MATLAB EXPO 2018

Master Class: Deep Learning
Del Prototipo a su Despliegue en Entornos Embarcados

Lucas García
Deep Learning Demo

Image Classification
Why MATLAB for Deep Learning?

- MATLAB is Productive
- MATLAB Integrates with Open Source
- MATLAB is Fast
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

- **Deploy**
  - Automate compilation to GPUs and CPUs using GPU Coder:
    - 5x faster than TensorFlow
    - 2x faster than MXNet
Deep Learning Applications

Voice assistants (speech to text)
Teaching character to beat video game
Automatically coloring black-and-white images
What is Deep Learning?
Deep Learning

Model learns to perform classification tasks directly from data.
Deep Learning is Versatile

Detection of cars and road in autonomous driving systems

Rain Detection and Removal

Iris Recognition – 99.4% accuracy

2. Source: An experimental study of deep convolutional features for iris recognition Signal Processing in Medicine and Biology Symposium (SPMB), 2016 IEEE Shervin Minaee ; Amirali Abdolrashidiy ; Yao Wang; An experimental study of deep convolutional features for iris recognition
How is deep learning performing so well?
Deep Learning Uses a Neural Network Architecture

Input Layer

Hidden Layers (n)

Output Layer
Thinking about Layers

- Layers are like blocks
  - Stack them on top of each other
  - Replace one block with a different one

- Each hidden layer processes the information from the previous layer
Thinking about Layers

- Layers are like blocks
  - Stack them on top of each other
  - Replace one block with a different one
- Each hidden layer processes the information from the previous layer
- Layers can be ordered in different ways
Deep Learning in 6 Lines of MATLAB Code

1. Read an image to classify
Why MATLAB for Deep Learning?

- MATLAB is Productive
- MATLAB integrates with Open Source
- MATLAB is Fast
“I love to label and preprocess my data”

~ Said no engineer, ever.
Caterpillar Case Study

- World’s leading manufacturer of construction and mining equipment.

- Similarity between these projects?
  - Autonomous haul trucks
  - Pedestrian detection
  - Equipment classification
  - Terrain mapping
Computer Must Learn from Lots of Data

- ALL data must first be labeled to create these autonomous systems.

“We were spending way too much time ground-truthing [the data]”
---Larry Mianzo, Caterpillar
How Did Caterpillar Do with Our Tools?

- Semi-automated labeling process
  - “We go from having to label 100 percent of our data to only having to label about 80 to 90 percent”

- Used MATLAB for entire development workflow.
  - “Because everything is in MATLAB, development time is short”
How Does MATLAB Come into Play?
ROI Label Definition

To label an ROI, you must first define one or more of the following label types:
- Rectangle label
- Pixel label

Scene Label Definition

To label a scene, you must first define a scene label.

Load images to start labeling.
MATLAB is Productive

- Image Labeler App semi-automates labeling workflow
- Bootstrapping
  - Improve automatic labeling by updating algorithm as you label more images correctly.
- Easy to load metadata even when labeling manually
Why MATLAB?

- MATLAB is Productive
- MATLAB Integrates with Open Source
- MATLAB is Fast
Used MATLAB and Open Source Together

- Used Caffe and MATLAB together
- Achieved significantly better results than an engineered rain model.
- Use our tools where it makes your workflow easier!

MATLAB Integrates with Open Source Frameworks

- Access to many pretrained models through add-ons
- Users wanted to import latest models
- Import models directly from TensorFlow or Caffe
  - Allows for improved collaboration

![Keras Importer](image)
Keras-TensorFlow Importer
MATLAB Integrates with Open Source Frameworks

- MATLAB supports entire deep learning workflow
  - Use when it is convenient for your workflow
- Access to latest models
- Improved collaboration with other users
Why MATLAB?

- MATLAB is Productive
- MATLAB Integrates with Open Source
- MATLAB is Fast
MATLAB is Fast

Performance

Training

Deployment
What is Training?

Feed labeled data into neural network to create working model
Speech Recognition Example

Audio signal → Spectrogram → Image Classification algorithm
Another Network for Signals - LSTM

- LSTM = Long Short Term Memory (Networks)
  - Signal, text, time-series data
  - Use previous data to predict new information
- I live in France. I speak ___________.

