MATLAB EXPO 2019

Deploying Deep Neural Networks to Embedded GPUs and CPUs

Abhijit Bhattacharjee
Deep Learning Workflow in MATLAB

Deep Neural Network Design + Training

Standalone Deployment
Deep Neural Network Design and Training

- Design in MATLAB
  - Manage large data sets
  - Automate data labeling
  - Easy access to models

- Training in MATLAB
  - Acceleration with GPU’s
  - Scale to clusters

MATLAB EXPO 2019
Application Design

Pre-processing $\rightarrow$ Process $\rightarrow$ Post-processing
Multi-Platform Deep Learning Deployment
Multi-Platform Deep Learning Deployment

Application logic

Desktop
- NVIDIA Jetson
- Raspberry pi
- Mobile
- Beaglebone

Data Center

Embedded

MATLAB EXPO 2019
Algorithm Design to Embedded Deployment Workflow

Conventional Approach

1. **Functional test**
   - High-level language
   - Deep learning framework
   - Large, complex software stack

2. **Deployment unit-test**
   - C/C++
   - Low-level APIs
   - Application-specific libraries

3. **Deployment integration-test**
   - C/C++
   - Target-optimized libraries
   - Optimize for memory & speed

4. **Real-time test**
   - C/C++

**Challenges**
- Integrating multiple libraries and packages
- Verifying and maintaining multiple implementations
- Algorithm & vendor lock-in

MATLAB EXPO 2019
Solution: Use MATLAB Coder & GPU Coder for Deep Learning Deployment

Target Libraries

- NVIDIA TensorRT & cuDNN Libraries
- Intel MKL-DNN Library
- ARM Compute Library
Deep Learning Deployment Workflows

**INFERENCE ENGINE DEPLOYMENT**

Trained DNN

```
cnncodegen
```

Portable target code

**INTEGRATED APPLICATION DEPLOYMENT**

Pre-processing

```
codegen
```

Post-processing

Portable target code
Workflow for Inference Engine Deployment

1. Generate the code for trained model
   >> cncodegen(net, 'targetlib', 'arm-compute')

2. Copy the generated code onto target board

3. Use hand written main function to call inference engine

4. Generate the exe and test the executable
   >> make -C ./ ......
Deep Learning Inference Deployment

Pedestrian Detection

Target Libraries
- NVIDIA TensorRT & cuDNN Libraries
- Intel MKL-DNN Library
- ARM Compute Library
  - Includes ARM Cortex-A support

MATLAB Coder

Application logic

MATLAB

Pedestrian Detection

Includes ARM Cortex-A support
Deep Learning Inference Deployment

Blood Smear Segmentation

Target Libraries

- NVIDIA TensorRT & cuDNN Libraries
- Intel MKL-DNN Library
- ARM Compute Library

MATLAB Coder

MATLAB EXPO 2019
Deep Learning Inference Deployment

Target Libraries

- NVIDIA TensorRT & cuDNN Libraries
- Intel MKL-DNN Library
- ARM Compute Library

Application logic

GPU Coder

Defect Classification & Detection

MATLAB EXPO 2019
How is the Performance?
Performance of Generated Code

- CNN inference (ResNet-50, VGG-16, Inception V3) on Titan V GPU

- CNN inference (ResNet-50) on Jetson TX2

- CNN inference (ResNet-50, VGG-16, Inception V3) on Intel Xeon CPU
Single Image Inference on Titan V using cuDNN

Inference Speed - ResNet-50 (Img/Sec)

- TensorFlow XLA: 120
- PyTorch JIT: 123
- GPU Coder: 289
TensorRT Accelerates Inference Performance on Titan V

Single Image Inference with ResNet-50 (Titan V)

- **cuDNN**
- **TensorRT (FP32)**
- **TensorRT (INT8)**

Images per second

- TensorFlow
- GPU Coder

Intel® Xeon® CPU 3.6 GHz - NVIDIA libraries: CUDA10.0/1 - cuDNN 7.5.0 - TensorRT 5.1.2 - Frameworks: TensorFlow 1.13.1
Single Image Inference on Jetson TX2

NVIDIA libraries: CUDA10.0/1 - cuDNN 7.5.0 - TensorRT 5.1.2 - Frameworks: TensorFlow 1.13.1
CPU Performance

CPU, Single Image Inference (Linux)

Images/Sec

0 50 100 150 200 250

ResNet-50  VGG-16  Inception-V3  SqueezeNet

MATLAB  TensorFlow  MXNet  MATLAB Coder  PyTorch
**Brief Summary**

**DNN libraries are great for inference,** ...

MATLAB Coder and GPU Coder generates code that takes advantage of:

- NVIDIA® CUDA libraries, including TensorRT & cuDNN
- Intel® Math Kernel Library for Deep Neural Networks (MKL-DNN)
- ARM® Compute libraries for mobile platforms
Brief Summary

DNN libraries are great for inference, ...

