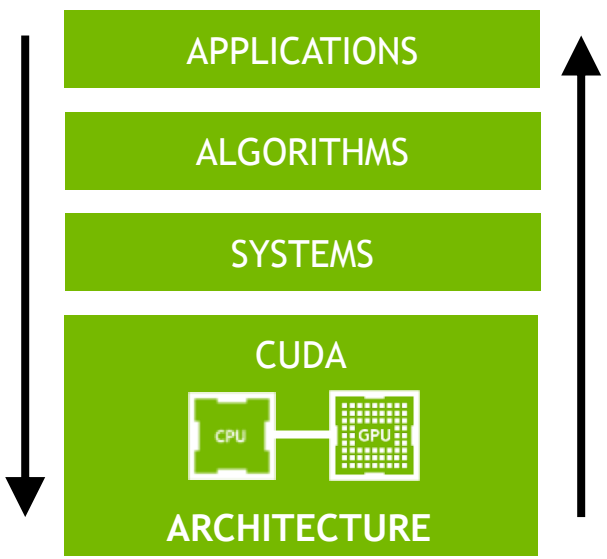
PROGRAMMING ENVIRONMENT FOR HPC+AI

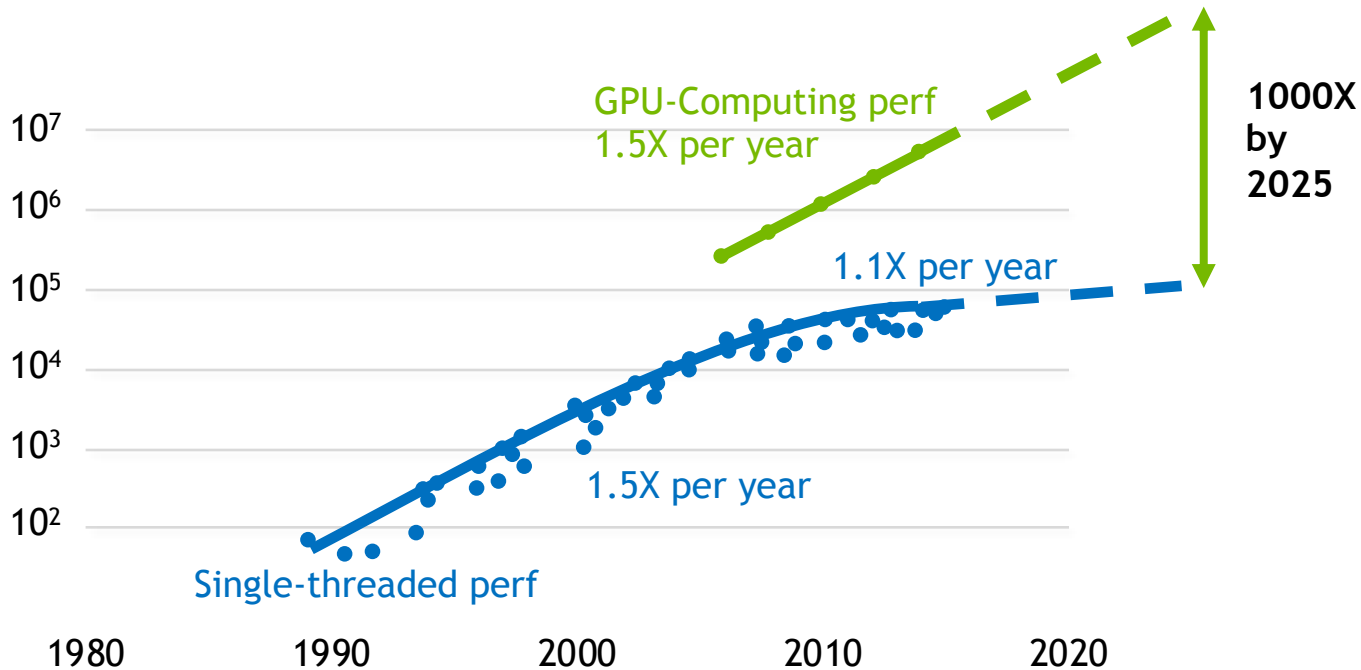
Oct 2019

AGENDA

- Introduction
- CUDA Development Ecosystem
- Mixed Precision Acceleration and Tensor Cores
- NGC

