PyArm NN Workshop

Karl Fezer, Henri Woodcock
22nd February 2021
## AI Virtual Tech Talks Series

<table>
<thead>
<tr>
<th>Date</th>
<th>Title</th>
<th>Host</th>
</tr>
</thead>
<tbody>
<tr>
<td>February 23rd</td>
<td>Hands-on with PyArmNN for object detection</td>
<td>Arm Workshop</td>
</tr>
<tr>
<td>March 9th</td>
<td>Automate tinyML Development &amp; Deployment with Qeexo AutoML</td>
<td>Qeexo</td>
</tr>
<tr>
<td>March 30th</td>
<td>Thermal Vision AI – Creating TensorFlow Models for Thermal Image Classification</td>
<td>OpenMV</td>
</tr>
<tr>
<td>April 13th</td>
<td>0The First No-Code Voice AI Platform for Arm Cortex-M-based Microcontrollers</td>
<td>Picovoice</td>
</tr>
</tbody>
</table>

Visit: developer.arm.com/solutions/machine-learning-on-arm/ai-virtual-tech-talks
Presenters – Arm AI Ecosystem Evangelism

Karl Fezer

Henri Woodcock
Arm Software Stack

Applications

High Level Frameworks
(e.g. TensorFlow Lite, PyTorch, Android NNAPI)

Tools

Dynamic Workload Compilers & Drivers

CPU
Cortex-A
Neoverse

GPU
Mali

NPU
Ethos-N

© 2021 Arm
For in depth on ArmNN & Compute Library
See Feb 9th Tech Talk

Debian Packages for Arm NN and ACL

- 20.08 Release of Arm SW stack available on Ubuntu Launchpad PPA. Formal release in Bullseye (Debian 11)
- Benefits:
  - Reliable: All build dependencies are taken care of by the robust Debian packaging infrastructure
  - Quick setup: Ready for prototyping with just a few ‘apt-get’ commands
  - Accessible: Ubuntu ‘snappy’ now available for Raspberry Pi with full Lubuntu desktop experience
Why Use the Arm NN and Arm Compute Library?

Versatile and Portable:
- Easily target multiple platforms (CPU, GPU and NPU) from a single code base
- Reduce overall development time, keep using existing framework and tools
- Deployable for Android, Linux and ‘bare metal’ applications

Superior Performance:
- Best in class across a wide range of popular networks
- Uses advanced network optimization techniques, workload tuning and GEMM heuristics

Arm Specific Optimizations:
- Outperforms generic math and ML libraries due to Arm specific optimization
- Specific architectures (e.g. dot product for Armv8.2A) and micro architecture optimizations (e.g. Cortex-A53)
- Quick adoption of new Arm technologies e.g. SVE, SVE2
Arm NN Integration Options with Neural Network Frameworks

- Multiple framework integration options for Linux and Android

- Model file support via parser
- Runtime support
Debian Packages for Arm NN and ACL

- 20.08 Release of Arm SW stack available on Ubuntu Launchpad PPA. Formal release in Bullseye (Debian 11)

- Benefits:
  - Reliable: All build dependencies are taken care of by the robust Debian packaging infrastructure
  - Quick setup: Ready for prototyping with just a few ‘apt-get’ commands
  - Accessible: Ubuntu ‘Groovy’ now available for Raspberry Pi with full Aarch64 desktop experience
PyArmNN

Rapid developing and prototyping

- Python APIs for Arm NN inference engine
- Compatible with Numpy arrays
- Minimal overhead – thin C++ API
- Easy to install using Debian packages

```python
Create a parser object and load your model file.
import pyarmnn as ann
import imageio
parser = ann.ITfLiteParser()
network = parser.CreateNetworkFromBinaryFile('./model.tflite')

Get the input binding information by using the name of the input layer.
input_binding_info = parser.GetNetworkInputBindingInfo(0, 'model/input')
options = ann.CreationOptions()
runtime = ann.IRuntime(options)

Choose preferred backends for execution and optimize the network.
preferredBackends = [ann.BackendId('CpuAcc'), ann.BackendId('CpuRef')]
opt_network, messages = ann.Optimize(network, preferredBackends,
runtime.GetDeviceSpec(),
ann.OptimizerOptions())
net_id, _ = runtime.LoadNetwork(opt_network)

Make workload tensors using input and output binding information.
img = imageio.imread('./image.png')
input_tensors = ann.make_input_tensors([input_binding_info], [img])
output_binding_info = parser.GetNetworkOutputBindingInfo(0, 'model/output')
output_tensors = ann.make_output_tensors([output_binding_info])

Perform inference and get the results back into a numpy array.
runtime.EnqueueWorkload(net_id, input_tensors, output_tensors)
results = ann.workload_tensors_to_ndarray(output_tensors)
print(results)
```
Arm Public Model Zoo

- Highly optimized networks specific to Arm architectures
- Clustering, pruning and quantization aware training used to produce the most efficient models
- Model, meta data and test data available
- Supported by code samples and How-To guides

<table>
<thead>
<tr>
<th>Model</th>
<th>Improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSD MobileNet v1</td>
<td>Revised for 30% Arm NN performance uplift</td>
</tr>
<tr>
<td>MobileNet v2</td>
<td>Clustering and pruning for better accuracy at the same sized</td>
</tr>
<tr>
<td>Wav2Letter</td>
<td>10x faster, tuned to mobile / embedded voice UI</td>
</tr>
<tr>
<td>DS-CNN</td>
<td>Revised for updated tooling and clustering</td>
</tr>
<tr>
<td>Yolo V3</td>
<td>Simplified classes and BB, detector quantization compatible</td>
</tr>
</tbody>
</table>
Useful Links

• Arm NN HAL for Android: https://github.com/ARM-software/android-nn-driver
• Arm NN TFLite Delegate: https://github.com/ARM-software/armnn/tree/branches/armnn_20_11/delegate
• Arm Model Zoo: https://github.com/ARM-software/ML-zoo/tree/master/models
• Arm ML Examples: https://github.com/ARM-software/ML-examples/
• How to Guides: https://developer.arm.com/solutions/machine-learning-on-arm/
• Ubuntu for Raspberry Pi: https://ubuntu.com/raspberry-pi
• Contributions: https://www.mlplatform.org/contributing/
Thank you!

Tweet us: @ArmSoftwareDev

Check out our Arm YouTube channel and our Arm Software Developers YouTube channel

Signup now for our next AI Virtual Tech Talk here

Attendees: don’t forget to fill out the survey to be in with a chance of winning an Arduino Nano 33 BLE board