Model Compiler Suite for Aries

Developers Guide

version 0.6

August 10, 2022

****

Important Notice

Mobilint Inc. reserves the right to make changes to the information in this publication at any time without prior notice. All information provided is for reference purpose only. Mobilint assumes no responsibility for possible errors or omissions, or for any consequences resulting from the use of the information contained herein.

This publication on its own does not convey any license, either express or implied, relating to any Mobilint and/or third-party products, under the intellectual property rights of Mobilint and/or any third parties.

Mobilint makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does Mobilint assume any liability arising out of the application or use of any product or circuit and specifically disclaims any and all liability, including without limitation any consequential or incidental damages.

Customers are responsible for their own products and applications. "Typical" parameters can and do vary in different applications. All operating parameters, including "Typicals" must be validated for each customer application by the customer's technical experts.

Mobilint products are not designed, intended, or authorized for use in applications intended to support or sustain life, or for any other application in which the failure of the Mobilint product could reasonably be expected to create a situation where personal injury or death may occur. Customers acknowledge and agree that they are solely responsible to meet all other legal and regulatory requirements regarding their applications using Mobilint products notwithstanding any information provided in this publication. Customer shall indemnify and hold Mobilint and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, expenses, and reasonable attorney fees arising out of, either directly or indirectly, any claim (including but not limited to personal injury or death) that may be associated with such unintended, unauthorized and/or illegal use.

**WARNING** No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electric or mechanical, by photocopying, recording, or otherwise, without the prior written consent of Mobilint. This publication is intended for use by designated recipients only. This publication contains confidential information (including trade secrets) of Mobilint protected by Competition Law, Trade Secrets Protection Act and other related laws, and therefore may not be, in part or in whole, directly or indirectly publicized, distributed, photocopied or used (including in a posting on the Internet where unspecified access is possible) by any unauthorized third party. Mobilint reserves its right to take any and all measures both in equity and law available to it and claim full damages against any party that misappropriates Mobilint’s trade secrets and/or confidential information.

****

| Address

7F, Teheran-ro 19-gil 5, Gangnam-guSeoul, Republic of Korea 06133

| Email

contact@mobilint.co

| Phone

+82) 2-552-9660

| Homepage

https://www.mobilint.co

**INTELLIGENCE EVERYWHERE**

Document Revision History

| Doc Revision Number | Date | Description |
| --- | --- | --- |
| 0.6 | August 10, 2022 | Revised for v0.6 |
| 0.5 | July 1, 2022 | Revised for v0.5 |
| 0.4 | February 23, 2022 | Revised for v0.4 |
| 0.3 | February 5, 2022 | Revised for v0.3 |
| 0.2 | December 1, 2021 | Revised for v0.2 |

Table of Contents

[1. Introduction 6](#_Toc130389932)

[2. Changelog 7](#_Toc130389933)

[2.1 qubee v0.6 (August 2022) 7](#_Toc130389934)

[2.2 qubee v0.5 (July 2022) 7](#_Toc130389935)

[2.3 qubee v0.4 (February 2022) 7](#_Toc130389936)

[2.4 qubee v0.3 (February 2022) 7](#_Toc130389937)

[2.5 qubee v0.2 (December 2021) 7](#_Toc130389938)

[3. Installation 8](#_Toc130389939)

[3.1 System requirements 8](#_Toc130389940)

[3.2 SDK installation 8](#_Toc130389941)

[3.2.1 Building docker image 8](#_Toc130389942)

[3.2.2 installation of qubee 8](#_Toc130389943)

[4. Tutorials 9](#_Toc130389944)

[4.1 Preparing calibration data 9](#_Toc130389945)

[4.2 Compiling ONNX models 9](#_Toc130389946)

[4.3 Compiling PyTorch models 10](#_Toc130389947)

[4.4 Compiling Keras models 10](#_Toc130389948)

[4.5 Compiling TensorFlow models 11](#_Toc130389949)

[5. Supported Frameworks 12](#_Toc130389950)

[5.1 Supported operations (ONNX) 12](#_Toc130389951)

[5.2 Supported operations (PyTorch) 12](#_Toc130389952)

[5.3 Supported operations (TensorFlow) 12](#_Toc130389953)

[5.4 Supported operations (Keras) 12](#_Toc130389954)

[6. API Reference 13](#_Toc130389955)

[6.1 Model\_Dict Class 13](#_Toc130389956)

[6.2 Method detail 13](#_Toc130389957)

[7. Open Source License Notice 14](#_Toc130389958)

[8. Copyright 15](#_Toc130389959)