RISE OF GPU COMPUTING

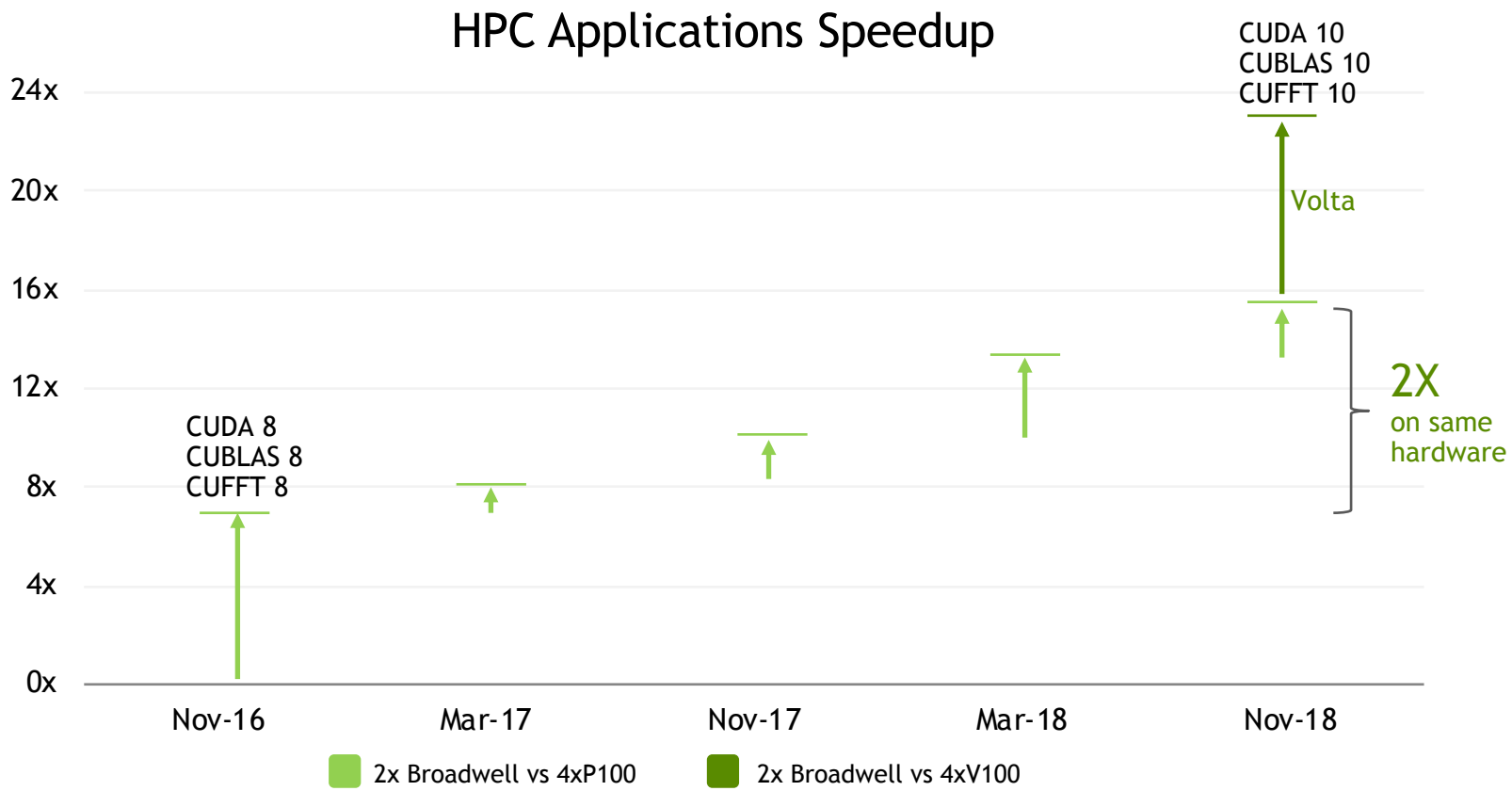




Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten New plot and data collected for 2010-2015 by K. Rupp

ACCELERATED COMPUTING IS FULL-STACK OPTIMIZATION

2X More Performance with Software Optimizations Alone

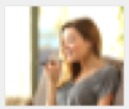


HPC Apps: AMBER, Chroma, GROMACS, GTC, LAMMPS, MILC, NAMD, QE, RTM, SPECFEM3D, VASP

NVIDIA DATA CENTER PLATFORM

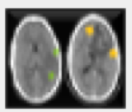
Single Platform Drives Utilization and Productivity

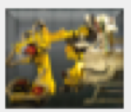
CUSTOMER USE CASES


Speech


Translate


Recommender


Healthcare


Manufacturing


Finance


Molecular Simulations


Weather Forecasting


Seismic Mapping


Creative & Technical


Knowledge Workers

CONSUMER INTERNET & INDUSTRY APPLICATIONS

SCIENTIFIC APPLICATIONS

VIRTUAL GRAPHICS

APPS & FRAMEWORKS

python · TensorFlow · mxnet · Chainer · ONNX · RAPIDS · PYTORCH

Amber · NAMD · +600 Applications

CATIA · Ps · AUTODESK 3DS MAX · Windows 10

CUDA-X & NVIDIA SDKs

MACHINE LEARNING: cuDF | cuML | cuGRAPH

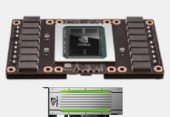
DEEP LEARNING: cuDNN | CUTLASS | TensorRT

HPC: OpenACC | cuFFT

VIRTUAL GPU: vDWS | vPC | vAPPS

CUDA & CORE LIBRARIES - cuBLAS | NCCL

TESLA GPUs & SYSTEMS


TESLA GPU


NVIDIA DGX FAMILY


NVIDIA HGX


EVERY OEM


EVERY MAJOR CLOUD

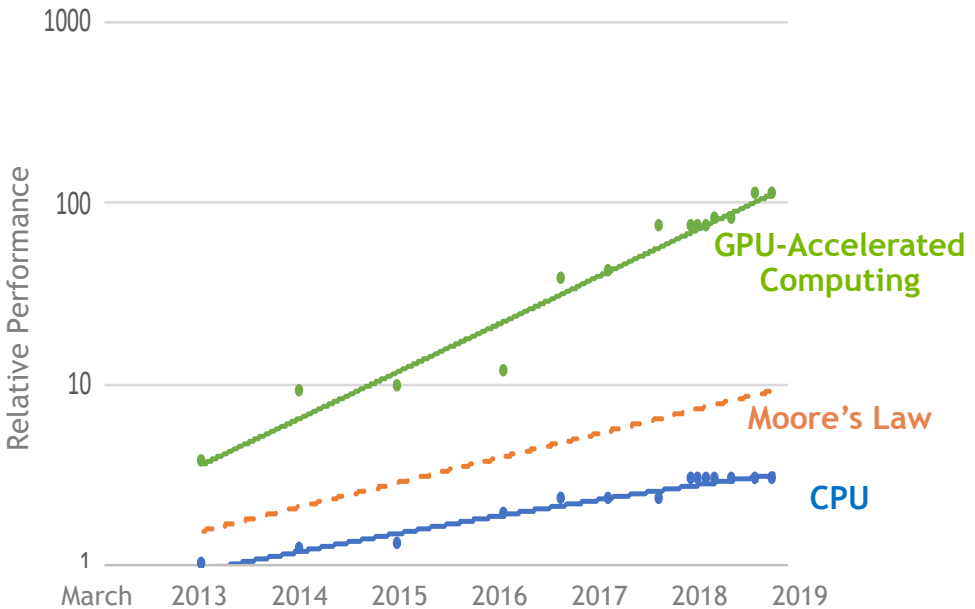
BEYOND MOORE'S LAW

Progress Of Stack In 6 Years

2013

- cuBLAS: 5.0
- cuFFT: 5.0
- cuRAND: 5.0
- cuSPARSE: 5.0
- NPP: 5.0
- Thrust: 1.5.3
- CUDA: 5.0
- Resource Mgr: r304
- Base OS: CentOS 6.2

Accelerated Server With Fermi



Measured performance of Amber, CHROMA, GTC, LAMMPS, MILC, NAMD, Quantum Espresso, SPECFEM3D

