PyNetsPresso and LaunchX

: An Integrated Toolchain for HW-Aware AI Model Optimization and Benchmarking

Jong Won Baek | NetsPresso Product Owner
# Table of Contents

1. Introduction  
2. Problems  
3. Solutions  
4. Success Cases  
5. Demo
Introduction: The Journey of Edge AI Engineer

It starts from AI model, to deployment

The Scope of Edge AI Engineer

1. Project Scope Definition
2. Data Acquisition
3. Modeling
4. Compress
5. Compile
6. Benchmark
7. Package
8. Deploy
Things Don’t Always Go As Planned

Source: lesswrong.com
Problems: What’s happening on the journey?

- Various types of SDKs
- Operation not supported
- Out of Memory
- Latency is too high
- Target Hardware A
- Target Hardware B
- Target Hardware C
What’s happening on the journey?

If the model does not work, you get back to square one
You need to go through the whole thing again to make your model work on the device.

Or, choose better solution
Solution: NetsPresso

The easiest tool for Edge AI engineers

Challenges

- Diverse Edge AI SDK
- Unsupported layers and operations
- Limited memory size
  - Latency is too high

Solution

- Unified Interface
- Operator Conversion
- AI Model Compression
NetsPresso® simplifies AI model optimization for target devices with automated processes.
Model Searcher Module: Automated Model Search for Your Device

The Model Searcher module provides models optimized for your target device.

- Optimized model design for target hardware
- Multiple models with various options
- 5X faster model
- Produce models close to production level based on actual hardware testing
- Create models with lower latency

Dataset → Model Searcher → Optimized AI Model
Model Compressor Module: Balancing Performance and Efficiency

Achieve optimal performance-efficiency tradeoffs with the Model Compressor module.

- Supports all CNN architectures (limitedly transformer models)
- Recommends optimal compression ratios
- Eliminates months of paper implementation period
- Minimal loss of information
Model Launcher Module: Swift Launch for Accelerated Models

Accelerate model deployment with the Model Launcher module.

- Unified interface for various SDKs
- Performance benchmarks and recommendations on actual devices
- Conversion to operations supported by target hardware
- Production-ready package

Model Launcher

Model converting

Benchmarks on device

<table>
<thead>
<tr>
<th>Latency</th>
<th>Memory</th>
<th>Power consumption</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Pre-processing
Inference engine
Post-processing

Production-ready Package

Optimized AI Model
Challenges vs NetsPresso

**Challenges**

- Various types of SDKs
- Diverse model conversion options
- Out of Memory
- Latency is too high
- Operation not supported

**Build & Search**
- Model Searcher

**Compress & Accelerate**
- Model Compressor

**Convert & Deploy**
- Model Launcher

- ✓
- ✓
- ✓
- ✓
- ✓
NetsPresso facilitates seamless transitions from optimization to deployment.
Success Cases

**With Renesas**
- Target: RA6M3 (ARM Cortex-M4)
- Model: YOLOv2
- Replace unsupported operation to be supported and HW-aware compression.
- Offering an extensive array of options to cater to diverse customer preferences.

**With Sony**
- Target: IMX500
- Model: YOLO based model
- IMX500 is commonly used for smart city use cases.
- While preserving latency, Nota maximizes the capacity of the model so we could train higher performing model.

**Success Cases**

<table>
<thead>
<tr>
<th>Without Nota</th>
<th>With Nota</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cannot operate DET model</td>
<td>Enabling</td>
</tr>
</tbody>
</table>

DET Model with 31.4% mAP

10% More Accurate

DE Model with 41.1% mAP
Success Cases

With STMicroelectronics

- Target: STM32H747I-DISCO
- Model: Mobilenet v2 (Tensorflow Flower Dataset)
- Significant benefits in terms of memory and latency optimization with minimal performance loss
- This experiment was conducted by engineers of STMicroelectronics (sample code will be released)
Success Cases

With STMicroelectronics

- 0.8% drop
- 14% lighter
- 61% lighter
- 24% faster

Basic pruning options were used
(Global pruning ratio 0.5, using recommended layer-wise pruning ratio)
Ecosystem for Edge AI Development

With integration to model/device ecosystem, NetsPresso can bring benefit to broader area.
Demo: PyNetsPresso and LaunchX

PyNetsPresso

Streamline your AI model optimization workflow

YOLOX | YOLOv8 | YOLOv7 | YOLOv5 | PIDNet | PyTorch-CIFAR-Models

python 3.8 | 3.9 | 3.10 | TensorFlow 2.3.x | 2.8.x | PyTorch 1.11.x | 1.13.x

NetsPresso | Open in Website | ModelZoo | Open in Github | Best Practice | Open in Colab

LaunchX

Convert
Convert AI models to the target framework to optimize the model or the device.
Check the Supported Model Range.

Benchmark
Measure the performance of the AI model on the target device.

