GRAPHCORE'S AI SOFTWARE STACK IS NOW CUSTOMER-DRIVEN

INTRODUCTION

In a paper published in May 2020, we examined the hardware and software offerings of Graphcore, detailing the development toolchain and noting the commitment to the open-source development community. After 18 months, Graphcore has released the latest version of the Poplar environment (SDK2.4). The company has created critical mass with an open-source development community across various applications to help mature and expand the Graphcore software stack. This updated research paper will examine advancements the company has introduced and the level of community involvement in ongoing software and model engineering.

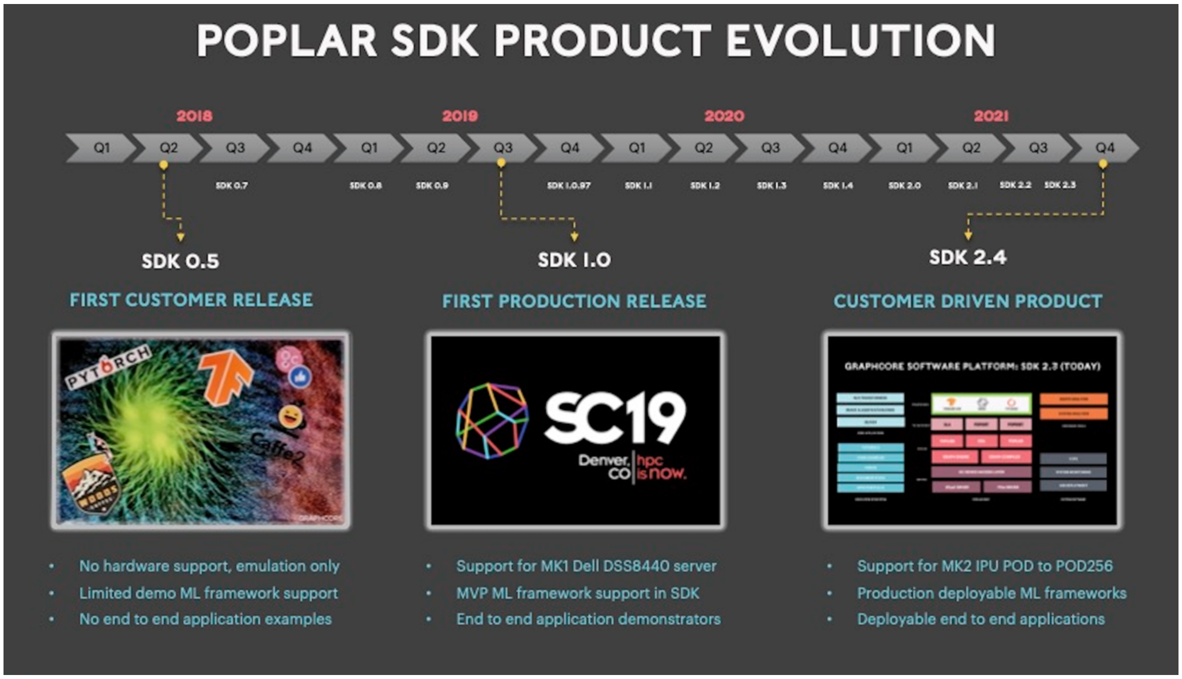
Software for new processor designs is critical to enabling application deployment and optimizing performance. UK-based startup Graphcore, a systems provider for application acceleration, emphasizes software, dedicating over half its engineering staff to the challenge. Graphcore's second-generation Intelligence Processing Unit (IPU) utilizes the expression of an algorithm as a directed graph, and the company's Poplar software stack translates models and algorithms into those graphs for execution. The software simplifies adapting the IPU-based system for AI and parallel computing, making it vital to success.

All AI hardware startups struggle to break through the adoption impasse: no software ➔ , no customers, ➔ no software. (By "software," we mean optimized models and kernels, in addition to frameworks and tools.) Graphcore, however, seems to be approaching a critical mass of company-engineered and open-source contributions, enabling a customer-centric mindset to drive enhancements. This community has a long way to go before remotely resembling NVIDIA's massive ecosystem to enjoy and harvest. But every journey begins with the first steps, and we are impressed with Graphcore's considerable progress.

THE MATURING OF THE GRAPHCORE SOFTWARE PLATFORM

The company has made significant strides since we published the original version of our research in May 2020. Beyond the Graphcore-engineered enhancements, the software ecosystem around the IPU is evolving from a push to a pull model: the AI community is now actively contributing to the open-source software Graphcore has released. Graphcore provides an extensive array of learning modules, reference designs, and tools; we would assert that only NVIDIA exceeds Graphcore in developer enablement.

The Graphcore stack is organized into four pillars: tools and apps for the developer ecosystem, the Graphcore engineered Poplar SDK, PopVision performance analyzer tools, and System software for deployment and management.


POPLAR SDK PRODUCT EVOLUTION

SDK 0.5

FIRST CUSTOMER RELEASE

- No hardware support, emulation only
- Limited demo ML framework support
- No end to end application examples

SDK 1.0

FIRST PRODUCTION RELEASE

- Support for MK1 Dell DSS8440 server
- MVP ML framework support in SDK
- End to end application demonstrators

SDK 2.4

CUSTOMER DRIVEN PRODUCT

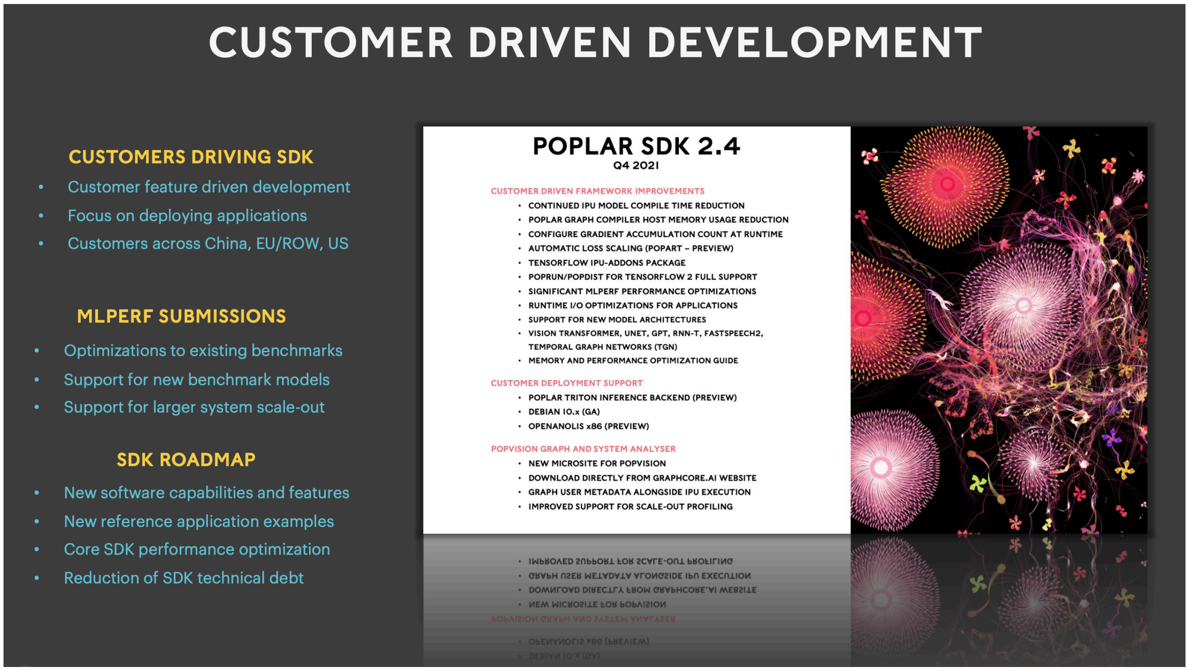
- Support for MK2 IPU POD to POD256
- Production deployable ML frameworks
- Deployable end to end applications

Figure 1: The Poplar SDK development timeline.

