Deep Learning – a cookbook view

or “Comparative Analysis of Different Deep Learning Solutions“
The evolution of artificial intelligence

**Small data sets**
- Early artificial intelligence
  - ENIAC Heralded the Giant Brain, used for WW II ballistics
  - Industrial robots

**Massive structured data sets**
- Machine learning
  - Deep Blue Beating World Chess Champion Kasparov
  - DARPA Challenge Autonomous vehicle drove 132 miles

**Massive unstructured big data**
- Deep Learning
  - Unsupervised training
  - Generic code
  - Pattern recognition

**Systems can**
- Observe
- Test
- Refine

**Successes**
- AlphaGO First Computer GO program to beat a human
- Deep Face Facial verification
- Libratus AI Poker App
- Digital virtual assistants Siri
- Google Self-driving cars

**Statistical and mathematical models applied to solve problems**
- Advanced Analytics and Heuristic

**Predictive models defined by machines based neural networks**

**Timeline**
- 1940 – 1980
- 1990 – 2000s
- Today
Traditional machine learning
Requires feature engineering

Training Data → Feature engineering → Machine learning algorithm

Training

Prediction

Data → Feature extraction → Learned model (prediction function) → Prediction
Deep learning
Efficient data representations, no more feature engineering

Training Data -> Deep learning algorithm

Training

Prediction (inference)

Data -> Learned model (transformation and prediction function) -> Prediction
Types of artificial neural networks
Topology to fit data characteristics

Convolutional:
Images

Fully connected:
Speech, text, sensor

Recurrent:
Speech, text, sensor
Terminology

- **Epoch**: Timing unit or parameter affected by the number of iterations of the training dataset through the model.
- **Training data**: Input data to the model.
- **Batch**: Protocol for training where the model is trained on a subset of the training data.
- **Model**: Algorithm that learns from and makes predictions on data.
- **Iteration**: In the context of training, a single pass through the training dataset.
- **Flower Predictions**: Predictions made by the model.
- **Errors**: Difference between true labels and predicted labels.
- **House**: True class labels.
- **Worker 1**
  - **Strong scaling**: Each worker 1 processes a batch of data.
  - **Weak scaling**: Each worker 2 processes a batch of data.
## Why deep learning?

### Applications

<table>
<thead>
<tr>
<th>Vision</th>
<th>Speech</th>
<th>Text</th>
<th>Other</th>
</tr>
</thead>
<tbody>
<tr>
<td>Search &amp; information extraction</td>
<td>Interactive voice response (IVR) systems</td>
<td>Search and ranking</td>
<td>Recommendation engines</td>
</tr>
<tr>
<td>Security/Video surveillance</td>
<td>Voice interfaces (Mobile, Cars, Gaming, Home)</td>
<td>Sentiment analysis</td>
<td>Advertising</td>
</tr>
<tr>
<td>Self-driving cars</td>
<td>Security (speaker identification)</td>
<td>Machine translation</td>
<td>Fraud detection</td>
</tr>
<tr>
<td>Medical imaging</td>
<td>Health care</td>
<td>Question answering</td>
<td>AI challenges</td>
</tr>
<tr>
<td>Robotics</td>
<td>People with disabilities</td>
<td></td>
<td>Drug discovery</td>
</tr>
</tbody>
</table>

### Why deep learning?

Deep learning is crucial in various applications due to its ability to process and learn from large amounts of data. This is particularly relevant in fields such as computer vision, speech recognition, natural language processing, and more. The following are some key reasons why deep learning is important:

- **Flexibility and Adaptability**: Deep learning models can adapt to new tasks and data, making them highly flexible.
- **Large Data Requirements**: Deep learning excels in scenarios where large datasets are available, enabling more accurate and nuanced predictions.
- **Hierarchical Learning**: Deep models capture hierarchical representations of data, useful for complex tasks.
- **End-to-End Learning**: Many deep learning systems can learn end-to-end without explicit feature engineering, simplifying the development process.
- **Reinforcement Learning**: Deep learning is prominent in reinforcement learning, where agents learn to make decisions based on rewards.