![Diagram of LSTM network]
1. Create Datastore

- Datastore creates reference for data
- Do not have to load in all objects into memory

```matlab
datafolder = fullfile(tempdir,'speech_commands_v0.01');
addpath(fullfile(matlabroot,'toolbox','audio','audiodemos'))
ads = audioexample.Datastore(datafolder, ...
    'IncludeSubfolders',true, ...
    'FileExtensions','.wav', ...
    'LabelSource','foldernames', ...
    'ReadMethod','File')```
2. Compute Speech Spectrograms

![Amplitude Graphs](image)

- **up**
- **go**
- **up**

![Frequency Graphs](image)

- **Frequency**
- **Time**
- **Frequency**
- **Time**
- **Frequency**
- **Time**
3. Split datastores

Training
- 70%
- Trains the model
- Computer “learns” from this data

Validation
- 15%
- Checks accuracy of model during training

Test
- 15%
- Tests model accuracy
- Not used until validation accuracy is good
4. Define Architecture and Parameters

Neural Network Architecture

Model Parameters

```matlab
layers = [ ... ]

model = trainNetwork(XTrain, YTrain, layers, miniBatchSize, ...);`
5. Train Network
Deep Learning on CPU, GPU, Multi-GPU and Clusters

**HOW TO TARGET?**

```
opts = trainingOptions('sgdm', ...
    'MaxEpochs', 100, ...  
    'MiniBatchSize', 250, ...  
    'InitialLearnRate', 0.00005, ...  
    'ExecutionEnvironment', 'auto' );
```

```
opts = trainingOptions('sgdm', ...
    'MaxEpochs', 100, ...  
    'MiniBatchSize', 250, ...  
    'InitialLearnRate', 0.00005, ...  
    'ExecutionEnvironment', 'multi-gpu' );
```

```
opts = trainingOptions('sgdm', ...  
    'MaxEpochs', 100, ...  
    'MiniBatchSize', 250, ...  
    'InitialLearnRate', 0.00005, ...  
    'ExecutionEnvironment', 'parallel' );
```
Training Performance

TensorFlow
MATLAB
MXNet
Batch size 32
Training is an Iterative Process

```matlab
miniBatchSize = 128;
validationFrequency = floor(numel(YTrain)/miniBatchSize);
options = trainingOptions('adam', ...
    'InitialLearnRate',5e-4, ...
    'MaxEpochs',25, ...
    'MiniBatchSize',miniBatchSize, ...
    'Shuffle','every-epoch', ...
    'Plots','training-progress', ...
    'Verbose',false, ...
    'ValidationData',{XValidation,YValidation}, ...
    'ValidationFrequency',validationFrequency, ...
    'ValidationPatience',Inf, ...
    'LearnRateSchedule','piecewise', ...
    'LearnRateDropFactor',0.1, ...
    'LearnRateDropPeriod',20);
```

Parameters adjusted according to performance
MATLAB is Fast for Deployment

- Target a GPU for optimal performance
- NVIDIA GPUs use CUDA code
- We only have MATLAB code. Can we translate this?
GPU Coder

- Automatically generates **CUDA** Code from MATLAB Code
  - can be used on NVIDIA GPUs

- CUDA extends C/C++ code with constructs for parallel computing
GPU Coder for Deployment

Deep Neural Networks
Deep Learning, machine learning

- 5x faster than TensorFlow
- 2x faster than MXNet

Image Processing and Computer Vision
Image filtering, feature detection/extraction

- 60x faster than CPUs for stereo disparity

Signal Processing and Communications
FFT, filtering, cross correlation,

- 20x faster than CPUs for FFTs

Accelerated implementation of parallel algorithms on GPUs & CPUs

Intel MKL-DNN Library
NVIDIA CUDA C/C++
ARM Compute Library

Intel XEON
ARM Compute Library
Intel MKL-DNN Library
ARM Compute Library
GPUs and CUDA

CUDA kernels

C/C++

GPU CUDA Cores

ARM Cortex

CUDA

C/C++

GPU Memory Space

CPU Memory Space

SECURITY ENGINES

4K60 VIDEO ENCODER

3G VIDEO DECODER

AUDIO ENGINE

2D ENGINE

DISPLAY ENGINES

128-bit LPDDR4

BOOT and PM PROC

GigE Ethernet MAC

IMAGE PROC (ISP)

