NATIONAL CENTER FOR SUPERCOMPUTING APPLICATIONS

Intro to Deep Learning on HAL

ILLINOIS
NCSA | National Center for Supercomputing Applications
What is AI / ML / DL?

Since an early flush of optimism in the 1950s, smaller subsets of artificial intelligence – first machine learning, then deep learning, a subset of machine learning – have created ever larger disruptions.
1. Artificial intelligence was born in the 1950s, when a handful of pioneers started asking whether computers could be made to “think” like human.

2. A concise definition of the field would be as follows: the effort to automate intellectual tasks normally performed by humans.
1. Machine learning arises from this question: could a computer go beyond “what we know how to order it to perform” and learn on its own how to perform a specified task?

2. The field of machine learning focuses on the study and construction of computer systems that can learn from data without being explicitly programmed.

3. Machine learning algorithms and techniques are used to build models to discover hidden patterns and trends in the data, allowing for data-driven decisions to be made.
What is DL?

1. Deep learning is a specific subfield of machine learning.
2. Deep learning is a new take on learning representations from data that puts an emphasis on learning successive layers of increasingly meaningful representations.
3. The “deep” in deep learning isn’t a reference to any kind of deeper understanding achieved by the approach.
4. Appropriate names for the field could have been layered representations learning and hierarchical representations learning.
What is DL?

Artificial intelligence, machine learning, and deep learning

What do the representations learned by a deep-learning algorithm look like? Let's examine how a network several layers deep (see figure 1.5) transforms an image of a digit in order to recognize what digit it is.

As you can see in figure 1.6, the network transforms the digit image into representations that are increasingly different from the original image and increasingly informative about the final result. You can think of a deep network as a multistage information-distillation operation, where information goes through successive filters and comes out increasingly purified (that is, useful with regard to some task).

So that's what deep learning is, technically: a multistage way to learn data representations. It's a simple idea—but, as it turns out, very simple mechanisms, sufficiently scaled, can end up looking like magic.

1.1.5 Understanding how deep learning works, in three figures

At this point, you know that machine learning is about mapping inputs (such as images) to targets (such as the label "cat"), which is done by observing many examples of input and targets. You also know that deep neural networks do this input-to-target mapping.
This is a Interdisciplinary Field
But Why Now?

- The two key ideas of deep learning for computer vision, convolutional neural networks and back-propagation, were already well understood in 1989.
- The Long Short-Term Memory (LSTM) algorithm, which is fundamental to deep learning for time series, was developed in 1997 and has barely changed since.

- **Hardware**: high-performance GPUs
- **Datasets**: large datasets collected from internet
- **Algorithmic advances**: activation functions, optimization schemes.
ML / DL Approaches

• **Classification**
  • Predict category given input data.

• **Regression**
  • Predict numeric value given input data.

• **Cluster Analysis**
  • Organize similar items into groups.

• **Association Analysis**
  • Find rules to capture co-occurrence relationships between items.
Supervised vs Unsupervised

- **Supervised Approaches**
  - Target (what you’re trying to predict) is provided.
  - “Labeled” data
  - Classification and regression approaches are supervised.

- **Unsupervised Approaches**
  - Target is unknown or unavailable.
  - “Unlabeled” data
  - Association analysis and cluster analysis are unsupervised.
Anatomy of a Neural Network

- In Deep Learning, the layered representations are (almost always) learned via models called neural networks, which include:
  - The input data and corresponding targets.
  - The layers, which are combined into a network (or model).
  - The loss function, which defines the feedback signal used for learning.
  - The optimizer, which determines how learning proceeds.
Convolutional Layer

```
CONVOLUTION

INPUT 6x6 IMAGE

FILTER 3x3

CONVOLUTION

3+1+2+0+0+0-1-8-2 = -5

OUTPUT 4x4 IMAGE
```
A Deep CNN

A lot of the work is figuring out hyperparameters = #filters, stride, padding etc.
Typically size → trend down,
# filters → trend up.

Typical ConvNet layers:
- Convolution
- Pooling
- Fully connected layers

Pooling (max):
Find max val in section
- Reduces size of repres.
- Speeds up computation
- Makes some of the detected feat. more robust.
Gradient-based Optimization

• Feed the **input data** to the neural network and optimize a set of **weights** which minimize the loss function’s feedback.

• This **gradual adjustment, also called training**, is basically the learning that machine learning is all about.

• There are parameters need to be set before the learning process begins (**learning-rate, batch-size**), which called hype-parameters.
**Batch Size**

**Mini-Batch Gradient Descent**

- Split your data into mini-batches.
- Do gradient descent after each batch.
- This way you can progress after just a short while.

