Chekuri S. Choudary
Cognitive Systems Infrastructure Specialist
IBM
Cognitive Landscape: Terms and Relationship

• Deep Neural Networks: Lot more hidden layers (order of tens/hundreds)
• Cognitive Computing: Intersects AI, ML, and DL
Training

- Data intensive: historical data sets
- Compute intensive: 100% accelerated
- Develop a model for use on the edge as inference

Inference

- Enables the computer to act in real time
- Low Power
- Out at the edge

Core ML functions

Training phase

- New data
- Dataset development
- Design NN candidates
- Data generator

Inference phase

- Testing / Deploy
- Out at the edge
Artificial Neural Networks

Cost Function:

\[ J(\Theta) = -\frac{1}{m} \sum_{i=1}^{m} \sum_{k=1}^{K} \left[ y_k^{(i)} \log(h_\Theta(x^{(i)}))_k + (1 - y_k^{(i)}) \log(1 - (h_\Theta(x^{(i)}))_k) \right]. \]
Then, What is Back Propagation?

Gradient Descent Iteration:

1. Calculate derivative of cost function w.r.t each parameter
   1. Option 1: Numerical Derivatives
      • Change the weight a little, calculate the change in cost function
      • Computationally intractable
   2. Option 2: Analytical Derivatives
      • Use Chain rule, i.e, Back propagation
      • Reasonable turnaround times, 1000s of times cheaper
2. Update the billions of parameters

Chain Rule:

\[
\frac{\partial z}{\partial x} = \frac{\partial z}{\partial y} \frac{\partial y}{\partial x}
\]

Back Propagation:
Neural network gradients

\[ z^{[1]} = W^{[1]}x + b^{[1]} \]

\[ a^{[1]} = \sigma(z^{[1]}) \]

\[ z^{[2]} = W^{[2]}x + b^{[2]} \]

\[ a^{[2]} = \sigma(z^{[2]}) \]

\[ L(a^{[2]}, y) \]

\[
\begin{align*}
dZ^{[2]} &= A^{[2]} - Y \\
dW^{[2]} &= \frac{1}{m} dZ^{[2]} A^{[1]^T} \\
&= \frac{1}{m} np.\, sum(dZ^{[2]}, axis = 1, keepdims = True) \\
dZ^{[1]} &= W^{[2]^T} dZ^{[2]} \ast g^{[1]'}(Z^{[1]}) \\
dW^{[1]} &= \frac{1}{m} dZ^{[1]} X^T \\
&= \frac{1}{m} np.\, sum(dZ^{[1]}, axis = 1, keepdims = True)
\end{align*}
\]
Deep Learning

- Neural networks with lot more hidden layers (tens or hundreds)
- Types of Deep Neural Networks
  - Convolutional Neural Networks
  - Recurrent Neural Networks
  - Autoencoders
  - Restricted Boltzmann Machines
  - Generative Adversarial Networks
- Why now?
  - Data explosion
  - GPUs and other SIMD architectures
  - Some advancements in neural networks
- Nobody knows why it works but it works
- End-to-end Learning
  - No need for feature engineering
Technologies for Democratization of Deep Learning

- HPC, Distributed Computing Clusters, Public and Private Clouds
  - Multicore-processors (Power9)
  - SIMD Parallel units (V100 GPUs)
- Software Frameworks and Libraries
  - MPI
  - Spark
  - CUDA
  - cuBLAS
  - cuDNN
- Open Source Deep Learning Frameworks
  - Caffe
  - TensorFlow
  - Torch
  - Chainer
  - Deeplearning4J
  - Keras
  - Theano
- Open Source Databases (OSDBs)
  - MongoDB, Cassandra, EnterpriseDB, MariaDB, Redis, Neo4J
unique innovation through OpenPower collaboration

POWER8 with NVLink delivers 2.8X the bandwidth

The system bottleneck shifts to PCIe-Express

Performance... Faster Training and Inferencing
• Shorter training times
• Facilitates distributed deep learning and large model support
Layers in AI Infrastructure Stack

- **Applications**
  - Segment Specific: Finance, Retail, Healthcare, etc.
  - Speech, Vision, NLP, Sentiment

- **Cognitive APIs (Eg: Watson)**
  - TensorFlow, Caffe, SparkML

- **In-House Cognitive APIs**
  - Spark, MPI

- **ML & DL Libraries & Frameworks**
  - Hadoop HDFS, NoSQL DBs

- **Distributed Computing**
  - Accelerated Infrastructure

- **Data Lake & Data Stores**

- **Transform & Prep Data (ETL)**

- **Accelerated Servers**

- **Storage**
PowerAI Platform

Deep Learning Frameworks
- Caffe
- NV-Caffe
- IBM Caffe
- Torch
- TensorFlow
- Chainer
- Theano