* You can see the real demo at Nota AI booth
import getpass
from netspresso.client import SessionClient
from netspresso.compressor import ModelCompressor

email = 'xxx.yyy@st.com'
print('Enter you password')
password = getpass.getpass()

session = SessionClient(email=email, password=password)
compressor = ModelCompressor(user_session=session)

Enter you password

2023-10-27 09:42:35.715 | INFO | netspresso.client::__login:50 - Login successfully
2023-10-27 09:42:37.787 | INFO | netspresso.client::__get_user_info:67 - successfully got user information
from netspresso.compressor import Task, Framework

model = compressor.upload_model(
    model_name='model',
    tasks=Task.IMAGE_CLASSIFICATION,
    framework=Framework.TENSORFLOW_KERAS,
    file_path='experiments_outputs/training/saved_models/best_model.h5',
    input_shapes=[{'batch': 1, 'channel': 3, 'dimension': [128, 128]}])

2023-10-27 09:52:50.120 | INFO | netspresso.compressor:upload_model:72 - Uploading Model...
import os
from netspresso.compressor import CompressionMethod
from netspresso.compressor import RecommendationMethod

if not os.path.exists('experiments_outputs/compressed_models'):
    os.makedirs('experiments_outputs/compressed_models')

compressed_model = compressor.recommendation_compression(
    model_id='94d358dc-0297-4964-aa4b-d033b93ffcc9f',
    model_name='compressed_model.h5',
    compression_method=CompressionMethod.PR_L2,
    recommendation_method=RecommendationMethod.SLAMP,
    recommendation_ratio=0.5,
    output_path='experiments_outputs/compressed_models/compressed_model.h5',
)

<table>
<thead>
<tr>
<th>Date</th>
<th>Time</th>
<th>Level</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>2023-10-27 00:53:59.084</td>
<td>INFO</td>
<td></td>
<td>netspresso.compressor:recommendation_compression:448 - Compressing recommendation-based model...</td>
</tr>
<tr>
<td>2023-10-27 00:54:19.376</td>
<td>INFO</td>
<td></td>
<td>netspresso.compressor:download_model:223 - Downloading model...</td>
</tr>
<tr>
<td>2023-10-27 00:54:31.058</td>
<td>INFO</td>
<td></td>
<td>netspresso.compressor:download_model:226 - Download model successfully. Local Path: experiments_outputs/compressed_models/compressed_model.h5</td>
</tr>
<tr>
<td>2023-10-27 00:54:31.063</td>
<td>INFO</td>
<td></td>
<td>netspresso.compressor:get_model:196 - Getting model...</td>
</tr>
<tr>
<td>2023-10-27 00:54:33.538</td>
<td>INFO</td>
<td></td>
<td>netspresso.compressor:recommendation_compression:505 - Recommendation compression successfully. Compressed Model ID: 49ac11c4-2c56-4575-b215-b7f184ca4693</td>
</tr>
<tr>
<td>2023-10-27 00:54:33.542</td>
<td>INFO</td>
<td></td>
<td>netspresso.compressor:recommendation_compression:506 - 50 credits have been consumed.</td>
</tr>
</tbody>
</table>
LaunchX

Convert
Convert AI models to the target framework to optimize the model on the device.
Check the Supported Model Range.

Select a Converting case

ONNX to TensorRT
Convert ONNX models for execution with TensorRT.

ONNX to TensorFlow Lite
Convert ONNX models for execution with TensorFlow Lite.

ONNX to OpenVINO
Convert ONNX models for execution with OpenVINO.

TensorFlow-Keras to TensorFlow Lite
Convert TensorFlow models for execution with TensorFlow Lite.

Benchmark
Measure the performance of the AI model on the target device.

Select a target device to benchmark

Renesas RA8D1 (Arm Cortex-M85)
Alif Ensemble DevKit Gen2 (Arm Cortex-M55+Ethos-U55)

Raspberry Pi Zero
Raspberry Pi 2W

Raspberry Pi 3B+
Raspberry Pi 4B

NVIDIA Jetson Nano
NVIDIA Jetson Xavier NX

NVIDIA Jetson TX2
NVIDIA Jetson AGX Orin

NVIDIA AWS-T4
LaunchX

AI Model Converter

File name: lightweight-yolox-nano.onnx

Input Shape
- Batch size: 1
- Channel: 3
- Input size: 416, 416

Target Framework
- TensorFlow Lite
- TensorRT
- OpenVINO

Target device: Ensemble-E7-DevKit-Gen2

Output datatype
- FP16
- INT8
LaunchX

AI Model Benchmarker

Result

File name: model_test.tflite

Output format

<table>
<thead>
<tr>
<th>Tensorflow</th>
<th>Data type</th>
<th>Batchsize</th>
<th>Input size</th>
<th>Channel</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lite</td>
<td>None</td>
<td>1</td>
<td>416, 416</td>
<td>3</td>
</tr>
</tbody>
</table>

Inference latency (avg)

64.391 ms

File size: 0.96 MB

Device Info

Ensemble-E7-DevKit-Gen2

- Library: Python 3.10.12 NumPy 1.26.1 TFLite 2.9.1
- CPU: Arm Dual-core CORTEX-M55 up to 400 MHz, with Helium, Dual-core ETHOS-U55 NPU
- GPU: N/A
- RAM: 15.827 GB

* You can do the same task with PyNetsPresso
Let’s do happy edge AI engineering with NetsPresso!
Copyright Notice

This presentation in this publication was presented as a tinyML® Asia Technical Forum. The content reflects the opinion of the author(s) and their respective companies. The inclusion of presentations in this publication does not constitute an endorsement by tinyML Foundation or the sponsors.

There is no copyright protection claimed by this publication. However, each presentation is the work of the authors and their respective companies and may contain copyrighted material. As such, it is strongly encouraged that any use reflect proper acknowledgement to the appropriate source. Any questions regarding the use of any materials presented should be directed to the author(s) or their companies.

tinyML is a registered trademark of the tinyML Foundation.

www.tinyml.org