Deep Learning
Alison B Lowndes | Solutions Architect EMEA | September 3, 2015
Agenda

➢ Deep Learning Intro
➢ Practical uses
➢ CNN => RNN
➢ Why GPU’s
➢ NVIDIA’s Deep Learning Platform
➢ DIGITS
➢ QWIKLABS - Hands ON
Deep Learning

Assumption: start from the beginning
DEEP LEARNING
PATTERN RECOGNITION
3 DRIVERS FOR DEEP LEARNING

More Data

Better Models

Powerful GPU Accelerators
Compound expressions: \[ f(x, y, z) = (x + y)z \]

\[ q = x + y \quad \frac{\partial q}{\partial x} = 1, \quad \frac{\partial q}{\partial y} = 1 \]

\[ f = qz \quad \frac{\partial f}{\partial q} = z, \quad \frac{\partial f}{\partial z} = q \]

Chain rule:

\[ \frac{\partial f}{\partial x} = \frac{\partial f}{\partial q} \frac{\partial q}{\partial x} \]

# set some inputs
x = -2; y = 5; z = -4

# perform the forward pass
q = x + y # q becomes 3
f = q * z # f becomes -12

# perform the backward pass (backpropagation) in reverse order:
# first backprop through f = q * z
dfdz = q # df/fz = q, so gradient on z becomes 3
dfdq = z # df/dq = z, so gradient on q becomes -4
# now backprop through q = x + y
dfdx = 1.0 * dfdq # dq/dx = 1. And the multiplication here is the chain rule!
dfdy = 1.0 * dfdq # dq/dy = 1

Fei-Fei Li & Andrej Karpathy  Lecture 5 - 12  21 Jan 2015
What is Deep Learning?

A family of methods that uses deep architectures to learn high-level feature representations.
What makes deep learning deep?

Application components:

- **Task objective**
  - e.g. Identify face

- **Training data**
  - 10-100M images

- **Network architecture**
  - ~10 layers
  - 1B parameters

- **Learning algorithm**
  - ~30 Exaflops
  - ~30 GPU days
CONVOLUTION

Center element of the kernel is placed over the source pixel. The source pixel is then replaced with a weighted sum of itself and nearby pixels.
Gradient Descent

**Oversimplified Gradient Descent:**

- Calculate slope at current position
- If slope is negative, move right
- If slope is positive, move left
- (Repeat until slope == 0)

@iamtrask Andrew Trask Digital Reason
Neural Networks

Modeling Neurons

Left: Sigmoid non-linearity squashes real numbers to range between $[0,1]$. Right: The tanh non-linearity squashes real numbers to range between $[-1,1]$.

Left: Rectified Linear Unit (ReLU) activation function, which is zero when $x < 0$ and then linear with slope 1 when $x > 0$. Right: A plot from Krizhevsky et al. (pdf) paper indicating the 6x improvement in convergence with the ReLU unit compared to the tanh unit.
CNNs- Stacked Repeating Triplets

Convolutions → Max Pooling (block-Wise max) → Activation (point-wise ReLu)

OverFeat Network 2014

<table>
<thead>
<tr>
<th>Layer</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stage</td>
<td>conv + max</td>
<td>conv + max</td>
<td>conv</td>
<td>conv</td>
<td>conv</td>
<td>conv</td>
<td>conv + max</td>
<td>full</td>
<td>full</td>
</tr>
<tr>
<td># channels</td>
<td>96</td>
<td>256</td>
<td>512</td>
<td>512</td>
<td>1024</td>
<td>1024</td>
<td>4096</td>
<td>4096</td>
<td>1000</td>
</tr>
<tr>
<td>Filter size</td>
<td>7x7</td>
<td>7x7</td>
<td>3x3</td>
<td>3x3</td>
<td>3x3</td>
<td>3x3</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Conv. stride</td>
<td>2x2</td>
<td>1x1</td>
<td>1x1</td>
<td>1x1</td>
<td>1x1</td>
<td>1x1</td>
<td>1x1</td>
<td>1x1</td>
<td>-</td>
</tr>
<tr>
<td>Pooling size</td>
<td>3x3</td>
<td>2x2</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>3x3</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Pooling stride</td>
<td>3x3</td>
<td>2x2</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>3x3</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Zero-Padding size</td>
<td>-</td>
<td>-</td>
<td>1x1x1</td>
<td>1x1x1</td>
<td>1x1x1</td>
<td>1x1x1</td>
<td>1x1x1</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Spatial input size</td>
<td>221x221</td>
<td>36x36</td>
<td>15x15</td>
<td>15x15</td>
<td>15x15</td>
<td>15x15</td>
<td>5x5</td>
<td>1x1</td>
<td>1x1</td>
</tr>
</tbody>
</table>
Convolutional Neural Networks