The original Graphcore software stack was a solid start, but as its engineers supported client projects, they evolved the stack to become more comprehensive and the tools easier to use. Notably, the company's commitment to open-source the entire stack (except the compiler) enabled clients to help build and expand the software stack and available algorithms/models. The end-user engagement and contributions provide a multiplier effect to the internal software engineering efforts in Graphcore.

Let's look at what has changed in the Poplar SDK since our last report. SDK 2.4 has added significant features. The SDK includes ten customer-driven enhancements to frameworks, including enhanced Pytorch integration, TensorFlow Lamb optimizer support, and enhanced scalability for the POD256. Additional support for deployment includes enhanced support for inference processing. We find this telling, so we expect more developments from Graphcore on the inferencing front this year.

One of the many additions that will help customers understand how models are optimized and deployed on IPU clusters are three core applications: open-source, optimized stacks for NLP/Transformers, Image Classification, and the MLPerf models, which Graphcore submitted to the MLPerf V1.1 training benchmarks. Large Language model optimizations were also engineered, covered later in this report.



CUSTOMER DRIVEN DEVELOPMENT

CUSTOMERS DRIVING SDK

- Customer feature driven development
- Focus on deploying applications
- Customers across China, EU/ROW, US

MLPERF SUBMISSIONS

- Optimizations to existing benchmarks
- Support for new benchmark models
- Support for larger system scale-out

SDK ROADMAP

- New software capabilities and features
- New reference application examples
- Core SDK performance optimization
- Reduction of SDK technical debt

POPLAR SDK 2.4
Q4 2021

CUSTOMER DRIVEN FRAMEWORK IMPROVEMENTS

- CONTINUED IPU MODEL COMPILE TIME REDUCTION
- POPLAR GRAPH COMPILER HOST MEMORY USAGE REDUCTION
- CONFIGURE GRADIENT ACCUMULATION COUNT AT RUNTIME
- AUTOMATIC LOSS SCALING (POPART – PREVIEW)
- TENSORFLOW IPU-ADDONS PACKAGE
- POPRUN/POPDIST FOR TENSORFLOW 2 FULL SUPPORT
- SIGNIFICANT MLPERF PERFORMANCE OPTIMIZATIONS
- RUNTIME I/O OPTIMIZATIONS FOR APPLICATIONS
- SUPPORT FOR NEW MODEL ARCHITECTURES
- VISION TRANSFORMER, UNET, GPT, RNN-T, FASTSPEECH2, TEMPORAL GRAPH NETWORKS (TGN)
- MEMORY AND PERFORMANCE OPTIMIZATION GUIDE

CUSTOMER DEPLOYMENT SUPPORT

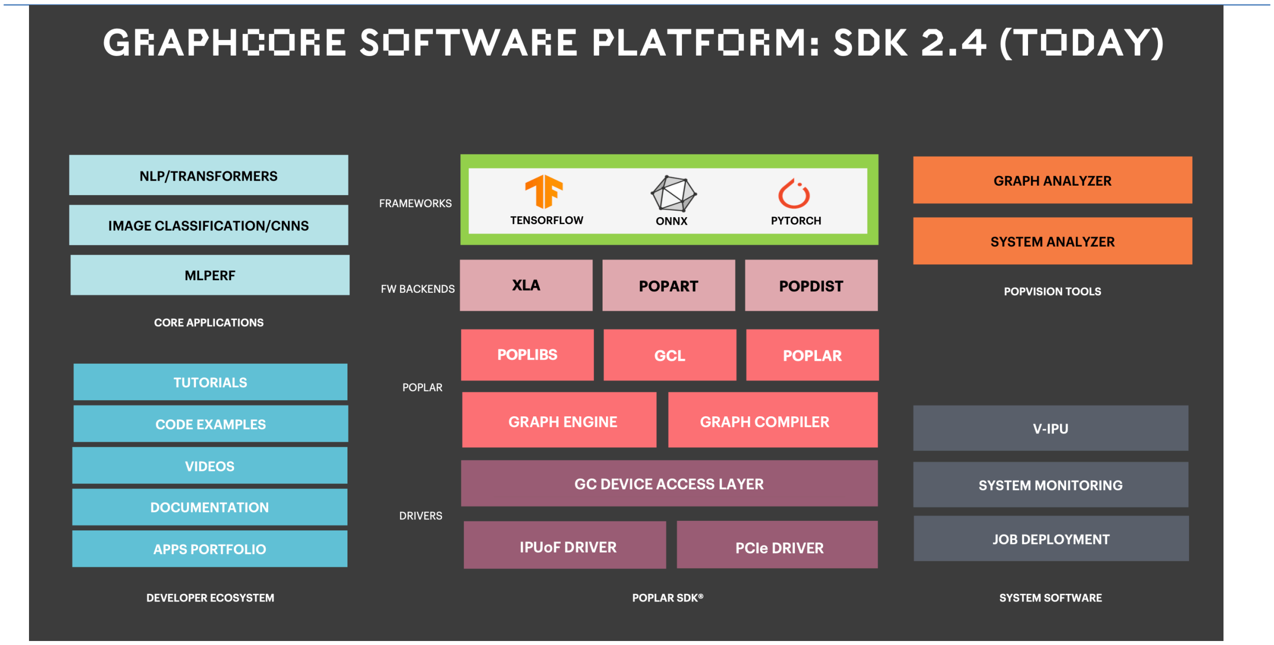
- POPLAR TRITON INFERENCE BACKEND (PREVIEW)
- DEBIAN 10.x (GA)
- OPENANOLIS x86 (PREVIEW)

POPVISION GRAPH AND SYSTEM ANALYSER

- NEW MICROSITE FOR POPVISION
- DOWNLOAD DIRECTLY FROM GRAPHCORE.AI WEBSITE
- GRAPH USER METADATA ALONGSIDE IPU EXECUTION
- IMPROVED SUPPORT FOR SCALE-OUT PROFILING

Figure 2: The 2.4 release includes some ten customer-driven enhancements.

Graphcore is addressing two challenges in software development for its IPU platforms: 1) make it easy to optimize and run existing ML software such as deep neural networks, or DNNs, expressed in high-level frameworks, and 2) enable research and development of entirely new fine-grained parallel workloads to run on an IPU infrastructure. The latter ability is central to the company's strategy and could enable Graphcore to address a much larger set of market segments, including finance and HPC applications.



