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

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

- **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.
Data Types for Deep Learning

Signal

Text

Image
Deep Learning is Versatile

Detection of cars and road in autonomous driving systems

Rain Detection and Removal

Deep Joint Rain Detection and Removal from a Single Image’ Wenhan Yang, Robby T. Tan, Jiashi Feng, Jiaying Liu, Zongming Guo, and Shuicheng Yan

1. 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

Iris Recognition – 99.4% accuracy

MATLAB EXPO 2018
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

Define new ROI label

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

Scene Label Definition

Define new scene label

Apply to Image
Remove from Image

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-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 ____________.
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

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
5. Train Network
Deep Learning on CPU, GPU, Multi-GPU and Clusters

**How to Target?**

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

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

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

Seconds / Epoch

Training (V100 GPU)

AlexNet | ResNet50

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

Accelerated implementation of parallel algorithms on GPUs & CPUs

Deep Neural Networks
Deep Learning, machine learning

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

Signal Processing and Communications
FFT, filtering, cross correlation,

5x faster than TensorFlow
2x faster than MXNet

60x faster than CPUs for stereo disparity

20x faster than CPUs for FFTs
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
GPU Coder Generates CUDA from MATLAB: saxpy

Scalarized MATLAB

for \( i = 1:length(x) \)
\[
z(i) = a \cdot x(i) + y(i);
\]
end

Vectorized MATLAB

\[ z = a \cdot x + y; \]

Loops and matrix operations are directly compiled into kernels

CUDA

```matlab
cudaMalloc(&gpu_z, 8388608UL);
cudaMalloc(&gpu_x, 4194304UL);
cudaMalloc(&gpu_y, 4194304UL);
cudaMemcpy((void *)gpu_y, (void *)y, 4194304UL, cudaMemcpyHostToDevice);
cudaMemcpy((void *)gpu_x, (void *)x, 4194304UL, cudaMemcpyHostToDevice);
saxpy_kernel1<<<dim3(2048U, 1U, 1U), dim3(512U, 1U, 1U)>>>(gpu_y, gpu_x, a, gpu_z);
cudaMemcpy((void *)z, (void *)gpu_z, 8388608UL, cudaMemcpyDeviceToHost);
cudaFree(gpu_y);
cudaFree(gpu_x);
cudaFree(gpu_z);
```

CUDA kernel for GPU parallelization

```c
static __global__ __launch_bounds__(512, 1) void saxpy_kernel1(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 + threadIdx.z) -
  (blockDim.x * blockDim.z) + threadIdx.x);

  if (i < 1048576) {
    z[i] = (real32_T)(a * x[i] + y[i]);
  }
}
```
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 *count, real_T *im)
{
    real_T z_im; 
    real_T y(10000000); 
    int32_T threadIdx;
    threadIdx = (int32_T)(blockDim.x * blockIdx.x + threadIdx.x);
    if (threadIdx.x >= 10000000)
    {
        z_im = z[threadIdx.x].re * z[threadIdx.x].im + z[threadIdx.x].im * z[threadIdx.x].re;
        z[threadIdx.x].re = (z[threadIdx.x].re) * z[threadIdx.x].re + z[threadIdx.x].im * z[threadIdx.x].im;
        z[threadIdx.x].im = z_im + z0[threadIdx.x].im;
    }

    count[threadIdx.x] += (real_T)(y[threadIdx.x] < 2.0);
}
```

CUDA

```
cudaMalloc(&gpu_xgrid, 8000000U);
cudaMalloc(&gpu_ygrid, 8000000U);

/* mandelbrot computation */
cudaMemcpy(gpu_ygrid, ygrid, 8000000U, cudaMemcpyHostToDevice);
cudaMemcpy(gpu_xgrid, xgrid, 8000000U, cudaMemcpyHostToDevice);
kernel<<<dim3(1564, 1U, 1U), dim3(512U, 1U, 1U)>>>(gpu_xgrid, gpu_ygrid, 
gpu_z, gpu_count, gpu_z0);
for (n = 0; n < (int32_T)(maxIterations + 1.0); n++) {
    kernel<<<dim3(1564, 1U, 1U), dim3(512U, 1U, 1U)>>>(gpu_z0, gpu_count, 
gpu_z); 
}
```

Mandelbrot space

Kernel data allocation is automatically optimized
Example: Fog Rectification
Algorithm Design to Embedded Deployment Workflow

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

MATLAB algorithm (functional reference)

GPU Coder

Build type

Call CUDA from MATLAB directly

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

Cross-compiled .lib

Desktop GPU

.mex

Desktop GPU

Desktop GPU

Embedded GPU

.C++

Embedded GPU

.C++
Demo: Alexnet Deployment with ‘mex’ Code Generation
Algorithm Design to Embedded Deployment on Tegra GPU

MATLAB algorithm (functional reference)

Functional test
(Test in MATLAB on host)

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

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

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

1
2
3
4

Build type

.mex
Tesla GPU

.library
Tesla GPU

Cross-compiled .lib
Tegra GPU

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

Call CUDA from (C++) hand-coded main()
Cross-compiled on host with Linaro toolchain

Call CUDA directly
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: \( y = ax^2 + bx + c \)

**Lane detection CNN** → **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

- GPU Coder: R2017b
- Networks: MNist, Alexnet, YOLO, VGG, Lane detection, Pedestrian detection

DAGNetwork

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

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)

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

Batch Size

TensorRT (2.1)
MATLAB GPU Coder (R2017b)
C++ Caffe (1.0.0-rc5)

1.15x
2x
Alexnet Inference on Jetson TX2: Memory Performance

- **C++ Caffe** (1.0.0-rc5)
- **MATLAB GPU Coder** (R2017b)
- **TensorRT 2.1** (using giexec wrapper)

The graph shows the peak memory usage (MB) against the batch size. The memory usage increases as the batch size increases for all the platforms.
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
Why MATLAB for Deep Learning?

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