Deep Learning on Hadoop

Scale out Deep Learning on YARN
Josh Patterson

Email: josh@pattersonconsultingtn.com
Twitter: @jpatanooga
Github: github.com/jpatanooga

Past
Published in IAAI-09:
“TinyTermite: A Secure Routing Algorithm”
Grad work in Meta-heuristics, Ant-algorithms
Tennessee Valley Authority (TVA)
Hadoop and the Smartgrid
Cloudera
Principal Solution Architect

Today: Patterson Consulting
Overview

• What Is Deep Learning?
• Neural Nets and Optimization Algorithms
• Implementation on Hadoop/YARN
• Results
What is Deep Learning?

Machine perception, pattern recognition.
What Is Deep Learning?

- Algorithms called neural nets that learn to recognize patterns:
  - Nodes learn smaller features of larger patterns
  - And combine them to recognize feature groups
  - Until finally they can classify objects, faces, etc.
  - Each node layer in net learns larger groups
Properties of Deep Learning

- Small training sets, they learn unsupervised data
- They save data scientists months of work
- Anything you can vectorize, DL nets can learn
- They can handle millions of parameters
- After training, DL models are one, small vector
Chasing Nature

- Learning sparse representations of auditory signals
- Leads to filters that correspond to neurons in early audio processing in mammals
- When applied to speech
  - Learned representations show a resemblance to cochlear filters in the auditory cortex.
Yann Lecun on Deep Learning

- DL is the dominant method for acoustic modeling in speech recognition
- It is becoming dominant in machine vision for:
  - object recognition
  - object detection
  - semantic segmentation.
Deep Neural Nets

“Deep” > 1 hidden layer
Restricted Boltzmann Machines

- RBMs are building blocks for deeper nets. They deal with Binary and Continuous data differently.

- Binary

\[
E(v, h) = -\frac{1}{\sigma^2} \left( \sum_i c_i v_i + \sum_j b_j h_j + \sum_{i,j} v_i w_{ij} h_j \right).
\]

- Continuous

\[
E(v, h) = \frac{1}{2\sigma^2} \sum_i v_i^2 - \frac{1}{\sigma^2} \left( \sum_i c_i v_i + \sum_j b_j h_j + \sum_{i,j} v_i w_{ij} h_j \right).
\]
What Is a Deep-Belief Network?

- A stack of restricted Boltzmann machines
- A generative probabilistic model
- 1) A visible (input) layer ...
- 2) Two or more hidden layers that learn more & more complex features...
- 3) An output layer that classifies the input.
A Recursive Neural Tensor Network?

- RNTN’s are top-down; DBN’s are feed-forward
- A tensor is 3d matrix
- RNTN’s handle multiplicity
- Scene and sentence parsing, windows of events
A Deep Autoencoder?

- DA’s are good for QA systems like Watson
- They encode lots of data in smaller number vectors
- Good for Image Search, Topic Modeling
A Convolutional Net?

- ConvNets slice up features with shared weights
- ConvNets learns images in patches from a grid
- Very good at generalization
DeepLearning4J

- The most complete, production-ready open-source DL lib
- Written in Java: Uses Akka, Hazelcast and Jblas
- Distributed to run fast, built for non-specialists
- More features than Theano-based tools
- Talks to any data source, expects 1 format
Nonspecialists can rely on its conventions to solve computationally intensive problems.

Usability first – DL4J follows ML tool conventions.

DL4J’s nets work equally well with text, image, sound and time-series.

DL4J will integrate with Python community through SDKs.
Vectorized Implementation

- Handles lots of data concurrently.
- Any number of examples at once, but the code does not change.
- Faster: Allows for native and GPU execution.
- One input format: Everything is a matrix.
- Image, sound, text, time series are vectorized.
DL4J vs Theano vs Torch

- DL4J’s distributed nature means problems can be solved by “throwing CPUs at them.”
- Java ecosystem has GPU integration tools.
- Theano is not distributed, and Torch7 has not automated its distribution like DL4J.
- DL4J’s matrix multiplication is native w/ Jblas.
What Are Good Applications for DL?

- Recommendation engines (e-commerce)
  - DL can model consumer and user behavior
- Anomaly detection (fraud, money laundering)
  - DL can recognize early signals of bad outcomes
- Signal processing (CRM, ERP)
  - DL has predictive capacity with time-series data
DL4J Vectorizes & Analyzes Text

- Sentiment analysis
- Logs
- News articles
- Social media
DL on Hadoop and AWS

Build Your Own Google Brain ...
Past Work: Parallel Iterative Algos on YARN

- Started with
  - Parallel linear, logistic regression
  - Parallel Neural Networks
  - "Metronome" packages DL4J for Hadoop
- 100% Java, ASF 2.0 Licensed, on Github
MapReduce vs. Parallel Iterative

MapReduce

- Input
  - Map
  - Map
  - Map
  - Reduce
  - Reduce

- Output

Parallel Iterative

- Superstep 1
  - Processor
  - Processor
  - Processor

- Superstep 2
  - Processor
  - Processor
  - Processor

...
SGD: Serial vs Parallel

Training Data → Model

Worker 1
- Partial Model

Worker 2
- Partial Model

Worker N
- Partial Model

Split 1

Split 2

Split 3

Master
- Global Model
Managing Resources

- Running through YARN on Hadoop is important
- Allows for workflow scheduling
- Allows for scheduler oversight
- Allows the jobs to be first-class citizens on Hadoop
- And shares resources nicely
Parallelizing Deep-Belief Networks

- Two-phase training
  - Pretrain
  - Fine-tune
- Each phase can do multiple passes over dataset
- Entire network is averaged at master
PreTrain and Lots of Data

We’re exploring how to better leverage the **unsupervised** aspects of the PreTrain phase of Deep-Belief Networks.

- Allows for the use of far more unlabeled data.
- Allows us to more easily model the massive amounts of structured data in HDFS.
Results

DL4J on Hadoop is fast and accurate
DBNs on IR Performance

- Faster to train.
- Parameter averaging is an automatic form of regularization.
- Adagrad with IR allows for better generalization of different features and even pacing.
Scale-out Metrics

- Batches of records can be processed by as many workers as there are data splits.
- Message passing overhead is minimal.
- Exhibits linear scaling.

Example: 3x workers, 3x faster learning.
Usage From Command Line

- Run Deep Learning on Hadoop
  - `yarn jar iterativereduce-0.1-SNAPSHOT.jar [props file]`

- Evaluate model
  - `./score_model.sh [props file]`
Handwriting Renders
Facial Renders
What’s Next?

- GPU integration in the cloud (AWS)
- Better vectorization tooling & data pipelines
- Move YARN version back over to JBLAS for matrices
- Spark
References

“A Fast-Learning Algorithm for Deep Belief Nets”

“Large Scale Distributed Deep Networks”
Dean, Corrado, Monga - NIPS (2012)

“Visually Debugging Restricted Boltzmann Machine Training with a 3D Example”
Yosinski, Lipson - Representation Learning Workshop (2012)
Parameter Averaging

- McDonald, 2010
  - Distributed Training Strategies for the Structured Perceptron
- Langford, 2007
- Vowpal Wabbit
- Jeff Dean’s Work on Parallel SGD
- DownPour SGD