

# S9500 - Deep Learning Framework Container Optimizations

Joey Conway, Senior Product Manager of Deep Learning Software  
Michael O'Connor, Director of Software, Optimized Frameworks  
Cliff Woolley, Director of Engineering, Optimized Frameworks



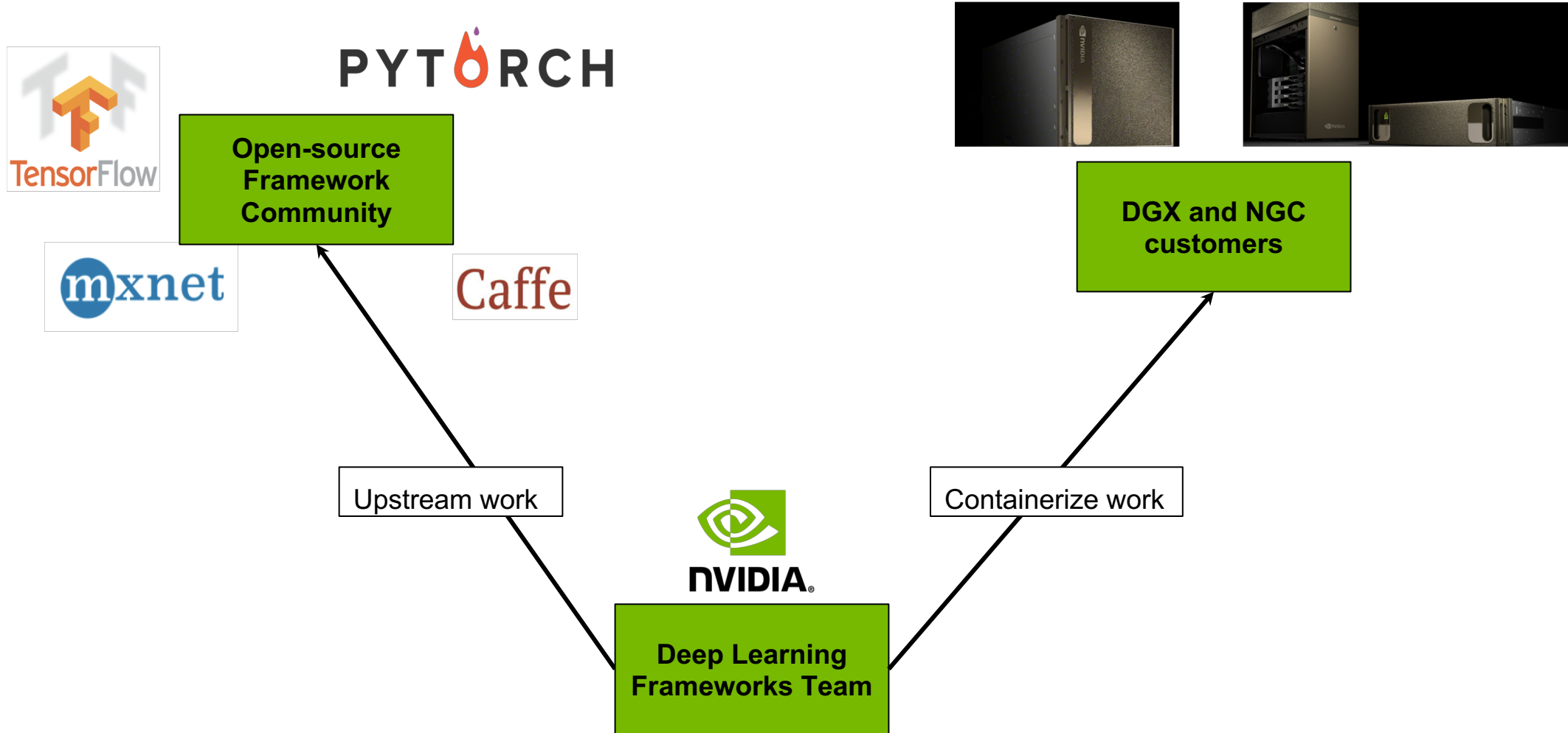
# AGENDA

## Deep Learning Framework Container Highlights

- Deep Learning Framework Team
- Overview
  - Best Performance
  - Latest Features
  - Best Practices
- Additional Resources

# NVIDIA Deep Learning Frameworks Team

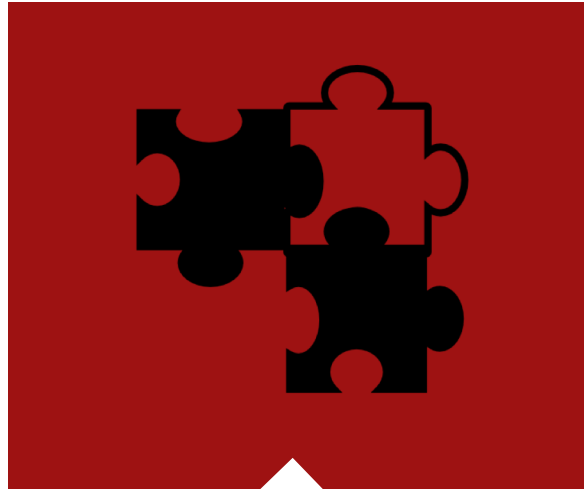
Overview of community interactions



# Challenges with Deep Learning



Performance



Version  
Compatibility



Resources &  
Best Practices

# Deep Learning Frameworks Highlights

## Best NVIDIA Performance

Deep Learning Frameworks optimizations for NVIDIA hardware

Volta Tensor Cores support for mixed-precision (FP16) across TensorFlow, MXNet, PyTorch, and NVCaffe

## Latest NVIDIA Features

Latest NVIDIA Deep Learning libraries incorporated cuDNN, cuBLAS, and NCCL

Automatic Mixed-Precision for TensorFlow, PyTorch and MXNet

## Best Practices & QA Verified

Improved documentation with best practices and monthly release notes

Thorough monthly quality assurance testing

Multi-node support updated

# Deep Learning Frameworks Highlights

## Best NVIDIA Performance

Deep Learning Frameworks optimizations for NVIDIA hardware

Volta Tensor Cores support for mixed-precision (FP16) across TensorFlow, MXNet, PyTorch, and NVCaffe