2019

- cuBLAS: 10.0
- cuFFT: 10.0
- cuRAND: 10.0
- cuSOLVER: 10.0
- cuSPARSE: 10.0
- NPP: 10.0
- Thrust: 1.9.0
- CUDA: 10.0
- Resource Mgr: r384
- Base OS: Ubuntu 16.04

Accelerated Server with Volta


CUDA DEVELOPMENT
ECOSYSTEM

CUDA DEVELOPMENT ECOSYSTEM

GPU Users

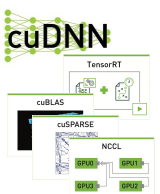
Domain Specialists

Problem Specialists

New Algorithm Developers and Optimization Experts

Applications

Frameworks



Libraries

Directives and Standard Languages

Extended Standard Languages

Ease of use ——— *Specialized Performance* →

CUDA: Programming Model, GPU Architecture, System Architecture

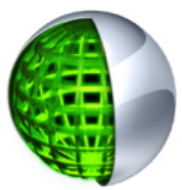
CUDA TOOLKIT

Libraries, Languages and Development Tools for GPU Computing

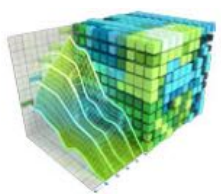
Programming Approaches

- **Libraries** — "Drop-in" Acceleration
- **Programming Languages** — Maximum Flexibility

Development Environment



Nsight IDE



NVIDIA Visual Profiler

CUDA Profiling Tools Interface

CUDA-GDB Debugger

CUDA MEMCHECK

Language Support

C C++ Fortran python

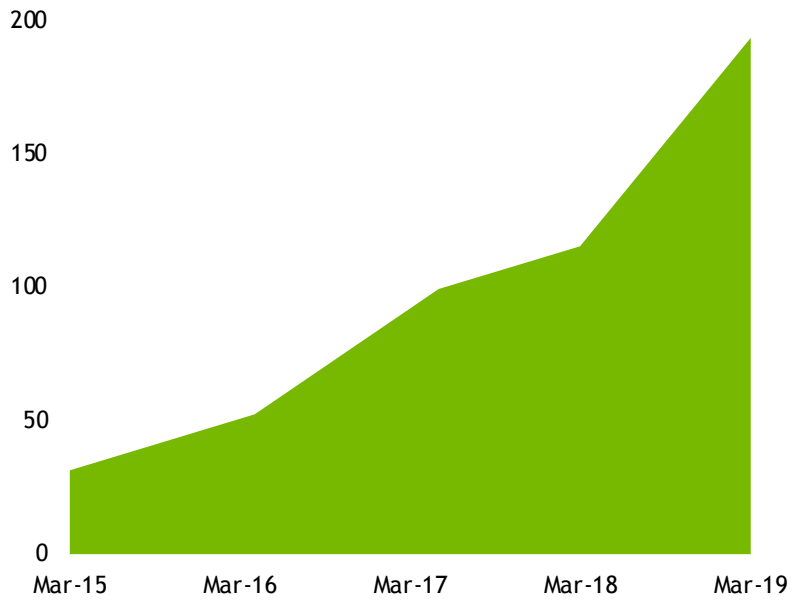
LLVM COMPILER INFRASTRUCTURE

Compile new languages to CUDA

RAPID ADOPTION OF OPENACC

Simple| Performant| Portable

ACROSS SCIENTIFIC DOMAINS



~200 APPS BEING ACCELERATED

DIRECTIVE-BASED HPC PROGRAMMING

Who's Using OpenACC?

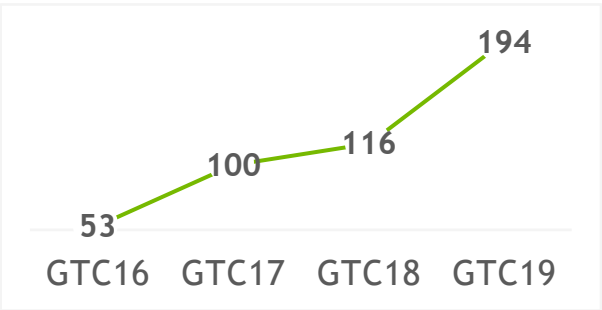
3 OF TOP 5 HPC APPS

Intersect360 RESEARCH

5 OF 13 CAAR CODES

summit

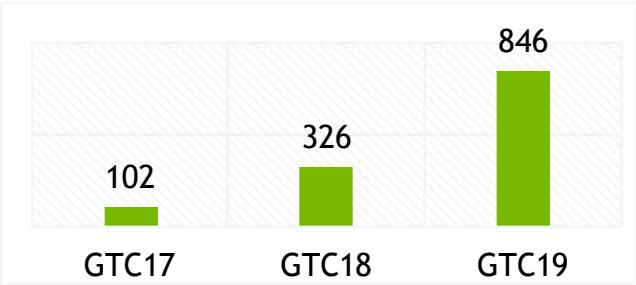
ACCELERATED APPS



725 TRAINED EXPERTS

OpenACC HACKATHON
Advancing High Performance Computing

SLACK MEMBERS



160,000+ DOWNLOADS

PGI®
Community EDITION

OPENMP OFFLOADING

NERSC

Powering Scientific Discovery Since 1974

Login
Site Map | My NERSC | Share
search...

HOME ABOUT SCIENCE AT NERSC SYSTEMS FOR USERS NEWS & PUBLICATIONS R & D EVENTS LIVE STATUS TIMELINE

NEWS & PUBLICATIONS

- News
  - Science News
  - Center News
    - NERSC 9 Perlmutter announcement
- Publications & Reports
- Journal Cover Stories
- Galleries
- Podcast
- User Announcements
- Staff Blogs

Home » News & Publications » News » Center News » NERSC, NVIDIA to Partner on Compiler Development for Perlmutter System

NERSC, NVIDIA TO PARTNER ON COMPILER DEVELOPMENT FOR PERLMUTTER SYSTEM

MARCH 21, 2019
Contact: Kathy Kincade, kkincade@lbl.gov, +1 510 495 2124

The National Energy Research Scientific Computing Center (NERSC) at Lawrence Berkeley National Laboratory (Berkeley Lab) has signed a contract with NVIDIA to enhance GPU compiler capabilities for Berkeley Lab's next-generation Perlmutter supercomputer.