GRAPHCORE SOFTWARE PLATFORM: SDK 2.4 (TODAY)

NLP/TRANSFORMERS
IMAGE CLASSIFICATION/CNNS
MLPERF
CORE APPLICATIONS

TUTORIALS
CODE EXAMPLES
VIDEOS
DOCUMENTATION
APPS PORTFOLIO
DEVELOPER ECOSYSTEM

FRAMEWORKS: TENSORFLOW, ONNX, PYTORCH
FW BACKENDS: XLA, POPART, POPDIST
POPLAR: POPLIBS, GCL, POPLAR, GRAPH ENGINE, GRAPH COMPILER
DRIVERS: GC DEVICE ACCESS LAYER, IPUoF DRIVER, PCIe DRIVER
POPLAR SDK®

GRAPH ANALYZER
SYSTEM ANALYZER
POPVISION TOOLS

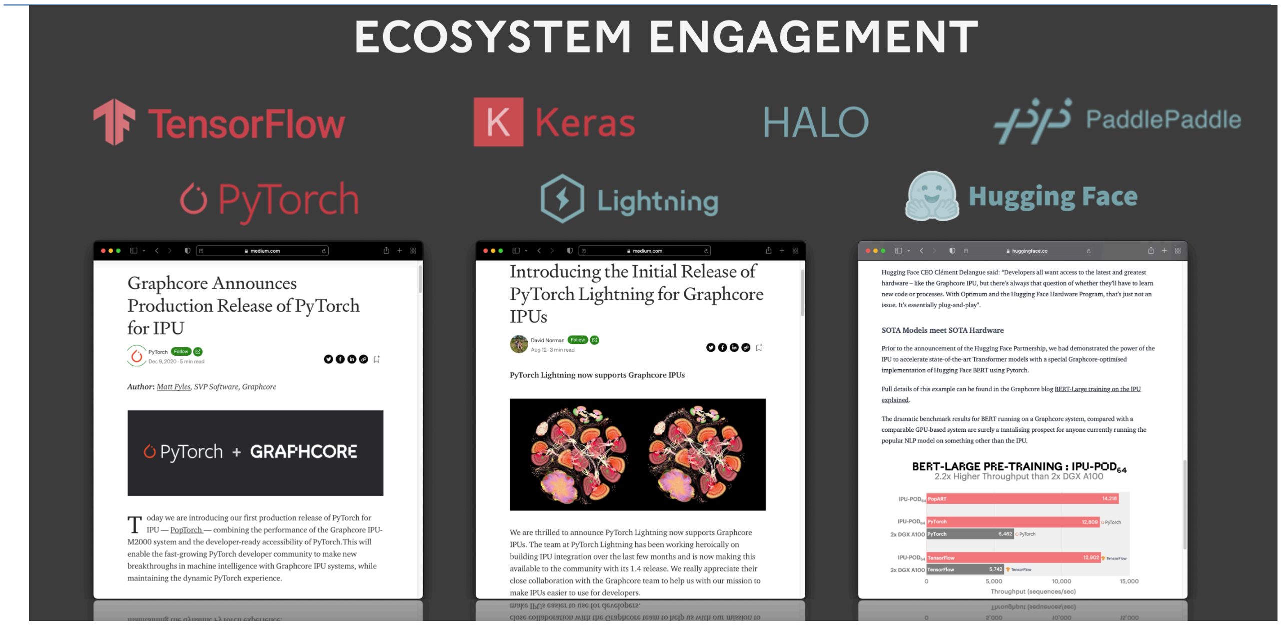
V-IPU
SYSTEM MONITORING
JOB DEPLOYMENT
SYSTEM SOFTWARE

Figure 3: The latest release of the Poplar SDK (V2.4) is complemented by system software, tools, applications, and developer enablement initiatives.

Partner Collaboration

While Graphcore's hardware and software are innovative and impressive, they alone are insufficient to build Graphcore into a material force on the acceleration landscape. To turn technology into valuable solutions, the company continually provides developer and partner programs to close the gap between potential and realized success. The company's near-constant release of new webinars, tools, tips, examples, and education are an industry best practice; we have not seen such quality and quantity from any AI startup. And the open-source software strategy provides the needed foundation to make this happen.

Graphcore is the only AI company that has released the source code of their ML framework backend implementations. Now partners such as Baidu (Paddle Paddle) and Alibaba (ODLA) have built their solutions on top of this. Graphcore enabled these projects by publicly releasing the source code to the PopART framework and Pytorch integration.



ECOSYSTEM ENGAGEMENT

Graphcore Announces Production Release of PyTorch for IPU

PyTorch Follow
Dec 9, 2020 · 5 min read

Author: Matt Fyles, SVP Software, Graphcore

Today we are introducing our first production release of PyTorch for IPU — PopTorch — combining the performance of the Graphcore IPU-M2000 system and the developer-ready accessibility of PyTorch.This will enable the fast-growing PyTorch developer community to make new breakthroughs in machine intelligence with Graphcore IPU systems, while maintaining the dynamic PyTorch experience.

Introducing the Initial Release of PyTorch Lightning for Graphcore IPUs

David Norman Follow
Aug 12 · 3 min read

PyTorch Lightning now supports Graphcore IPUs

We are thrilled to announce PyTorch Lightning now supports Graphcore IPUs. The team at PyTorch Lightning has been working heroically on building IPU integration over the last few months and is now making this available to the community with its 1.4 release. We really appreciate their close collaboration with the Graphcore team to help us with our mission to make IPUs easier to use for developers.

Hugging Face CEO Clément Delangue said: "Developers all want access to the latest and greatest hardware – like the Graphcore IPU, but there's always that question of whether they'll have to learn new code or processes. With Optimum and the Hugging Face Hardware Program, that's just not an issue. It's essentially plug-and-play".

