THE RACE FOR FASTER MACHINE LEARNING - INTEL ARTIFICIAL INTELLIGENCE TECHNICAL UPDATE

Paweł Gepner

Robert Adamski

Pascal Lassaigne

Legal Notices and Disclaimers

Legal Disclaimer & Optimization Notice

Optimization Notice

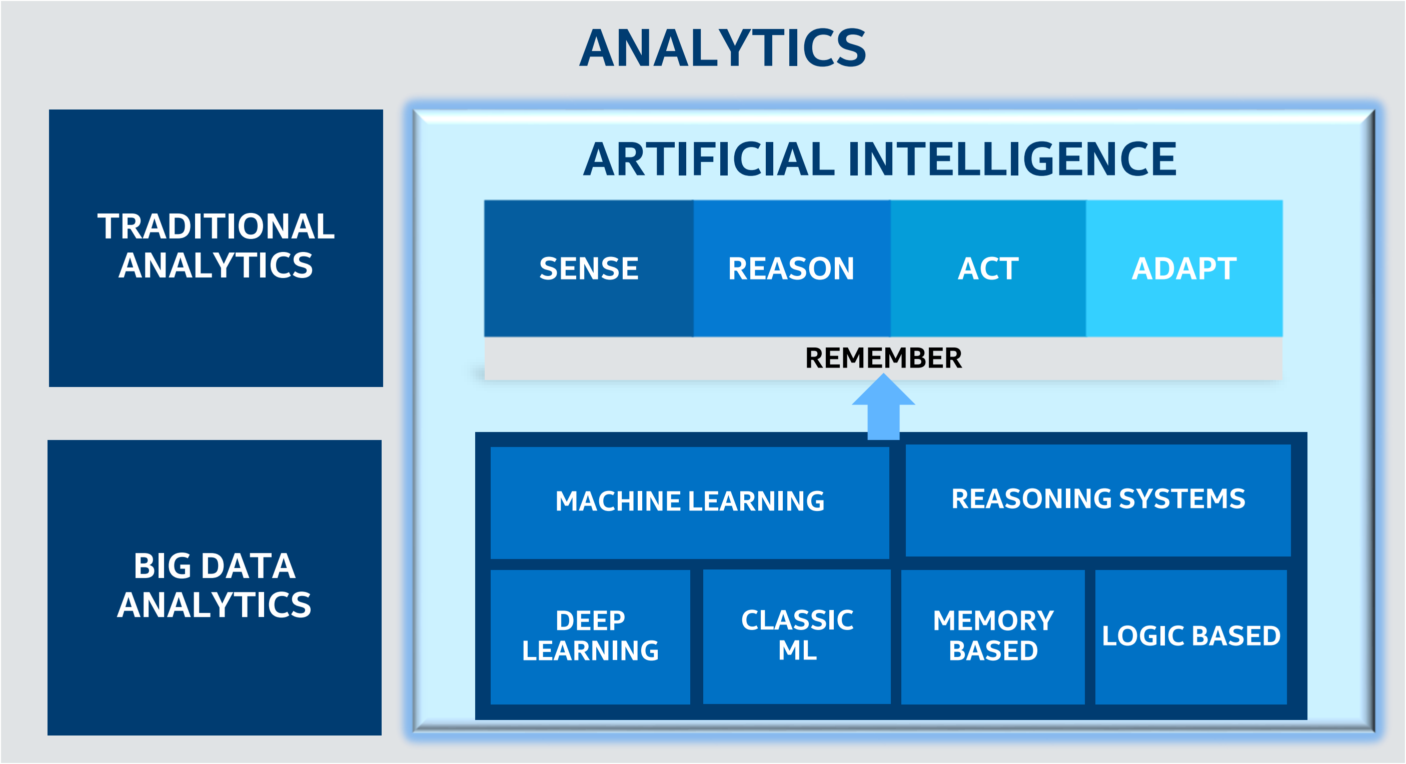
Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice.

Notice revision #20110804

Agenda

- What is AI?
- Intel AI portfolio
- MKL-DNN
- Reinforcement Learning on IA
  - Atari Games experiment on Xeon/Xeon Phi
  - Environment - Open AI Gym, A3C, PLGRID
  - First results

Focus on AI – Part of Analytics



What is Machine Learning?

