IBM’s Open-Source Based AI Developer Tools

Sumit Gupta
VP, AI, Machine Learning & HPC
IBM Cognitive Systems
@SumitGup
guptasum@us.ibm.com

March 2019
AI Software Portfolio Strategy
Deliver a comprehensive platform that enables data science at all skill levels

- Prepare Data
- Build AI Models (Machine / Deep Learning)
- Train AI Models: Interactive or Batch
  - With GPU acceleration
- Deploy & Manage Model Lifecycle
  - Inference on CPU, GPU, FPGA
- AI Model Performance Monitoring

Scale to Enterprise-wide Deployment
- Multiple data scientists
- Shared Cluster / Hardware Infrastructure

Hybrid Cloud: Common experience on-premise and in public cloud
IBM Open Source Based AI Stack

**Auto-AI software: PowerAI Vision, IBM Auto-AI**

**Watson Studio**
- WML CE
  - Data Preparation
  - Model Development Environment

**Watson Machine Learning**
- Watson ML Accelerator
- Watson ML CE
  - Runtime Environment
  - Train, Deploy, Manage Models

**Watson OpenScale**
- Model Metrics, Bias, and Fairness Monitoring

---

**Previous Names:**
- WML Accelerator = PowerAI Enterprise
- WML Community Ed. = PowerAI-base

**Accelerated AC922 Power9 Servers**
- Runs on x86 & other storage too
- Available on Private Cloud or Public Cloud

**Storage**
- (Spectrum Scale ESS)
Our Focus: Ease of Use & Faster Model Training Times

- **Watson ML Accelerator**
  - Distributed Deep Learning (DDL)
  - Auto Hyper-Parameter Optimization (HPO)
  - Elastic Distributed Training (EDT) & Elastic Distributed Inference (EDI)

- **IBM Spectrum Conductor**
  - Apache Spark, Cluster Virtualization, Job Orchestration

- **Watson ML**
  - Model Management & Execution
  - Model Life Cycle Management

- **Watson ML Community Edition (WML CE)**
  - **WML CE: Open Source ML Frameworks**
    - TensorFlow
    - PyTorch
    - Chainer
    - Snap ML
  - Large Model Support (LMS)
  - DDL-16

- **Infrastructure Designed for AI**
  - Power9 or x86 Servers with GPU Accelerators
  - Storage (ESS)
Snap ML
Distributed High Performance Machine Learning Library

Snap Machine Learning (ML) Library

- **GPU Accelerated**
  - Logistic Regression
  - Linear Regression
  - Ridge / Lasso Regression
  - Support Vector Machines

- **Multi-Core CPU**
  - Decision Trees
  - Random Forests
  - More coming ....

- **Multi-Core, Multi-Socket & GPU Acceleration**

- **CPU-GPU Memory Management**

- **Distributed Training: Multi-CPU & Multi-GPU**

APIs for Popular ML Frameworks:
- Python
- Apache Spark
- ml

New
Most Popular Data Science Methods

Performance Matters for
- Online Re-training of Models
- Model Selection & Hyper-Parameter Tuning
- Fast Adaptability to Changes

Scalability to Large Datasets useful for
- Recommendation engines, Advertising, Credit Fraud
- Space Exploration, Weather

Source: Kaggle Data Science Survey 2017
Logistic Regression: 46x Faster
Snap ML (GPU) vs TensorFlow (CPU)

- 90 x86 Servers (CPU-only)
- 4 Power9 Servers With GPUs

Google TensorFlow: 1.1 Hours
Snap ML: 1.53 Minutes

46x Faster

Ridge Regression: 3x Faster
Power-NVLink-GPUs vs x86-PCIe-GPUs

- x86 Server with 1 GPU: RunTime (s)
  - Runs 1: 106.71
  - Runs 2: 86.87
  - Runs 3: 94.94
  - Runs 4: 86.22
  - Runs 5: 102.10

- Power Server with 1 GPU: RunTime (s)
  - Runs 1: 30.75
  - Runs 2: 30.64
  - Runs 3: 30.66
  - Runs 4: 29.83
  - Runs 5: 30.63

Predict volatility of stock price, 10-K textual financial reports, 482,610 examples x 4.27M features

Advertising Click-through rate prediction
Criteo dataset: 4 billion examples, 1 million features
Snap ML is 2-4x Faster than scikit-learn - CPU-only

**Decision Trees**

- **3.8x faster**
- Snap ML on Power vs sklearn on x86

**Random Forests**

- **4.2x faster**
- Snap ML on Power vs sklearn on x86
### Summary of Performance Results for Snap ML

<table>
<thead>
<tr>
<th>Comparison</th>
<th>Model Comparison</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>GPU vs CPU</td>
<td>Snap ML vs scikit-learn: Linear Models</td>
<td>20-40x</td>
</tr>
<tr>
<td>Power vs x86 with GPUs</td>
<td>Snap ML: Linear Models</td>
<td>3x</td>
</tr>
<tr>
<td>CPU Only: Power vs x86</td>
<td>Snap ML vs scikit-learn: Tree Models</td>
<td>2-4x</td>
</tr>
</tbody>
</table>
Large Model Support (LMS) Enables Higher Accuracy via Larger Models