Saul Perlmutter

In October 2018, the U.S. Department of Energy (DOE) announced that NERSC had signed a contract with Cray for a pre-exascale supercomputer named "Perlmutter," in honor of Berkeley Lab's Nobel Prize-winning astrophysicist Saul Perlmutter. The Cray Shasta machine, slated to be delivered in 2020, will be a heterogeneous system comprising both CPU-only and GPU-accelerated cabinets. It will include a new Cray system interconnect designed for data-centric computing; NVIDIA GPUs with new Tensor Core technology; CPU-only nodes based on next-generation AMD EPYC CPUs; direct liquid cooling; and an all-flash scratch filesystem that will move data at a rate of more than 4 terabytes/sec.

Under the new non-recurring engineering contract with NVIDIA, worth approximately $4 million, Berkeley Lab researchers will work with NVIDIA engineers to enhance the NVIDIA's PGI C, C++ and Fortran compilers to enable OpenMP applications to run on NVIDIA GPUs. This collaboration will help NERSC users, and the HPC community as a whole, efficiently port suitable applications to target GPU hardware in the Perlmutter system.

"We are excited to work with NVIDIA to enable OpenMP GPU computing using their PGI compilers," said Nick Wright, the

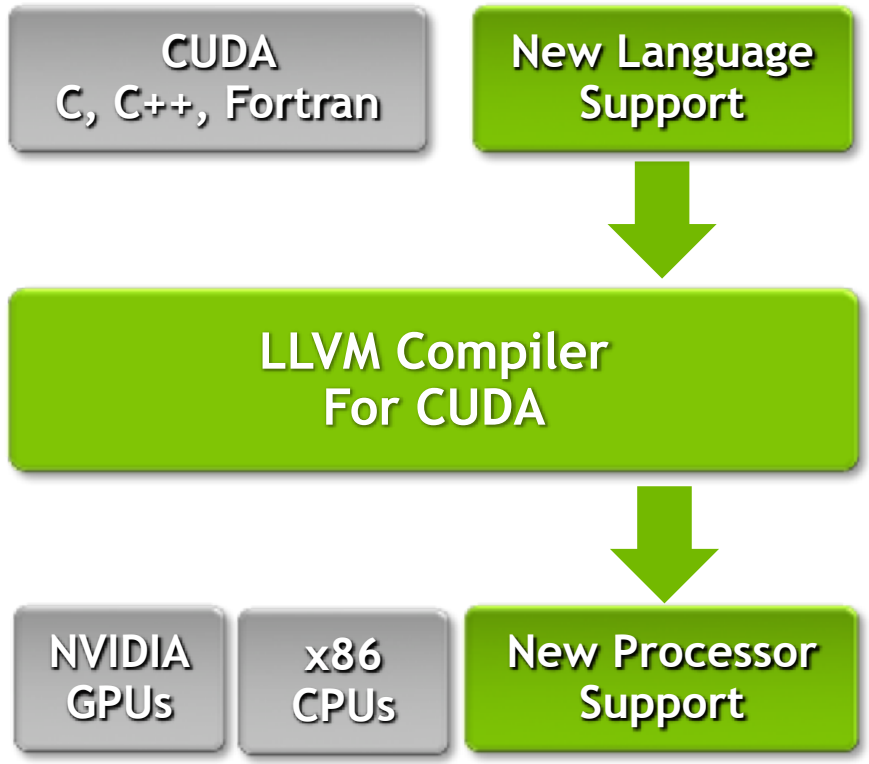
NEW LANGUAGES

Developers want to build front-ends for:

- Java, Python, R, DSLs

Target other processors like:

- ARM, FPGA, GPUs, x86



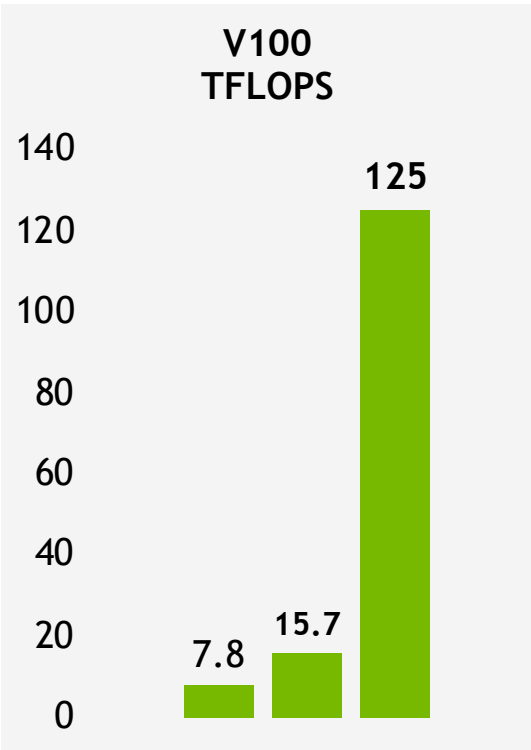
CUDA Compiler Contributed to Open Source LLVM