Architecture

C layers are convolutions,

S layers pool/sample

FIGURE 2 Conceptual example of convolutional neural network. The input image is convolved with three trainable filters and biases as in Figure 1 to produce three feature maps at the C1 level. Each group of four pixels in the feature maps are added, weighted, combined with a bias, and passed through a sigmoid function to produce the three feature maps at S2. These are again filtered to produce the C3 level. The hierarchy then produces S4 in a manner analogous to S2. Finally these pixel values are rasterized and presented as a single vector input to the “conventional” neural network at the output.
Practical DEEP LEARNING Examples

Image Classification, Object Detection, Localization, Action Recognition, Scene Understanding

Speech Recognition, Speech Translation, Natural Language Processing

Pedestrian Detection, Traffic Sign Recognition

Breast Cancer Cell Mitosis Detection, Volumetric Brain Image Segmentation
Practical uses

- ConvNet learns to convert raw pixels directly into suitable steering commands.
- The task was purely learned; no hand tuning of algorithms or parameters.
- 2 convolutional layers followed by 2x2 averaging each, fully connected output layer
- 3 million connections, 72,000 free parameters.

Training Data

- 225,000 example images
- Extracted from 3,200 data collection runs

Human: turn right

Human: go straight

Human: turn right

Human: turn left
DeepDriving: Learning Affordance for Direct Perception in Autonomous Driving

Points within the zero-height plane

Points above the zero-height plane

CNN estimation (ours)

DPM estimation

Ground truth

Princeton DeepDrive
Running on Drive PX and developed in just 3 weeks!
Recurrent Neural Networks

Christopher Olah - DL child genius - aged 22

An unrolled recurrent neural network.
Recurrent neural networks

\[ x - \text{input}; \ h - \text{hidden state vector}; \ y - \text{output} \]

\( f \) – maps input and previous hidden state into new hidden state; \( g \) – maps hidden state into \( y \)

\( f \) – can be a huge feed-forward network

“classical” feed-forward network with shared weights \( f \)

Most commonly trained by back-propagation

To put \( f \)’s weight update together:
sum with much smaller learning rate
averaging
The problem with RNN

Exploding gradient – large increase of the gradient’s norm due to long term dependencies. Results in large increase of the cost function during training.

Address with Rectified-linear (ReLu) activation function

Vanishing gradient – opposite behaviour, long term components go exponentially fast to 0. Results in bad prediction of long term dependencies.
Long short-term memory (LSTM)

Hochreiter (1991) analysed vanishing gradient “LSTM falls out of this almost naturally”

Gates control importance of the corresponding activations

Training via backprop unfolded in time

Long time dependencies are preserved until input gate is closed (-) and forget gate is open (O)

Fig from Vinyals et al, Google April 2015 NIC Generator

Fig from Graves, Schmidhuber et al, Supervised Sequence Labelling with RNNs
LSTM Use Cases

“The RNN’s memory is necessary to deal with ambiguous sensory inputs from repetitively visited states”

- **Metalearning** - general purpose learning algorithms that can learn better learning algorithms! Applying Genetic Programming to itself.