## Latest NVIDIA Features

Latest NVIDIA Deep Learning libraries incorporated cuDNN, cuBLAS, and NCCL

Automatic Mixed-Precision for TensorFlow, PyTorch and MXNet

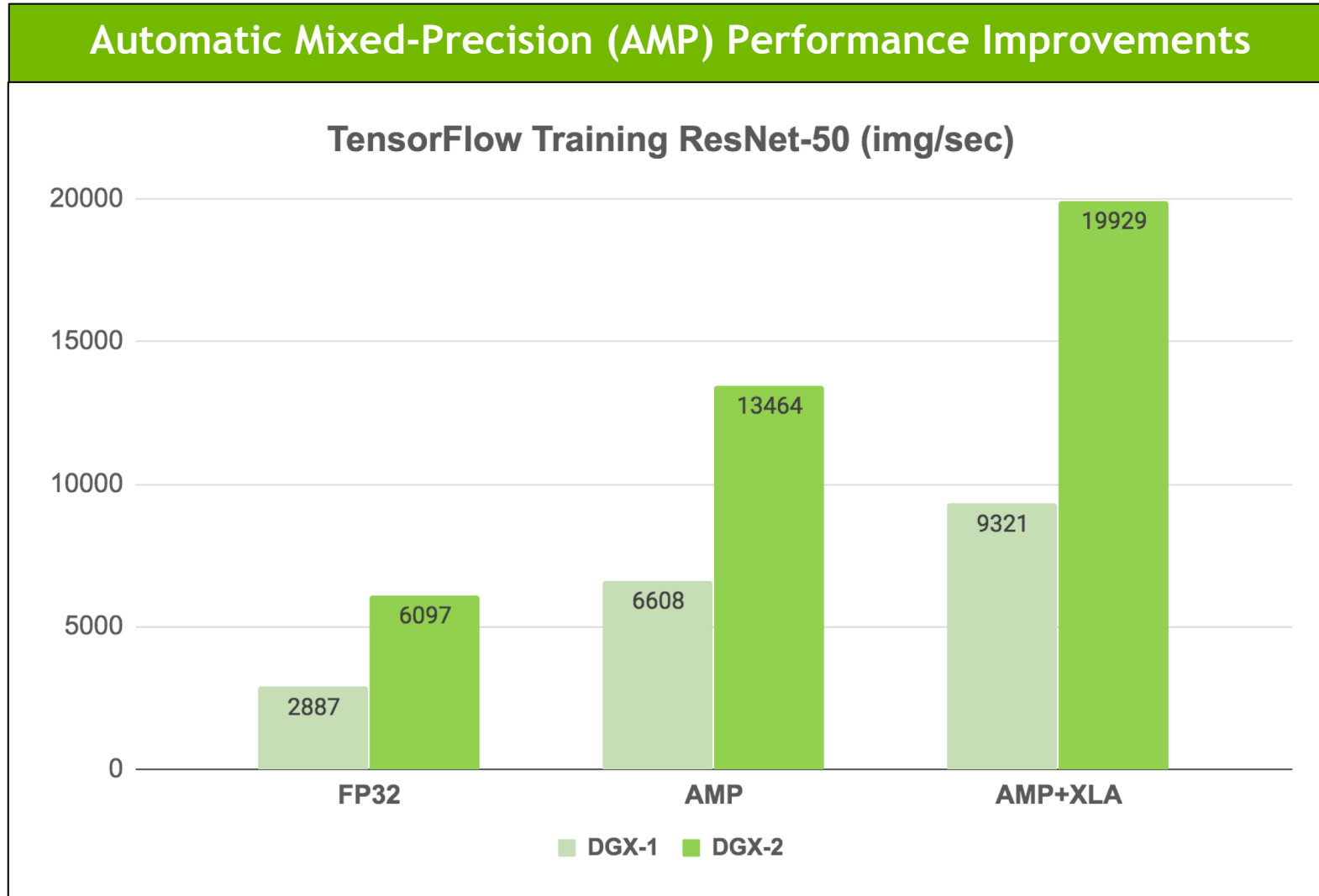
## Best Practices & QA Verified

Improved documentation with best practices and monthly release notes

Thorough monthly quality assurance testing

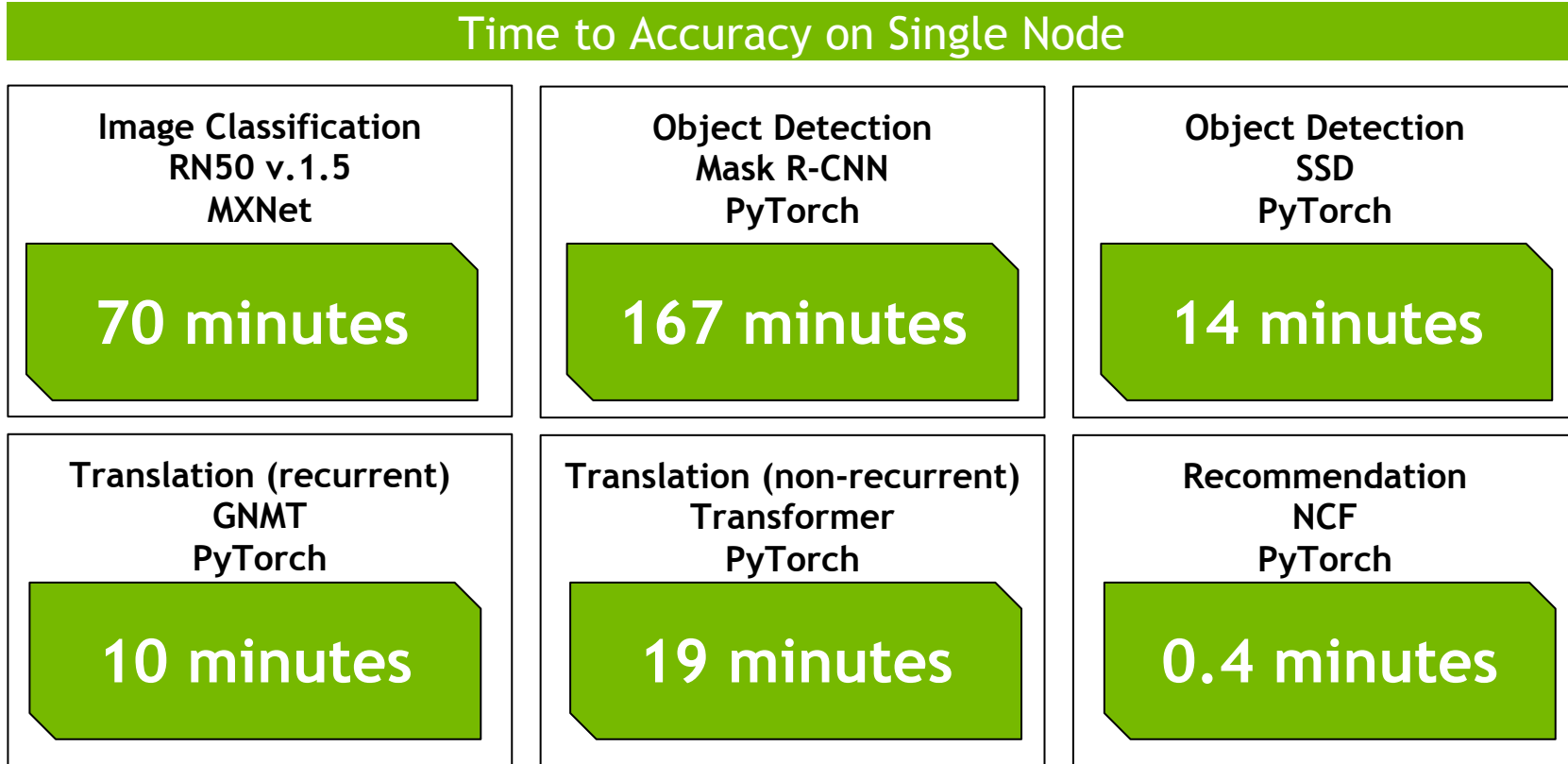
Multi-node support updated

# TensorFlow Performance on ResNet-50 with DGX



# DGX Mixed-Precision Led MLPerf

World's Fastest Industry-Wide AI Benchmark Achieved on NVIDIA GPUs



Test Platform: DGX-2H - Dual-Socket Xeon Platinum 8174, 1.5TB system RAM, 16 x 32 GB Tesla V100 SXM-3 GPUs connected via NVSwitch