SOTA Models meet SOTA Hardware

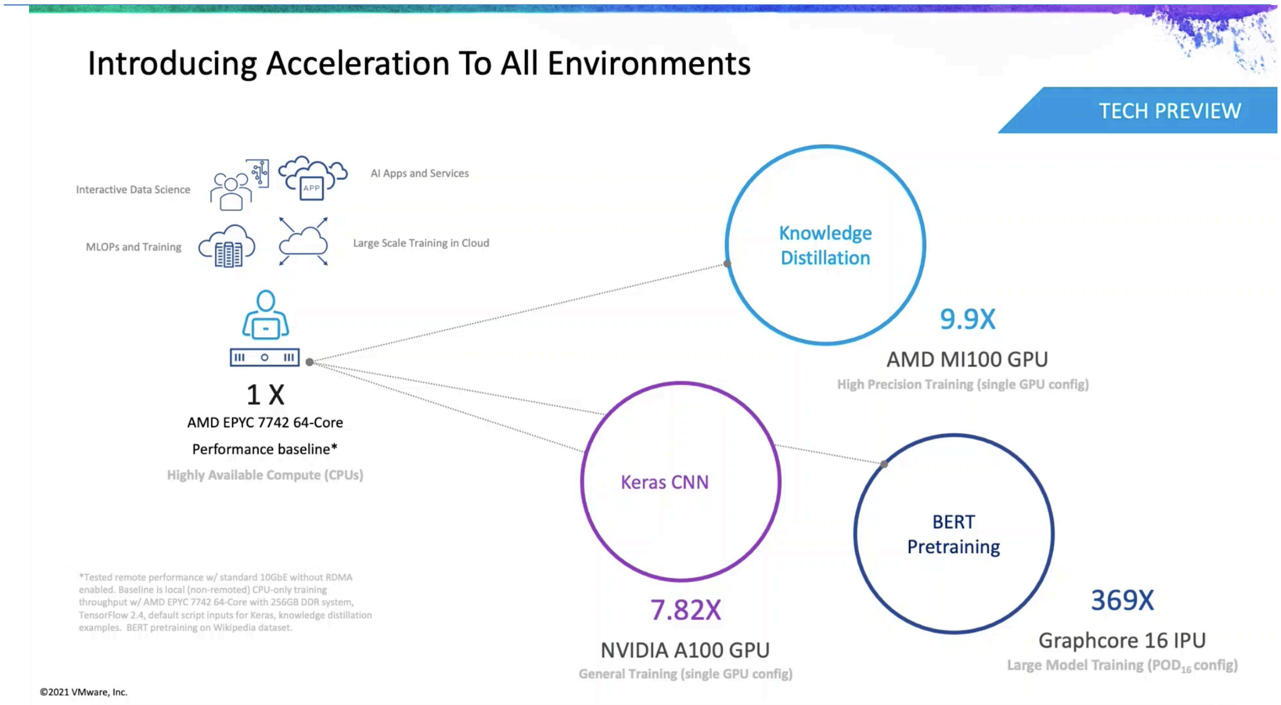
Prior to the announcement of the Hugging Face Partnership, we had demonstrated the power of the IPU to accelerate state-of-the-art Transformer models with a special Graphcore-optimised implementation of Hugging Face BERT using Pytorch.

Full details of this example can be found in the Graphcore blog BERT-Large training on the IPU explained.

The dramatic benchmark results for BERT running on a Graphcore system, compared with a comparable GPU-based system are surely a tantalising prospect for anyone currently running the popular NLP model on something other than the IPU.

BERT-LARGE PRE-TRAINING : IPU-POD64
2.2x Higher Throughput than 2x DGX A100
IPU-POD64 PopART 14,218
IPU-POD64 PyTorch 12,809 PyTorch
2x DGX A100 PyTorch 6,462 PyTorch
IPU-POD64 TensorFlow 12,902 TensorFlow
2x DGX A100 TensorFlow 5,742 TensorFlow
0
5,000
10,000
15,000
Throughput (sequences/sec)

VMWare and Pure Storage recently became Graphcore partners as well. VMWare and Graphcore have collaborated on virtualizing IPU's to enable pooling and sharing of IPU resources, providing familiar management and deployment constructs to data center managers. VMWare offers a valuable extension of the IPU-POD architecture for Enterprises that want to share these parallel-processing resources across the organization, using tools they are already using for virtualized multi-tenant hosting. This technology alliance is a meaningful indicator of traction and demand for Graphcore solutions instead of substantial deployment announcements, which customers tend to avoid.



Introducing Acceleration To All Environments

TECH PREVIEW

Interactive Data Science

AI Apps and Services

MLOPs and Training

Large Scale Training in Cloud

1 X

AMD EPYC 7742 64-Core

Performance baseline*

Highly Available Compute (CPUs)

Knowledge Distillation

9.9X

AMD MI100 GPU

High Precision Training (single GPU config)

Keras CNN

7.82X

NVIDIA A100 GPU

General Training (single GPU config)

BERT Pretraining

369X

Graphcore 16 IPU

Large Model Training (POD$_{16}$ config)

*Tested remote performance w/ standard 10GbE without RDMA enabled. Baseline is local (non-remoted) CPU-only training throughput w/ AMD EPYC 7742 64-Core with 256GB DDR system, TensorFlow 2.4, default script inputs for Keras, knowledge distillation examples. BERT pretraining on Wikipedia dataset.

©2021 VMware, Inc.

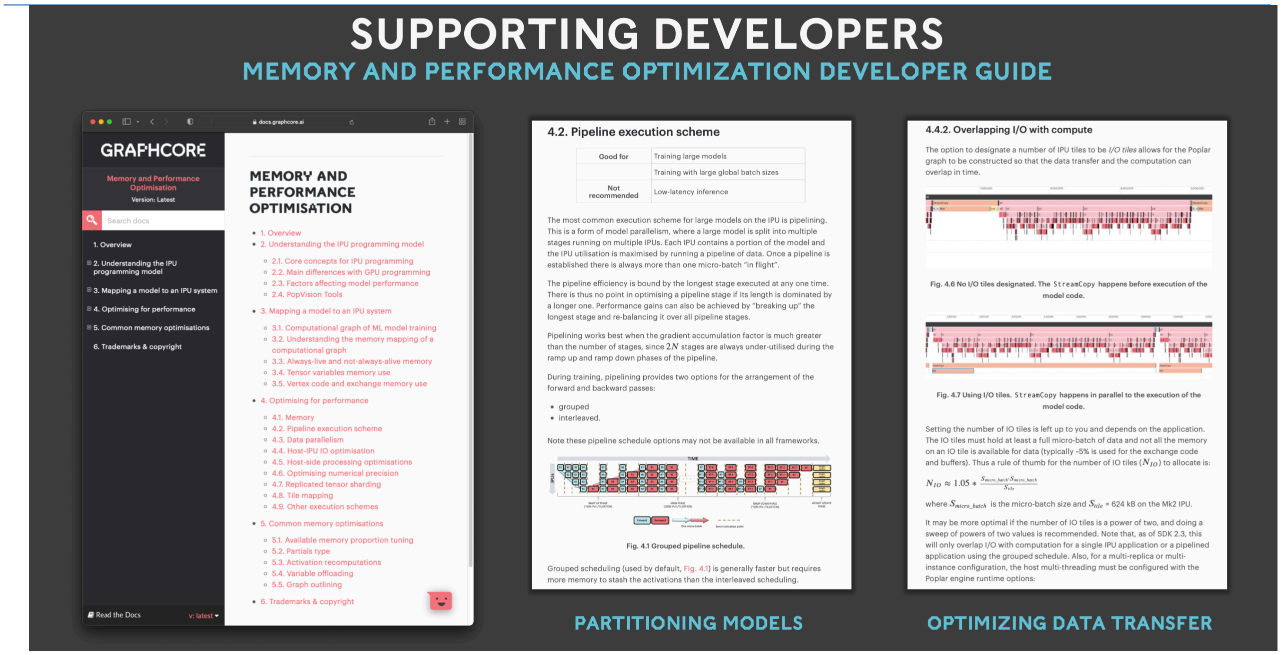
Figure 4: VMWare and Graphcore have collaborated to bring virtualization to IPU users focused on BERT NLP.

