TABLE OF CONTENTS

Chapter 1. C++ API........................................................................................................ 1
Chapter 2. Python API.................................................................................................. 2
  2.1. Graph Surgeon API............................................................................................ 2
  2.2. UFF API............................................................................................................ 2
Chapter 1.
C++ API

The C++ API allows developers to import, calibrate, generate and deploy networks using C++. Networks can be imported directly from NVCaffe™, or from other frameworks via the UFF format. They may also be created programmatically by instantiating individual layers and setting parameters and weights directly.

Within the core C++ API in NvInfer.h, the following APIs are included:

- Builder API
- Execution API
- Network Definition API
- Plugin API

To view this API, see TensorRT C++ API.

For more information about the C++ API, including sample code, see TensorRT Developer Guide.
The TensorRT Python API enables developers, (in Python based development
environments and those looking to experiment with TensorRT) to easily parse models
(for example, from NVcaffe, TensorFlow™, Open Neural Network Exchange™
(ONNX), and NumPy compatible frameworks) and generate and run PLAN files.
Currently, all functionality except for Int8Calibrators and RNNs are available to use in
Python.

To view this API, see TensorRT Python API.

For more information about the Python API, including sample code, see TensorRT
Developer Guide.

2.1. Graph Surgeon API

Included within the Python API is the Graph Surgeon API; which enables you to
transform TensorFlow graphs.

The Graph Surgeon API is located in graphsurgeon/graphsurgeon.html and
contains three classes, Node Creation, Static Graph, and Dynamic Graph.

To view this API, see Graph Surgeon API.

For more information about the Graph Surgeon API, see TensorRT Developer Guide.

2.2. UFF API

Included within the Python API is the UFF API; a package that contains a set of utilities
to convert trained models from various frameworks to a common format.

The UFF API is located in uff/uff.html and contains two conversion type tool classes
called Tensorflow Modelstream to UFF and Tensorflow Frozen Protobuf
Model to UFF.

To view this API, see UFF API.
For more information about the UFF API, see TensorRT Developer Guide.
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, DALI, 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

© 2018 NVIDIA Corporation. All rights reserved.