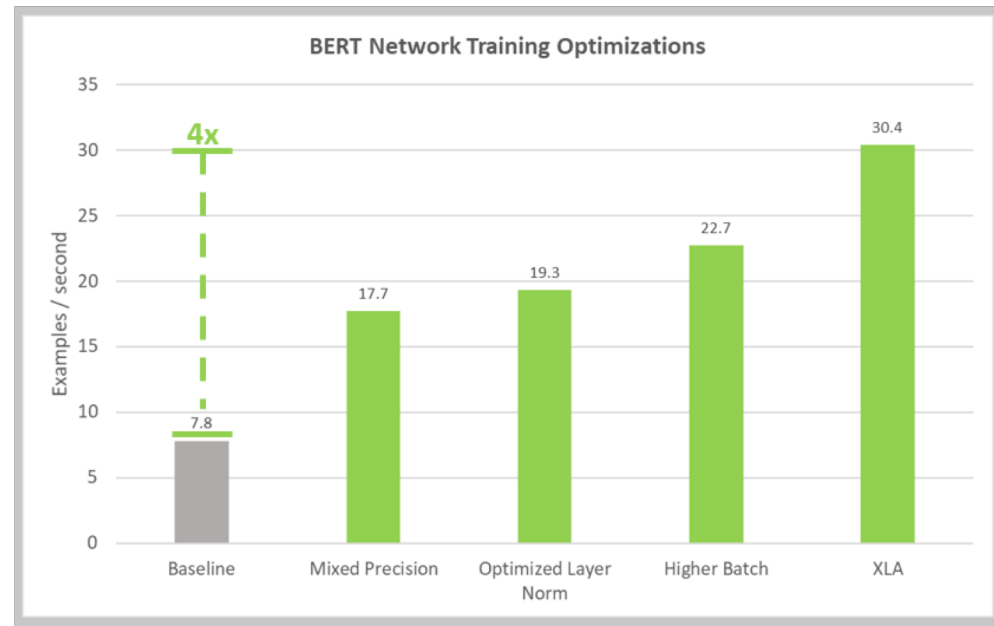
# BERT

Performance improvements from MLPerf Transformer carries over to BERT

- State-of-the-art model for NLP tasks
- Compute intensive Transformer-like workload
- Optimizations from MLPerf carry over in both PyTorch and TF
- TF Training scripts released here:

<https://github.com/NVIDIA/DeepLearningExamples/tree/master/TensorFlow/LanguageModeling/BERT>

- Pretraining (Wikipedia)
- Q&A fine-tuning (SQuAD)
- Mixed Precision using Tensor Cores



# Tensor Core Examples: Developer Page

<https://developer.nvidia.com/deep-learning-examples>

## New Deep Learning Training Scripts

- Tensor Core optimized performance
- State-of-the-art accuracy using Tensor Cores

## Serve as a quick start guide

- How we implemented mixed-precision
- Exposing hyperparameters for further adjustment

## Code examples on

- GitHub  
<https://www.github.com/NVIDIA/deeplearningexamples>
- NGC DL Framework containers
- NGC Model Scripts registry

<https://developer.nvidia.com/deep-learning-examples>

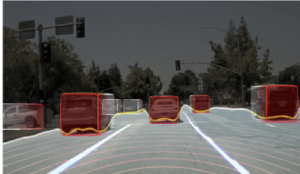
## Models by Application Areas

Click on the application area to jump directly to that section:

[Computer Vision >](#) [Speech and NLP >](#) [Recommender Systems >](#) [Generative Adversarial Networks >](#)

### Computer Vision

Computer vision deals with algorithms and techniques for computers to understand the world around us using image and video data or in other words, teaching machines to automate the tasks performed by human visual systems. Common computer vision tasks include image classification, object detection in images and videos, image segmentation, and image restoration. In recent years, deep learning has revolutionized the field of computer vision with algorithms that deliver super-human accuracy on the above tasks. Below is a list of popular deep neural network models used in computer vision and their open-source implementation.



**ResNet50:** Residual network architecture introduced "skip connections" and won the 1st place on the ILSVRC 2015 classification task

- [\[TensorFlow\]](#)

**Inception v3:** Version 3 of the Inception architecture, which was the winning architecture of the ILSVRC 2014 classification task. It introduced the inception module to drastically reduce the number of parameters in the network.

- [\[TensorFlow\]](#)

**VGG16/19:** Runner-up at the ILSVRC 2014 classification task.

- [\[TensorFlow\]](#)

**LeNet:** Image classification network used to recognize hand-written digits. LeNet is the first successful applications of Convolutional Neural Networks, developed by Yann LeCun.

- [\[Caffe2\]](#)

**CIFAR10:** CIFAR10 is a dataset of images with 10 classes. This model architecture, based on AlexNet, is designed to achieve good accuracy (not state-of-the-art) and can be used as a starting point to experiment alternate approaches.

- [\[Caffe2\]](#)

**FastPhotoStyle:** Fast photorealistic style transfer network. Takes as input a content photo and a style photo, and transfers the style of the style photo to the

# Tensor Core Examples: Developer Page

<https://developer.nvidia.com/deep-learning-examples>

## Available model training scripts

- Image Classification
  - ResNet-50v1.5
- Object Detection:
  - SSD with RN50
  - Mask R-CNN with RN50
- Translation
  - GNMT
  - Transformer
- Recommender
  - NCF
- Text-to-Speech
  - Tacotron2 and Waveglow

The screenshot shows the GitHub repository page for NVIDIA/DeepLearningExamples. The repository has 52 stars, 327 forks, and 113 issues. It contains 29 commits, 2 branches, 0 releases, and 8 contributors. The repository is currently on the master branch. A recent pull request #11 by nvpstr is merged, adding MNIST and CIFAR10 to the Caffe2/Classification folder. Other recent commits include adding ResNet50v1.5 to the MxNet/Classification folder, updating the NCF README, and cleaning up the TensorFlow imagenet readme.

Commit	Message	Time
nvpstr Merge pull request #11 from tgre/master	Adding MNIST, CIFAR10	11 months ago
MxNet/Classification/RN50v1.5	Adding ResNet50v1.5 to MxNet/Classification	2 months ago
PyTorch	update NCF README	24 days ago
TensorFlow	Cleanup TF imagenet readme.	11 months ago
.gitmodules	Add TensorFlow examples	11 months ago
README.md	Update README.md	8 months ago

# PyTorch GNMT Performance

<https://developer.nvidia.com/deep-learning-examples>

**DGX-1V 16G**

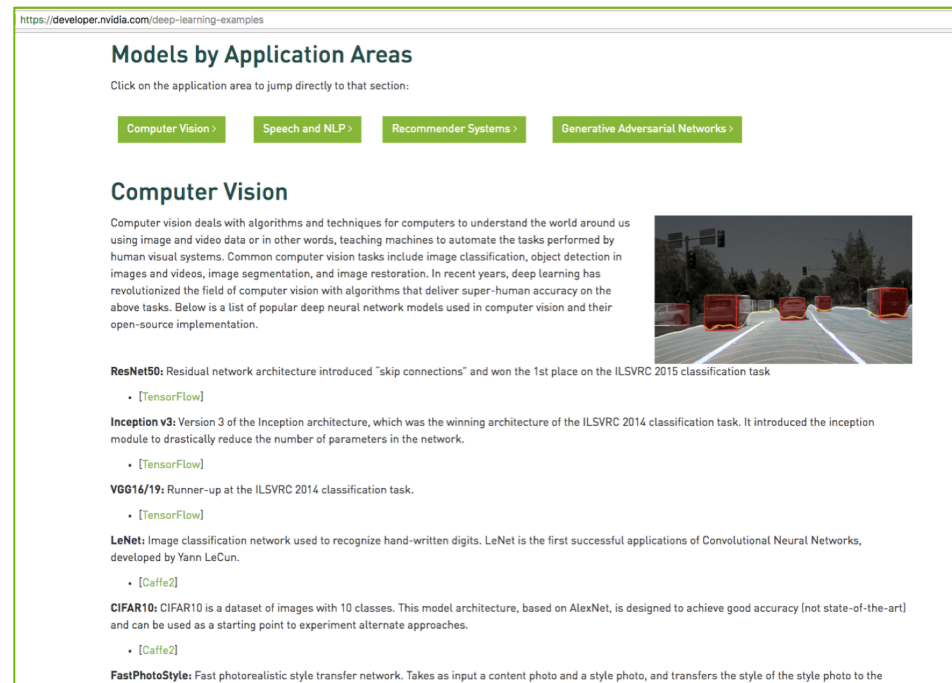
Time to Accuracy: 46 minutes

BLEU score (accuracy): 24.45

Tokens per second: 387,282

Data set: WMT16 English to German

NGC 19.01 PyTorch container



<https://developer.nvidia.com/deep-learning-examples>

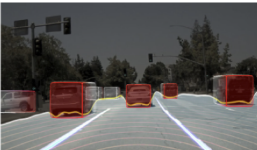
## Models by Application Areas

Click on the application area to jump directly to that section:

- Computer Vision >
- Speech and NLP >
- Recommender Systems >
- Generative Adversarial Networks >

### Computer Vision

Computer vision deals with algorithms and techniques for computers to understand the world around us using image and video data or in other words, teaching machines to automate the tasks performed by human visual systems. Common computer vision tasks include image classification, object detection in images and videos, image segmentation, and image restoration. In recent years, deep learning has revolutionized the field of computer vision with algorithms that deliver super-human accuracy on the above tasks. Below is a list of popular deep neural network models used in computer vision and their open-source implementation.



