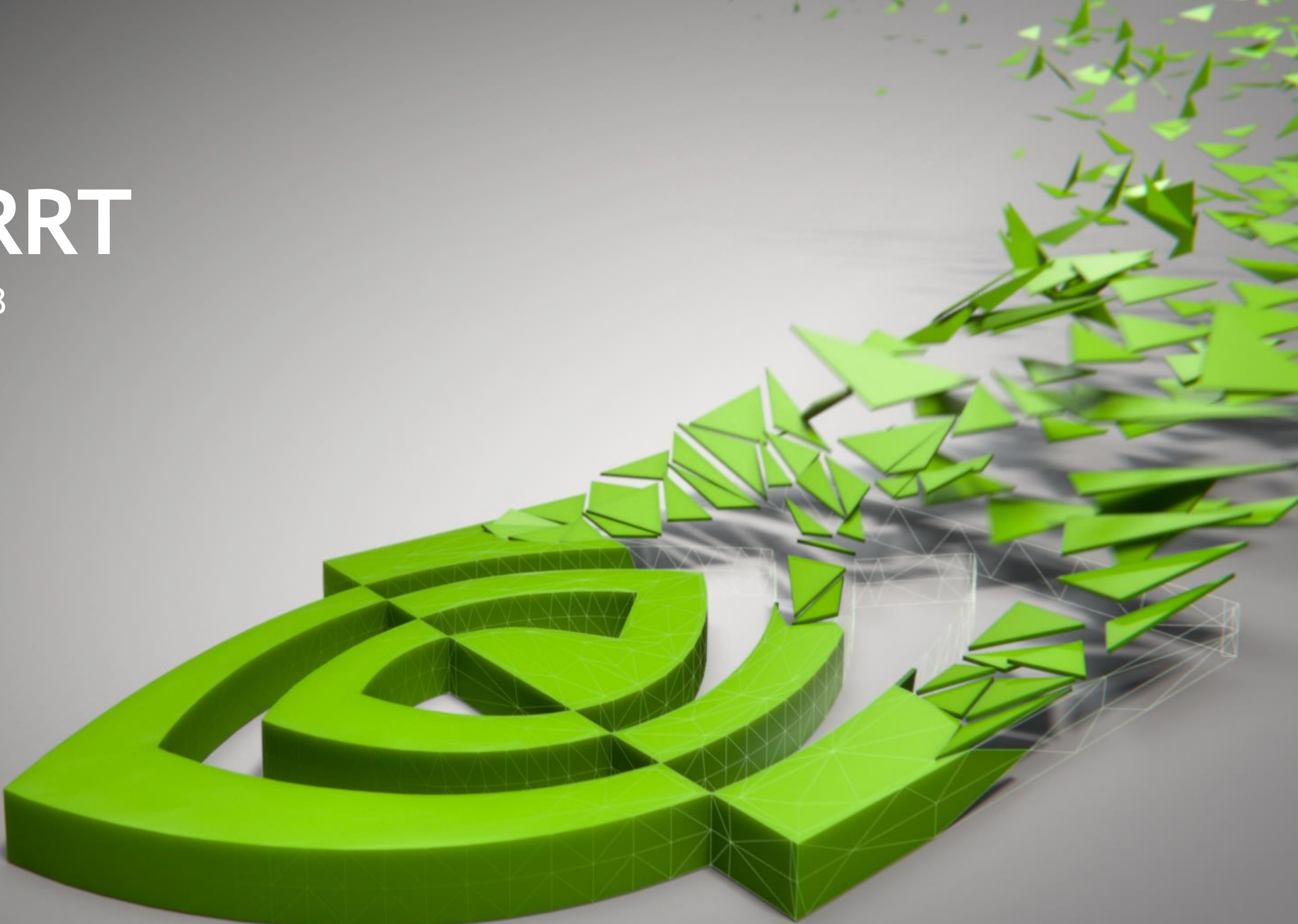
TENSORRT

Ken He, 2018.04.18

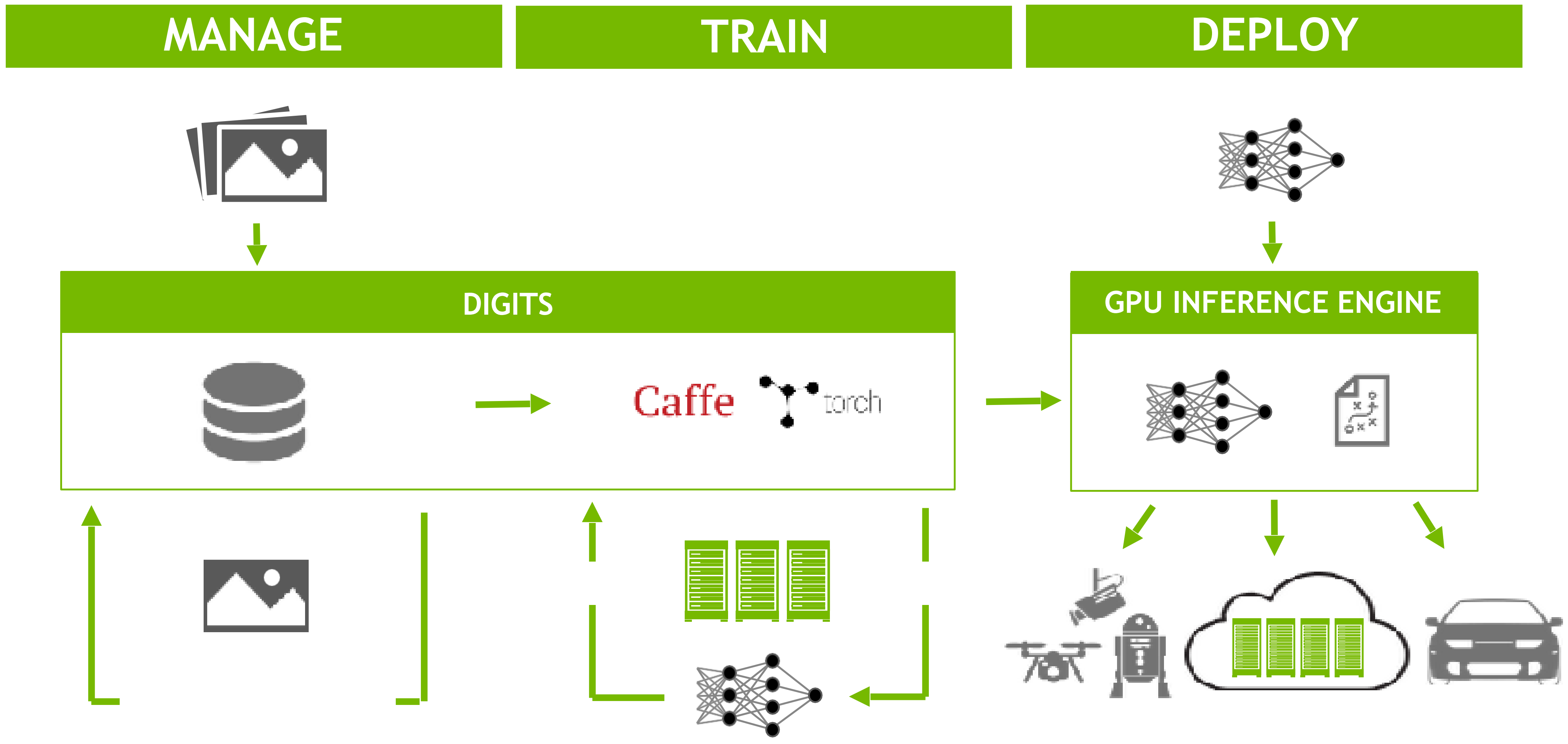


INFERENCE WITH TENSORRT

- ❑ Introduction of TensorRT
- ❑ TensorRT 4: What’s New
- ❑ TensorRT workflow
- ❑ Inference with FP16 and int8