Supporting Libraries
- OpenBLAS
- Bazel
- Distributed Frameworks
- NCCL
- DIGITS

Accelerated Servers and Infrastructure for Scaling

Cluster of NVLink Servers

Spectrum Scale: High-Speed Parallel File System

Scale to Cloud
Distributed Deep Learning

Model Parallelism

- Machine 4
- Machine 2
- Machine 1
- Machine 3

Data Parallelism

- Machine 1
- Machine 2
- Machine 3
- Machine 4

- Sample Distribution of Training Set:
- 16 nodes with 4 GPUs each
- # of training samples is 64000
- Batch size is 6400
- 1 epoch requires 10 iterations
- Each GPU processes 100 training samples

PowerAI Deep Learning Software Stack

**Input Data**
- Data Prep & ETL via Spectrum Conductor with Spark

**AI Vision**
- Computer Vision App Development Toolkit

**DL Insight**
- Tuning Engine

**DL Frameworks**
- (TF, Caffe, etc)
- Distributed Training

**Deep Learning GUI**
- Data & Model Management, ETL Tools, Monitor, Visualize, Advise

**IBM Spectrum Conductor with Spark**
- System mgmt, Distributed ETL, Distributed Training, Hyper-Parameter Optimization
Distributed Deep Learning (DDL)
Using the Power of 100s of Servers

16 Days Down to 7 Hours:
58x Faster

Near Ideal Scaling to 256 GPUs and Beyond

1 System
64 Systems

ResNet-101, ImageNet-22K, Caffe with PowerAI DDL, Running on Minsky (S822Lc) Power System
PowerAI Rel. 4 with Large Model Support tech. preview

Performance…
Faster Training and Inferencing

Traditional Model Support
(Competitors)
Limited memory on GPU forces trade-off in model size / data resolution

Large Model Support
(PowerAI)
Use system memory and GPU to support more complex models and higher resolution data
Deep Learning / AI Enterprise Use Cases

- **AUTOMOTIVE**
  - Auto sensors reporting location, problems

- **COMMUNICATIONS**
  - Location-based advertising

- **CONSUMER PACKAGED GOODS**
  - Sentiment analysis of what's hot, problems

- **FINANCIAL SERVICES**
  - Risk & portfolio analysis
  - New products

- **EDUCATION & RESEARCH**
  - Experiment sensor analysis

- **HIGH TECHNOLOGY / INDUSTRIAL MFG.**
  - Mfg. quality
  - Warranty analysis

- **LIFE SCIENCES**
  - Clinical trials

- **MEDIA/ENTERTAINMENT**
  - Viewers / advertising effectiveness

- **ON-LINE SERVICES / SOCIAL MEDIA**
  - People & career matching

- **OIL & GAS**
  - Drilling exploration sensor analysis

- **RETAIL**
  - Consumer sentiment

- **TRAVEL & TRANSPORTATION**
  - Sensor analysis for optimal traffic flows

- **UTILITIES**
  - Smart Meter analysis for network capacity,

- **LAW ENFORCEMENT & DEFENSE**
  - Threat analysis - social media monitoring, photo analysis

- **HEALTH CARE**
  - Patient sensors, monitoring, EHRs
Data Preparation Tools

Tumor Proliferation Assessment – mitosis detection
Images from electron-microscope
Size of image - 70K * 60K

Data Transformation

<table>
<thead>
<tr>
<th>Framework</th>
<th>Format</th>
<th>Input Size (Faster R-CNN)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Caffe</td>
<td>LMDB</td>
<td>1K*1K</td>
</tr>
<tr>
<td>TensorFlow</td>
<td>TensorRecord</td>
<td>1K*1K</td>
</tr>
</tbody>
</table>

Data Distribution among training, validation and testing

Data Shuffle
ELINAR Sales Order processing

- Traditional capture is difficult on Sales Orders (SO)
  - Sales orders contain line data; one SO can have hundreds or thousands of different line items
  - Large enterprises might have tens of thousands of clients ordering items or services by email
  - Each client might have multiple locations that each has unique order template(s)
- Sample calculation: 40 000 clients x 20 locations -> 800 000 unique Sales Order templates
- To implement using traditional capture by templating:
  - 10 hours / template -> 8 million hour exercise -> very bad business case!
- Each order could have hundreds of complex order items
Sales Order Processing by ELINAR using PowerAI

Old orders / invoices + extracted information

AI Training

= Several weeks of Super Computer capacity (Power8 Minsky + power.ai)

Incoming Order / Invoice

Datacap OCR / Layout

Trained AI Model

Datacap Validation & Verification

Datacap Extraction

Customer ERP / Finance
Backup
Which Deep Learning Framework to Pick?