- ResNet50**: Residual network architecture introduced "skip connections" and won the 1st place on the ILSVRC 2015 classification task
  - [\[TensorFlow\]](#)
- Inception v3**: Version 3 of the Inception architecture, which was the winning architecture of the ILSVRC 2014 classification task. It introduced the inception module to drastically reduce the number of parameters in the network.
  - [\[TensorFlow\]](#)
- VGG16/19**: Runner-up at the ILSVRC 2014 classification task.
  - [\[TensorFlow\]](#)
- LeNet**: Image classification network used to recognize hand-written digits. LeNet is the first successful applications of Convolutional Neural Networks, developed by Yann LeCun.
  - [\[Caffe2\]](#)
- CIFAR10**: CIFAR10 is a dataset of images with 10 classes. This model architecture, based on AlexNet, is designed to achieve good accuracy (not state-of-the-art) and can be used as a starting point to experiment alternate approaches.
  - [\[Caffe2\]](#)
- FastPhotoStyle**: Fast photorealistic style transfer network. Takes as input a content photo and a style photo, and transfers the style of the style photo to the

# PyTorch GNMT Performance

<https://developer.nvidia.com/deep-learning-examples>

**DGX-2 32G**

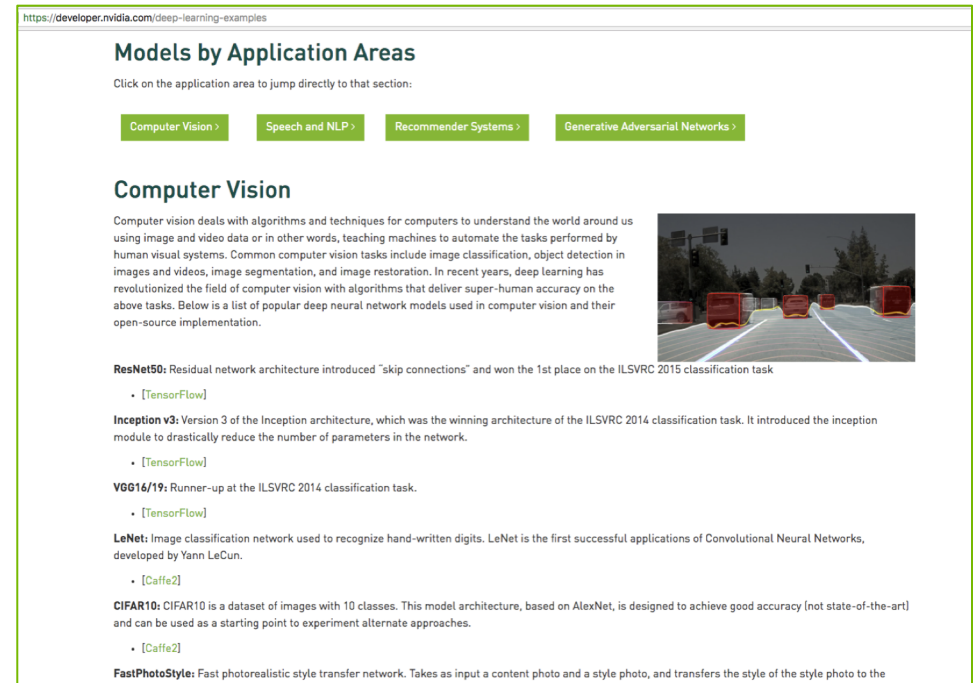
Time to Accuracy: 26.3 minutes

BLEU score (accuracy): 24.22

Tokens per second: 738,521

Data set: WMT16 English to German

NGC 19.01 PyTorch container



The screenshot shows the 'Models by Application Areas' page on the NVIDIA developer website. It features a navigation bar with buttons for 'Computer Vision', 'Speech and NLP', 'Recommender Systems', and 'Generative Adversarial Networks'. The 'Computer Vision' section is active, displaying a list of models with brief descriptions and links to their respective GitHub repositories. An image of a road with orange traffic cones is visible on the right side of the page.

<https://developer.nvidia.com/deep-learning-examples>

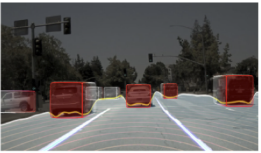
## Models by Application Areas

Click on the application area to jump directly to that section:

- Computer Vision >
- Speech and NLP >
- Recommender Systems >
- Generative Adversarial Networks >

### Computer Vision

Computer vision deals with algorithms and techniques for computers to understand the world around us using image and video data or in other words, teaching machines to automate the tasks performed by human visual systems. Common computer vision tasks include image classification, object detection in images and videos, image segmentation, and image restoration. In recent years, deep learning has revolutionized the field of computer vision with algorithms that deliver super-human accuracy on the above tasks. Below is a list of popular deep neural network models used in computer vision and their open-source implementation.



**ResNet50:** Residual network architecture introduced "skip connections" and won the 1st place on the ILSVRC 2015 classification task

- [\[TensorFlow\]](#)

**Inception v3:** Version 3 of the Inception architecture, which was the winning architecture of the ILSVRC 2014 classification task. It introduced the inception module to drastically reduce the number of parameters in the network.

- [\[TensorFlow\]](#)

**VGG16/19:** Runner-up at the ILSVRC 2014 classification task.

- [\[TensorFlow\]](#)

**LeNet:** Image classification network used to recognize hand-written digits. LeNet is the first successful applications of Convolutional Neural Networks, developed by Yann LeCun.

- [\[Caffe2\]](#)

**CIFAR10:** CIFAR10 is a dataset of images with 10 classes. This model architecture, based on AlexNet, is designed to achieve good accuracy (not state-of-the-art) and can be used as a starting point to experiment alternate approaches.