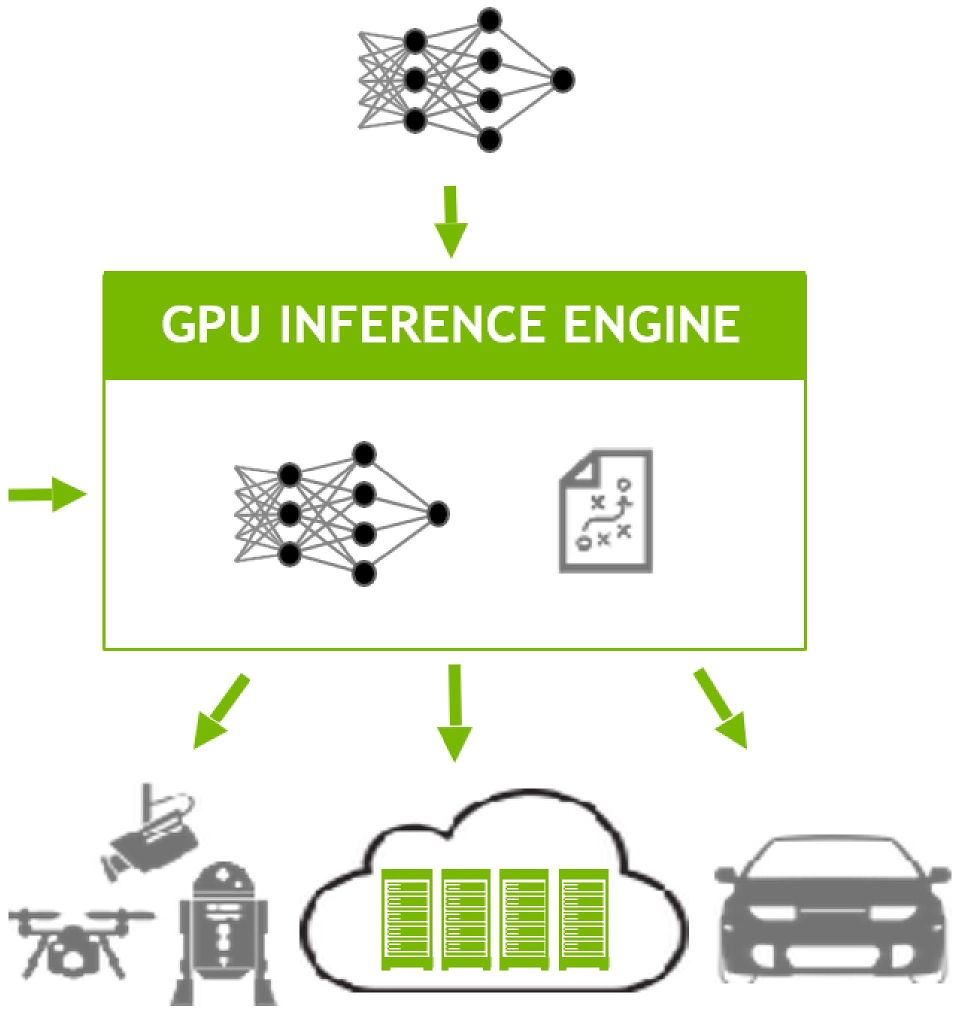
TENSORRT: GPU INFERENCE ENGINE

A COMPLETE DL PLATFORM



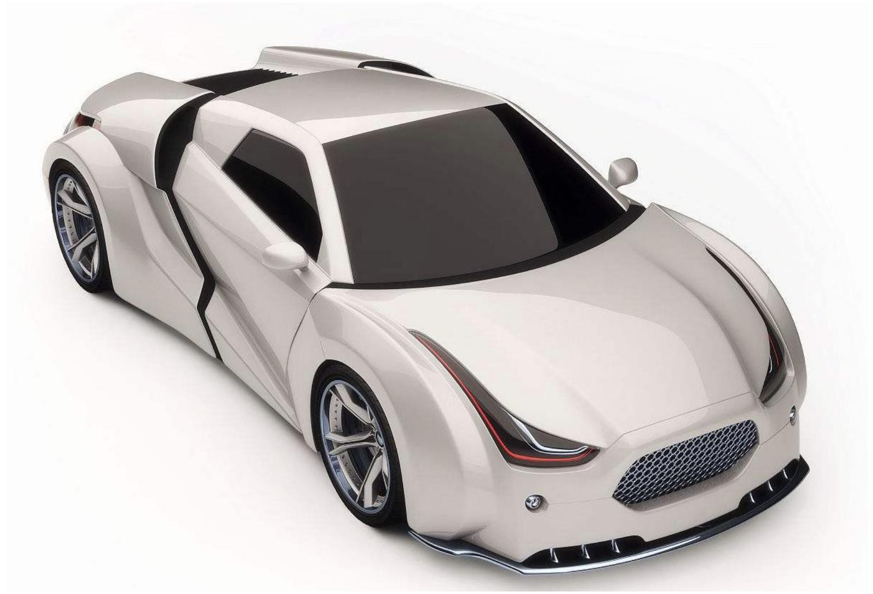
DEPLOY

TensorRT works at deploy stage



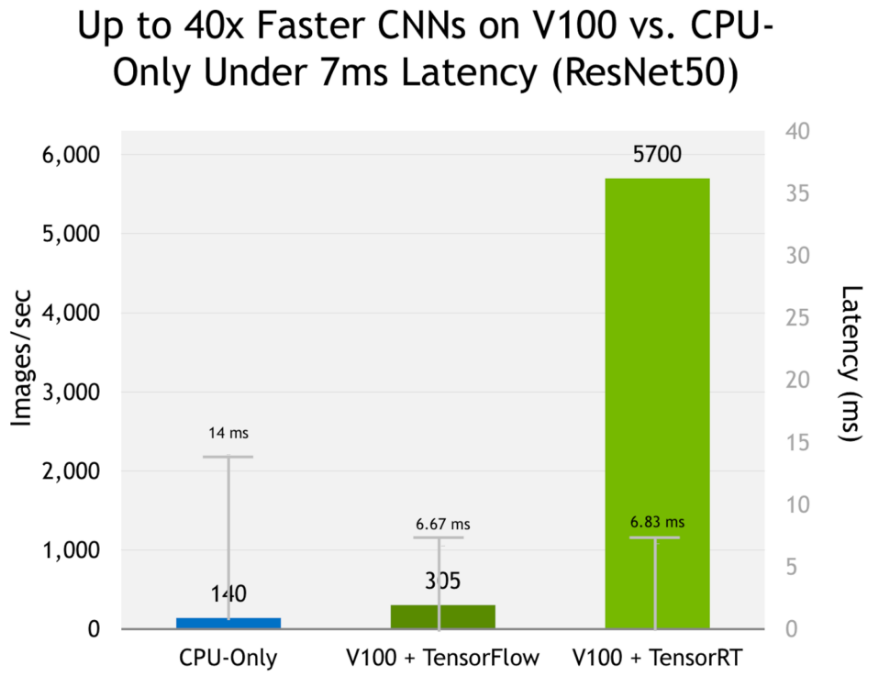
Why TensorRT ?

1.6L Engine



V100 + FP16 (TensorCore) + TensorRT 3.0

ResNet-50 inference performance



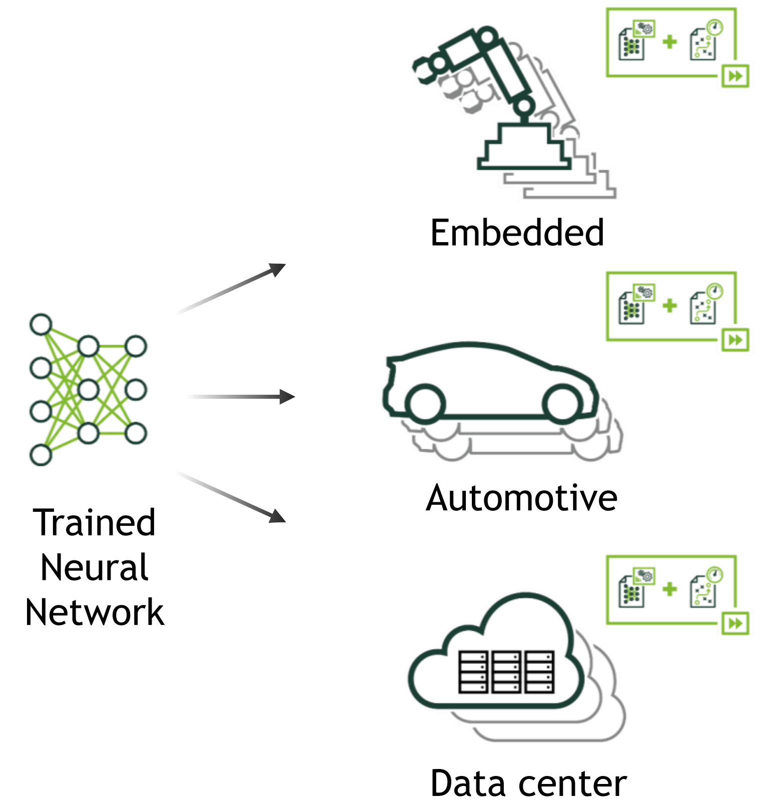
Inference throughput (images/sec) on ResNet50. **V100 + TensorRT**: NVIDIA TensorRT (FP16), batch size 39, Tesla V100-SXM2-16GB, E5-2690 v4@2.60GHz 3.5GHz Turbo (Broadwell) HT On **V100 + TensorFlow**: Preview of volta optimized TensorFlow (FP16), batch size 2, Tesla V100-PCIE-16GB, E5-2690 v4@2.60GHz 3.5GHz Turbo (Broadwell) HT On. **CPU-Only:** Intel Xeon-D 1587 Broadwell-E CPU and Intel DL SDK. Score doubled to comprehend Intel's stated claim of 2x performance improvement on Skylake with AVX512.

TENSORRT
GPU INFERENCE ENGINE (GIE)

High-performance deep learning inference for production deployment

High performance neural network inference engine for production deployment

- Automatically generate performance optimized and deployment-ready models for data-center, embedded and automotive platforms
- Easily deploy optimized models for inference with GIE runtime
- Deploy larger models with FP16/INT8, that deliver higher throughput than FP32 on Pascal GPUs



https://developer.nvidia.com/tensorrt

TENSORRT 4: WHAT'S NEW?

TENSORRT 4: WHAT'S NEW

TensorRT 4 now provides capabilities to accelerate speech recognition, neural machine translation and recommender systems. The native ONNX parser in TensorRT 4 provides an easy path to import models from frameworks such as PyTorch, Caffe2, MxNet, CNTK and Chainer.