Driven by Modeling capabilities:
• Interfaces available
• Algorithms supported
• Ecosystem
• Model deployment
• Cross platform
• Ease of use

Data Scientist Challenges (Source: NVIDIA)

<table>
<thead>
<tr>
<th>Deep Learning Needs</th>
<th>Why</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Scientists</td>
<td>New computing model</td>
</tr>
<tr>
<td>Latest Algorithms</td>
<td>Rapidly evolving</td>
</tr>
<tr>
<td>Fast Training</td>
<td>Impossible -&gt; Practical</td>
</tr>
<tr>
<td>Deployment Platforms</td>
<td>Must be available everywhere</td>
</tr>
</tbody>
</table>

Metrics can also play a role:
• Speed of execution
• Resources required (CPU and Memory capacity)
• GPU support and performance
• Distributed systems capability
## Comparison of Deep Learning Frameworks

<table>
<thead>
<tr>
<th>Framework</th>
<th>Languages</th>
<th>Tutorials and training materials</th>
<th>CNN modeling capability</th>
<th>RNN modeling capability</th>
<th>Architecture: easy-to-use and modular front end</th>
<th>Speed</th>
<th>Multiple GPU support</th>
<th>Keras compatible</th>
</tr>
</thead>
<tbody>
<tr>
<td>Theano</td>
<td>Python, C++</td>
<td>++</td>
<td>++</td>
<td>+</td>
<td>++</td>
<td>++</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>TensorFlow</td>
<td>Python</td>
<td>+++</td>
<td>+++</td>
<td>+++</td>
<td>+++</td>
<td>++</td>
<td>++</td>
<td>++</td>
</tr>
<tr>
<td>Torch</td>
<td>Lua, Python (new)</td>
<td>+</td>
<td>+++</td>
<td>++</td>
<td>+++</td>
<td>++</td>
<td>+</td>
<td>++</td>
</tr>
<tr>
<td>Caffe</td>
<td>C++</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>MXNet</td>
<td>R, Python, Julia, Scala</td>
<td>++</td>
<td>+</td>
<td>++</td>
<td>++</td>
<td>+++</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Neon</td>
<td>Python</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>++</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>CNTK</td>
<td>C++</td>
<td>+</td>
<td>+++</td>
<td>+</td>
<td>+</td>
<td>++</td>
<td>+</td>
<td>+</td>
</tr>
</tbody>
</table>

Scalability vs Flexibility

Scalability-oriented

- Use-cases in mind
  - Image/speech recognition system
  - Fast DL as a service in cloud
- Problem type
  - A few general applications
  - 10+ million training samples
  - 10+ nodes cluster w/ fast network
- Possible bottleneck
  - Tuning of well-known algorithms
  - Distributed computation for model/data-parallel training

Flexibility-oriented

- Use-cases in mind
  - New algorithm research
  - R&D projects for AI products
- Problem type
  - Various specific applications
  - 10+ k training samples
  - 1 node with multiple GPUs
- Possible bottleneck
  - Trial-and-error in prototyping
  - Debugging, profiling & refactoring
  - (wait time during compilation)

Source: Preferred Networks presentation, 2017 OpenPOWER Developer Congress
OSDB Types

• Relational
  – MariaDB and EnterpriseDB
  – Utility databases for applications

• NoSQL Databases
  – Document Databases
  – Graph Databases
  – Key-Value Databases
  – Wide Column Stores
Enterprise-Ready Open-Source Databases

While there are myriad open-source databases, consider these six for their enterprise versions and technical support.

**MongoDB**

mongodb.com  
**Classification:** NoSQL document store  
**Optimized for:** Document model and document stores; semi-structured or unstructured data  
**Technical support:** docs.mongodb.com/manual/support

**EnterpriseDB**

enterprisedb.org  
**Classification:** Open-source object relational database  
**Optimized for:** Variety of transactional work; relational structured queries to object store and retrieval  
**Technical support:** enterprisedb.com/services/support

**MariaDB**

mariadb.org  
**Classification:** Open-source relational database  
**Optimized for:** Transactional SQL-based queries and updates  
**Technical support:** Community support available

**Cassandra**

cassandra.apache.org; Enterprise version available at datastax.com  
**Classification:** NoSQL wide column store  
**Optimized for:** NoSQL environments with high data volumes that require high performance and scalability  
**Technical support:** datastax.com

**Redis**

redis.io; Enterprise version available at redislabs.com  
**Classification:** NoSQL in-memory key value store  
**Optimized for:** Data queues, strings, lists, counts, caching, statistics, text, session IDs, videos  
**Technical support:** redis.io/support

**Neo4J**

neo4j.com  
**Classification:** NoSQL graph store  
**Optimized for:** Graph database, data stored as edges, nodes or attributes  
**Technical support:** support.neo4j.com

Information contributed by Rick Murphy, migration solution architect, IBM Lab Services, and Mark Short, lead migration consultant, IBM Lab Services Migration Factory.