- [\[Caffe2\]](#)

**FastPhotoStyle:** Fast photorealistic style transfer network. Takes as input a content photo and a style photo, and transfers the style of the style photo to the

# MXNet RN50 Performance

<https://developer.nvidia.com/deep-learning-examples>

## DGX-1V 16G

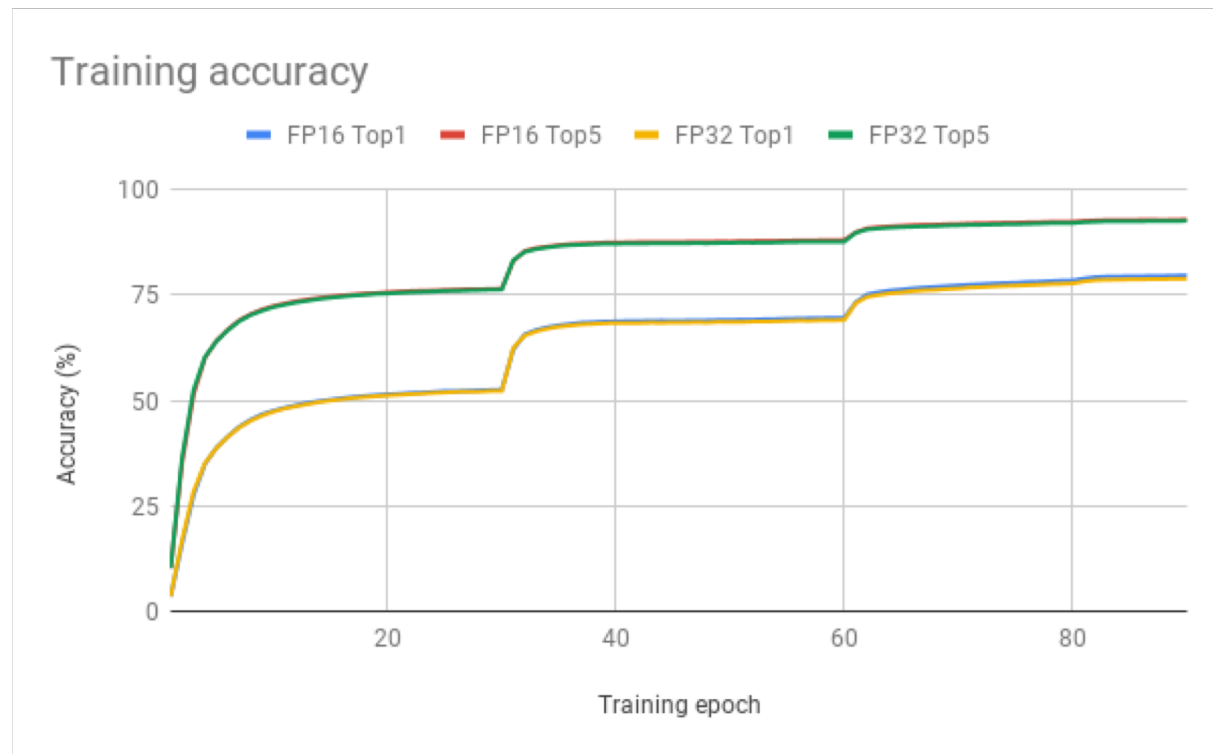
Time to Train: 3.3 hours

Top 1% (accuracy): 76.49

Images per second: 10,263

Data set: ImageNet

NGC 18.12 MXNet container



# PyTorch NCF Performance

<https://developer.nvidia.com/deep-learning-examples>

**DGX-1V 16G**

Time to Accuracy: < 1 minute

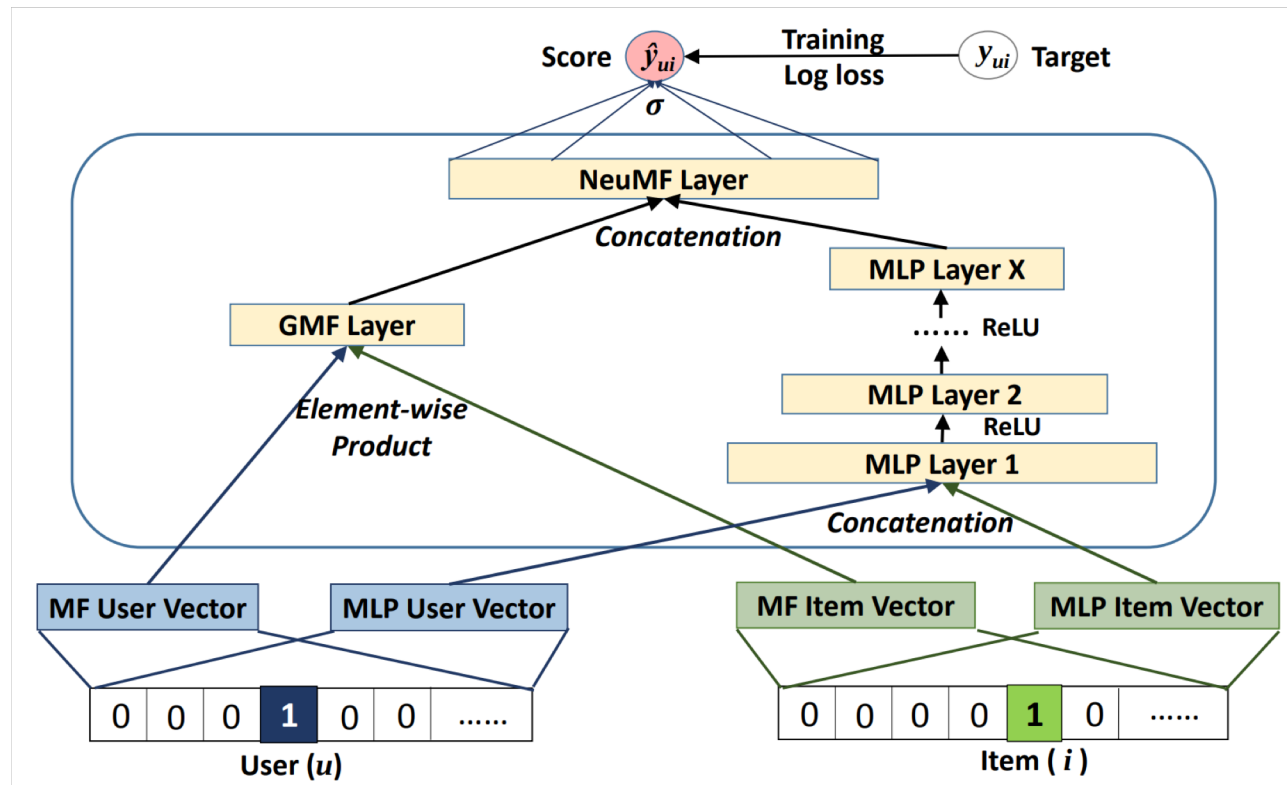
Hit Rate at 10 (accuracy): 0.96

Samples per second:

99,332,230

Data set: MovieLens 20M

NGC 18.12 PyTorch container



# Tensor Core Examples: Coming next

## Top Use Cases

- Adding existing models to more frameworks
- Optimizing more models for Tensor Cores
- Releasing externally and maintaining

More efforts in-progress!

## Top Use Cases

Classification

Object detection

Segmentation: Medical Imaging

Segmentation: Manufacturing

Audio speech recognition (ASR)

Text to speech (TTS)

Natural Language Processing (NLP)

Recommendation System



# Deep Learning Frameworks Highlights

## Best NVIDIA Performance

Deep Learning Frameworks optimizations for NVIDIA hardware

Volta Tensor Cores support for mixed-precision (FP16) across TensorFlow, MXNet, PyTorch, and NVCaffe

## Latest NVIDIA Features

Latest NVIDIA Deep Learning libraries incorporated cuDNN, cuBLAS, and NCCL

Automatic Mixed-Precision for TensorFlow, PyTorch and MXNet

## Best Practices & QA Verified

Improved documentation with best practices and monthly release notes

Thorough monthly quality assurance testing

Multi-node support updated

# Latest NVIDIA Features

## MXNet

- Multi-node support w/ Horovod
- NHWC support
- MXNet-AMP
- Mixed Precision support to TensorRT