List of Figures

Figure Title Page

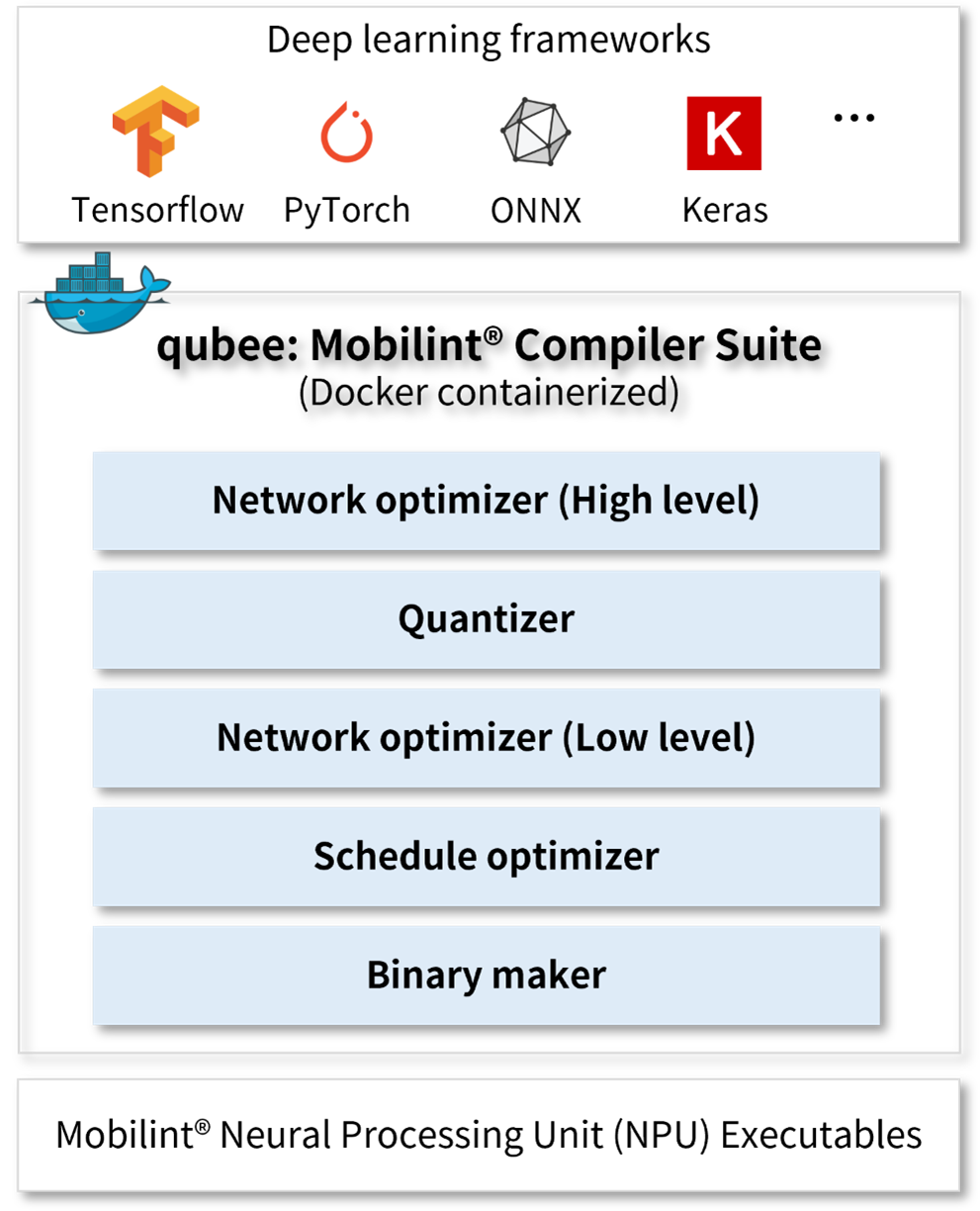
[Figure 1‑1. SDK Components 6](#_Toc130389960)

[Figure 1‑2. Input and output of qubee 6](#_Toc130389961)

[Figure 5‑1. Supported deep-learning frameworks 12](#_Toc130389962)

# Introduction

Mobilint® Model Compiler (i.e., Compiler) is a tool that converts models fromdeep learning frameworks (ONNX, PyTorch, Keras, TensorFlow, etc...) into Mobilint®Model eXeCUtable (i.e., MXQ), a format executable by Mobilint® Neural ProcessingUnit (NPU). This is the manual for the qubee, Mobilint's SDK. In this manual,you can leran how to use the SDK, what kind of frameworks does it support, etc.A set of functions that can be used to interact with the SDK will be given below.



1. Figure 1‑1. SDK Components

Input to the SDK is a trained deep learning model, its input shape, andcalibration data. SDK will return MXQ (compiled model) as an output.