These advantages make deep learning a cornerstone of modern artificial intelligence, impacting industries from healthcare to autonomous vehicles.
Applications break down

- **Images**
  - Image analysis
- **Video**
  - Video surveillance
- **Speech**
  - Speech recognition
- **Text**
  - Sentiment analysis
- **Sensor**
  - Predictive maintenance
- **Other**
  - Fraud detection

**Detection**
Look for a known object/pattern

**Generation**
Generate content

**Classification**
Assign a label from a predefined set of labels

**Anomaly detection**
Look for abnormal, unknown patterns
How an individual customer’s AI evolves

Explore
How can AI help me?

- **Do things better**
  - Product development
  - Customer experience
  - Productivity
  - Employee experience

- **Do new things**
  - New disruptions

Experiment
How can I get started?

- **Boundary** constraints
  (regulations, etc.)

- **Data**
  Data model? Location?
  How to **create** a model?
  - Homegrown solution or open source?
  - Simple ML or scalable DL?

- **Design**
  How to design and deploy the PoC?
  - On-prem, cloud?
  - How to think about inference

- **Performance**
  What is the best config to run?
  How to tune the model to improve accuracy?

Scale up and Optimize
How can I scale and optimize?

- **Provisioning** for inference

- **Infrastructure scale up**
  - Training
  - Inference
  - On-prem / cloud / hybrid

- **Data management**
  - Between edge and core
  - Security
  - Updates
  - Regulations
  - Tracing
Key IT challenges are constraining deep learning adoption
Limited knowledge, resources and capabilities

How to get started?

“I need simple, infrastructure and software capabilities to rapidly and efficiently support deep learning app development.”

Immature, sub-optimal foundation

How to go to production?

“I could use more expert advice and tailored solutions for migrating and integrating apps in a production environment.”

Inability to scale and integrate

How to optimize?

“I need help integrating the latest technologies into my deep learning environment to accelerate actionable insights.”

Lack of technology integration capabilities

Content under embargo until Oct 10, 2017
What about AI consumers?

Do it yourself
Current wave of AI / Machine Learning is core to their business. All in-house

Google, Baidu, Facebook, Microsoft, Apple, etc.

How do I do it?
Could benefit from better data science, machine learning, but it is not historically their core-competency

Banks, advertisers, healthcare, manufacturing, food, automotive, etc.

I know better
Super-Experts – current wave is woefully inadequate

Government – DoD, DoE, NSA, NASA, etc.

Not ready for an ASIC. Don’t know what they need exactly. Many still developing on CPUs. Can’t use solutions that can’t be verified or understood

Begging for higher performance ASICs. Know exactly what they want to do. Strong technology pull.
**Where to start?**
Recommend DL stack by vertical application

<table>
<thead>
<tr>
<th>Verticals</th>
<th>Voice interfaces</th>
<th>Social media</th>
<th>Manufacturing</th>
<th>Oil &amp; gas</th>
<th>Connected cars</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data type</td>
<td>Speech</td>
<td>Images</td>
<td>Video</td>
<td>Sensor data</td>
<td></td>
</tr>
<tr>
<td>Data</td>
<td>Small</td>
<td>Moderate</td>
<td>Large</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Typical layers</td>
<td>Convolutional</td>
<td>Fully-connected</td>
<td>Recurrent</td>
<td></td>
<td>Neural Network sits here</td>
</tr>
<tr>
<td>Frameworks</td>
<td>TensorFlow</td>
<td>Caffe 2</td>
<td>CNTK</td>
<td>Torch</td>
<td></td>
</tr>
<tr>
<td>Infrastructure</td>
<td>x86</td>
<td>GPUs</td>
<td>FPGAs</td>
<td>TPU ?</td>
<td></td>
</tr>
</tbody>
</table>
# Neural Network: Popular Networks

<table>
<thead>
<tr>
<th>Network</th>
<th>Model size (# params)</th>
<th>Model size (MB)</th>
<th>GFLOPs (forward pass)</th>
</tr>
</thead>
<tbody>
<tr>
<td>AlexNet</td>
<td>60,965,224</td>
<td>233 MB</td>
<td>0.7</td>
</tr>
<tr>
<td>GoogleNet</td>
<td>6,998,552</td>
<td>27 MB</td>
<td>1.6</td>
</tr>
<tr>
<td>VGG-16</td>
<td>138,357,544</td>
<td>528 MB</td>
<td>15.5</td>
</tr>
<tr>
<td>VGG-19</td>
<td>143,667,240</td>
<td>548 MB</td>
<td>19.6</td>
</tr>
<tr>
<td>ResNet50</td>
<td>25,610,269</td>
<td>98 MB</td>
<td>3.9</td>
</tr>
<tr>
<td>ResNet101</td>
<td>44,654,608</td>
<td>170 MB</td>
<td>7.6</td>
</tr>
<tr>
<td>ResNet152</td>
<td>60,344,387</td>
<td>230 MB</td>
<td>11.3</td>
</tr>
</tbody>
</table>
## Today’s scale
Model size, data size, compute requirements