## PyTorch

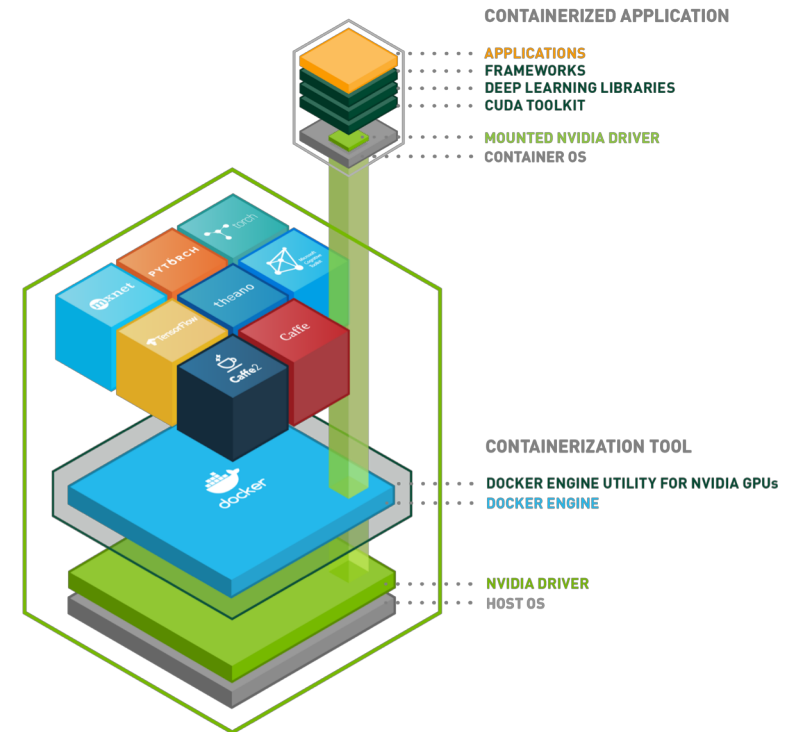
- PyTorch-AMP: unified mixed precision interface
- Automatic fusion for elementwise ops

## TensorFlow

- TensorFlow-AMP
- More TensorRT op coverage
- Added cuDNN RNN features
- Jetson releases

## Overall

- Mixed Precision tools
- Tensor Core optimized examples with trained models
- TensorRT integration
- Added Jupyter & JupyterLab



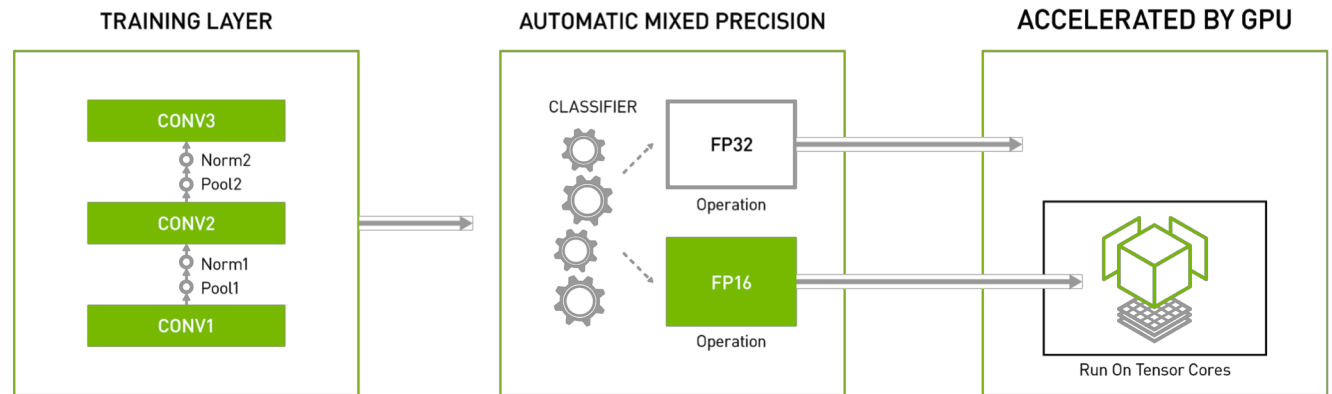
# AUTOMATIC MIXED PRECISION (AMP)

## Utilize Tensor Cores for Mixed Precision Training

Insert two lines of code to introduce Automatic Mixed-Precision in your training layers for up to a 3x performance improvement.

The Automatic Mixed Precision feature uses a graph optimization technique to determine FP16 operations and FP32 operations

Available in TensorFlow, PyTorch and MXNet via our NGC Deep Learning Framework Containers

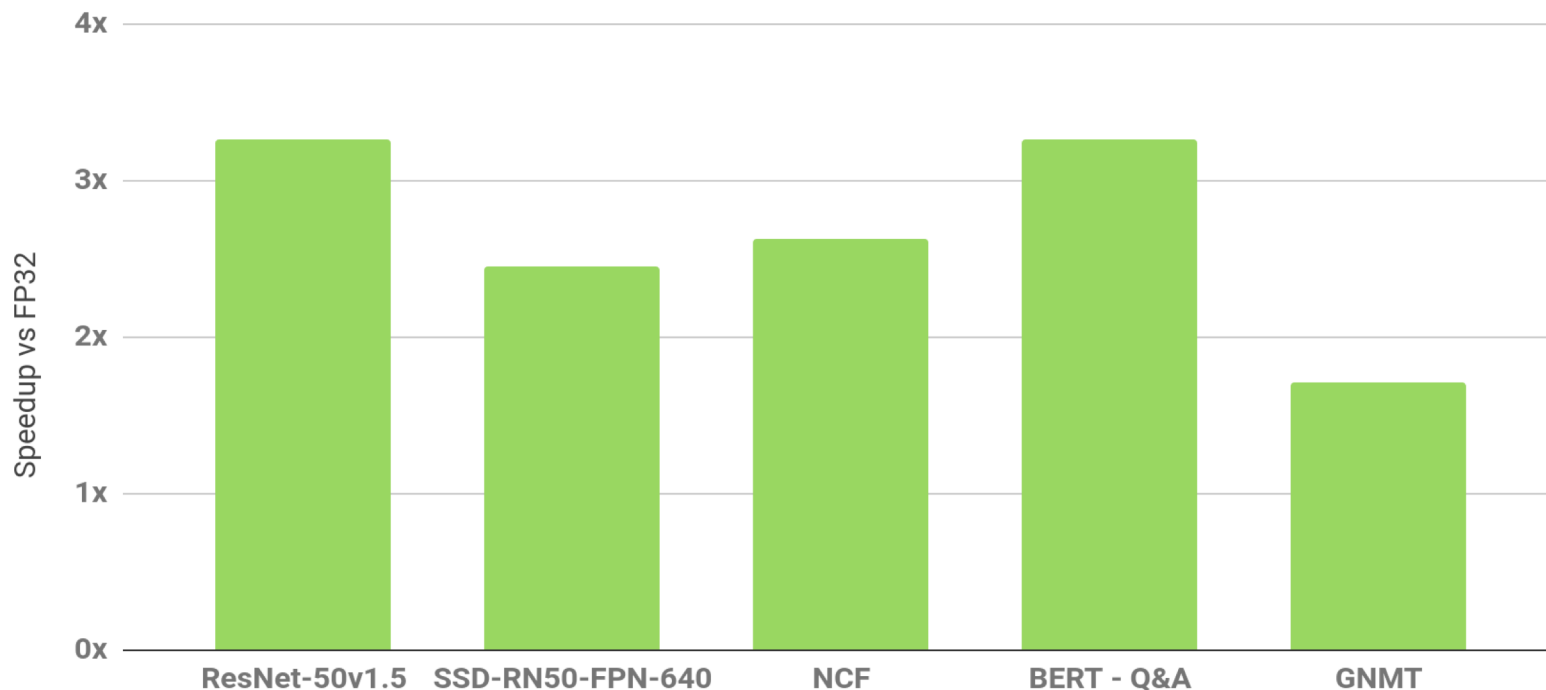


More details: <https://developer.nvidia.com/automatic-mixed-precision>

# Automatic Mixed-Precision Performance for Common Workloads

## TensorFlow Performance Improvements on 1 x V100 on DGX-1V w/XLA

Automatic Mixed Precision Speedups on Common Workloads



All models can be found at <https://github.com/NVIDIA/DeepLearningExamples/tree/master/TensorFlow>, except for `ssd-rn50-fpn-640`.

