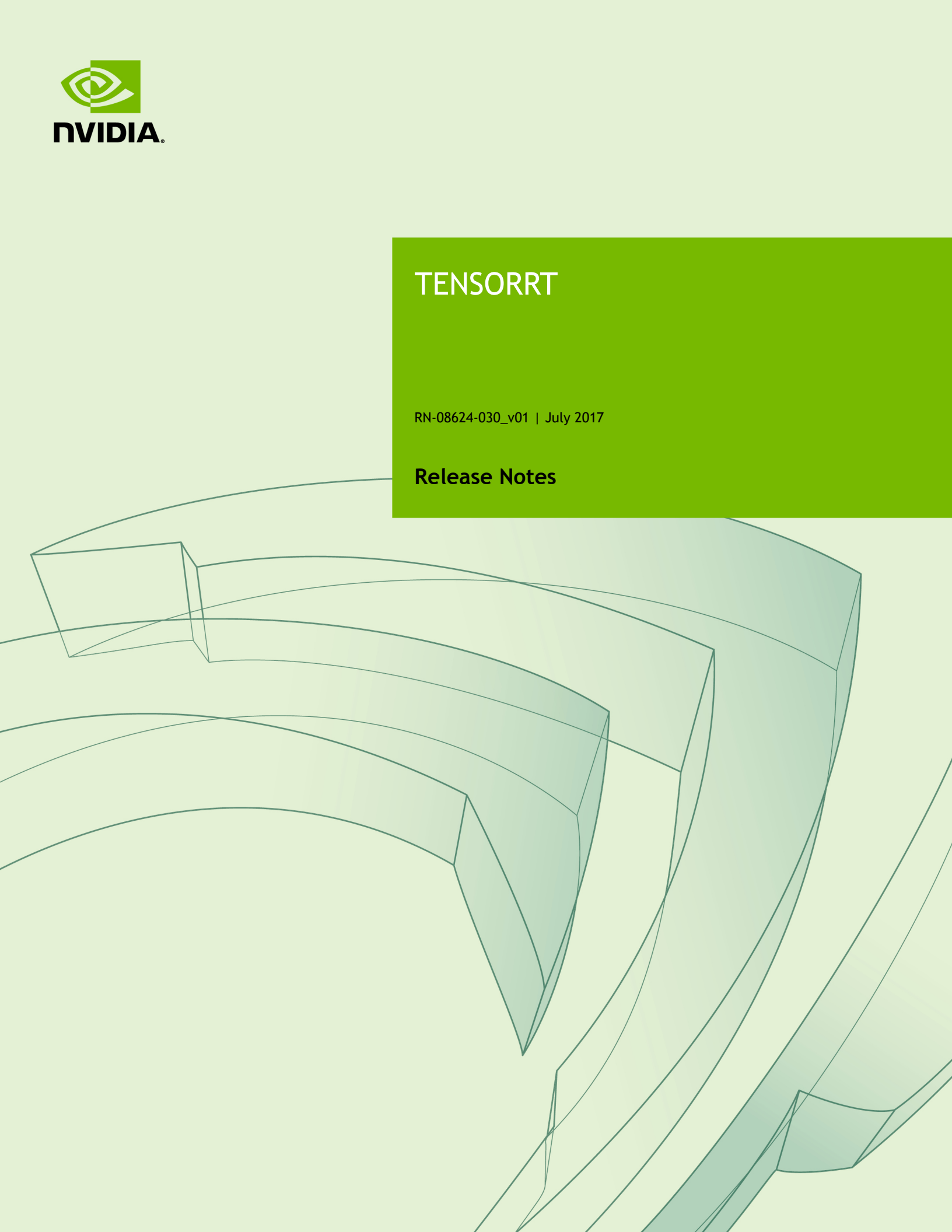

TENSORRT

RN-08624-030_v01 | July 2017

Release Notes

TABLE OF CONTENTS

Chapter 1. TENSORRT OVERVIEW

NVIDIA® TensorRT™ is a C++ library that facilitates high performance inference on NVIDIA GPUs. TensorRT takes a network definition and optimizes it by merging tensors and layers, transforming weights, choosing efficient intermediate data formats, and selecting from a large kernel catalog based on layer parameters and measured performance.

TensorRT consists of import methods to help you express your trained deep learning model for TensorRT to optimize and run. It is an optimization tool that applies graph optimization and layer fusion and finds the fastest implementation of that model leveraging a diverse collection of highly optimized kernels, and a runtime that you can use to execute this network in an inference context.

TensorRT includes a full infrastructure that allows you to leverage high speed reduced precision capabilities of Pascal™ GPUs as an optional optimization.

TensorRT is built with gcc 4.8.

Chapter 2. TENSORRT RELEASE 3.0 PREVIEW (EA)

This is a technology preview release of the future version of TensorRT. Production use of TensorRT should continue to use 2.1 for the time being.

Key Features and Enhancements

This TensorRT release includes the following key features and enhancements.

Streamlined export for models trained in TensorFlow to TensorRT
With this release you can take a TensorFlow trained model saved in a TensorFlow protobuf and convert it to run in TensorRT. The TensorFlow to UFF converter creates an output file in a format called UFF (Universal Framework Format) which can then be read into TensorRT.

Currently the export path is expected to support the following:

- Tensorflow 1.0
- FP32 CNNs
- FP16 CNNs

The TensorFlow export path is currently not expected to support the following:

- Other versions of TensorFlow (0.9, 1.1, etc..)
- RNNs
- INT8 CNNs

TensorFlow convenience functions
NVIDIA provides convenience functions so that when using UFF and TensorRT to export a model and run inference, only a few lines of code is needed.

