MATLAB EXPO 2018

Deploying Deep Learning Networks to Embedded GPUs and CPUs

Rishu Gupta, PhD
Senior Application Engineer, Computer Vision
MATLAB Deep Learning Framework

- Manage large image sets
- Automate image labeling
- Easy access to models

- Acceleration with GPU’s
- Scale to clusters
Multi-Platform Deep Learning Deployment

Desktop

Data-center

Nvidia TX1, TX2, TK1

Raspberry pi

Mobile

Embedded

Beagle bone
Multi-Platform Deep Learning Deployment

- Need code that takes advantage of:
  - NVIDIA® CUDA libraries, including cuDNN and TensorRT
  - Intel® Math Kernel Library for Deep Neural Networks (MKL-DNN) for Intel processors
  - ARM® Compute libraries for ARM processors
Multi-Platform Deep Learning Deployment

- Need code that takes advantage of:
  - NVIDIA® CUDA libraries, including cuDNN and TensorRT
  - Intel® Math Kernel Library for Deep Neural Networks (MKL-DNN) for Intel processors
  - ARM® Compute libraries for ARM processors
Algorithm Design to Embedded Deployment Workflow

**Conventional Approach**

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

2. **C++**
   - Low-level APIs
   - Application-specific libraries

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

**Challenges**

- Integrating multiple libraries and packages
- Verifying and maintaining multiple implementations
- Algorithm & vendor lock-in
Solution - GPU Coder for Deep Learning Deployment

Target Libraries

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

Application logic

MATLAB EXPO 2018
Deep Learning Deployment Workflows

INFERENCE ENGINE DEPLOYMENT

Trained DNN

\[ \text{cnncodegen} \]

Portable target code

INTEGRATED APPLICATION DEPLOYMENT

Pre-processing

Trained DNN

\[ \text{cnncodegen} \]

Portable target code

Post-processing
**Workflow for Inference Engine Deployment**

**Steps for inference engine deployment**

1. Generate the code for trained model
   ```
   >> cnnncodegen(net, 'targetlib', 'cudnn')
   ```

2. Copy the generated code onto target board

3. Build the code for the inference engine
   ```
   >> make -C ./codegen -f ...mk
   ```

4. Use hand written main function to call inference engine

5. Generate the exe and test the executable
   ```
   >> make -C ./ ......
How to get a Trained DNN into MATLAB?

1. Idea
2. Train in MATLAB
3. Model importer
4. Reference model
5. Transfer learning
6. Trained DNN

MATLAB EXPO 2018
Deep Learning Inference Deployment

Train in MATLAB

Reference model

Transfer learning

Trained DNN

Target Libraries

Intel MKL-DNN Library

NVIDIA TensorRT & cuDNN Libraries

ARM NEON™ Compute Library
Building DNN from Scratch

Load Training Data

Build Layer Architecture

Set Training Options

Train Network

%% Create a datastore
imds = imageDatastore('Data',...
  'IncludeSubfolders',true,'LabelSource','foldernames');
num_classes = numel(unique(imds.Labels));

%% Build layer architecture
layers = [
  imageInputLayer([64 32 3])
  convolution2dLayer(5,20)
  reluLayer()
  maxPooling2dLayer(2,'Stride',2)
  fullyConnectedLayer(512)
  fullyConnectedLayer(2)
  softmaxLayer()
  classificationLayer()];

%% Set Training Options
trainOpts = trainingOptions('sgdm',...
  'MiniBatchSize', miniBatchSize,...
  'Plots', 'training-progress');

%% Train Network
net = trainNetwork(imds, layers, trainOpts);
Pedestrian Detection DNN Deployment on ARM Processor

```
layers = [imageInputLayer([64 32 3])
    convolution2dLayer(5,20)
    reluLayer()
    maxPooling2dLayer(2,'Stride',2)
    CrossChannelNormalizationLayer(5,'K',1);
    convolution2dLayer(5,20)
    reluLayer()
    maxPooling2dLayer(2,'Stride',2)
    fullyConnectedLayer(512)
    fullyConnectedLayer(2)
    softmaxLayer()
    classificationLayer()];
```
Pedestrian Detection DNN Deployment on ARM Processor

- ARM Neon instruction set architecture
  - Example: ARM Cortex A

- ARM Compute Library
  - Low-level Software functions
  - Computer vision, machine learning etc…

- Pedestrian detection on Raspberry pi
Deep Learning Inference Deployment

Train in MATLAB

Model importer

Trained DNN

Target Libraries

NVIDIA TensorRT & cuDNN Libraries

Intel MKL-DNN Library

ARM NEON™ ARM Compute Library

Keras TensorFlow Caffe

Reference model

Transfer learning
Importing DNN from Open Source Framework

Caffe Model Importer (including Caffe Model Zoo)

- `importCaffeLayers`
- `importCaffeNetwork`