LLVM COMPILER INFRASTRUCTURE

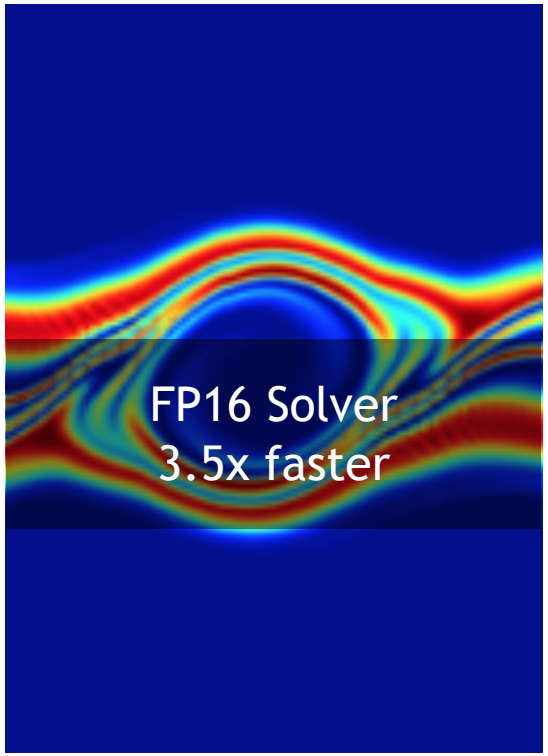

MIXED PRECISION
ACCELERATION AND
TENSOR CORES

TENSOR CORES FOR SCIENCE

Mixed-Precision Computing

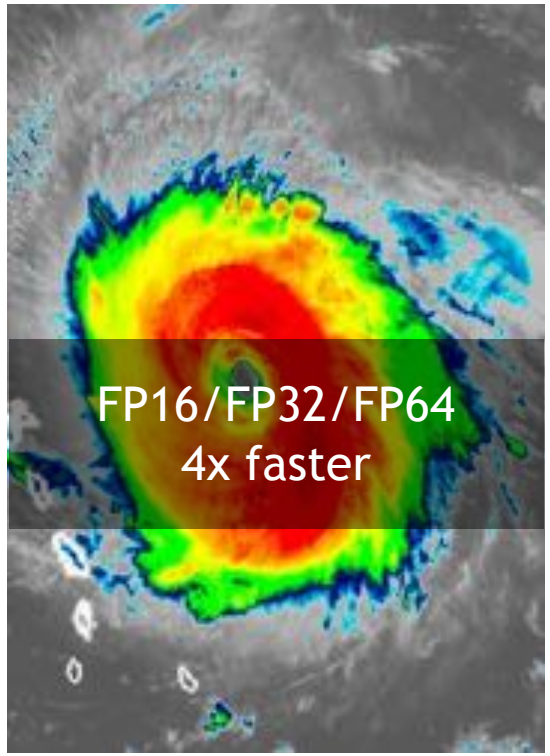


FP64+ MULTI-PRECISION



PLASMA FUSION APPLICATION

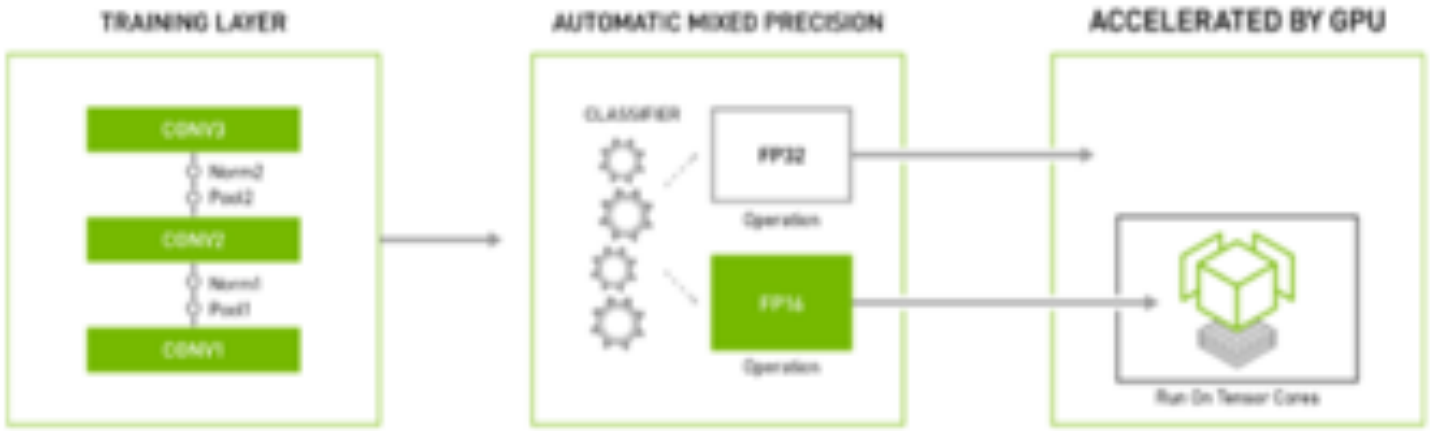
EARTHQUAKE SIMULATION



MIXED PRECISION WEATHER PREDICTION

AUTOMATIC MIXED PRECISION

Easy to Use, Greater Performance and Boost in Productivity



Insert ~ two lines of code to introduce Automatic Mixed-Precision and get upto 3X speedup

AMP uses a graph optimization technique to determine FP16 and FP32 operations

Support for TensorFlow, PyTorch and MXNet

Unleash Next-Generation AI Performance and Accelerate Time to ROI

ENABLING AUTOMATIC MIXED PRECISION

Add Just A Few Lines of Code, Get Upto 3X Speedup

TensorFlow	`os.environ['TF_ENABLE_AUTO_MIXED_PRECISION'] = '1'`
	OR
	`export TF_ENABLE_AUTO_MIXED_PRECISION=1`
	
	Explicit optimizer wrapper available inNVIDIA Container 19.07+, TF 1.14+, TF 2.0:
	
	`opt = tf.train.experimental.enable_mixed_precision_graph_rewrite(opt)`
		GA

PyTorch	`model, optimizer = amp.initialize(model, optimizer, opt level="O1")`
	`with amp.scale loss(loss, optimizer) as scaled_loss:`
	`scaled_loss.backward()`
		GA

MXNet	`amp.init()`
	`amp.init trainer(trainer)`
	`with amp.scale loss(loss, trainer) as scaled_loss:`
	`autograd.backward(scaled_loss)`
		GA
		Coming Soon

More details: https://developer.nvidia.com/automatic-mixed-precision

ENABLING AUTOMATIC MIXED PRECISION

Add Just A Few Lines of Code

- TensorFlow
  - NVIDIA container 19.03+:
    - **`export TF_ENABLE_AUTO_MIXED_PRECISION=1`** `[automatic casting` *and* `automatic loss scaling]`
  - Available in NVIDIA container 19.07+, TF 1.14+, TF 2.0:
    - We provide an explicit optimizer wrapper to perform loss scaling - which can also enable auto-casting for you:

```
import tensorflow as tf

opt = tf.train.GradientDescentOptimizer(0.5)

opt = tf.train.experimental.enable_mixed_precision_graph_rewrite(opt)
```

ENABLING AUTOMATIC MIXED PRECISION

Add Just A Few Lines of Code

- PyTorch
  - Two steps: initialization and wrapping backpropagation

```
from apex import amp
model = …
optimizer = SomeOptimizer(model.parameters(), …)
# …
model, optimizer = amp.initialize(model, optimizer, opt_level="O1")
# …
for train_loop():
  loss = loss_fn(model(x), y)
  with amp.scale_loss(loss, optimizer) as scaled_loss:
    scaled_loss.backward()
  # Can manipulate the .grads if you'd like
  optimizer.step()
```