Store Large Models & Dataset in System Memory
Transfer One Layer at a Time to GPU

TensorFlow with LMS

4.7x Faster

Images / sec

Power + 4 GPUs

x86 + 4 GPUs

500 Iterations of Enlarged GoogleNet model on Enlarged ImageNet Dataset (2240x2240), mini-batch size = 15
Both servers with 4 NVIDIA V100 GPUs

IBM AC922 Power9 Server

CPU-GPU NVLink 5x Faster than Intel x86 PCI-Gen3
Deep learning training takes days to weeks

DDL in WML CE extends TensorFlow & enables scaling to 100s of servers

Automatically distribute and train on large datasets to 100s of GPUs

Near Ideal (95%) Scaling to 256 GPUs

- Ideal Scaling
- DDL Actual Scaling

Runs for Days

Runs within Hours

ResNet-50, ImageNet-1K
Caffe with PowerAI DDL, Running on S822LC Power System
Auto Hyper-Parameter Optimization (HPO) in WML Accelerator

Manual Process

Change Hyperparameters

Train Model

Manually Choose Parameters

WML Accelerator Auto-Hyperparameter Optimizer (Auto-HPO)

Lots of Hyperparameters:
Learning rate, Decay rate, Batch size, Optimizers (Gradient Descent, Momentum, ..)

Auto-HPO has 3 search approaches
Random, Tree-based Parzen Estimator (TPE), Bayesian

IBM Spectrum Conductor running Spark
Elastic Distributed Training (EDT)

Dynamically reallocates GPUs within milliseconds

Increases job throughput and server/GPU utilization

Works with Spark & AI jobs

Works with Hybrid x86 & Power cluster

2 Servers with 4 GPUs each: total 8 GPUs

Available policies: Fair share, Preemption, Priority

Graph showing:

- T0: Job 1 starts, uses all available GPUs
- T1: Job 2 starts, Job 1 gives up 4 GPUs
- T2: Job 2 gets higher priority, Job 1 gives up GPUs
- T3: Job 1 finishes, Job 2 uses all GPUs

Graph legend:

- Job 1
- Job 2

Graph time frame: 8:09 to 8:21

Graph X-axis: Time

Graph Y-axis: GPU Slots

Graph bars show allocation of GPUs over time.
PowerAI Vision: “Point-and-Click” AI for Images & Video

Label Image or Video Data

Auto-Train AI Model

Package & Deploy AI Model

- **Create Dataset**: Start by adding images and video files to a dataset.
- **Prepare Data**: Label objects or assign categories to images or videos, then use auto-labeling to complete the entire dataset.
- **Train Model**: Select a few custom options to create your model.
- **Deploy Model**: Deploy the trained model and receive an API link for an inference device.
Core use cases

Image Classification

Object Detection

Image Segmentation
Automatic Labeling using PowerAI Vision

- Manually Label Some Image / Video Frames
- Train DL Model
- Auto-Label Full Dataset with Trained DL Model
- Manually Correct Labels on Some Data

Repeat Till Labels Achieve Desired Accuracy
Retail Analytics
Track how customers navigate store, identify fraudulent actions, detect low inventory

Worker Safety Compliance
Zone monitoring, heat maps, detection of loitering, ensure worker safety compliance

Remote Inspection & Asset Management
Identify faulty or worn-out equipment in remote & hard to reach locations
Quality Inspection Use Cases

- Semiconductor Manufacturing
- Electronics Manufacturing
- Travel & Transportation
- Oil & Gas
- Utilities Inspection
- Robotic Manufacturing
- Steel Manufacturing
- Aerospace & Defense
AI Developer Box & AI Starter Kit

**Power AI DevBox**

Free 30-Day Licenses for PowerAI Vision & WML Accelerator (free to Academia)

Power9 + GPU Desktop PC: $3,449
Order from: [https://raptorcs.com/POWERAI/](https://raptorcs.com/POWERAI/)

**AI Starter Kit**

WML Accelerator Pre-installed (formerly called PowerAI Enterprise)

2 AC922 Accelerated Servers + 1 P9 Linux Storage Server
500+ Clients using AI on Power Systems

Power AI Clients at THINK 2019
IBM AI Meetups Community Grew 10x in 9 Months

From 6K to 85K Members in 9 Months

https://www.meetup.com/topics/powerai/
Summary


Snap ML: Fast Machine Learning Framework

Power AI DevBox & AI Starter Kit
Get Started Today with Machine & Deep Learning

IBM PowerAI

Build a Data Science Team
Your Developers Can Learn
http://cognitiveclass.ai

Identify a Low Hanging Use Case

Figure Out Data Strategy

Consider Pre-Built AI APIs

Hire Consulting Services

Get Started Today at
www.ibm.biz/poweraideveloper
Additional Details
Why are Linear & Tree Models Useful?

**Fast Training**
GLMs can scale to datasets with billions of examples and/or features & still train in minutes to hours

**Need Less Data**
Machine learning models can train to “good-enough” accuracy with much less data than deep learning requires

**Interpretability**
Linear models explicitly assign an importance to each input feature
Tree models explicitly illustrate the path to a decision.

**Less Tuning**
Linear models involve much fewer parameters than more complex models (GBMs, Neural Nets)