All performance collected on 1xV100-16GB, except `bert-squadqa` on 1xV100-32GB.

Batch sizes measured as follows. `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.

# CUDA COMPATIBILITY - CUDA 9.x

*Newer CUDA Version **DID NOT** Run on Older Display Driver*

CUDA driver API is backward compatible but not forward compatible

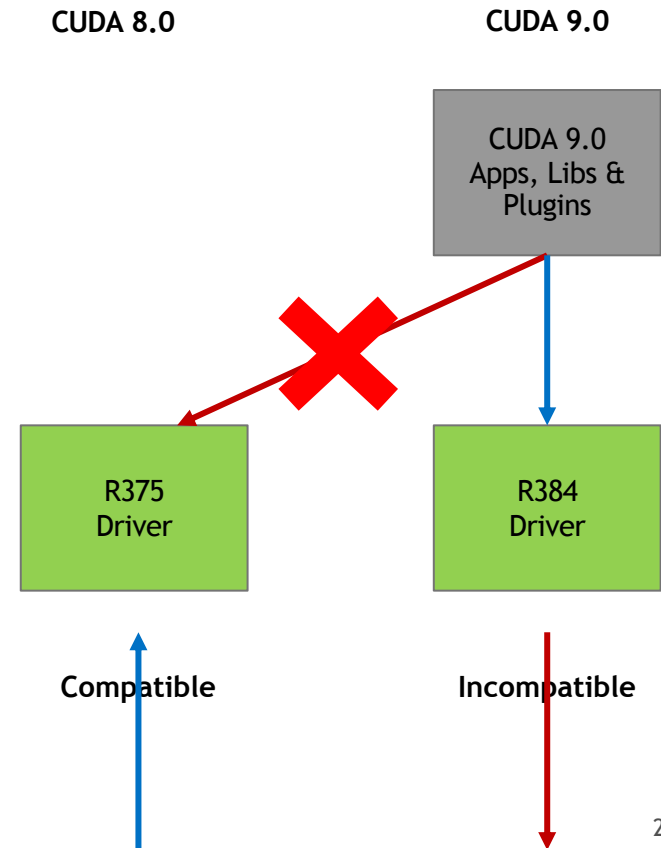
Each CUDA release has a minimum driver requirement

Applications compiled against a particular version of CUDA API will work on later driver releases

E.g.

CUDA 8.0 needs  $\geq$  R375

CUDA 9.0 needs  $\geq$  R384



# CUDA COMPATIBILITY - CUDA 10.x

Starting with CUDA 10.0

New compatibility platform upgrade path available

Use newer CUDA toolkits on older driver installs

Compatibility only with specific older driver versions

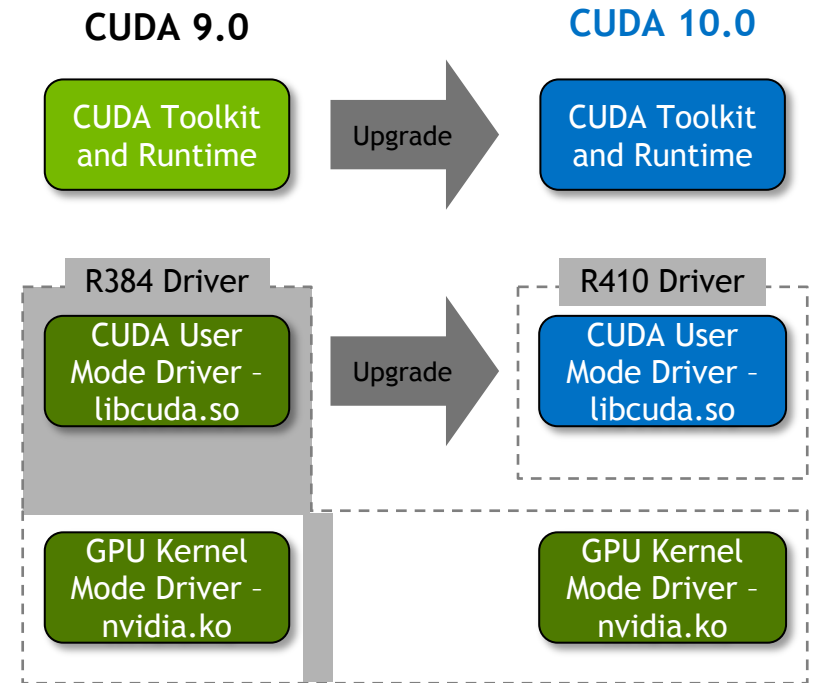
System requirements

Tesla GPU support only - no Quadro or GeForce

Only available on Linux

**NEW Forward Compatibility Option**

Upgrade *only* user-mode CUDA components\*



\*requires new 'cuda-compat-10-0' package

# ALWAYS UP-TO-DATE

Monthly releases and CUDA 10.1 in 19.03 containers

		19.03	18.09	18.08
<b>Supported Platform</b>	DGX OS	4.0 and 3.1.2	4.0.1 and 3.1.2+	4.0.1 and 3.1.2+
	NVIDIA Driver	418, 410, and 384	410 and 384	384
<b>Base Image</b>	Ubuntu	16.04	16.04	16.04
	CUDA	10.1.105	10.0.130	9.0.176
	cuBLAS	10.1.0.105	10.0.130	9.0.425
	cuDNN	7.5.0.56	7.3.0	7.2.1
	NCCL	2.4.3	2.3.4	2.2.13
<b>NVIDIA Optimized Frameworks</b>	NVCaffe	0.17.3	0.17.1	0.17.1
	MXNet	1.4.0	1.3.0	1.2.0
	PyTorch	1.1.0a0	0.4.1+	0.4.1
	TensorFlow	1.13.1+	1.10.0	1.9.0
	TensorRT	5.1.2.2	5.0.0	4.0.1
	TensorRT Server	1.0	0.6	0.5.0 Beta

# Deep Learning Frameworks Highlights

## Best NVIDIA Performance

Deep Learning Frameworks optimizations for NVIDIA hardware

Volta Tensor Cores support for mixed-precision (FP16) across TensorFlow, MXNet, PyTorch, and NVCaffe

## Latest NVIDIA Features

Latest NVIDIA Deep Learning libraries incorporated cuDNN, cuBLAS, and NCCL

Automatic Mixed-Precision for TensorFlow, PyTorch and MXNet

## Best Practices & QA Verified

Improved documentation with best practices and monthly release notes

Thorough monthly quality assurance testing

Multi-node support updated



# Significant Documentation Updates

Customers requested more documentation for deep learning containers

## Monthly Release Notes

Release Notes for: TensorFlow, PyTorch, MXNet, and NVCAffe

## TensorFlow 19.02 Release Notes

The screenshot shows the NVIDIA Deep Learning Frameworks Documentation page for TensorFlow 19.02. The page is titled "DEEP LEARNING FRAMEWORKS DOCUMENTATION" and "TensorFlow Release 19.02". It includes sections for "Contents of TensorFlow", "Driver Requirements", "GPU Requirements", and "Key Features and Enhancements".

**Contents of TensorFlow**

This container image contains the complete source of the version of NVIDIA TensorFlow in /opt/tensorflow. It is pre-built and installed as a system Python module.

To achieve optimum TensorFlow performance, for image based training, the container includes a sample script that demonstrates efficient training of convolutional neural networks (CNNs). The sample script may need to be modified to fit your application.