ENABLING AUTOMATIC MIXED PRECISION

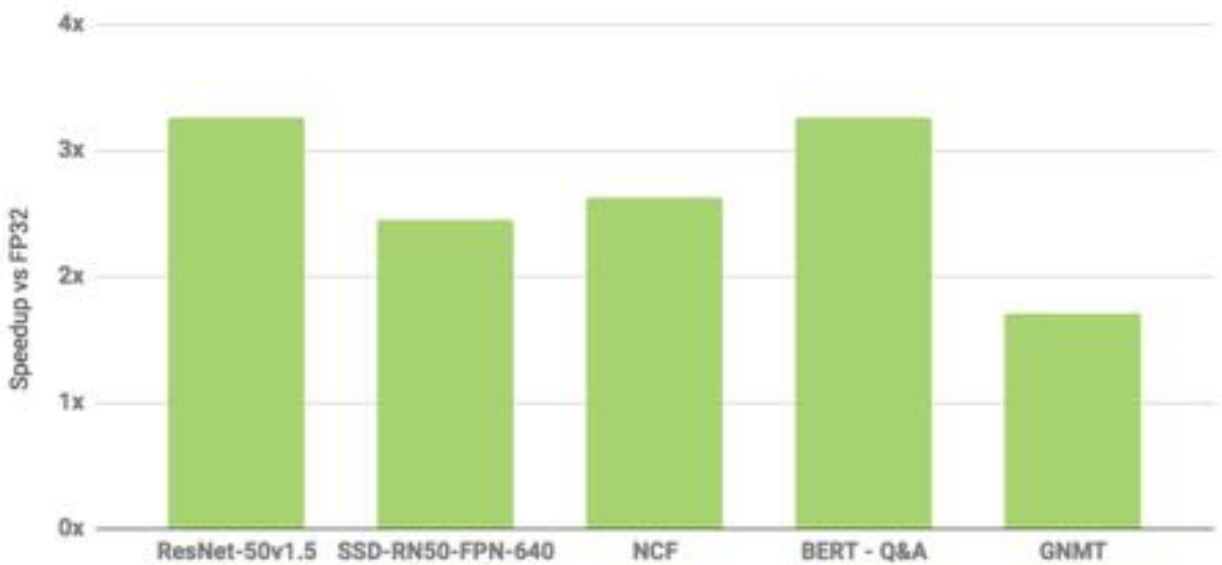
Add Just A Few Lines of Code

- MXNET
  - NVIDIA container 19.03+ and MXNET 1.5:

```
from mxnet.contrib import amp
amp.init()
net = get_network()
trainer = mx.gluon.Trainer(...)
amp.init_trainer(trainer)
for data in dataloader:
    with autograd.record(True):
        out = net(data)
        l = loss(out)
        with amp.scale_loss(l, trainer) as scaled_loss:
            autograd.backward(scaled_loss)
    trainer.step()
```

AUTOMATIC MIXED PRECISION IN TENSORFLOW

Upto 3X Speedup



TensorFlow Medium Post: Automatic Mixed Precision in TensorFlow for Faster AI Training on NVIDIA GPUs

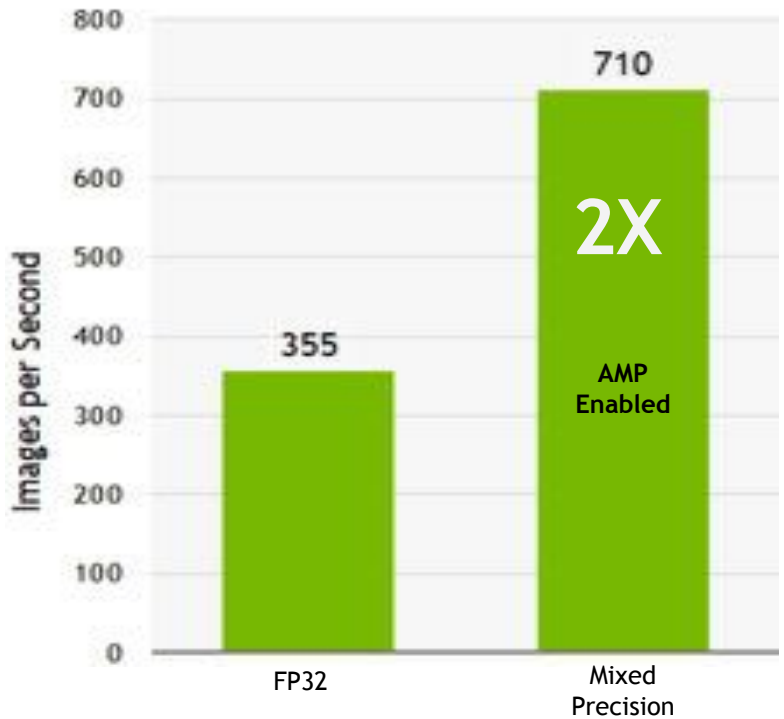
All models can be found at:
https://github.com/NVIDIA/DeepLearningExamples/tree/master/TensorFlow, except for ssd-rn50-fpn-640, which is here: https://github.com/tensorflow/models/tree/master/research/object_detection
All performance collected on 1xV100-16GB, except bert-squadqa on 1xV100-32GB.
Speedup is the ratio of time to train for a fixed number of epochs in single-precision and Automatic Mixed Precision. Number of epochs for each model was matching the literature or common practice (it was also confirmed that both training sessions achieved the same model accuracy).
Batch sizes:. rn50 (v1.5): 128 for FP32, 256 for AMP+XLA; ssd-rn50-fpn-640: 8 for FP32, 16 for AMP+XLA; NCF: 1M for FP32 and AMP+XLA; bert-squadqa: 4 for FP32, 10 for AMP+XLA; GNMT: 128 for FP32, 192 for AMP.