- **Blues Improv** with LSTM Recurrent Networks; Doug Eck

- **Text-to-speech synthesis** (Fan et al., Microsoft, Interspeech 2014)

- **English to French translation** (Sutskever et al., Google, NIPS 2014)

- **Arabic handwriting recognition** (Bluche et al., DAS 2014)

---

**iCub (Intelligent Cub)** http://robotics.idsia.ch/robots

Courtesy of J Schmidhuber, IDSIA
Video

wheelchair basketball: 0.829
basketball: 0.114
streetball: 0.020
Prototype
Why GPUs?
GPUs make deep learning accessible

**GOOGLE DATACENTER**

- 1,000 CPU Servers
- 2,000 CPUs • 16,000 cores
- $5,000,000

**STANFORD AI LAB**

- 3 GPU-Accelerated Servers
- 12 GPUs • 18,432 cores
- $33,000
Why are GPUs good for Deep Learning

- same *or better* prediction accuracy
- faster results
- smaller footprint
- lower power

<table>
<thead>
<tr>
<th>Inherently Parallel</th>
<th>Neural Networks</th>
<th>GPUs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Matrix Operations</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>FLOPS</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

[Image: Comparison of Neural Networks and GPUs for Deep Learning tasks, showing better performance with GPUs.]
WHO IS USING DEEP LEARNING?

**Industry**
- Adobe
- Baidu
- flickr
- Netflix
- Yandex

**Use Cases**
- Image Recognition
- Face Recognition
- Gesture Recognition
- Video Search & Analytics
- Speech Recognition & Translation
- Recommendation Engines
- Indexing & Search

**Research**
- facebook
- stanford
- DARPA
- NYU
- Carnegie Mellon University
- DENSO
- Massachusetts Institute of Technology
- Berkeley

**Who is using Deep Learning?**
- Image Analytics for Creative Cloud
- Speech/Image Recognition
- Image Classification
- Recommendation
- Search Rankings
- Search Rankings
- Search Rankings
NVIDIA’s Deep Learning Platform
NVIDIA DEEP LEARNING PLATFORM

DEVELOPMENT

Applications

DIGITS Tools

Deep Learning Frameworks

Hardware

Systems

Software

cuDNN

DIGITS DevBox

Titan X

DEPLOYMENT

Software

System Management

Systems

Hardware

Tesla
Deep Learning Platform

- World changing applications
  Many domains

- Quick start to leverage DL ecosystem

- GPU accelerated Frameworks
  Quick start for GPU acceleration

- Best performance
  Robust, easy to use, reliable

- Parallel computing platform
  Great Features

- World’s Best DL HW – Kepler, Maxell, Pascal & NVLINK, DevBox

- NVIDIA Application
  Customer Application

- DIGITS

- DL Frameworks (Caffe, Torch, Theano)

- Performance libraries (cuDNN, cuBLAS)

- CUDA

- GPU
Frameworks
What is Caffe?

An open framework for deep learning developed by the Berkeley Vision and Learning Center (BVLC)

- Pure C++/CUDA architecture
- Command line, Python, MATLAB interfaces
- Fast, well-tested code
- Pre-processing and deployment tools, reference models and examples
- Image data management
- Seamless GPU acceleration
- Large community of contributors to the open-source project

caffe.berkeleyvision.org
http://github.com/BVLC/caffe
What is Caffe?
End-to-end Deep Learning for the practitioner and developer
Caffe features

Deep Neural Network training

Network training also requires no coding - just define a “solver” file

```plaintext
net: "lenet_train.prototxt"
base_lr: 0.01
momentum: 0.9
max_iter: 10000
snapshot_prefix: "lenet_snapshot"
Solver_mode: GPU
```

Multiple optimization algorithms available: SGD (+momentum), ADAGRAD, NAG
Caffe features

Deep Neural Network sharing

Caffe Model Zoo hosts community shared models

Benefit from networks that you could not practically train yourself

https://github.com/BVLC/caffe/wiki/Model-Zoo

Caffe comes with unrestricted use of BVLC models:

AlexNet
R-CNN
GoogLeNet
Caffe model mobile deployment

- Jetson TK1
  - Inference at 35ms per image*
    - Alexnet, 10 crops per input image
  - No need to change code
- Simply compile Caffe and copy a trained .caffemodel to TK1

*Source: http://petewarden.com/2014/10/25/how-to-run-the-caffe-deep-learning-vision-library-on-nvidias-jetson-mobile-gpu-board/
## GPU-ACCELERATED DEEP LEARNING FRAMEWORKS