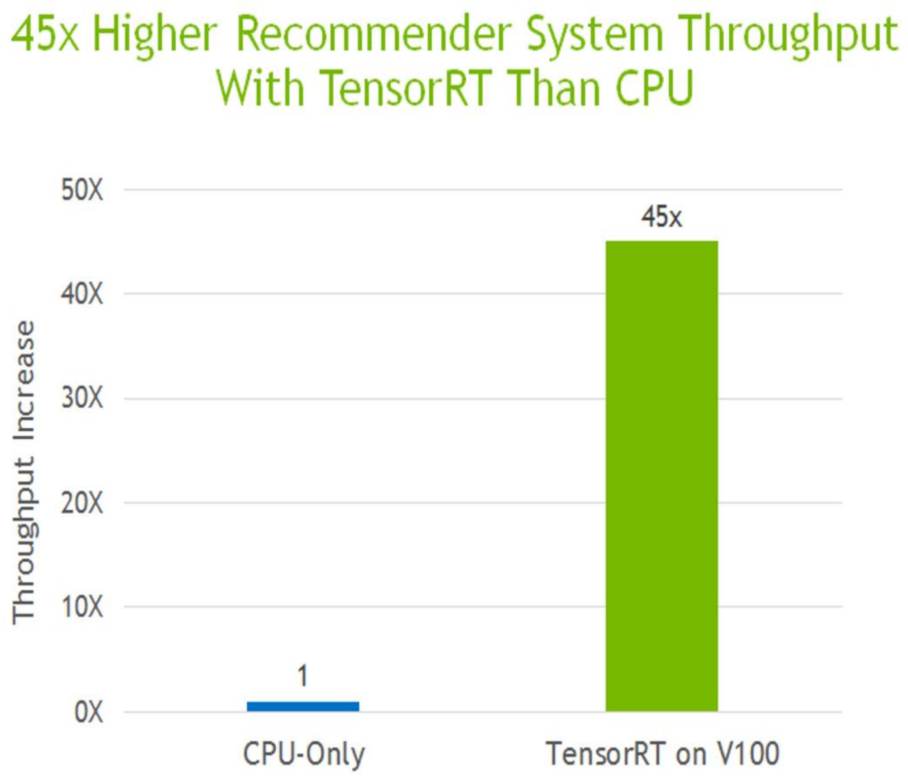
Highlights include:

45x higher throughput vs. CPU with new layers for Multilayer Perceptrons (MLP) and Recurrent Neural Networks (RNN)

50x faster inference performance on V100 vs. CPU-only for ONNX models imported with ONNX parser in TensorRT

Support for NVIDIA DRIVE™ Xavier - AI Computer for Autonomous Vehicles

3x inference speedup for FP16 custom layers with APIs for running on Volta Tensor Cores



SAP recommender system performance using TensorRT on NVIDIA Tesla V100 GPUs compared with TensorFlow running on Intel Xeon E5-2698 v4 at 2.20GHz.

FRAMEWORK INTEGRATIONS

NVIDIA works closely with deep learning framework developers to achieve optimized performance for inference on AI platforms using TensorRT. If your training models are in the ONNX format or other popular frameworks such as TensorFlow and MATLAB, there are easy ways for you to import models into TensorRT for inference. Below are few integrations with information on how to get started.

