Quadra AI Python Inference API

Quick Start Guide V4.5

NETINT Technologies Inc.

LEVEL B: CONFIDENTIAL –DISTRIBUTION RESTRICTED

Table of Contents

NETINT Technologies Inc.
Level B: Confidential - Distribution
Restricted

NETINT © 2023. LEVEL B: CONFIDENTIAL-DISTRIBUTION RESTRICTED

1. Legal Notices

Information in this document is provided in connection with NETINT products. No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted by this document. Except as provided in NETINT's terms and conditions of sale for such products, NETINT assumes no liability whatsoever and NETINT disclaims any express or implied warranty, relating to sale and/or use of NETINT products including liability or warranties relating to fitness for a particular purpose, merchantability, or infringement of any patent, copyright, or other intellectual property right.

A "Mission Critical Application" is any application in which failure of the NETINT Product could result, directly or indirectly, in personal injury or death. Should you purchase or use NETINT's products for any such mission critical application, you shall indemnify and hold NETINT and its subsidiaries, subcontractors and affiliates, and the directors, officers, and employees of each, harmless against all claims costs, damages, and expenses and reasonable attorney's fees arising out of, directly or indirectly, any claim of product liability, personal injury, or death arising in any way out of such mission critical application, whether or not NETINT or its subcontractor was negligent in the design, manufacture, or warning of the NETINT product or any of its parts.

NETINT may make changes to specifications, technical documentation, and product descriptions at any time, without notice. The information here is subject to change without notice. Do not finalize a design with this information. The products described in this document may contain design defects or errors known as errata which may cause the product to deviate from published specifications.

NETINT™, Codensity™, Quadra™ and the NETINT Logo are trademarks of NETINT Technologies Inc. All other trademarks or registered trademarks are the property of their respective owners.

© 2023 NETINT Technologies Inc. All rights reserved.

NETINT Technologies Inc. Level B: Confidential - Distribution Restricted

2. Introduction

NETINT provides high density and efficient video transcoding solutions using the powerful video processing engines inside the Codensity/Quadra Application-Specific Integrated Circuit (ASIC). We can provide multiple stream transcoding functions and services, directly to video content providers. We can also provide Transcoding as a Service (TaaS), which can be integrated into video streaming systems and services. Our functions and services can be used for highly efficient Video-on-Demand file transcoding, as well as real-time, live video streaming applications.

The NETINT AI Toolkit offers Python Inference APIs for developing and prototyping deep learning-based AI applications using Quadra NPU. Quadra AI Python Inference APIs have similar interface to TensorFlow Lite, which can be used to execute a Network Binary Graph (NBG) model on the Quadra chip to make predictions based on input data.

NETINT Technologies Inc. Level B: Confidential - Distribution Restricted

3. Install the Python Inference API

This section describes the Quadra AI Python Inference API installation.

Prerequisites

- The Quadra card should be installed correctly with Xcoder Codec Library in your host.
- The <*NETINT_toolkit*> package has already been unpacked at the home directory.

Note: Only the network_wrapper needs to be installed.

Procedure

1. Installing network_wrapper.

```
cd ~/NETINT_toolkit/network_wrapper

pip3 install ./dist/<netint_inference_api>.tar.gz
```

2. Changing the permission for the device's nodes libxcoder.

```
sudo ./scripts/change_permission.sh
```

3. Running the demo to verify the Python Inference API installation (Optional).

```
cd ./scripts

python3 ./demo.py
```

NETINT Technologies Inc. Level B: Confidential Distribution Restricted

4. API Workflow

The Quadra AI Python Inference API typically follows the following steps.

4.1. Creating an Inference Object

You need to create an interpreter and load the NBG (*.nb) model into memory.

```
import netint.network

interpreter = netint.network.Interpreter(model_path = model_path, dev_id = device_id)

# Get input and output tensors
input_details = interpreter.get_input_details()
output_details = interpreter.get_output_details()
```

Loading Inference Data

You need to load inference data and allocate the tensors.

```
interpreter.set_tensor(0, input_data)
```

Note: If the model requires multiple inputs, use the *set_tensor(Index, Input_data)* with the increasing input index to meet the requirements.

To use Python Inference API, the input data types must be one of the following types:

Data	Description
Uint8	Requires an uint8 type input data. **Note:** The input data will be fed into the NBG directly. Thus, the NBG may need to include pre-processing node.
Float32	Requires float32 input data. **Note:** The input data should be processed with pre-processing step.

NETINT Technical Level B: Confidential Restricted Distribution

NETINT © 2023. LEVEL B: CONFIDENTIAL-DISTRIBUTION RESTRICTED

Input data for the model generally does not match the input data format expected by the model. For example, you might need to resize an image or change the image format/ layout to be compatible with the model. The pre-processing script depends on different model requirements.

4.2. Running Inference

This step involves using the Inference API to execute the model.

```
interpreter.invoke()
```

4.3. Interpreting Output

The results of the inference can be obtained by *interpreter.get_tensor()*:

```
output_list = []
for output_index in range(len(output_details)):
    tensor_output = interpreter.get_tensor(output_index)
    output_list.append(tensor_output)
```

The output tensors generated by inference API can be used in the post-processing parts depending on different scenarios.

NETINT Technologies Level B: Confidential - Distribution Restricted

5. Example of Python Inference API Programming

This section provides an example of using the Quadra AI Python Inference API.

In the demo case, we use yolov4-tiny model to present whole procedures.

```
import netint.network
img = pre_process(img,mean,scale)

interpreter = netint.network.Interpreter(model_path=network_binary,
        dev_id=device_id)

input_details = interpreter.get_input_details()
output_details = interpreter.get_output_details()

# input format is image
interpreter.set_tensor(0, img)

interpreter.invoke()

data_list = []
for output_index in range(len(output_details)):
    tensor_output = interpreter.get_tensor(output_index)
    data_list.append(tensor_output)
    np.savetxt('{}/output_{}.tensor'.format(output_dir, output_index),
            tensor_output.reshape(-1), fmt='%.6f')
```

NETINT © 2023. LEVEL B: CONFIDENTIAL-DISTRIBUTION RESTRICTED

6. Reference

For detailed information about NETINT AI Toolkit, please refer to the user guide.

The demo code of Quadra AI Python Inference API can be found at:

```
~/NETINT_toolkit/netint_wrapper/scripts/
```

A set of model examples of using NETINT AI Toolkit can be found at:

```
~/NETINT_toolkit/examples/
```

NETINT Technologies Inc. Level B: Confidential - Distribution Restricted