<table>
<thead>
<tr>
<th>Description</th>
<th>CAFFE</th>
<th>TORCH</th>
<th>THEANO</th>
<th>CUDA-CONVNET2</th>
<th>KALDI</th>
</tr>
</thead>
<tbody>
<tr>
<td>cuDNN</td>
<td>Version 3.0</td>
<td>Version 2.0</td>
<td>Version 2.0</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>Multi-GPU</td>
<td>In Progress</td>
<td>In Progress</td>
<td>In Progress</td>
<td>✓</td>
<td>✓ (nnet2)</td>
</tr>
<tr>
<td>Multi-CPU</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✓ (nnet2)</td>
</tr>
<tr>
<td>License</td>
<td>BSD-2</td>
<td>BSD</td>
<td>BSD</td>
<td>Apache 2.0</td>
<td>Apache 2.0</td>
</tr>
<tr>
<td>Interface(s)</td>
<td>Text-based definition files, C++, Python, MATLAB</td>
<td>Python, Lua, MATLAB</td>
<td>Python</td>
<td>C++</td>
<td>C++, Shell scripts</td>
</tr>
<tr>
<td>Embedded (TK1)</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
</tbody>
</table>

Explore the CUDA (libraries) Ecosystem

developer.nvidia.com/cuda-tools-ecosystem
CUDA Parallel Computing Platform

**Programming Approaches**
- Libraries: “Drop-in” Acceleration
- OpenACC Directives: Easily Accelerate Apps
- Programming Languages: Maximum Flexibility

**Development Environment**
- Parallel Nsight IDE: Linux, Mac and Windows GPU Debugging and Profiling
- CUDA-GDB debugger NVIDIA Visual Profiler

**Open Compiler Tool Chain**
- Enables compiling new languages to CUDA platform, and CUDA languages to other architectures

**Hardware Capabilities**
- SMX
- Dynamic Parallelism
- HyperQ
- GPUDirect
NVIDIA® cuDNN Roadmap

**Release v1**
September 2014
- Layers:
  - Convolutional
  - MaxPooling
  - Softmax
  - Rectified/Sigmoid
- High performance convolution

**Release v2**
March 2015
- Layers:
  - Arbitrary dimension API
  - 3D convolution
- Up to 2x faster on convolutional layers

**Release v3**
Summer 2015
- Mixed precision support
- 2D FFT support
- Normalization layers (LRN, LCN)
- Full 3D images support
- Improved tuning for Maxwell GPUs

**Release v4**
Fall 2015 (Tentative)
- Under NDA

**Timeline**
- Q3'14
- Q1'15
- Q2'15
- Q3'15
- Q4'15

**Features**
- High performance convolution
- Up to 2x faster on convolutional layers
- Mixed precision support
- 2D FFT support
- Normalization layers (LRN, LCN)
- Full 3D images support
- Improved tuning for Maxwell GPUs

**Performance**
- High performance convolution
- Up to 2x faster on convolutional layers
- Mixed precision support
- 2D FFT support
- Normalization layers (LRN, LCN)
- Full 3D images support
- Improved tuning for Maxwell GPUs
GPU acceleration

-gpu N flag tells caffe which gpu to use

Alternatively, specify solver_mode: GPU in solver.prototxt

Benchmark: Train Caffenet model, 20 iterations, 5120 256x256 images, mini-batch size 256

- 17.1s
  - Tesla K80, cuDNN v2
  - ECC off, autoboost on

- 381s
  - 4 x Intel® Xeon® E5-2690 v3
  - @ 2.60GHz (48 cores total), 128GB RAM, multi-threading
DIGITS
4 TITAN X GPUs with 7 TFLOPS of single precision, 336.5 GB/s of memory bandwidth, and 12 GB of memory per board

NVIDIA DIGITS software providing powerful design, training, and visualization of deep neural networks for image classification

Pre-installed standard Ubuntu 14.04 with Caffe, Torch, Theano, BIDMach, cuDNN v2, and CUDA 7.0

A single deskside machine that plugs into standard wall plug, with superior PCIe topology
DIGITS™ Devbox
How to build your own DevBox

Get the specific Hardware - See detailed specs on https://developer.nvidia.com/devbox