TensorFlow

ONNX

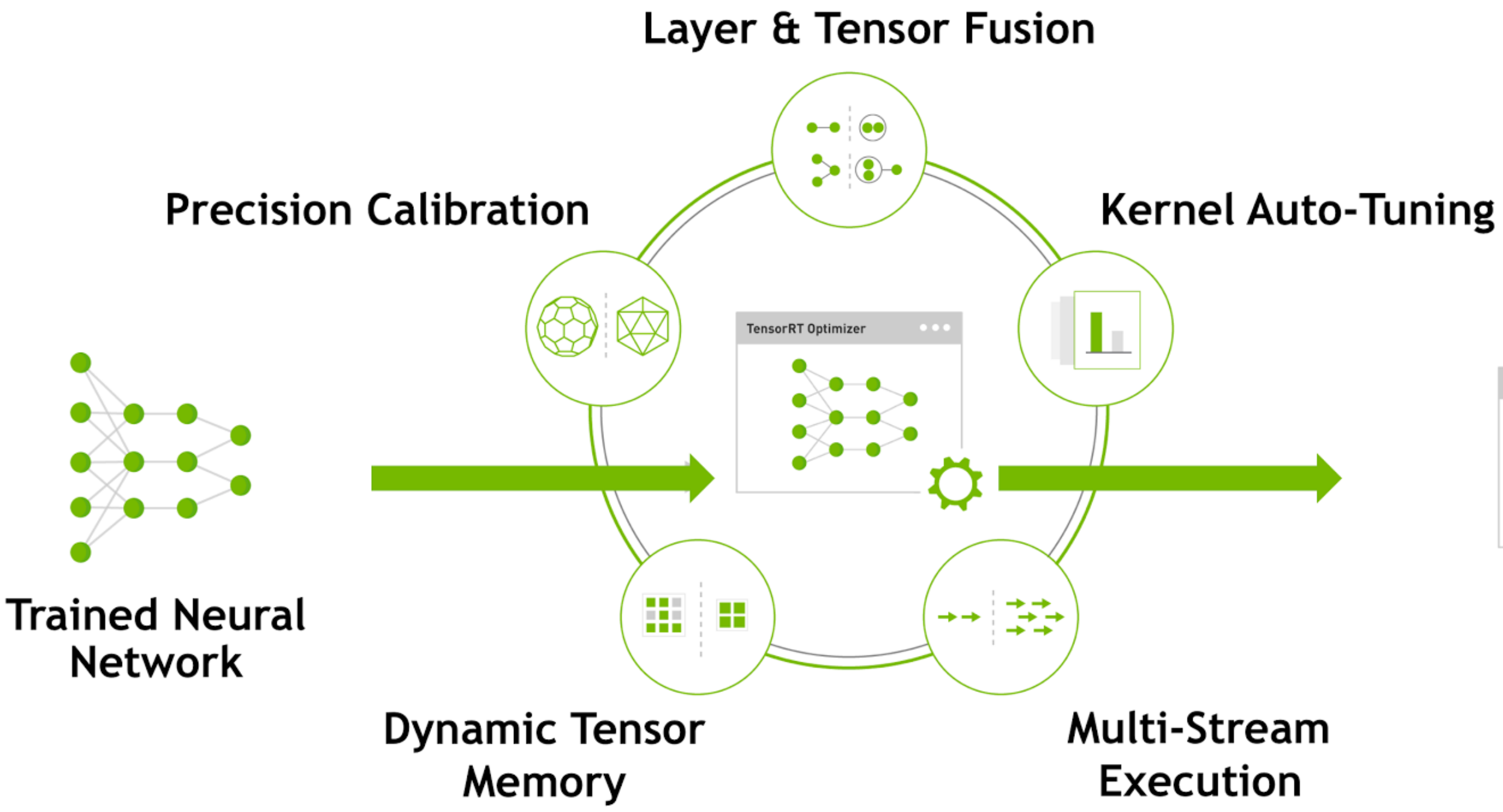
Caffe2 Chainer Microsoft CNTK

mxnet PYTORCH

MathWorks

TENSORRT: WORK FLOW

TENSORRT: WORK FLOW



TENSORRT WORKFLOW

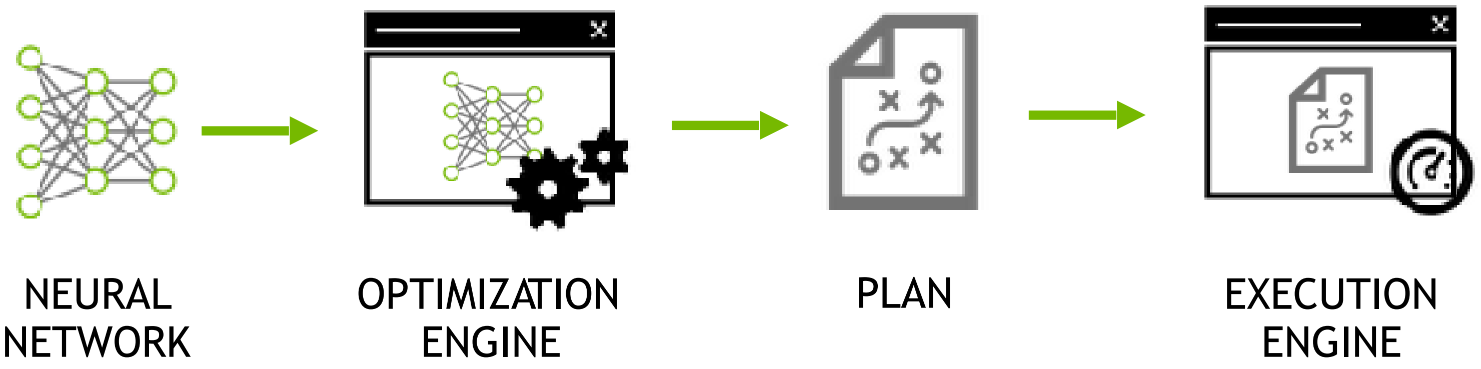
Input

- Pre-trained FP32 model and network

Output

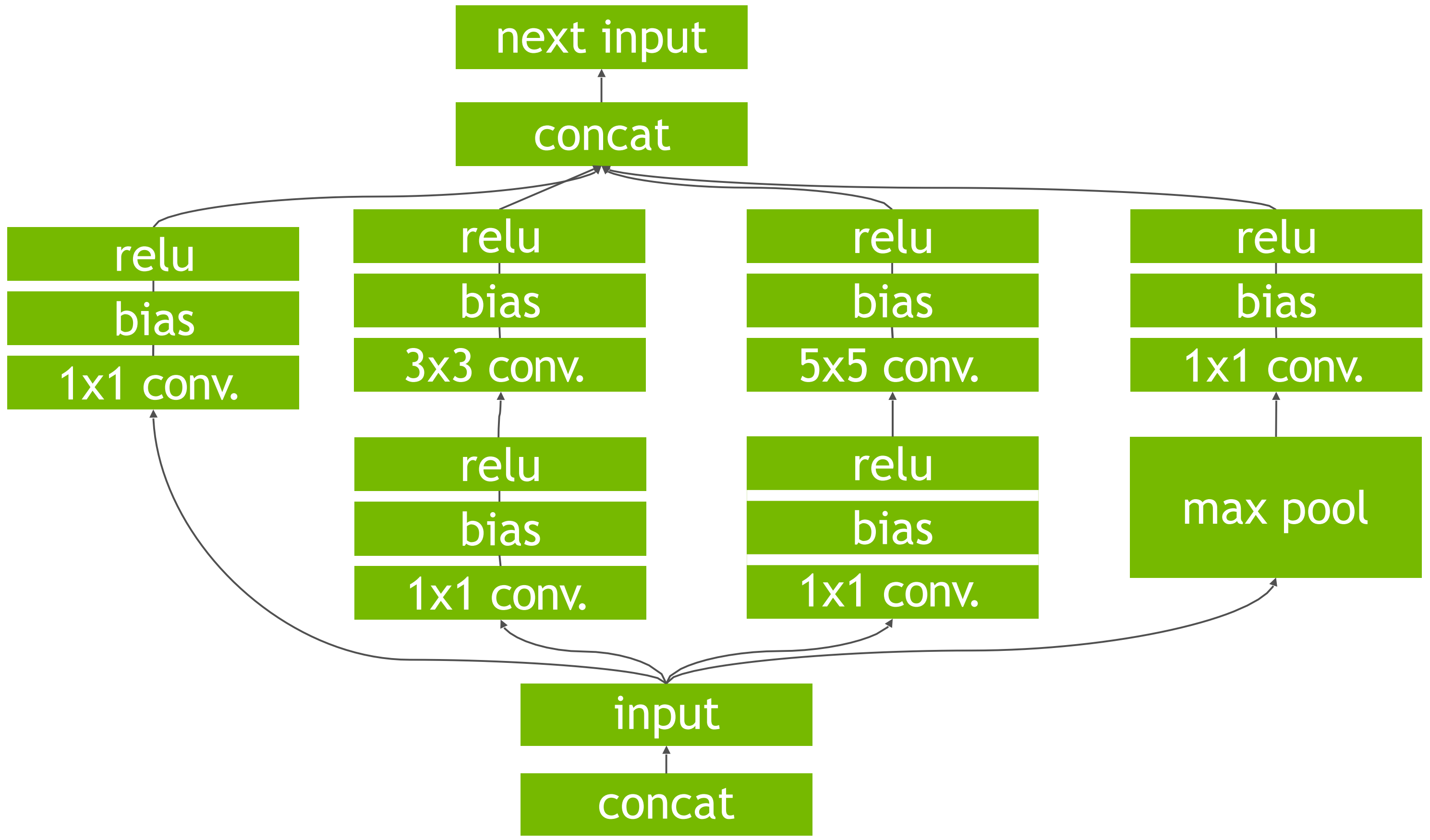
- Optimized execution engine on GPU for deployment

Serialized a PLAN can be reloaded from the disk into the TensorRT runtime. There is no need to perform the optimization step again.