MATLAB Coder and GPU Coder generates code that takes advantage of:

- NVIDIA® CUDA libraries, including TensorRT & cuDNN
- Intel® Math Kernel Library for Deep Neural Networks (MKL-DNN)
- ARM® Compute libraries for mobile platforms

But, Applications Require More than just Inference
Deep Learning Workflows: Integrated Application Deployment

Pre-processing \[\rightarrow\] codegen \[\rightarrow\] Post-processing

Portable target code
Lane and Object Detection using YOLO v2

**Workflow:**
1) Test in MATLAB on CPU
2) Generate code and test on desktop GPU
3) Generate code and test on Jetson AGX Xavier GPU
(1) Test in MATLAB on CPU

- **AlexNet-based Lane Detection**
- **YOLO v2 Object Detection**
- **Post-processing**
- **Strongest Bounding Box**

MATLAB EXPO 2019
(2) Generate Code and Test on Desktop GPU

MATLAB

- AlexNet-based Lane Detection
- Post-processing
- YOLO v2 Object Detection
- Strongest Bounding Box

CUDA optimized code

cuDNN/TensorRT optimized code
(3) Generate Code and Test on Jetson AGX Xavier GPU

MATLAB

AlexNet-based
Lane Detection

Post-processing

YOLO v2
Object Detection

Strongest Bounding Box

CUDA optimized code

cuDNN/TensorRT optimized code
Lane and Object Detection using YOLO v2

1) Running on CPU
2) 7X faster running generate code on desktop GPU
3) Generate code and test on Jetson AGX Xavier GPU
Accessing Hardware

Access Peripheral from MATLAB

Deploy Standalone Application

Processor-in-Loop Verification
Deploy to Target Hardware via Apps and Command Line

%% Define HSP settings

%% setup hardware object

%% setup codegen config object

cfg_hsp = coder.gpuConfig('exe');
cfg_hsp.Hardware = coder.hardware(hwObj.BoardPref);
buildDir = '~/buildDir';
cfg_hsp.Hardware.BuildDir = buildDir;

codegen -config cfg_hsp -args {im, coder.Constant(cnnMatFile)) alexnet_test

%% copy input and run the executable

%% execute on Jetson

%% copy the output file back to host machine
Inference Speed - ResNet-50 (Img/Sec)

TensorFlow XLA: 120
PyTorch JIT: 123
GPU Coder: 289
How does MATLAB Coder and GPU Coder achieve these results?
Coders Apply Various Optimizations

MATLAB

- Traditional compiler optimizations
- MATLAB

Loop optimizations
- Library function mapping
- Scalarization
- Loop perfectization
- Loop interchange
- Loop fusion
- Scalar replacement

CUDA kernel lowering
- Parallel loop creation
- CUDA kernel creation
- cudaMemcpy minimization
- Shared memory mapping
- CUDA code emission
Coders Apply Various Optimizations

- Traditional compiler optimizations
- MATLAB Library function mapping
- Parallel loop creation
- CUDA kernel creation
- cudaMemcpy minimization
- Shared memory mapping
- CUDA code emission

Optimized Libraries
- Loop optimizations
  - Loop fusion
  - Scalar replacement
  - Parallel loop
  - CUDA kernel
  - cudaMemcpy
  - Shared memory mapping
  - CUDA code emission

Network Optimization

Coding Patterns
- Loop optimizations
  - Loop fusion
  - Scalar replacement
  - Parallel loop
  - CUDA kernel
  - cudaMemcpy
  - Shared memory mapping
  - CUDA code emission
Generated Code Calls Optimized Libraries

Performance
1. Optimized Libraries
2. Network Optimizations
3. Coding Patterns

cuFFT, cuBLAS, cuSolver, Thrust Libraries

Pre-processing

Post-processing

Intel MKL-DNN Library

NVIDIA TensorRT & cuDNN Libraries

ARM Compute Library
Deep Learning Network Optimization

1. Optimized Libraries
2. Network Optimizations
3. Coding Patterns
Coding Patterns: Stencil Kernels

- Automatically applied for image processing functions (e.g. imfilter, imerode, imdilate, conv2, …)
- Manually apply using `gpucoder.stencilKernel()`

**Performance**
1. Optimized Libraries
2. Network Optimizations
3. Coding Patterns
Coding Patterns: Matrix-Matrix Kernels

- Automatically applied for many MATLAB functions (e.g. matchFeatures SAD, SSD, pdist, …)
- Manually apply using `gpucoder.matrixMatrixKernel()`
Deep Learning Workflow in MATLAB

1. Deep Neural Network Design + Training
   - Trained DNN

2. Application Design
   - Application logic

3. Standalone Deployment
   - ARM Compute Library
   - MKL-DNN Library
   - TensorRT and cuDNN Libraries
Deep Learning Workflow in MATLAB

Deep Neural Network Design + Training

Train in MATLAB

Trained DNN

Transfer learning

Model importer

Application Design

Application logic

Standalone Deployment

Coders

MKL-DNN Library

TensorRT and cuDNN Libraries

ARM Compute Library

Reference model
Call to action

- Visit the Deep Learning Booth!
- Related upcoming talks:
  - AI Techniques for Signals, Time-series, and Text Data
  - Sensor Fusion and Tracking for Autonomous Systems
  - Deploying Deep Neural Networks to Embedded GPUs and CPUs