Build SW and start using it- Instructions are given at https://github.com/NVIDIA/DIGITS/blob/master/README.md
DIGITS
Deep Learning GPU Training System

Select from a variety of industry standard DNNs or design your own

Import your data from disk or web

Monitor multiple trainings in parallel

Visualize and debug DNN accuracy
DIGITS
Deep Learning GPU Training System

Available at http://developer.nvidia.com/digits

Free to use, Source Code available at Github, latest branch v2.0 https://github.com/NVIDIA/DIGITS

Current release supports classification on images

Future versions: More problem types and data formats (video, speech)
DIGITS
Installation Instructions

Get the latest Source Code from GitHub, https://github.com/NVIDIA/caffe

You may also download the more stable version from NVIDIA webpage https://developer.nvidia.com/digits

DIGITS is only officially supported on Ubuntu 14.04, You can try on other Linux Variants 😊

Prerequisites

CUDA toolkit (>= CUDA 6.5) with Latest cuDNN

Deep Learning framework - At least one is needed (Caffe/Torch ...)
DIGITS
Using web application

Open link for DIGITS access by browser
http://localhost:5000

Home Screen looks like
Training a Model

Advanced Options

Solver Options
- Training epochs: 30
- Snapshot interval (in epochs): 1
- Validation interval (in epochs): 1
- Batch size: [network defaults]
- Base Learning Rate: 0.01
- Policy: Step Down

New Image Classification Model
- Select Dataset: DIGITS
- Data Transformations: Crop Size: [network defaults]
DIGITS
Training a Model-Advanced Options
DIGITS
NEW Features in DIGITS 2.0

Ability to do Multi-GPU training

Assign work to the fastest GPUs

New Solver- GoogLeNet

2014 Image Net LSVRC competition

The Custom Network edit box has settings for the layers, activation function (ReLU, TANH, or sigmoid), and bias value.
NVIDIA DIGITS

Digits 2.0 Improved Layer Visualization

Statistical Information
NVIDIA DIGITS
DIGITS 2.0 - Top N predictions per Category
Create Account at https://nvidia.qwiklab.com/ to do Hands-On
More about DEEP Learning?

Check out our Free Deep Learning Courses https://developer.nvidia.com/deep-learning-courses

<table>
<thead>
<tr>
<th>Date</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>7/22</td>
<td>Class #1 - Introduction to Deep Learning</td>
</tr>
<tr>
<td>8/5</td>
<td>Class #2 - Getting Started with DIGITS interactive training system for image classification</td>
</tr>
<tr>
<td>8/19</td>
<td>Class #3 - Getting Started with the Caffe Framework</td>
</tr>
<tr>
<td>9/2</td>
<td>Class #4 - Getting Started with the Theano Framework</td>
</tr>
<tr>
<td>9/16</td>
<td>Class #5 - Getting Started with the Torch Framework</td>
</tr>
</tbody>
</table>

Office hours for Q&A too!
Hands-on lab

- Use data pre-processing tools
- Edit a network definition
- Train a model
- Improve classification accuracy by modifying network parameters
- Visualize trained network weights
- Deploy a model using Python
DIGITS-2.0
Performance Results

Train models up to 2x faster with automatic multi-GPU scaling

Scaling is nearly linear

DL requires very Dense GPU systems
NVIDIA® DIGITS Roadmap

Version 1
March 2015

Support for image classification networks
Visualize layer-wise responses
Run locally, manage single-GPU jobs

Caffe

Version 2

Additional image analysis network types
Richer visual analysis tools
Run locally, more job management options

Additional frameworks

Version 3

Continued improvement to visualization tools
Front end to cluster task scheduler

API for easy framework integration

2015

Features
Framework Support

2016
Where to get DIGITS 2

Easy to use web installer https://developer.nvidia.com/digits

github - https://github.com/NVIDIA/DIGITS

Remember to install NVIDIA’s Caffe branch - https://github.com/NVIDIA/caffe

User support

DIGITS Users Google group - https://groups.google.com/forum/#!forum/digits-users

For more information on getting started with DIGITS


Getting started guide - https://github.com/NVIDIA/DIGITS/blob/master/docs/GettingStarted.md
The future
Questions?