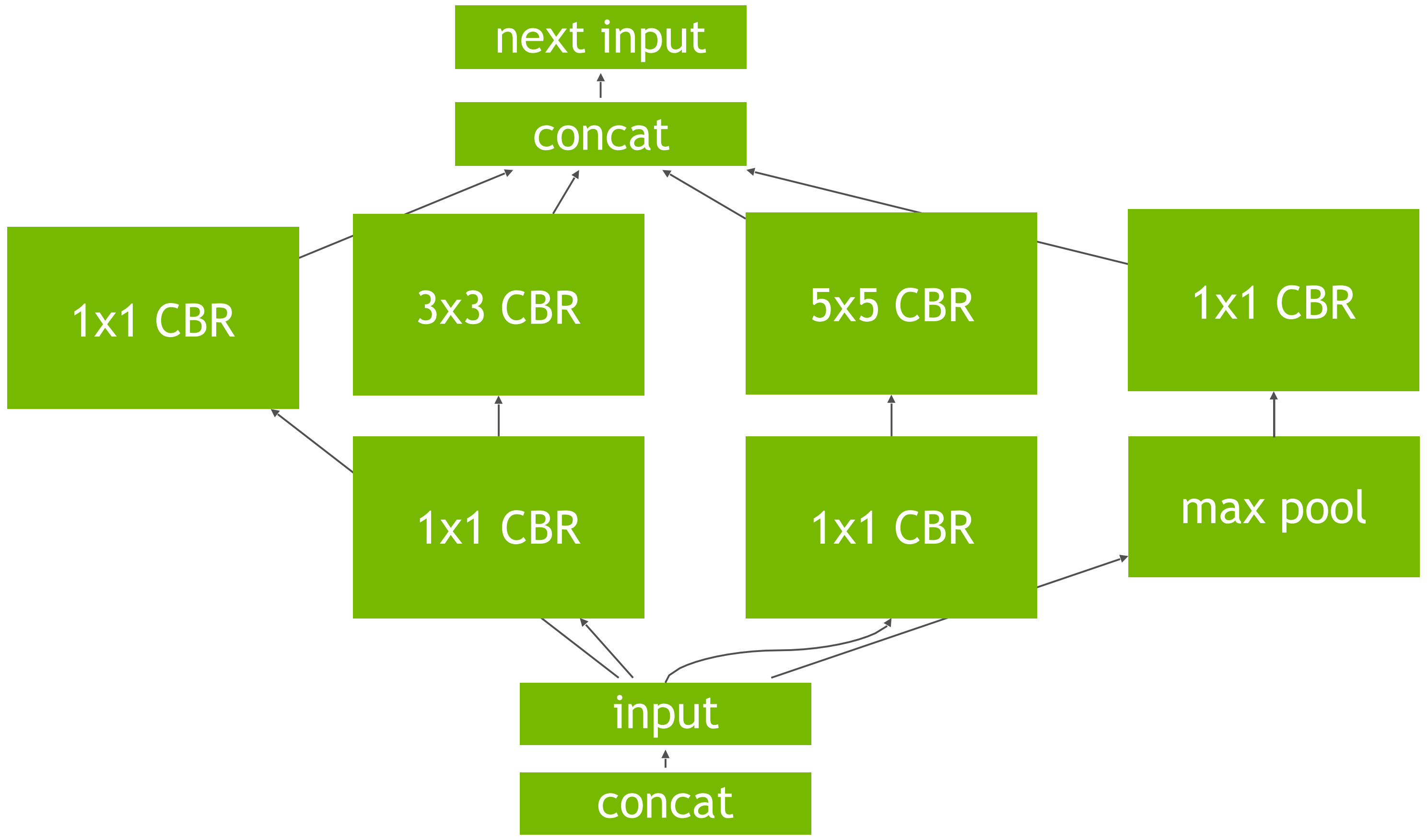
FUSE NETWORK LAYERS

Inception structure in GoogleNet



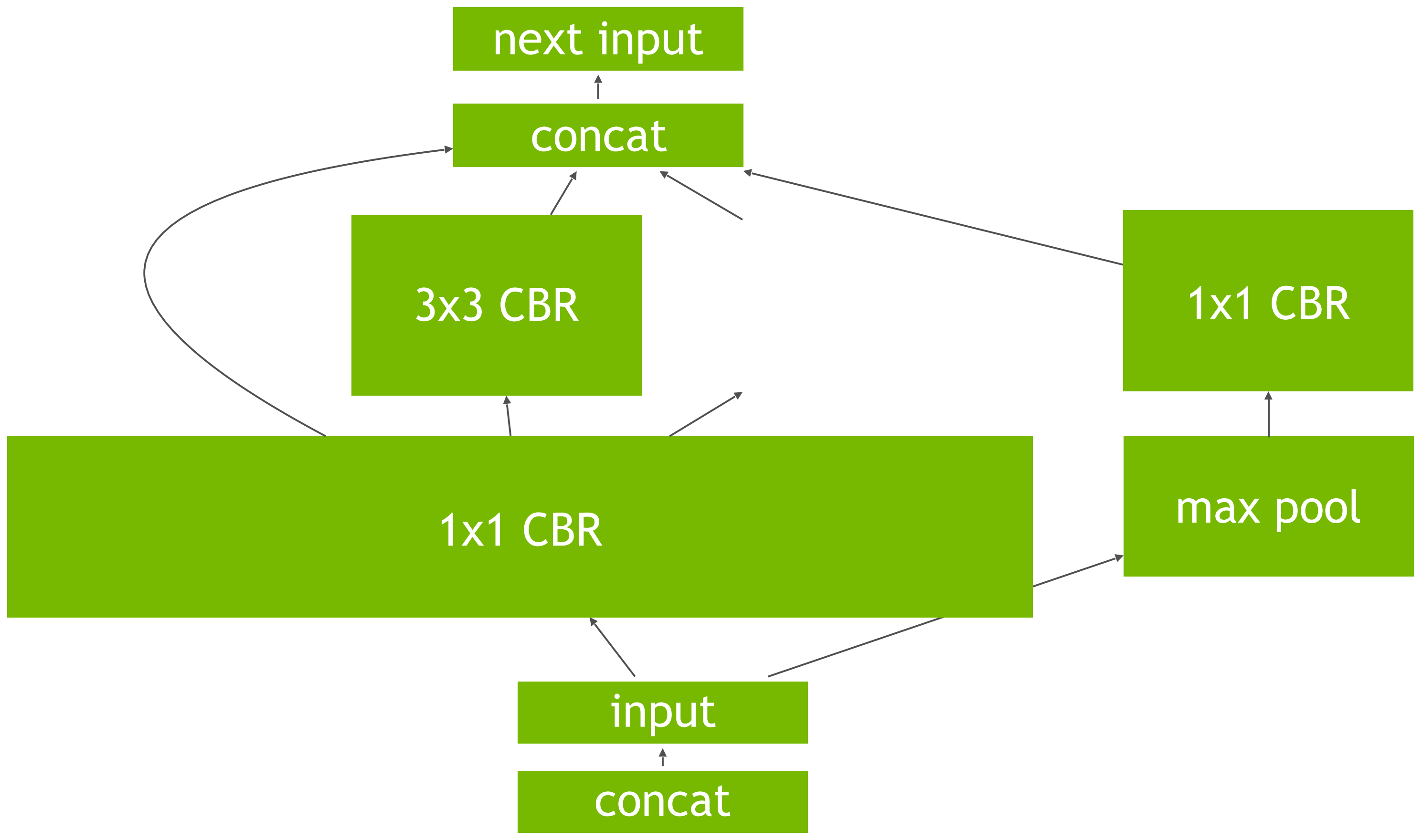
FUSE NETWORK LAYERS

Vertical fusion



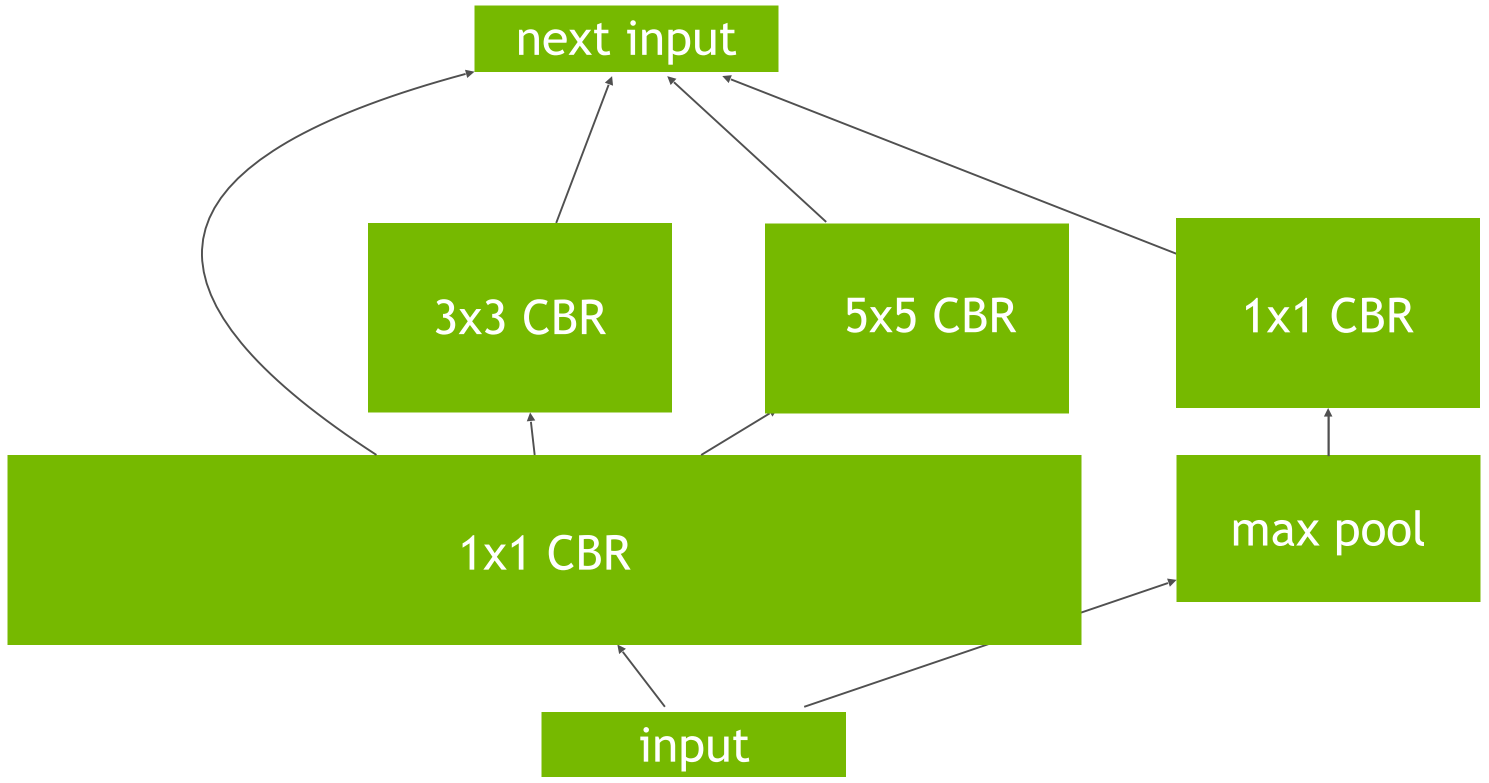
FUSE NETWORK LAYERS

Horizontal fusion



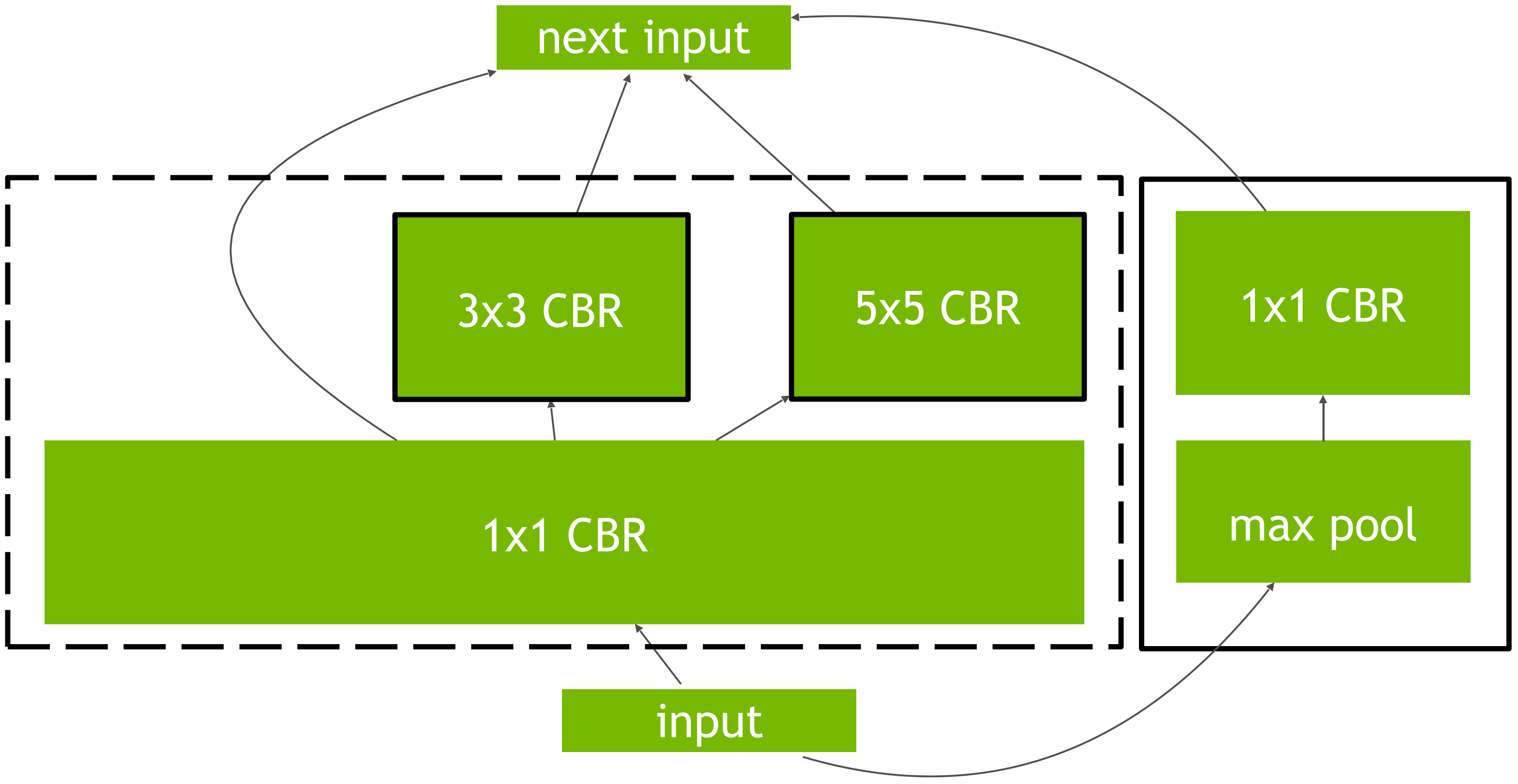
FUSE NETWORK LAYERS

Concat elision



FUSE NETWORK LAYERS

Concurrency



FP16/INT8 PERFORMANCE

Peak performance on Tesla GPU

TensorRT leverages high speed reduced precision capabilities of Pascal GPUs for better performance

- FP16
  - Pascal P100: **21** Tflops FP16 vs **10.6** Tflops FP32
  - Volta V100: **120** Tflops FP16 vs **15** Tflops FP32
- INT8
  - P4 : **22** INT8 TOPS* vs **5.5** Tflops FP32
  - P40: **47** INT8 TOPS* vs **12** Tflops FP32

FP16 INFERENCE

No extra input is needed

Same input as FP32 inference

- A pre-trained FP32 model

FP16 inference is highly automatized in TensorRT

- Set the model datatype to ***DataType::kHALF*** when parsing the model
- Set the inference mode to FP16 by builder->setHalf2Mode(true)

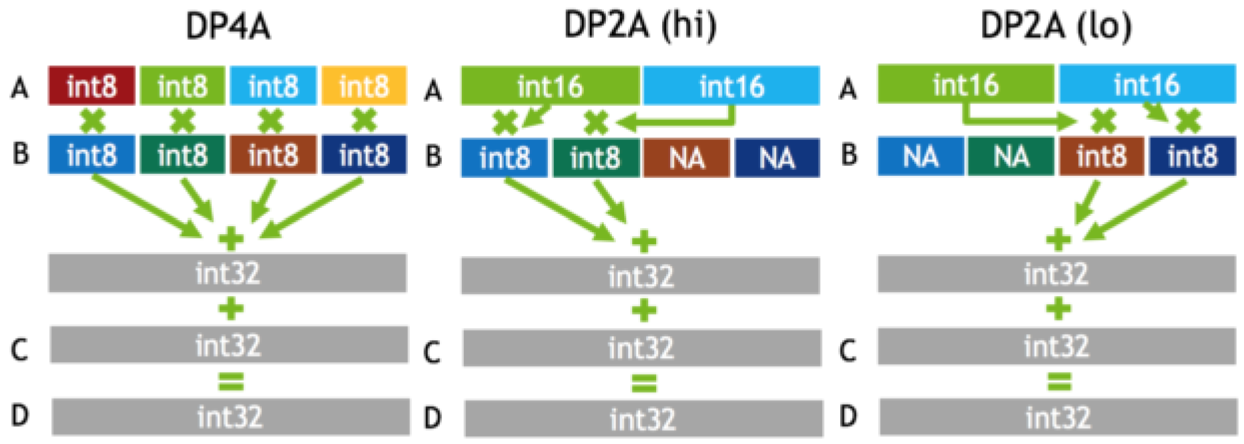