<table>
<thead>
<tr>
<th>Application</th>
<th>Model</th>
<th>Training data</th>
<th>FLOPs per epoch</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vision</td>
<td>$1.7 \times 10^9$&lt;br&gt;$\sim 6.8$ GB</td>
<td>$14 \times 10^6$ images&lt;br&gt;$\sim 2.5$ TB (256x256)&lt;br&gt;$\sim 10$ TB (512x512)</td>
<td>$6 \times 1.7 \times 10^9 \times 14 \times 10^6$&lt;br&gt;$\sim 1.4 \times 10^{17}$</td>
</tr>
<tr>
<td>Speech</td>
<td>$60 \times 10^6$&lt;br&gt;$\sim 240$ MB</td>
<td>$100K$ hours of audio&lt;br&gt;$\sim 34 \times 10^9$ frames&lt;br&gt;$\sim 50$ TB</td>
<td>$6 \times 60 \times 10^6 \times 34 \times 10^9$&lt;br&gt;$\sim 1.2 \times 10^{19}$</td>
</tr>
<tr>
<td>Text</td>
<td>$6.5 \times 10^6$&lt;br&gt;$\sim 260$ MB</td>
<td>$856 \times 10^6$ words</td>
<td>$6 \times 6.5 \times 10^6 \times 856 \times 10^6$&lt;br&gt;$\sim 3.3 \times 10^{16}$</td>
</tr>
<tr>
<td>Signals</td>
<td>$1.2 \times 10^6$&lt;br&gt;$\sim 4.8$ MB</td>
<td>$3 \times 10^6$ frames</td>
<td>$6 \times 1.2 \times 3 \times 10^6 \times 3 \times 10^6$&lt;br&gt;$6.5 \times 10^{13}$</td>
</tr>
</tbody>
</table>
Today’s hardware
Model size, data size, compute requirements

<table>
<thead>
<tr>
<th>Application</th>
<th>Model</th>
<th>Training data</th>
<th>FLOPs per epoch</th>
</tr>
</thead>
</table>
| Vision      | 1.7 * 10^9
~6.8 GB  | 14*10^6 images
~2.5 TB (256x256)
~10 TB (512x512) | 6*1.7*10^9*14*10^6
~1.4*10^17 |

1 epoch per hour:~39 TFLOPS

Today’s hardware:
Google TPU2: 180 TFLOPS Tensor ops (FP16 ??)
NVIDIA Tesla V100: 15 TFLOPS SP (30 TFLOPS FP16 , 120 TFLOPS Tensor ops), 12 GB memory
NVIDIA Tesla P100: 10.6 TFLOPS SP, 16 GB memory
NVIDIA Tesla K40: 4.29 TFLOPS SP, 12 GB memory
NVIDIA Tesla K80: 5.6 TFLOPS SP (8.74 TFLOPS SP with GPU boost), 24 GB memory
INTEL Xeon Phi: 2.4 TFLOPS SP

Superdome X: ~21 TFLOPS SP, 24 TB memory
So what to recommend?

Software

- Caffe
- theano
- torchvision
- KALDI

Hardware

- NVIDIA
- Intel
- ARM
- AMD
- Google
Building performance models

- Alex Net
- GoogleNet
- VGG-16, VGG-19
- ResNet 50, 101, 152
- Eng Acoustic Model

TensorFlow
Caffe
Tensor RT
BVLC Caffe

Hardware
Scalable, automated real-time intelligence
Populated with 8 GPUs
TensorFlow – Weak Scaling – Training – Different models performance in TensorFlow. Scaling up to 8 GPUs

Speedup for up to 8 GPUs

TensorFlow - Inference (Inferences per Second) - Different Models with different Batch numbers

HOW TO ANALYZE ALL THE DIFFERENT NUMBERS.