AUTOMATIC MIXED PRECISION IN PYTORCH

https://developer.nvidia.com/automatic-mixed-precision

- Plot shows ResNet-50 result with/without automatic mixed precision(AMP)
- More AMP enabled model scripts coming soon:

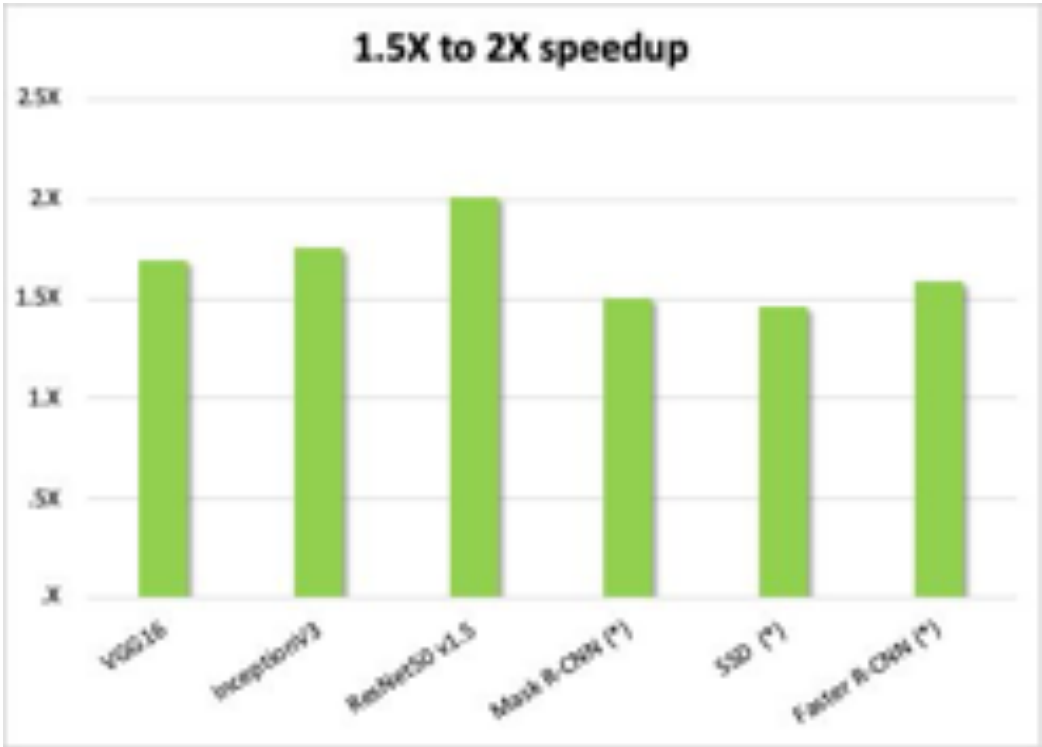
  Mask-R CNN, GNMT, NCF, etc.



Source: https://github.com/NVIDIA/apex/tree/master/examples/imagenet

AUTOMATIC MIXED PRECISION IN MXNET

AMP speedup ~1.5X to 2X in comparison with FP32



(*) based on ResNet50 v1.5

https://github.com/apache/incubator-mxnet/pull/14173

Matching Accuracy for FP32 and Mixed Precision

Model Script	Framework	Data Set	Automatic or Manual Mixed-Precision	FP32 Accuracy	Mixed-Precision Accuracy	FP32 Throughput	Mixed-Precision Throughput	Speedup
BERT Q&A (2)	**TensorFlow**	SQuaD	AMP	90.83 Top 1	90.99 Top 1	66.65 sentences/sec	129.16 sentences/sec	**1.94**
SSD w/RN50 (1)	**TensorFlow**	COCO 2017	AMP	0.268 mAP	0.269 mAP	569 images/sec	752 images/sec	**1.32**
GNMT (3)	**PyTorch**	WMT16 English to German	Manual	24.16 BLEU	24.22 BLEU	314,831 tokens/sec	738,521 tokens/sec	**2.35**
Neural Collaborative Filter (1)	**PyTorch**	MovieLens 20M	Manual	0.959 HR	0.960 HR	55,004,590 samples/sec	99,332,230 items/sec	**1.81**
U-Net Industrial (1)	**TensorFlow**	DAGM 2007	AMP	0.965-0.988	0.960-0.988	445 images/sec	491 images/sec	**1.10**
ResNet-50 v1.5 (1)	**MXNet**	ImageNet	Manual	76.67 Top 1%	76.49 Top 1%	2,957 images/sec	10,263 images/sec	**3.47**
Tacotron 2 / WaveGlow 1.0 (1)	**PyTorch**	LJ Speech Dataset	AMP	0.3629/ -6.1087	0.3645/ -6.0258	10,843 tok/s 257,687 smp/s	12,742 tok/s 500,375 smp/s	**1.18/ 1.94**

Values are measured with model running on (1) **DGX-1V 8GPU 16G,** (2) **DGX-1V 8GPU 32G or** (3) **DGX-2V 16GPU 32G**