Safety Engine

I/O
Challenges of Programming in CUDA for GPUs

- Learning to program in CUDA
  - Need to rewrite algorithms for parallel processing paradigm

- Creating CUDA kernels
  - Need to analyze algorithms to create CUDA kernels that maximize parallel processing

- Allocating memory
  - Need to deal with memory allocation on both CPU and GPU memory spaces

- Minimizing data transfers
  - Need to minimize while ensuring required data transfers are done at the appropriate parts of your algorithm
GPU Coder Helps You Deploy 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 kernel for GPU parallelization**

```c
static __global__ __launch_bounds__((512, 1)) void saxpy_kernel4(const real32_T *y,
const real32_T *x, real32_T a, real32_T *z)
{
    int32_T i;
    i = (int32_T)(((gridDim.x * blockDim.z + blockIdx.z + blockDim.x * blockDim.y) + blockDim.x) * blockDim.y + blockIdx.x * blockDim.y + threadIdx.y * blockDim.x + threadIdx.x);
    if (i < 1024)
    {
        z[i] = (real_T)(a * x[i] + y[i]);
    }
}
```

Loops and matrix operations are directly compiled into kernels.
Kernel data allocation is automatically optimized

Mandelbrot space

Generated CUDA Optimized for Memory Performance

CUDA kernel for GPU parallelization

```
static __global__ __launch_bounds__(512, 1) void kernel3(real_T *z0, real_T *z)
{
    real_T z_im;
    real_T yIm(0.000000);
    int32_T threadIdx;
    threadIdx = (int32_T)(blockIdx.x * blockDim.x + threadIdx.x);
    if ((threadIdx.x >= 0) && threadIdx.x < (1000000)) {
        z_im = z[threadIdx].re * z[threadIdx].im + z[threadIdx].im * z[threadIdx].re;
        z[threadIdx].re = (z[threadIdx].re - z[threadIdx].im) * z[threadIdx].im * z[(threadIdx).re] + z[threadIdx].im + z[0][threadIdx].im;
        yIm[threadIdx] = hypot(z[threadIdx].re, z[threadIdx].im);
        count[threadIdx] = (real_T)yIm[threadIdx] <= 2.0;
    }
}
```

CUDA

```matlab
... ...
```

```
cudamalloc(&gpu_xgrid, BUU00000U);
cudamalloc(&gpu_ygrid, BUU00000U);

/* mandelbrot computation */
cudamemcpy(gpu_ygrid, ygrid, BUU00000U, cudaMemcpyHostToDevice);
cudamemcpy(gpu_xgrid, xgrid, BUU00000U, cudaMemcpyHostToDevice);
kernel<<<dim3(1954, 1U, 1U), dim3(512U, 1U, 1U)>>>(gpu_xgrid, gpu_xgrid, gpu_ygrid, gpu_xgrid, gpu_ygrid, gpu_ygrid, &n, &z0);  
for (n = 0; n < (int32_T)(maxIterations + 1.0); n++) {
    kernel<<<dim3(1954, 1U, 1U), dim3(512U, 1U, 1U)>>>(gpu_z0, gpu_count, gpu_ygrid);
}
```

```
cudamemcpy(count, gpu_count, BUU00000U, cudaMemcpyDeviceToHost);
cudafree(gpu_ygrid);
```

... ...

Mandelbrot space

GPU Coder
Example: Fog Rectification
Algorithm Design to Embedded Deployment Workflow

MATLAB algorithm (functional reference)

1. Functional test
2. Deployment unit-test
3. Deployment integration-test
4. Real-time test

GPU Coder

Build type

Call CUDA from MATLAB directly

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

Cross-compiled .lib

Desktop GPU

C++

Embedded GPU

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

Desktop GPU

C++

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

Desktop GPU

C++

Build type

Call CUDA from MATLAB directly

Call CUDA from (C++) hand-coded main()
Demo: Alexnet Deployment with ‘mex’ Code Generation
Algorithm Design to Embedded Deployment on Tegra GPU