Another example of partner collaboration has brought state-of-the-art flash storage into the growing ecosystem. PureStorage has worked with Graphcore to define and deploy a reference architecture for the PureStorage Flashblade infrastructure. The PureStorage eXternal Fabric Modules scale linearly from one to ten chassis and 7 to 150 blades, providing up to 40x17 TB of storage for Graphcore IPOD customers.

Developer Collaboration & Support

As a result of the open approach and ecosystem enablement, end-users are increasingly embracing the Graphcore platform. The Graphcore Model Garden is an excellent example of ecosystem investments gaining traction with a repository of tested and optimized ML models ready for deployment. Much of this work is a natural outcome of the work done to submit MLPerf benchmark runs.

One of the most challenging tasks developers face when exploring new hardware is optimizing their code and models for an entirely new hardware architecture than what they might be accustomed to. In addition to the Model Zoo, webinars, documentation, etc., Graphcore offers an in-depth memory and performance optimization guide to help developers on this journey. This includes model pipelining and overlapping I/O with computation, among other topics. By providing this level of support, much of which undoubtedly emerged from client engagements, Graphcore can efficiently support the development community and enlarge its ecosystem.



SUPPORTING DEVELOPERS

MEMORY AND PERFORMANCE OPTIMIZATION DEVELOPER GUIDE

docs.graphcore.ai

GRAPHCORE

Memory and Performance Optimisation

Version: Latest

Search docs

1. Overview
2. Understanding the IPU programming model
3. Mapping a model to an IPU system
4. Optimising for performance
5. Common memory optimisations
6. Trademarks & copyright

Read the Docs v: latest

MEMORY AND PERFORMANCE OPTIMISATION

- 1. Overview
- 2. Understanding the IPU programming model
  - 2.1. Core concepts for IPU programming
  - 2.2. Main differences with GPU programming
  - 2.3. Factors affecting model performance
  - 2.4. PopVision Tools
- 3. Mapping a model to an IPU system
  - 3.1. Computational graph of ML model training
  - 3.2. Understanding the memory mapping of a computational graph
  - 3.3. Always-live and not-always-alive memory
  - 3.4. Tensor variables memory use
  - 3.5. Vertex code and exchange memory use
- 4. Optimising for performance
  - 4.1. Memory
  - 4.2. Pipeline execution scheme
  - 4.3. Data parallelism
  - 4.4. Host-IPU IO optimisation
  - 4.5. Host-side processing optimisations
  - 4.6. Optimising numerical precision
  - 4.7. Replicated tensor sharding
  - 4.8. Tile mapping
  - 4.9. Other execution schemes
- 5. Common memory optimisations
  - 5.1. Available memory proportion tuning
  - 5.2. Partials type
  - 5.3. Activation recomputations
  - 5.4. Variable offloading
  - 5.5. Graph outlining
- 6. Trademarks & copyright

4.2. Pipeline execution scheme

Good for	Training large models
	Training with large global batch sizes
Not recommended	Low-latency inference

The most common execution scheme for large models on the IPU is pipelining. This is a form of model parallelism, where a large model is split into multiple stages running on multiple IPUs. Each IPU contains a portion of the model and the IPU utilisation is maximised by running a pipeline of data. Once a pipeline is established there is always more than one micro-batch "in flight".

The pipeline efficiency is bound by the longest stage executed at any one time. There is thus no point in optimising a pipeline stage if its length is dominated by a longer one. Performance gains can also be achieved by "breaking up" the longest stage and re-balancing it over all pipeline stages.

Pipelining works best when the gradient accumulation factor is much greater than the number of stages, since $2N$ stages are always under-utilised during the ramp up and ramp down phases of the pipeline.

During training, pipelining provides two options for the arrangement of the forward and backward passes:

- grouped
- interleaved.

Note these pipeline schedule options may not be available in all frameworks.

TIME
IPUs
RAMP UP PHASE (~50% IPU UTILIZATION)
MAIN PHASE (100% IPU UTILIZATION)
RAMP DOWN PHASE (~50% IPU UTILIZATION)
WEIGHT UPDATE PHASE
Forward
Backward
One micro-batch
Synchronisation point

Fig. 4.1 Grouped pipeline schedule.

Grouped scheduling (used by default, Fig. 4.1) is generally faster but requires more memory to stash the activations than the interleaved scheduling.

PARTITIONING MODELS

4.4.2. Overlapping I/O with compute

The option to designate a number of IPU tiles to be *I/O tiles* allows for the Poplar graph to be constructed so that the data transfer and the computation can overlap in time.

Fig. 4.6 No I/O tiles designated. The `StreamCopy` happens before execution of the model code.

Fig. 4.7 Using I/O tiles. `StreamCopy` happens in parallel to the execution of the model code.

Setting the number of IO tiles is left up to you and depends on the application. The IO tiles must hold at least a full micro-batch of data and not all the memory on an IO tile is available for data (typically ~5% is used for the exchange code and buffers). Thus a rule of thumb for the number of IO tiles (N_{IO}) to allocate is:

$$N_{IO} \approx 1.05 * \frac{S_{micro\_batch} \cdot S_{micro\_batch}}{S_{tile}}$$

where S_{micro_batch} is the micro-batch size and S_{tile} = 624 kB on the Mk2 IPU.

It may be more optimal if the number of IO tiles is a power of two, and doing a sweep of powers of two values is recommended. Note that, as of SDK 2.3, this will only overlap I/O with computation for a single IPU application or a pipelined application using the grouped schedule. Also, for a multi-replica or multi-instance configuration, the host multi-threading must be configured with the Poplar engine runtime options:

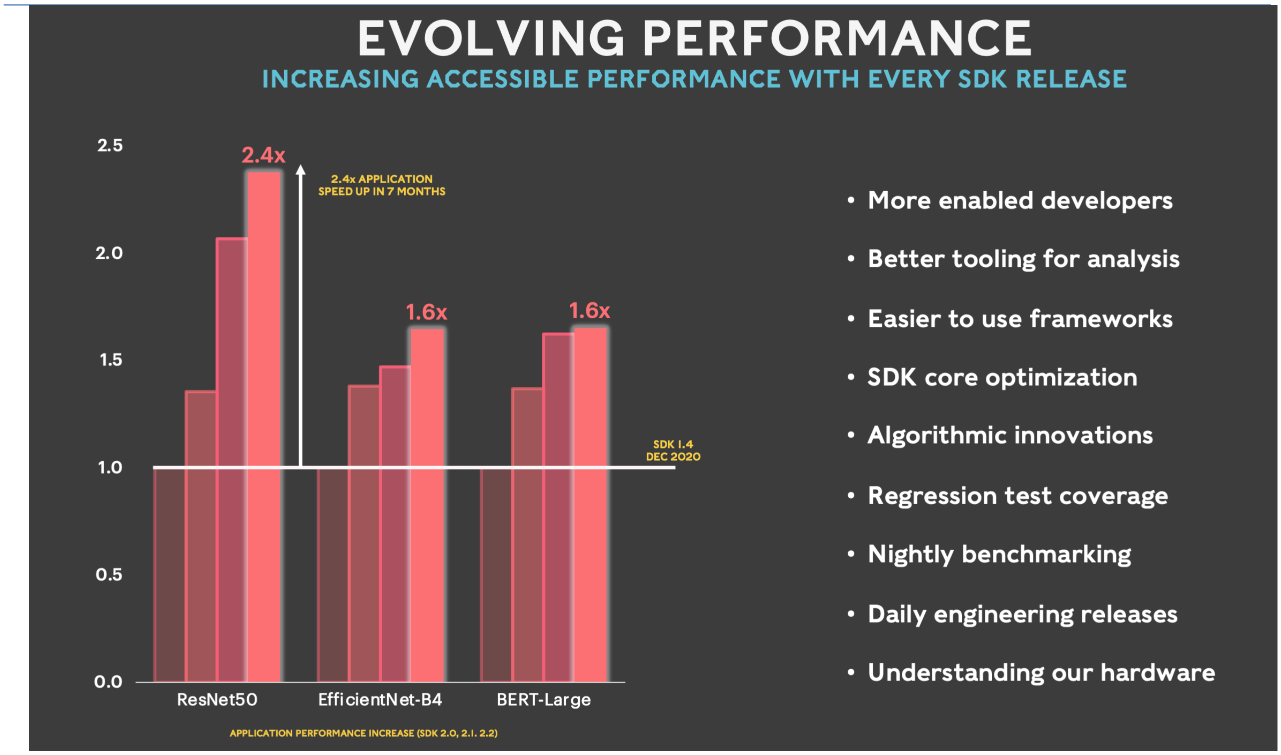
OPTIMIZING DATA TRANSFER

Figure 5: Graphcore has produced a guide to help developers learn how to tune and optimize their parallel processing applications running on the IPU.

Graphcore is also adept at leveraging the tremendous body of work from MLPerf submissions and turning the experience and code into learning tools, such as the webinar on BERT-large tuning and Jupyter notebook.

SOFTWARE PERFORMANCE ENHANCEMENTS

Like other players in the AI Acceleration market, Graphcore has continually delivered significant performance improvements as their software matures, with up to 2.4X improvement in image processing over the last year and a 60% decrease in BERT-Large training time. We anxiously await performance data on inference processing; running models on the same platform used for training has potential synergies and cost savings.



EVOLVING PERFORMANCE

INCREASING ACCESSIBLE PERFORMANCE WITH EVERY SDK RELEASE

- More enabled developers
- Better tooling for analysis
- Easier to use frameworks
- SDK core optimization
- Algorithmic innovations
- Regression test coverage
- Nightly benchmarking
- Daily engineering releases
- Understanding our hardware

Figure 6: Graphcore has continually improved the performance of MLPerf and other benchmarks, and we expect them to continue to make progress. The company has also improved the scalability of these and other models with the POD256.

LARGE LANGUAGE MODEL TRAINING

As we have noted, the updated SDK 2.4 comes with a rich set of tools, training, webinars, videos, podcasts, and documentation. Graphcore also provides a model garden to enable experimentation and adoption. These all assist developers and deliver content for Graphcore marketing to attract developers to the platform.

One of the most recent Graphcore communication campaigns promoted Large Language Model (LLMs) development on IPU-PODs. LLMs have become one of the latest innovations for AI, spurred by Open.AI GPT-3, and the subject of a media frenzy. While many of these multi-billion-parameter models have been trained on thousands of GPUs, the Graphcore architecture provides a cost-effective solution thanks to the exchange memory on each IPU-Machine, complementing the fast on-die memory of each IPU.

A recent Graphcore blog discussed how the 128- and 256-node clusters support models with tens of billions of parameters in the IPU-POD$_{256}$ utilizing the 16 TB of memory in these configurations, which are now fully supported and characterized by Graphcore. Graphcore shared the scalability of training the ViT vision transformer and GPT-2 (which, unlike GPT-3, has been open sourced), demonstrating roughly a 12X performance increase with 16X more IPU nodes. Beyond that, Graphcore believes the memory capacity of the POD256, which supports 16 terabytes of IPU Exchange Memory, opens the possibility to support models with trillions of parameters. These brain-scale models can also benefit from the new compute approaches made possible by our IPU processors.

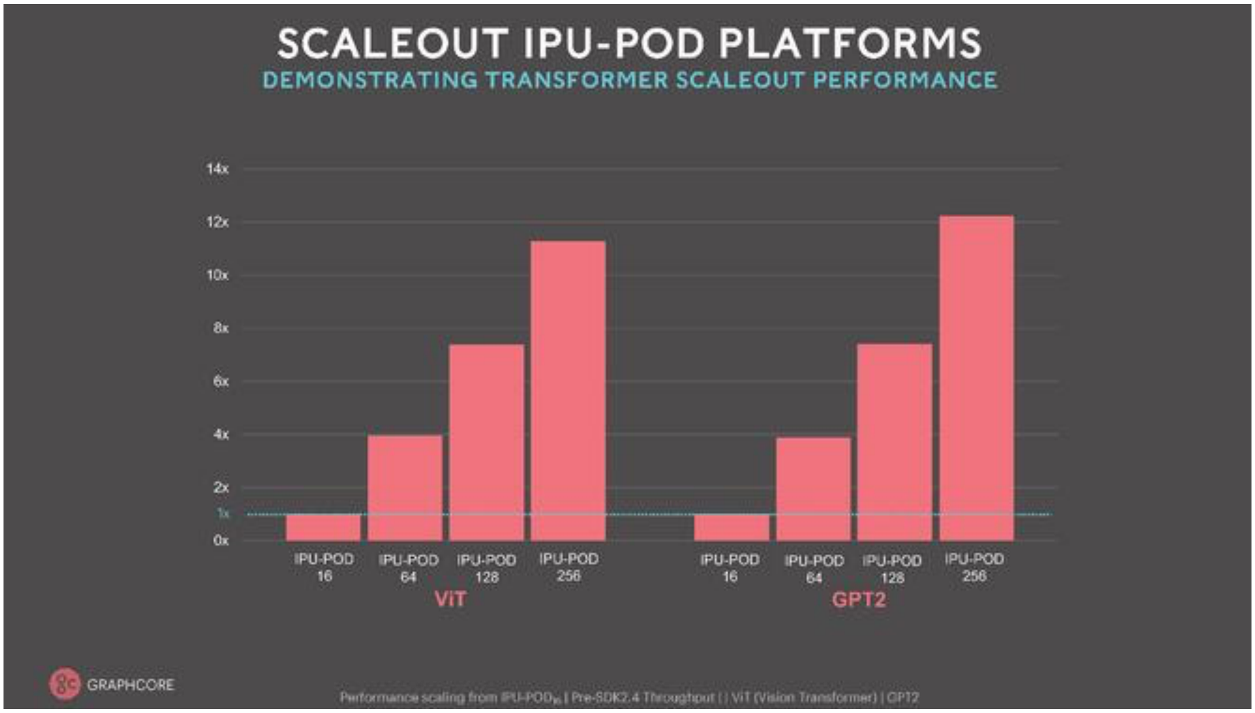


Figure 7: Graphcore has recently demonstrated excellent performance and scalability when training Large Language Models such as the Vision Transformer and GPT-2.