```python
network = importCaffeNetwork(protofile, 'yolo.caffemodel');
```

TensorFlow-Keras Model Importer

- `importKerasLayers`
- `importKerasNetwork`
Deep Learning Inference Deployment

Reference model

Object Detection

Model importer

Trained DNN

Target Libraries

NVIDIA
TensorRT & cuDNN Libraries

Intel MKL-DNN Library

ARM Compute
Library

MATLAB EXPO 2018
Deep Learning Inference Deployment

Train in MATLAB

Reference model → Transfer learning → Trained DNN

Model importer

Target Libraries

NVIDIA TensorRT & cuDNN Libraries

ARM Compute Library

Intel MKL - DNN Library

Keras TensorFlow Caffe

MATLAB EXPO 2018
Layered Architecture for Segnet- Semantic Segmentation

Complete Layer Graph

DAG Network
Total number of layers: 91
NVIDIA TensorRT
PROGRAMMABLE INference ACCELERATOR

TensorRT

Layer & Tensor Fusion
Precision Calibration
Kernel Auto-Tuning

Trained Neural Network
Dynamic Tensor Memory
Multi-Stream Execution

Optimized Inference Engine

TESLA P4
DRIVE PX 2
JETSON TX2
NVIDIA DLA
TESLA V100

MATLAB EXPO 2018
Performance Summary (VGG-16) on TitanXP

MATLAB on TitanXP: 72 Fps
GPU Coder (cuDNN) on TitanXP: 175 Fps
GPU Coder (TensorRT) on TitanXP: 210 Fps
GPU Coder (TensorRT-int8) on TitanXP: 345 Fps
How Good is Generated Code Performance?

- Performance of CNN inference (Alexnet) on Titan XP GPU

- Performance of CNN inference (Alexnet) on Jetson (Tegra) TX2
Alexnet Inference on NVIDIA Titan Xp

**Testing platform**

- **CPU**: Intel(R) Xeon(R) CPU E5-1650 v4 @ 3.60GHz
- **GPU**: Pascal Titan Xp
- **cuDNN**: v7

**Software versions**

- **TensorFlow**: 1.6.0
- **MXNet**: 1.1.0
- **GPU Coder + cuDNN**: TensorFlow
- **GPU Coder + TensorRT (3.0.1)**
- **GPU Coder + TensorRT (3.0.1, int8)**

**Graph**

- Frames per second vs Batch Size
VGG-16 Inference on NVIDIA Titan Xp

Frames per second vs Batch Size

- GPU Coder + TensorRT (3.0.1, int8)
- GPU Coder + TensorRT (3.0.1)
- GPU Coder + cuDNN
- MXNet (1.1.0)
- TensorFlow (1.6.0)

Testing platform:
- CPU: Intel(R) Xeon(R) CPU E5-1650 v4 @ 3.60GHz
- GPU: Pascal Titan Xp
- cuDNN: v7
Alexnet Inference on Jetson TX2: Frame-Rate Performance

Frames per second

Batch Size

GPU Coder + TensorRT
GPU Coder + cuDNN
C++ Caffe (1.0.0-rc5)
Brief Summary

DNN libraries are great for inference, ...

- GPU coder generates code that takes advantage of:
  - NVIDIA® CUDA libraries, including cuDNN, and TensorRT
  - Intel® Math Kernel Library for Deep Neural Networks (MKL-DNN)
  - ARM® Compute libraries for mobile platforms
Brief Summary

DNN libraries are great for inference, ...

- GPU coder generates code that takes advantage of:

  NVIDIA® CUDA libraries, including cuDNN, and TensorRT

  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

INTEGRATED APPLICATION DEPLOYMENT

Pre-processing → Post-processing

codegen

Portable target code
Traffic sign detection and recognition

- **YOLO**
  - Object detection DNN

- **Recognition net**
  - Strongest Bounding Box
  - Classifier DNN
Traffic sign detection and recognition
Traffic sign detection and recognition

Frame Rate: 58.6075

Design phase: 25 Fps
Deployement: 60 Fps
GPU Coder Helps You Deploy Applications to GPUs Faster

- CUDA Kernel creation
- Memory allocation
- Data transfer minimization

- Library function mapping
- Loop optimizations
- Dependence analysis

- Data locality analysis
- GPU memory allocation

- Data-dependence analysis
- Dynamic memcpy reduction
CUDA Code Generation from GPU Coder app

Integrated editor and simplified workflow for code generation
Summary- GPU Coder

MATLAB algorithm (functional reference)

Build type

Call CUDA from MATLAB directly

Call CUDA from (C++) hand-coded main()

Deployment unit-test

Desktop GPU

.mex

Cross-compiled .lib

Embedded GPU

Functional test

Call CUDA from (C++) hand-coded main().

Deployment integration-test

Real-time test

Call CUDA from MATLAB directly

Call CUDA from (C++) hand-coded main()
MATLAB Deep Learning Framework

Access Data
- Manage large image sets
- Automate image labeling
- Easy access to models

Design + Train
- Acceleration with GPU’s
- Scale to clusters

DEPLOYMENT
- Intel MKL-DNN Library
- NVIDIA TensorRT & cuDNN Libraries
- ARM NEON Compute Library
• **Share your experience with MATLAB & Simulink on Social Media**
  - Use #MATLABEXPO
  - I use #MATLAB because………………………… Attending #MATLABEXPO
  - Examples
    - I use #MATLAB because it helps me be a data scientist! Attending #MATLABEXPO
    - Learning new capabilities in #MATLAB and #Simulink at #MATLABEXPO.

• **Share your session feedback:**
  Please fill in your feedback for this session in the feedback form