MATLAB algorithm (functional reference)

1. Functional test
   (Test in MATLAB on host)

2. Deployment unit-test
   (Test generated code in MATLAB on host + GPU)

3. Deployment integration-test
   (Test generated code within C/C++ app on host + GPU)

4. Real-time test
   (Test generated code within C/C++ app on Tegra target)

GPU Coder

Build type

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

Cross-compiled .lib

- Call CUDA from (C++) hand-coded main()
  Cross-compiled on host with Linaro toolchain
Alexnet Deployment to Tegra: Cross-Compiled with ‘lib’

**Two small changes**

1. Change build-type to ‘lib’
2. Select cross-compile toolchain
End-to-End Application: Lane Detection

Alexnet

Transfer Learning

Output of CNN is lane parabola coefficients according to: \( y = ax^2 + bx + c \)

Image CNN

Lane detection

Left lane coefficients

Post-processing (find left/right lane points)

Right lane coefficients

Image with marked lanes

GPU coder generates code for whole application
Deep Learning Network Support (with Neural Network Toolbox)

**SeriesNetwork**
- Single-in/single-out
- Networks: MNist, Alexnet, YOLO, VGG, Lane detection, Pedestrian detection
- GPU Coder: R2017b

**DAGNetwork**
- Networks: GoogLeNet, ResNet, SegNet, DeconvNet
- Object detection, Semantic segmentation
- GPU Coder: R2018a
Semantic Segmentation

Running in MATLAB

Generated Code from GPU Coder
Deploying to CPUs

Deep Learning Networks → GPU Coder → Intel MKL-DNN Library

→ NVIDIA TensorRT & cuDNN Libraries

→ ARM Compute Library
Deploying to CPUs

Deep Learning Networks

GPU Coder

Desktop CPU

NVIDIA TensorRT & cuDNN Libraries

Raspberry Pi board

23.88 FPS
89.7% computer keyboard
8.6% space bar
1.7% typewriter key
0.0% mouse
0.0% notebook
How Good is Generated Code Performance

- Performance of image processing and computer vision
- Performance of CNN inference (Alexnet) on Titan XP GPU
- Performance of CNN inference (Alexnet) on Jetson (Tegra) TX2
GPU Coder for Image Processing and Computer Vision

- Fog removal: 5x speedup
- Distance transform: 8x speedup
- Frangi filter: 3x speedup
- Stereo disparity: 50x speedup
- Ray tracing: 18x speedup
- SURF feature extraction: 700x speedup
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

**Frames per second vs Batch Size**
- **TensorFlow (1.6.0)**
- **MXNet (1.1.0)**
- **GPU Coder + cuDNN**
- **GPU Coder + TensorRT (3.0.1)**
- **GPU Coder + TensorRT (3.0.1, int8)**
VGG-16 Inference on NVIDIA Titan Xp

<table>
<thead>
<tr>
<th>Testing platform</th>
<th>CPU</th>
<th>GPU</th>
<th>cuDNN</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Intel(R) Xeon(R) CPU E5-1650 v4 @ 3.60GHz</td>
<td>Pascal Titan Xp</td>
<td>v7</td>
</tr>
</tbody>
</table>

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)**
Alexnet Inference on Jetson TX2: Frame-Rate Performance

Frames per second vs. Batch Size

- TensorRT (2.1) with a 1.15x increase in performance
- MATLAB GPU Coder (R2017b) with a 2x increase in performance
- C++ Caffe (1.0.0-rc5)
Alexnet Inference on Jetson TX2: Memory Performance

### MATLAB GPU Coder
- **Version:** R2017b
- **Description:**

### C++ Caffe
- **Version:** 1.0.0-rc5
- **Description:**

### TensorRT 2.1
- **Description:** (using giexec wrapper)
MATLAB Deep Learning Framework

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

- Acceleration with GPU’s
- Scale to clusters

- Automate compilation to GPUs and CPUs using GPU Coder:
  - 5x faster than TensorFlow
  - 2x faster than MXNet
Why MATLAB for Deep Learning?

- MATLAB is Productive
- MATLAB Integrates with Open Source (Frameworks)
- MATLAB is Fast (Performance)