Note1: No extra input is needed compared with FP32 inference

Note2: Only Tesla P100/V100 supports the native FP16 math

INT8 INFERENCE

NEED AN EXTRA INPUT: CALIBRATION TABLE

Floating point numbers combine high dynamic range with high precision, but there are also cases where dynamic range is not necessary, so that integers may do the job. There are even applications where the data being processed has low precision so very low-precision storage (such as C short or char/byte types) can be used.



Optimization 2: FP16 and INT8 Precision Calibration

TensorRT can deploy models in FP32, FP16 and INT8, and switching between them is as easy as specifying the data type in the `uff_to_trt_engine` function:

- For FP32, `trt.infer.DataType.FLOAT`.
- For FP16 in and FP16 Tensor Cores on Volta GPUs,
  `trt.infer.DataType.HALF`
- For INT8 inference, `trt.infer.DataType.INT8`.

SAMPLE

SampleINT8 - Calibration And 8-bit Inference

1. Defining The Network
2. Building The Engine
3. Configuring the Builder
4. Running The Engine

SAMPLE

SampleINT8 - Calibration And 8-bit Inference

1.Defining The Network

```
const IBlobNameToTensor* blobNameToTensor =
parser->parse(locateFile(deployFile).c_str(),
locateFile(modelFile).c_str(), *network,
DataType::kFLOAT);
```

SAMPLE

SampleINT8 - Calibration And 8-bit Inference

2.Building The Engine

```
bool getBatch(void* bindings[], const char* names[], int nbBindings) override
{
         if (!mStream.next()) return false;
        CHECK(cudaMemcpy(mDeviceInput, mStream.getBatch(), mInputCount * sizeof(float),
                                        cudaMemcpyHostToDevice));
        assert(!strcmp(names[0], INPUT_BLOB_NAME));
        bindings[0] = mDeviceInput; return true;
}
```

SAMPLE

SampleINT8 - Calibration And 8-bit Inference

3. Configuring the Builder

When building an INT8 engine, the builder performs the following steps:

1. Builds a 32-bit engine, runs it on the calibration set, and records a histogram for each tensor of the distribution of activation values.
1. Builds a calibration table from the histograms.
2. Builds the INT8 engine from the calibration table and the network definition.

SAMPLE

SampleINT8 - Calibration And 8-bit Inference

4.Running The Engine

NVIDIA