The container also includes the following:

- [Ubuntu 16.04](#)

**Note:** Container image 19.02-py2 contains [Python 2.7](#); 19.02-py3 contains [Python 3.5](#).

- [NVIDIA CUDA 10.0.130](#) including [CUDA® Basic Linear Algebra Subroutines library™ \(cuBLAS\) 10.0.130](#)
- [NVIDIA CUDA® Deep Neural Network library™ \(cuDNN\) 7.4.2](#)
- [NVIDIA Collective Communications Library \(NCCL\) 2.3.7](#) (optimized for [NVLink™](#))
- [Horovod 0.19.1](#)
- [DistMPI 3.1.3](#)
- [TensorBoard 1.12.2](#)
- [MLNX\\_OFED 3.4](#)
- [OpenSSL/CURL 1.18.12](#) at [commit 59c70e7](#)
- [TensorRT 5.0.2](#)
- [DAL 0.0.1 Python](#)
- [Jupyter and JupyterLab:](#)
  - [Jupyter Client 5.2.4](#)
  - [Jupyter Core 4.6.0](#)
  - [JupyterLab 0.35.4](#)
  - [JupyterLab Server 0.2.0](#)

**Driver Requirements**

Release 19.02 is based on CUDA 10, which requires [NVIDIA Driver](#) release 410.xx. However, if you are running on Tesla (Tesla V100, Tesla P4, Tesla P40, or Tesla P100), you may use NVIDIA driver release 384. For more information, see [CUDA Compatibility and Upgrades](#).

**GPU Requirements**

Release 19.02 supports CUDA compute capability 6.0 and higher. This corresponds to GPUs in the Pascal, Volta, and Turing families. Specifically, for a list of GPUs that this compute capability corresponds to, see [CUDA GPUs](#). For additional support details, see [CUDA Learning Frameworks Support Matrix](#).

**Key Features and Enhancements**

This TensorFlow release includes the following key features and enhancements.

- TensorFlow container image version 19.02 is based on [TensorFlow 1.13.0rc0](#).

## User Guides and Best Practices

User Guides for: Keras, TensorFlow, NVCAffe, and DIGITS  
Best Practices: Containers, Frameworks, DGX, and NGC

## Deep Learning Software Stack Matrix

Table 1. Software stack packaged with the 18.xx container images

Supported Platform	Container Image	18.02	18.01	17.12	17.11	17.10
Supported Platform	DGX OS	3.1.2+ and 2.1.1+	3.1.2+ and 2.1.1+	3.1.2+ and 2.1.1+	3.1.2+ and 2.1.1+	3.1.2+ and 2.1.1+
	NVIDIA Driver	384	384	384	384	384
Base Image	Ubuntu	16.04	16.04	16.04	16.04	16.04
	CUDA	9.0.176	9.0.176	9.0.176	9.0.176	9.0.176
Base Image	cuBLAS	9.0.282 Patch 2 <<and cuBLAS 9.0.234 Patch 1>>	9.0.282 Patch 2	16.04	16.04	16.04
	cuDNN	7.0.5	7.0.5	9.0.176	9.0.176	9.0
Base Image	NCCL	2.1.2	2.1.2	9.0.234	9.0.234	
	NVCAffe	0.16.5 including Python 2.7	0.16.5 including Python 2.7	7.0.5	7.0.4	7.0.3
NVIDIA Optimized Frameworks	Caffe2	0.8.1 including Python 2.7 or Python 3.5	0.8.1 including Python 2.7 or Python 3.5	2.1.2	2.1.2	2.0.5
	DIGITS	<<6.1.0>>	6.0.0	0.16.4	0.16.4	0.16.4
NVIDIA Optimized Frameworks	Microsoft Cognitive Toolkit	2.3.1 including Python 3.4 and OpenMPI 3.0.0	2.3.1 including Python 3.4 and OpenMPI 3.0.0	6.0.0	6.0.0	6.0.0
	MXNet	1.0.0 including Python 2.7 or Python 3.5	1.0.0 including Python 2.7 or Python 3.5	2.2 and OpenMPI 3.0.0	2.2 and OpenMPI 3.0.0	2.2
NVIDIA Optimized Frameworks	PyTorch	0.3.0 including Python 3.6	0.3.0 including Python 3.6	1.0.0	0.12.0	0.11.0
	TensorFlow	1.4.0 including Python 2.7 or Python 3.5 and Horovod 0.11.2	1.4.0 including Python 2.7 or Python 3.5 and Horovod 0.11.2	0.2.0	0.2.0	0.2.0
NVIDIA Optimized Frameworks	TensorRT	<<3.0.4>> including Python 2.7	3.0.1 including Python 2.7	1.4.0	1.3.0	1.3.0
	Theano	1.0.1 including Python 2.7	1.0.1 including Python 2.7	3.0.1		
NVIDIA Optimized Frameworks	Torch	1.0.0rc1	1.0.0rc1	1.0.0rc1	1.0.0rc1	0.10beta3
	Torch	7 including Python 2.7	7 including Python 2.7	Z	Z	Z

# DL Training with Tensor Cores: More resources

## Examples

New mixed-precision model examples: <https://developer.nvidia.com/deep-learning-examples>

GitHub: <https://github.com/NVIDIA/DeepLearningExamples>

TensorFlow ResNet-50 mixed-precision video: <https://www.youtube.com/watch?v=i1flBtdhjlg>

PyTorch GNMT mixed-precision how-to video: <https://www.youtube.com/watch?v=Dkzp05cpdpw>

## Tools

TensorFlow and MXNet Automatic Mixed-Precision: <https://developer.nvidia.com/automatic-mixed-precision>

PyTorch APEX: [https://nvidia.github.io/apex/fp16\\_utils.html](https://nvidia.github.io/apex/fp16_utils.html) & [NVIDIA developer news article](#)

## Further information

Mixed-precision blog: <https://devblogs.nvidia.com/mixed-precision-training-deep-neural-networks/>

Mixed-precision best practices: <https://docs.nvidia.com/deeplearning/sdk/mixed-precision-training/index.html>

Mixed-precision arXiv paper: <https://arxiv.org/abs/1710.03740>

GTC 2018 Sessions: [Training with Mixed Precision: Theory and Practice](#) and [Training with Mixed Precision: Real Examples](#)

## Available today

**[DGX/NGC Registry](#)**: Latest versions of the software stack and Tensor Core optimized examples (<https://ngc.nvidia.com>)

# Improved Multi-node Support

## Additional updates for multi-node support

### Support added:

- Tensorflow and MXNet distributed training via Horovod.
  - Partnered w/ Amazon to port Horovod to MXNet and to optimize Horovod NCCL integration.
- PyTorch distributed training via NVIDIA Apex [DistributedDataParallel](#) or native PyTorch.
- NVIDIA Caffe distributed training via OpenMPI+NCCL.

### Bundled inside the containers:

- Horovod+OpenMPI 3.x pre-installed for use with TensorFlow, MXNet, and NVIDIA Caffe.
- Containers pre-configured w/ Mellanox OpenFabrics drivers to enable GPUDirect RDMA.

# Connect with Deep Learning Experts

## Deep Learning Basics

- Tuesday at 3:00
- Wednesday at 12:00 AND 5:00

## Deep Learning with Tensor Cores

- Tuesday at 2:00
- Wednesday at 2:00

## Deep Learning Libraries (cuDNN, cuBLAS, CUTLASS)

- Tuesday at 1:00
- Wednesday at 1:00

## Deep Learning with Fast Data Pre-Processing (DALI)

- Tuesday at 3:00
- Wednesday at 2:00

## Advanced Deep Learning

- Tuesday at 1:00
- Wednesday at 4:00

## Deep Learning with TensorRT

- Tuesday at 5:00
- Thursday at 11:00

## Deep Learning Inference with Reduced Precision

- Tuesday at 4:00

## Deep Learning Inference Solutions on Windows

- Tuesday at 2:00

## Deep Learning with TensorRT Inference Server

- Wednesday at 12:00

## Deep Learning Deployment

- Tuesday at 4:00
- Wednesday at 3:00

AND MORE

# Thank you