**Choosing the Minibatch Size**

- $\text{size}=m \rightarrow \text{Batch Grad Desc.}$
- $\text{size}=1 \rightarrow \text{Stochastic Grad Desc.}$

**Graphs**

- **Standard**
  - Cost vs. #ITER
- **Mini-Batch**
  - Cost vs. #ITER

**Batch**

- Too long per iteration
- Shorter iteration
- Uses vectorization

**Stochastic**

- Loss almost all speed from vectorization
1. Draw a batch of training samples \( x \) and corresponding targets \( y \).
2. Run the network on \( x \) to obtain predictions \( y_{\text{pred}} \).
3. Compute the loss of the network on the batch, a measure of the mismatch between \( y_{\text{pred}} \) and \( y \).
4. Compute the gradient of the loss with regard to the network’s parameters (a *backward pass*).
5. Move the parameters a little in the opposite direction from the gradient (for example \( W -= \text{step} \times \text{gradient} \)) thus reducing the loss on the batch a bit.
• QUESTIONS SO FAR?
HAL System Overview

• NSF-funded IBM cluster for Deep Learning applications
• 16x computing nodes
• 160x CPU cores (4x SMT), 4x NVIDIA V100 GPUs
• 72TB of storage on 100Gb/s network
• The origin of the name
  • 2001: a space odyssey
  • Early concept of an Artificial Intelligence system
Major Tool

• Watson Machine Learning Community Edition: WMLCE-1.6.1 (former PowerAI)
  • WMLCE is an enterprise software distribution that combines popular open source deep learning frameworks, efficient AI development tools, and accelerated IBM Power Systems servers.
    • Caffe: ML / DL framework developed by Berkeley AI Research
    • Pytorch: ML / DL framework developed by Facebook
    • Tensorflow: ML / DL framework developed by Google
    • Rapids cuDF and cuML: ML / DL framework incubated by NVidia
    • Distributed Deep Learning (or DDL): MPI-based communication library
Slurm Workload Manager
How to properly use Slurm?

• Original Slurm command could be complex
  • `srun --partition=gpux4 --time=72:00:00 --nodes=1 --ntasks-per-node=144 --sockets-per-node=2 --cores-per-socket=18 --threads-per-core=4 --mem-per-cpu=1500 --wait=0 --export=ALL --gres=gpu:v100:4 --pty /bin/bash`

• We have a easier solution!
  • Slurm Wrapper Suite v0.3
Slurm Wrapper Suite

“swrun” Usage

• Only 4 options
  • Partition (required)
  • CPUs Per GPU (optional)
  • Wall Time (optional)
  • Singularity Container (optional)

• Restrictions
  • Partitions vary by GPU number (gpux1, gpux2, gpux3, …)
  • CPU Per GPU (12 <= c <= 36, default 12)
  • Wall Time (1 <= t <= 72, default 24 hours)

• Default if selecting 1 gpu
  • gpux1(required), 12x CPUs, 18GB Memory, 1x GPU, 24 Hrs

• Full Node Example
  • swrun -p gpux4 -c 36 -t 72
Slurm Wrapper Suite

• “swbatch” Usage
  • Only 7 options
    • Partition (**required**)
    • CPUs Per GPU (**optional**)
    • Wall Time (**optional**)
    • Job name (**optional**)
    • Output file (**optional**)
    • Error file (**optional**)
    • Singularity Container (**optional**)

• Restrictions
  • Partitions vary by GPU number (x1, x2, x3, …)
  • CPU Per GPU (12 <= c <= 36, default 12)
  • Wall Time (1 <= t <= 72, default 24 hours)
Slurm Wrapper Suite

- swqueue (new): show cluster usage within terminal
Tensorboard

• Start a interactive session
  • swrun –p gpux1

• Load powerai on remote node (hal compute node)
  • module load powerai

• Start a Tensorboard session on remote node (hal compute node)
  • export TMPDIR=/tmp/$USER;
  • mkdir -p $TMPDIR; echo $TMPDIR
  • tensorboard --logdir=./tensorflow_logs --port <port_num>

• Start a ssh Tunnel from local machine (your laptop)
  • ssh -L <port_number>:<remote_name>:<port_number> <netid>@hal.ncsa.illinois.edu

• Access the Tensorboard webpage from your local web browser
  • https://localhost:<port_number>
Jupyter Notebook

• Start a interactive session
  • swrun –p gpux1

• Load powerai on remote node (hal compute node)
  • module load powerai

• Start a jupyter session on remote node (hal compute node)
  • unset XDG_RUNTIME_DIR
  • jupyter-notebook --ip=0.0.0.0 --port=<port_number> (random port number)

• Start a ssh Tunnel from local machine (your laptop)
  • ssh -L <port_number> :<remote_name>:<port_number> <netid>@hal.ncsa.illinois.edu

• Access the jupyter webpage from your local web browser
  • https://localhost:<port_number>

• Jupyter nodebook password can be found
  • $HOME/.jupyter/.jupyter_pass
Live Demo

- Pytorch: MNIST Digits Classification.
- Tensorflow: Cifar10 Image Classification.
QUESTIONS?

HAL Tickets: help+isl@ncsa.illinois.edu
THANK YOU FOR YOUR TIME!