AS WE ADD MORE OPTIONS and MORE TECHNOLOGIES IT WOULD BE IMPOSSIBLE TO USE
HPE demystifies deep learning for faster intelligence across all organizations

New IT expertise, blueprints and technologies to get started, scale, integrate and optimize

**Get started rapidly:**
Develop deep learning models

**Scale and Integrate:**
Deliver attractive returns

**Optimize Environment:**
Enhance competitive advantage

**IT expertise and solutions**
to “get started” with deep learning models

- Expertise
  - Rapid technology selection guides
  - State of the art training

- Solutions
  - Integrated purpose-built solutions
  - Out of the box solutions

**Proven blueprints and services**
for “scalable” production deployments

- Proven Blueprints
  - Reference Architectures
  - Innovation labs for best practices

- Services
  - Deploy, integrate and support
  - Flexible, on-demand capacity

**Technology integration**
capabilities to maximize performance

- Integration capabilities
  - Enhanced global Centers of Excellence
  - Next gen technology integration
Select ideal technology configurations with HPE Deep Learning Cookbook

### “Book of recipes” for deep learning workloads

- **Comprehensive tool set** based on extensive benchmarking
- **Includes** 11 workloads with 8 DL frameworks and 8 HPE hardware systems
- **Estimates workload performance** and recommends an optimal HW/SW stack for that workload

### Expert advice to get you started

- **Informed decision making** - optimal hardware and software configurations
- **Eliminates the “guesswork”** - validated methodology and data
- **Improves efficiency** - detects bottlenecks in deep learning workloads

### Availability of complete toolset

- **Deep Learning Benchmarking Suite**: available on GitHub Dec 2018
- **Deep Learning Performance Analysis Tool**: planned to be released in the beginning of 2018.
- **Reference configurations**: available soon on HPE.com website
Deep Learning Cookbook helps to pick the right HW/SW stack

**Benchmarking Suite**
- Benchmarking scripts
- Reference models
- Performance metrics

**Knowledgebase**
- Performance results
  - 11 reference models
  - 8 frameworks
  - 8 hardware systems

**Performance and scalability models**
- Machine learning (SVR) to predict performance of core operations
- Analytical communication models
- Analytical models for overall performance

**Reporting tool**
- Performance results
- Performance prediction for arbitrary ANNs
- Scalability prediction
- Optimal HW/SW configuration for a given workload

**Reference configurations**
- Image classification
- Others to come

will be available externally
internal assets
## Deep Learning Cookbook

### Automatic Meeting Notes  |  Video Surveillance  |  Hospital Smart Care Unit  |  Custom

- 🎬 Images
- 🎬 Videos
- 🎬 Text
- 🎬 Speech
- 🎬 Sensor Data

### Data and Model

<table>
<thead>
<tr>
<th>Data size</th>
<th>Epochs</th>
<th>Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>100000000</td>
<td>50</td>
<td>VGG19</td>
</tr>
</tbody>
</table>

### Hardware

- **Server**: Apollo 6500
- **Processor unit**: NVIDIA P100
- **Cluster size**: 2
- **Interconnect**: InfiniBand QDR

### Software

- **Framework**: Caffe2
- **Batch size**: 1024
- **Scaling**: strong

### Training performance

<table>
<thead>
<tr>
<th>Data</th>
<th>Hardware</th>
<th>Software</th>
<th>Time (hours)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Server</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Apollo 6500</td>
<td>Mode:</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Caffe2</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>30000000</td>
<td>Epochs:</td>
<td>Model:</td>
<td></td>
</tr>
<tr>
<td>50</td>
<td></td>
<td>Apollo 6500</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **Framework**: Caffe2
- **Interconnect**: 1024(Gig)

- **Server**: Apollo 6500
- **Cluster size**: 2
- **Interconnect**: IB

- **Framework**: Caffe2
- **Batch**: 1024(Gig)

- **Server**: Apollo 6500
- **Epochs**: 50
- **Model**: VGG19

- **Framework**: Caffe2
- **Batch**: 1024(Gig)
Thank you

Natalia Vassilieva
nvassilieva@hpe.com

Sergey Serebryakov
sergey.serebryakov@hpe.com

Sorin Cheran
sorin.cheran@hpe.com

Bruno Monnet
bruno.monnet@hpe.com