The Graphcore Bulk Synchronous Parallel Processing approach lends itself well to large models, with optimized data communications and phased execution enabling efficient distributed processing of hundreds of billions of parameters. By combining pipelining, tensor, and model parallelism, Graphcore can support massive models without burdening the developers with the job of manually managing these parallel queues.

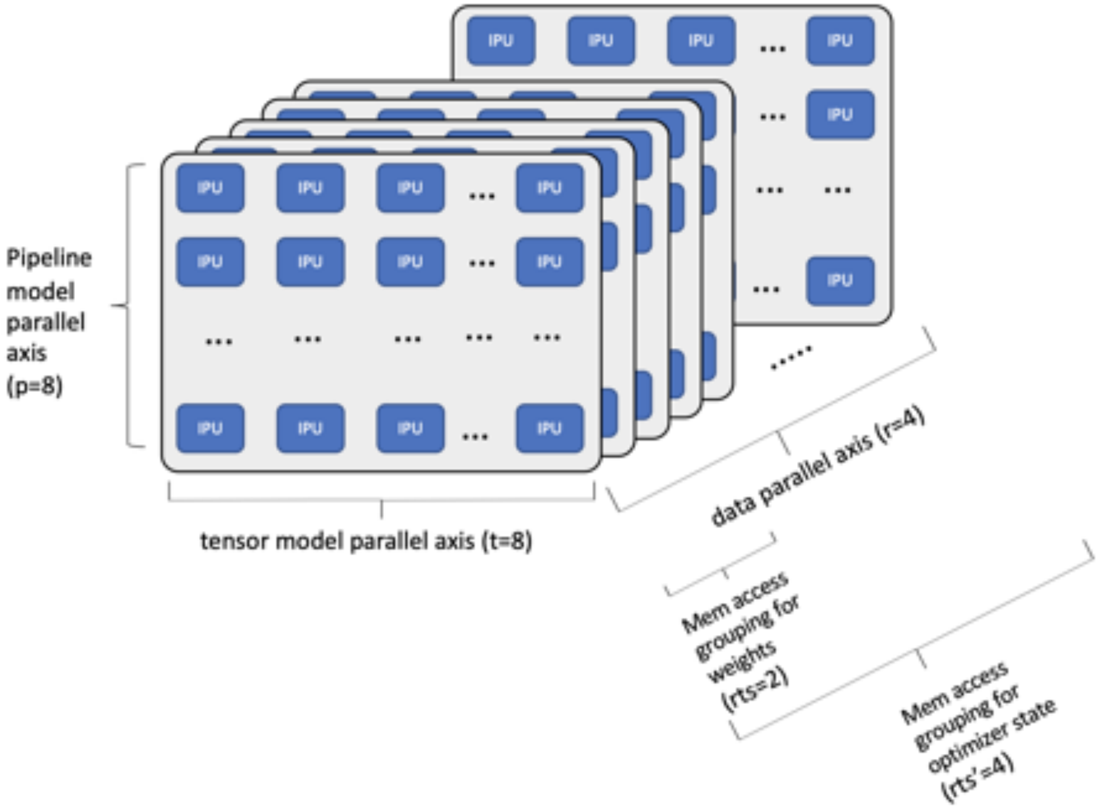

IPU
Pipeline model parallel axis (p=8)
tensor model parallel axis (t=8)
data parallel axis (r=4)
Mem access grouping for weights (rts=2)
Mem access grouping for optimizer state (rts'=4)

Figure 8: Large Language Model deployment on Graphcore IPUs.

THE BENEFITS OF HOST DISAGGREGATION

Depending on the scalar processing and management required, AI models and other parallel applications may require vastly different accelerator/server ratios. Consequently, instead of designing an IPU server on a PCIe card inserted into a 1- or 2-socket server, as is the case for almost all AI accelerators, the IPU-Machines are disaggregated from the host server(s), communicating over 100Gb Ethernet. Graphcore recently shared some data to illustrate the potential saving of this approach, and it is compelling. Why spend more money on CPUs if your application doesn't need them. And the infrastructure is dynamic, allowing different CPU to IPU ratios concurrently and re-configurations on the fly.

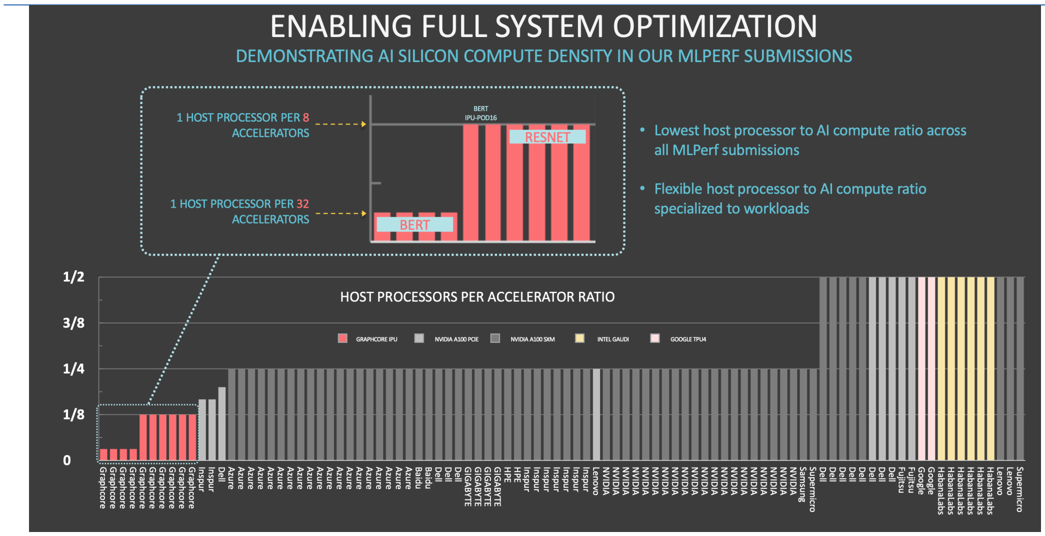


Figure 9: Graphcore illustrates how server disaggregation enables a more cost-effective training infrastructure, with a variable number of CPUs per accelerator.