1. Figure 1‑2. Input and output of qubee

# Changelog

## qubee v0.6 (August 2022)

Minor updates

## qubee v0.5 (July 2022)

Docker

Conda -> Virtualenv

Python: 3.7.7 -> 3.8.10

torch: 1.8.1 -> 1.10.1

tensorflow: 1.15.0 -> 2.3.0

onnx:1.6.0 -> 1.11.0

Parser

Code refactoring

API

Enable saving sample inference results (inputs and outputs)

## qubee v0.4 (February 2022)

Optimizer

Minor updates in fusing reshape

## qubee v0.3 (February 2022)

Parser

Identify preprocess and postprocess of the model

Exclude preprocess and postprocess if they are unsupported by the NPU

API

Simulate integer inference in Python API

## qubee v0.2 (December 2021)

First release

# Installation

## System requirements

In order to use the qubee, the NVIDIA GPU is required. CPU version qubee willbe provided in the future.

Reference System

|  |
| --- |
| Ubuntu 18.04.6 LTS  NVIDIA Graphics Driver 465.19.01 |

Requirement packages

|  |
| --- |
| NVIDIA Graphics Driver 450.80.02 **or** Above  Docker  nvidia-docker |

## SDK installation

We recommend installing qubee on the mobilint docker container.(Docker image: (mobilint/qbcompiler:v0.4)[https://hub.docker.com/r/mobilint/qbcompiler])

### Building docker image

Run the following commands to build the docker image.

|  |
| --- |
| $ *# Docker image download*  $ docker pull mobilint/qbcompiler:v0.4  $ *# Make a docker container*  $ docker run -it --gpus all --name mxq\_compiler -v $(pwd):/data mobilint/qbcompiler:v0.4 |

### installation of qubee

Run the following commands to install qubee on the docker container.

|  |
| --- |
| $ *# Download qubee-0.6-py3-none-any.whl file*  $ *# Copy qubee-0.6-py3-none-any.whl file to Docker*  $ docker cp /path/to/qubee-0.6-py3-none-any.whl mxq\_compiler:/  $ *# Start docker*  $ docker start mxq\_compiler  $ *# Attach docker*  $ *# Install qubee*  $ cd /  $ python -m pip install qubee-0.6-py3-none-any.whl |

# Tutorials

The tutorials below go through preparing calibration dataset, model compileand inference steps.

## Preparing calibration data

This step makes calibration data txt file for quantization. This step isrequired before compiling the model.

|  |
| --- |
| **from** qubee.utils **import** list\_np\_files\_in\_txt  target\_folder = 'imagenet\_cali\_npy' *# path to the folder with NumPy calibration data*  dataset\_txt\_path = 'cal\_image\_test.txt' *# path where calibration .txt file will be saved*  **list\_np\_files\_in\_txt**(dir\_path=target\_folder, save\_txt\_path=dataset\_txt\_path) |

## Compiling ONNX models

ONNX model can be parsed in two different ways. The first one just directlyparses the ONNX model, converts it to Mobilint IR. The second one converts theONNX model to TVM, parses it, and converts it to Mobilint IR. Once the model isconverted into Mobilint IR, then it will be compiled into MXQ.

|  |
| --- |
| *""" Compile ONNX model, first way """*  **import** qubee  **import** os  **import** wget  input\_shape = (224, 224, 3)  data\_path = 'cal\_image\_test.txt'  *### get resnet18 onnx model*  model\_url = 'https://github.com/onnx/models/raw/main/vision/classification/resnet/model/resnet18-v1-7.onnx'  onnx\_model\_path = wget.**detect\_filename**(model\_url)  **if** os.path.**isfile**(onnx\_model\_path):  **print**('Found cached model: {}'.**format**(onnx\_model\_path))  **else**:  **print**('Downloading model: {}'.**format**(model\_url))  onnx\_model\_path = wget.**download**(model\_url)  *### parse ONNX model and compile it*  model = qubee.**Model\_Dict**(onnx\_model\_path, backend='onnx')  model.**compile**(model\_nickname='resnet18', calib\_txt\_path=data\_path,  save\_path='resnet18.mxq') |
| *""" Compile ONNX model, second way """*  **import** qubee  **import** os  **import** wget  input\_shape = (224, 224, 3)  data\_path = 'cal\_image\_test.txt'  *### get resnet18 onnx model*  model\_url = 'https://github.com/onnx/models/raw/main/vision/classification/resnet/model/resnet18-v1-7.onnx'  onnx\_model\_path = wget.**detect\_filename**(model\_url)  **if** os.path.**isfile**(onnx\_model\_path):  **print**('Found cached model: {}'.**format**(onnx\_model\_path))  **else**:  **print**('Downloading model: {}'.**format**(model\_url))  onnx\_model\_path = wget.**download**(model\_url)  *### convert ONNX model to TVM IR, parse it and compile it*  model = qubee.**Model\_Dict**(onnx\_model\_path, backend='tvm')  model.**compile**(model\_nickname='resnet18', calib\_txt\_path=data\_path,  save\_path='resnet18.mxq') |