Universal Framework Format 0.1
UFF format is designed as a way of storing the information about a neural network that is needed to create an inference engine based on that neural network.

Python API

TensorRT 3.0 introduces the TensorRT Python API, allowing developers to access:

- the NvCaffeParser
- the NvUffParser
- the nvinfer graph definition API
- the inference engine builder
- the inference-time interface for engine execution within Python

TensorRT also introduces a workflow to include C++ custom layer implementations in Python based TensorRT applications.

Using TensorRT 3.0

Ensure you are familiar with the following notes when using this release.

- Although networks can use NHWC and NCHW, TensorFlow users are encouraged to convert their networks to use NCHW data ordering explicitly in order to achieve the best possible performance.
- Average pooling behavior changed to exclude the padding from the computation. The padding is now excluded from the computation in all of the pooling modes. This results in incorrect behavior for networks which rely on average pooling which includes padding, such as inceptionV3. This issue will be addressed in a future release.
- The **`libnvcaffe_parsers.so`** library file is now called **`libnvparsers.so`**. The links for **`libnvcaffe_parsers`** are updated to point to the new **`libnvparsers`** library. The static library **`libnvcaffe_parser.a`** is also linked to the new **`libnvparsers`**. For example:
  - **Old structure:** **`libnvcaffe_parsers.4.0.0.so`** links to **`libnvcaffe_parsers.4.so`** which links to **`libnvcaffe_parsers.so`**.
  - **New structure:** **`libnvcaffe_parsers.4.0.0.so`** links to **`ibnvcaffe_parsers.4.so`** which links to **`libnvcaffe_parsers.so`** which links to **`libnvparsers.so`**(actual file) .

Known Issues

- TensorRT does not support asymmetric padding.
- Some TensorRT optimizations disabled just for this Early Release (EA) to ensure that the UFF model runs properly. This will be addressed in TensorRT 3.0.
- The TensorFlow conversion path is not fully optimized.
- INT8 Calibration is not available in Python.

Notice

THE INFORMATION IN THIS GUIDE AND ALL OTHER INFORMATION CONTAINED IN NVIDIA DOCUMENTATION REFERENCED IN THIS GUIDE IS PROVIDED "AS IS." NVIDIA MAKES NO WARRANTIES, EXPRESSED, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO THE INFORMATION FOR THE PRODUCT, AND EXPRESSLY DISCLAIMS ALL IMPLIED WARRANTIES OF NONINFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE. Notwithstanding any damages that customer might incur for any reason whatsoever, NVIDIA's aggregate and cumulative liability towards customer for the product described in this guide shall be limited in accordance with the NVIDIA terms and conditions of sale for the product.

THE NVIDIA PRODUCT DESCRIBED IN THIS GUIDE IS NOT FAULT TOLERANT AND IS NOT DESIGNED, MANUFACTURED OR INTENDED FOR USE IN CONNECTION WITH THE DESIGN, CONSTRUCTION, MAINTENANCE, AND/OR OPERATION OF ANY SYSTEM WHERE THE USE OR A FAILURE OF SUCH SYSTEM COULD RESULT IN A SITUATION THAT THREATENS THE SAFETY OF HUMAN LIFE OR SEVERE PHYSICAL HARM OR PROPERTY DAMAGE (INCLUDING, FOR EXAMPLE, USE IN CONNECTION WITH ANY NUCLEAR, AVIONICS, LIFE SUPPORT OR OTHER LIFE CRITICAL APPLICATION). NVIDIA EXPRESSLY DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY OF FITNESS FOR SUCH HIGH RISK USES. NVIDIA SHALL NOT BE LIABLE TO CUSTOMER OR ANY THIRD PARTY, IN WHOLE OR IN PART, FOR ANY CLAIMS OR DAMAGES ARISING FROM SUCH HIGH RISK USES.

NVIDIA makes no representation or warranty that the product described in this guide will be suitable for any specified use without further testing or modification. Testing of all parameters of each product is not necessarily performed by NVIDIA. It is customer's sole responsibility to ensure the product is suitable and fit for the application planned by customer and to do the necessary testing for the application in order to avoid a default of the application or the product. Weaknesses in customer's product designs may affect the quality and reliability of the NVIDIA product and may result in additional or different conditions and/or requirements beyond those contained in this guide. NVIDIA does not accept any liability related to any default, damage, costs or problem which may be based on or attributable to: (i) the use of the NVIDIA product in any manner that is contrary to this guide, or (ii) customer product designs.

Other than the right for customer to use the information in this guide with the product, no other license, either expressed or implied, is hereby granted by NVIDIA under this guide. Reproduction of information in this guide is permissible only if reproduction is approved by NVIDIA in writing, is reproduced without alteration, and is accompanied by all associated conditions, limitations, and notices.

Trademarks

NVIDIA, the NVIDIA logo, and cuBLAS, CUDA, cuDNN, cuFFT, cuSPARSE, DIGITS, DGX, DGX-1, Jetson, Kepler, NVIDIA Maxwell, NCCL, NVLink, Pascal, Tegra, TensorRT, and Tesla are trademarks and/or registered trademarks of NVIDIA Corporation in the Unites States and other countries. Other company and product names may be trademarks of the respective companies with which they are associated.

Copyright