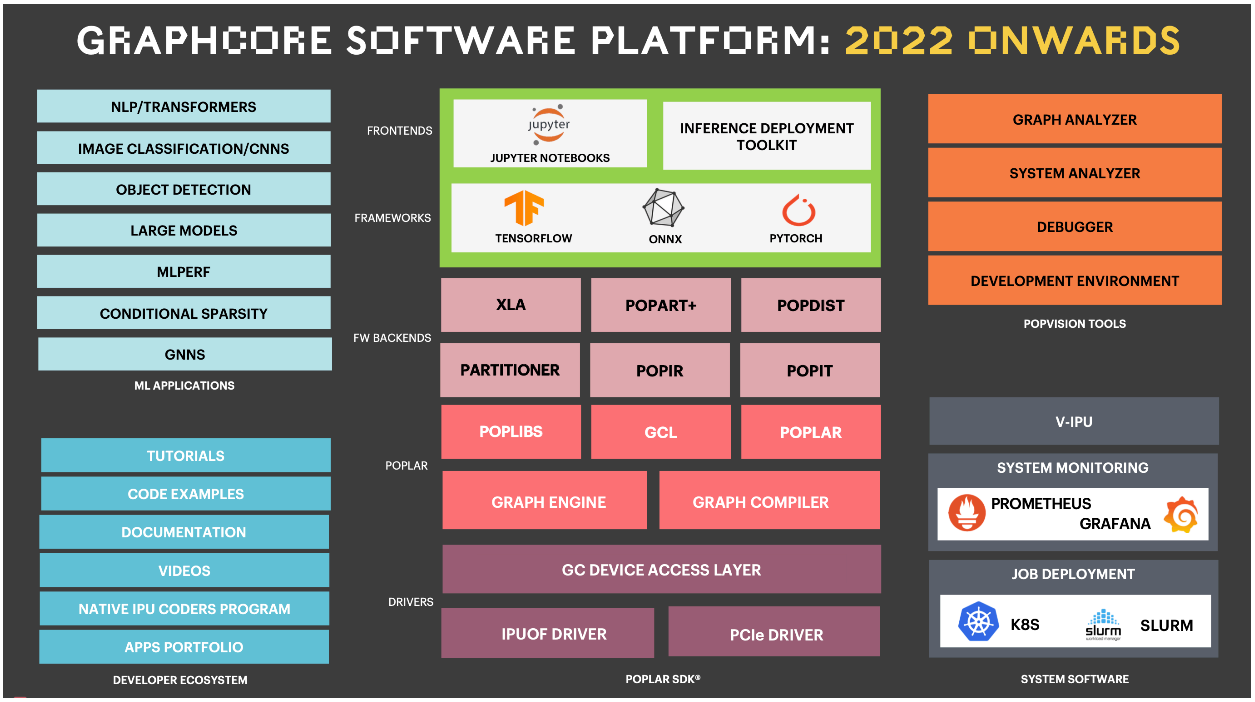
This plug-and-play approach of CPUs and IPUs enables:

1. Optimized performance for models that require more servers than, say, two CPUs for 4 or 8 accelerators,
2. Lower costs for models that require fewer CPUs by avoiding over-provisioning,
3. A flexible data center infrastructure that can handle both, and
4. Servers can reside in shared utility racks for optimal rack power utilization and serviceability.

The Road Ahead

Innovation in AI hardware and software presents one of the fastest-paced technologies today, and Graphcore has ambitious plans for both. In the software area, the company plans to add more applications such as conditional sparsity and GNNs. On the framework front, Graphcore has invested significantly in supporting the two main machine learning frameworks of Pytorch and Tensorflow alongside providing a lightweight and flexible runtime that supports ONNX and experimentation in PopART. These projects are open-source and can be leveraged as an underlying platform to support further ecosystem engagement and growth for programming frameworks in 2022.

New and enhanced support for Keras, Huggingface, TIMM, JAX, Pytorch Lightning, ODLA, and Paddle Paddle are all planned to increase IPU platforms' adoption further and make Graphcore products available to more developers. Alongside these, there will be product enhancements to support inference deployments with an inference-focused toolkit and broader developer productivity support for Jupyter notebooks that can be used with standard ML ops platforms.



GRAPHCORE SOFTWARE PLATFORM: 2022 ONWARDS

NLP/TRANSFORMERS
IMAGE CLASSIFICATION/CNNS
OBJECT DETECTION
LARGE MODELS
MLPERF
CONDITIONAL SPARSITY
GNNS
ML APPLICATIONS

TUTORIALS
CODE EXAMPLES
DOCUMENTATION
VIDEOS
NATIVE IPU CODERS PROGRAM
APPS PORTFOLIO
DEVELOPER ECOSYSTEM

FRONTENDS
JUPYTER NOTEBOOKS
INFERENCE DEPLOYMENT TOOLKIT
FRAMEWORKS
TENSORFLOW
ONNX
PYTORCH
FW BACKENDS
XLA
POPART+
POPDIST
PARTITIONER
POPIR
POPIT
POPLAR
POPLIBS
GCL
POPLAR
GRAPH ENGINE
GRAPH COMPILER
DRIVERS
GC DEVICE ACCESS LAYER
IPUOF DRIVER
PCIe DRIVER
POPLAR SDK®

GRAPH ANALYZER
SYSTEM ANALYZER
DEBUGGER
DEVELOPMENT ENVIRONMENT
POPVISION TOOLS

V-IPU
SYSTEM MONITORING
PROMETHEUS
GRAFANA
JOB DEPLOYMENT
K8S
SLURM
SYSTEM SOFTWARE

Figure 10: Graphcore has shared a high-level roadmap of software it is currently developing.

Conclusion: Graphcore has enabled a self-evolving ecosystem for parallel programming on their hardware and is consequently gaining momentum.

Developers require fast, scalable accelerators to handle the massive computational loads of larger models such as natural language processing and conversational interfaces to train large neural networks. But AI developers also need a robust development environment that meets them in their AI development journey. The latest version of Graphcore software delivers and is benefiting from client contributions to the open-source base and is being picked up by end-users and partners.

More broadly, computationally intensive applications are emerging which require a high-speed parallel processor that goes beyond the matrix-multiplication operations common to neural networks. Graphcore appears to have developed just such a general-purpose, high-performance hardware and software platform to meet these needs. The emergence of Large Language Models with performance data indicates that Graphcore intends to stay on the leading edge of model development and optimization.

Important Information About This Paper

Author

Karl Freund, Founder and Principal Analyst at Cambrian-AI Research

Publisher

Karl Freund, Cambrian-AI Research, LLC.

Inquiries

Contact us if you would like to discuss this report, and Cambrian-AI Research will respond promptly.

Citations

This paper can be cited by accredited press and analysts but must be mentioned in the context, displaying the author's name, author's title, and "Cambrian-AI Research." Non-press and non-analysts must receive prior permission from Cambrian-AI Research for any citations.

Licensing

Disclosures

Graphcore Inc. Cambrian-AI Research commissioned this paper to provide research, analysis, advising, and consulting to many high-tech companies mentioned in this paper. No employees at the firm hold any equity positions with any companies cited in this document.

Disclaimer

The information presented in this document is for informational purposes only and may contain technical inaccuracies, omissions, and typographical errors. Cambrian-AI Research disclaims all warranties as to the accuracy, completeness, or adequacy of such information and shall have no liability for errors, omissions, or inadequacies in such information. This document consists of the opinions of Cambrian-AI Research and should not be construed as statements of fact. The views expressed herein are subject to change without notice.

Cambrian-AI Research provides forecasts and forward-looking statements as directional indicators and not as precise predictions of future events. While our forecasts and forward-looking statements represent our current judgment on what the future holds, they are subject to risks and uncertainties that could cause actual results to differ materially. You are cautioned not to place undue reliance on these forecasts and forward-looking statements, which reflect our opinions only as of the date of publication for this document. Please keep in mind that we are not obligating ourselves to revise or publicly release the results of any revision to these forecasts and forward-looking statements considering new information or future events.