## Compiling PyTorch models

PyTorch model can be parsed in two different ways. First, one converts to ONNX,parses it, and converts to Mobilint IR. The second one converts to TVM, parses it,and converts to Mobilint IR. Once the model is converted to Mobilint IR, then itwill be compiled into MXQ.

|  |
| --- |
| *""" Compile PyTorch model, first way """*  **import** qubee  **from** qubee.utils **import** convert\_pytorch\_to\_onnx  **import** torchvision  input\_shape = (224, 224, 3)  data\_path = 'cal\_image\_test.txt'  *### get resnet18 from torchvision and convert it to ONNX*  torch\_model = torchvision.models.**resnet18**(pretrained=True)  onnx\_model\_path = 'resnet18.onnx'  **convert\_pytorch\_to\_onnx**(torch\_model, input\_shape, onnx\_model\_path)  *### parse ONNX model and compile it*  model = qubee.**Model\_Dict**(onnx\_model\_path, backend='onnx')  model.**compile**(model\_nickname='resnet18', calib\_txt\_path=data\_path,  save\_path='resnet18.mxq') |
| *""" Compile PyTorch model, second way """*  **import** qubee  **import** torchvision  input\_shape = (224, 224, 3)  data\_path = 'cal\_image\_test.txt'  *### get resnet18 from torchvision*  torch\_model = torchvision.models.**resnet18**(pretrained=True)  *### convert PyTorch model to TVM IR, parse it and compile it*  model = qubee.**Model\_Dict**(torch\_model, backend='tvm', input\_shape=input\_shape)  model.**compile**(model\_nickname='resnet18', calib\_txt\_path=data\_path,  save\_path='resnet18.mxq') |

## Compiling Keras models

Keras model will be to TVM, which will be parsed and converted to Mobilint IR.Once the model is converted to Mobilint IR, then it will be compiled into MXQ.

|  |
| --- |
| *""" Compile Keras model """*  **import** qubee  **import** tensorflow.keras as keras  input\_shape = (224, 224, 3)  data\_path = 'cal\_image\_test.txt'  *### get resnet50 from keras model zoo*  keras\_model = keras.applications.resnet50.**ResNet50**()  *### convert Keras model to TVM IR, parse it and compile it*  model = qubee.**Model\_Dict**(keras\_model, backend='tvm')  model.**compile**(model\_nickname='resnet50', calib\_txt\_path=data\_path,  save\_path='resnet50.mxq') |

## Compiling TensorFlow models

qubee supports TensorFlow up to version 1.15. So, it requires a frozenTensorFlow PB graph as input, which will be parsed and converted to Mobilint IR.Once the model is converted to Mobilint IR, then it will be compiled into MXQ.

|  |
| --- |
| *""" Compile Tensorflow model """*  **import** qubee  **import** wget  **import** os  input\_shape = (224, 224, 3)  data\_path = 'cal\_image\_test.txt'  *### download tensorflow resnet50 from zenodo website*  tf\_model = 'resnet50\_v1.pb'  **if** os.path.**isfile**(tf\_model):  **print**('Found cached model: {}'.**format**(tf\_model))  **else**:  **print**('Downloading model: {}'.**format**(tf\_model))  tf\_model = wget.**download**('https://zenodo.org/record/2535873/files/resnet50\_v1.pb')  *### parse tensorflow model and compile it*  model = qubee.**Model\_Dict**(tf\_model, backend='tf')  model.**compile**(model\_nickname='resnet50', calib\_txt\_path=data\_path,  save\_path='resnet50.mxq') |

# Supported Frameworks

We support almost all the commonly used Machine Learning frameworks & libraries,such as ONNX, TVM, PyTorch, Keras, and TensorFlow.



1. Figure 5‑1. Supported deep-learning frameworks

## Supported operations (ONNX)

## Supported operations (PyTorch)

## Supported operations (TensorFlow)

## Supported operations (Keras)

# API Reference

## Model\_Dict Class

## Method detail

# Open Source License Notice

Apache TVM

https://github.com/apache/tvm

Apache 2.0 License

PyTorch

https://github.com/pytorch/pytorch

BSD-like License

TensorFlow

https://github.com/tensorflow/tensorflow

Apache 2.0 License

ONNX

https://github.com/onnx/onnx

Apache 2.0 License

ONNX Runtime

https://github.com/microsoft/onnxruntime

MIT License

Keras

https://github.com/keras-team/keras

Apache 2.0 License

# Copyright

Copyrightⓒ 2019-present, Mobilint, Inc. All rights reserved.