- Fix handling of multi input and multi output IPU Keras models.
- Fix the grouping in the IPU Keras `InstanceNormalization` and `LayerNormalization`.
- Fix handling of datasets with nested structures passed to the IPU Keras models.
- Fix groups in IPU Keras Layer and Instance norm
- Fix initializer colocation in IPUMultiWorkerStrategy.

- Fix to recomputation in pipelining when a pipeline stage contains stateful operations which cannot be re-computed.
- Prevent `TemporaryVariable` operations from being generated.
- Fix edge cases where a transposed constant could be used as-is without transposition.
- Support `--log_cycle_count` for cached executables.

CHAPTER

SEVEN

KNOWN ISSUES

The following section will detail known issues in v.
Each product will be detailed separately.

7.1 Driver & Utilities known issues

7.1.1 1.0.50

- Killing an application whilst it is in the process of attaching to an IPU within a POD can cause the IPUs to reject further connections, requiring a partition reset to recover.

7.2 PopART known issues

7.2.1 2.0.0

None.

7.3 PopTorch known issues

7.3.1 2.0.0

None.

7.4 Poplar known issues

7.4.1 2.0.0

- A bootloader is now used by default on Mk2 targets which reduces the tle memory available slightly
- Deprecated the following dangerous API's:
  - `Engine::Engine(Graph &&, . . . )`
  - `Engine::readTensor(StringRef, void *)`
  - `Engine::writeTensor(StringRef, const void *)`
  - `Engine::connectStream(const DataStream &stream, void *, void *)`
  - `Engine::connectStream(const DataStream &stream, void *)`
  - `Engine::connectStreamToCallback(const DataStream &, StreamCallbackHandle)`
  - `Engine::connectStreamToCallback(const DataStream &, unsigned, StreamCallbackHandle)`
  - `Engine::copyFromRemoteBuffer(const RemoteBuffer &, void *, int, unsigned)`
  - `Engine::copyToRemoteBuffer(const RemoteBuffer &, void *, int, unsigned)`
  - `compileGraph(Graph &&)`
- Deprecated the `poplar::cycleCount` and `poplar::cycleStampt` API's that do not take an explicit sync type.
- When running TensorFlow CNN applications on POD128 the Poplar Engine option `"target.maxStreamCallbackThreadsPerNumaNode": "auto"` must be set to ensure convergence to reference accuracy and throughput.
- A deadlock will occur during replicated training if the fastest replica is executing sufficiently fast that it empties its data feed queue, when the slowest replica has a full queue.
- Some tables do not appear in the HTML Poplar documents. See the documents published on https://docs.graphcore.ai for the correct versions.

7.5 Poplar Libraries known issues

7.5.1 2.0.0

- The following API's have been deprecated:
  - `popops::reduceScatter(. . . , popops::Operation, . . . )`
  - `popops::allReduce(. . . , popops::Operation, . . . )`
  - `poplin::createTriangularSolveInputLHS(. . . , std::size_t blockSize, . . . )`
  - `poplin::createTriangularSolveInputRHS(. . . , std::size_t blockSize, . . . )`
  - `poplin::triangularSolve(. . . , std::size_t blockSize, . . . )`
  - `poplin::getTriangularSolveMatMulPrePlanParameters(. . . , std::size_t blockSize, . . . )`

7.6 PopDist known issues

7.6.1 2.0.0

None.

7.7 PopRun known issues

7.7.1 2.0.0

None.

7.8 Libpva Library known issues

7.8.1 2.0.0

None.

7.9 TensorFlow known issues

7.9.1 2.0.0

- IPU Keras API ignores the `shuffle` and `verbose` arguments.

CHAPTER

EIGHT

COMPATIBILITY CHANGES

The following section will detail compatibility changes in v

Product	Paragraph
Driver & Utilities	*Driver & Utilities compatibility changes* (page 24)
PopART	*PopART compatibility changes* (page 24)
PopTorch	*PopTorch compatibility changes* (page 25)
Poplar	*Poplar compatibility changes* (page 25)
Poplar Libraries	*Poplar Libraries compatibility changes* (page 25)
PopDist/PopRun	*PopRun/PopDist compatibility changes* (page 25)
Libpva Library	*Libpva Library compatibility changes* (page 25)
TensorFlow	*TensorFlow compatibility changes* (page 25)

8.1 Driver & Utilities Compatibility changes

8.1.1 1.0.50

None.

8.2 PopART Compatibility changes

8.2.1 2.0.0

- [API] Deprecate accumulationReductionType SessionOption
- [API] Remove previously deprecated LegacyOpFactoryFunction
- [API] Deprecate `0px::debugPrefix`

8.3 PopTorch Compatibility changes

8.3.1 2.0.0

None.

8.4 Poplar Compatibility changes

8.4.1 2.0.0

None.

8.5 Poplar Libraries Compatibility changes

8.5.1 2.0.0

None.

8.6 PopDist Compatibility changes

8.6.1 2.0.0

None.

8.7 PopRun Compatibility changes

8.7.1 2.0.0

None.

8.8 Libpva Library Compatibility changes

8.8.1 2.0.0

None.

8.9 TensorFlow Compatibility changes

8.9.1 2.0.0

- TensorFlow packages have been renamed from `gc-tensorflow` to `tensorflow`. This ensures that the Graphcore port of the TensorFlow framework can be detected by `pip` when using other libraries dependent on TensorFlow.
- The default Poplar IPU Model used for tests has changed to IPU version 2.

- Expose `offload_weight_update_variables` to the IPU Keras Pipelining API.
- Custom IPU operations metadata API (now level 4) has changed to include parameters `input_to_output_tensor_aliasing` and `is_hashable`. See the `Custom IPU operations` documentation section for further details.
- Updated optimizer namespace structure to allow simpler import statements.
- `ipu.keras.SequentialPipelineModel` has been renamed to `ipu.keras.PipelineSequential`. The old name is deprecated and will be removed in a future release.
- The `accumulation_count` and `accumulation_dtype` arguments in the constructors of the `ipu.keras.Model` and `ipu.keras.Sequential` classes have been renamed to `gradient_accumulation_count` and `gradient_accumulation_dtype`. The old names are deprecated and will be removed in a future release.
- Autosharding has been deprecated and will be removed in a future release.
- For a comprehensive list of all the API calls which have been deprecated or removed, and how to update them, see the `API changes` section in the documentation.

CHAPTER

NINE

APPENDIX

9.1 Appendix A : Additional requirements

9.1.1 PopVision Graph Analyser

- To be able to view profiling reports generated by SDK v1.3.0, PopVision Graph Analyser v2.1 is required.

9.1.2 TensorFlow

To correctly execute TensorFlow code please ensure:

Intel platforms

- Use Python 3.6 as minimum version
- A CPU compatible with the AVX-512 instruction set is needed.

AMD plaforms

- Use Python 3.6 as minimum version
- A CPU compatible with the Znver1 instruction set is needed.