NGC

NGC: GPU-OPTIMIZED SOFTWARE HUB

Ready-to-run GPU Optimized Software, Anywhere

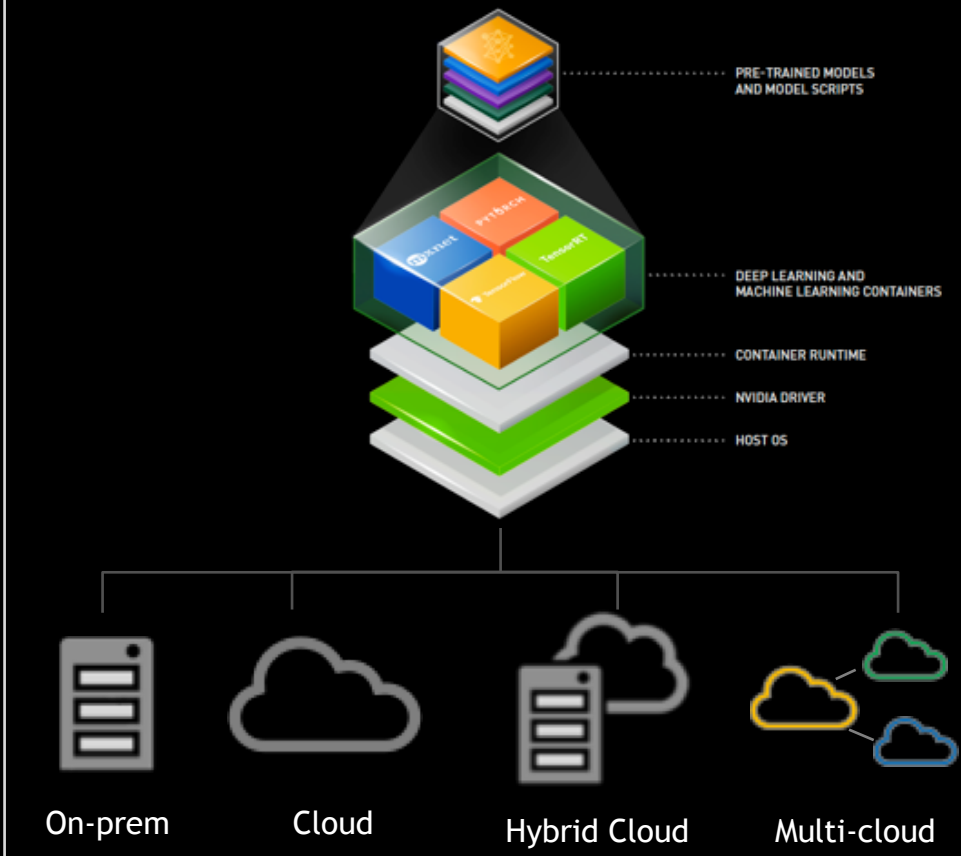
50+ Containers
DL, ML, HPC

15+ Model Training Scripts
NLP, Image Classification, Object Detection & more

NGC

60 Pre-trained Models
NLP, Image Classification, Object Detection & more

Industry Workflows
Medical Imaging, Intelligent Video Analytics



NGC MODEL SCRIPTS

Tensor Core Optimized Deep Learning Examples

18 Available

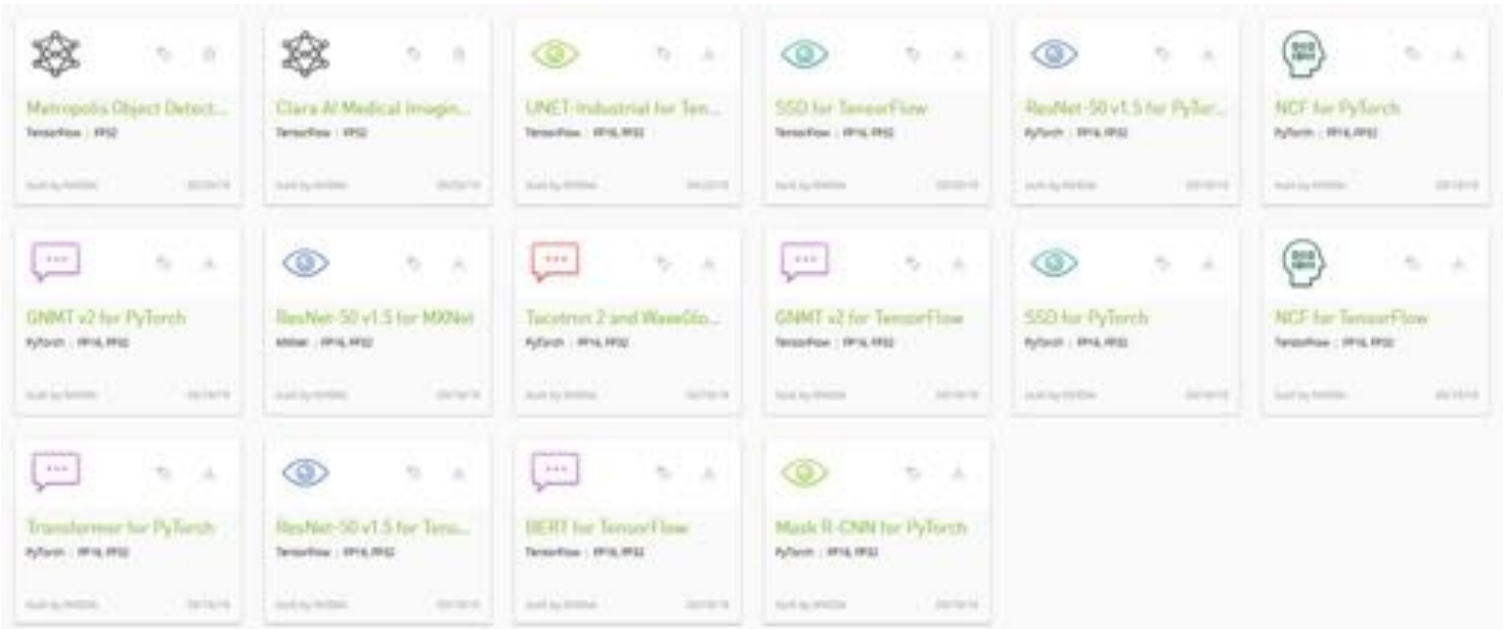
- Tensor Core optimized for greater performance
- Test drive automatic mixed precision
- Actively updated by NVIDIA
- State-of-the-art accuracy using Tensor Cores
- Serves as a reference implementation
- Exposes hyperparameters and source code for further adjustment

Accessible via:

- NVIDIA NGC https://ngc.nvidia.com/catalog/model-scripts
- GitHub https://www.github.com/NVIDIA/deeplearningexamples
- NVIDIA NGC Framework containers https://ngc.nvidia.com/catalog/containers

NVIDIA NGC MODEL SCRIPTS

Tensor Core Examples Built for Multiple Use Cases and Frameworks



A dedicated hub to download Tensor Core Optimized Deep Learning Examples on NGC

https://ngc.nvidia.com/catalog/model-scripts?quickFilter=deep-learning

NGC PRE-TRAINED MODEL FILES AVAILABLE

- Published for multiple frameworks: TensorRT, TensorFlow, PyTorch, and MXNet.
- Trained on multiple industry data sets: ImageNet, MSCOCO, LibreSpeech, Wikipedia/BookCorpus, etc.
- Provided in the most popular precisions: FP32, FP16, and INT8
- Key customer benefits:
  - Quick experimentation with our trained models to understand inference performance/latency
  - Building blocks for model ensembling pipelines, such as Speech AI: ASR->BERT Q&A->TTS
  - Reproducibility of our inference benchmarks

AGENDA

- Introduction
- CUDA Development Ecosystem
- Mixed Precision Acceleration and Tensor Cores
- NGC


NVIDIA