CLASSIC ML

Using functions or algorithms
to extract insights from new data

Training Data*

Functions
$(f_1, f_2, \dots, f_K)$

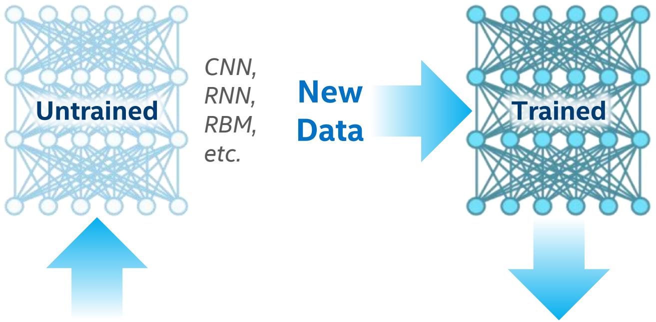
Random Forest
Decision Trees
Graph Analytics
Regression
More…

Inference

New Data*

DEEP LEARNING

Using massive data sets to train deep (neural)
graphs that can extract insights from new data



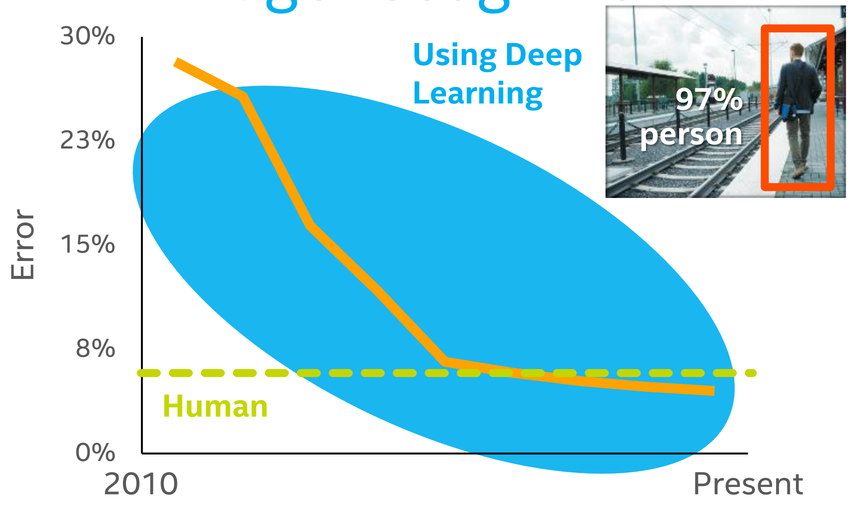
Step 1: Training
Hours to Days
in Cloud

Step 2: Inference
Real-Time
at Edge/Cloud

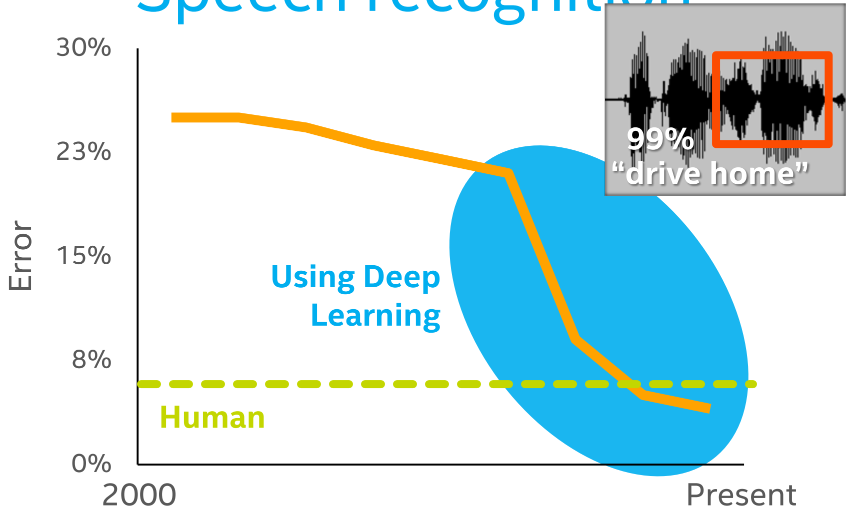
*Not all classical machine learning algorithms require separate training and new data sets

Deep Learning Breakthroughs

Image recognition



Speech recognition

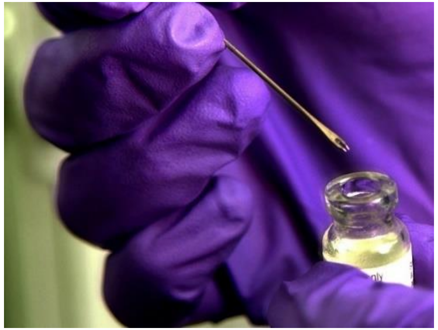


enabling improved and all new applications !

AI Will Usher in a Better World

on the scale of the agricultural, industrial and digital revolutions

ACCELERATE
Large-Scale Solutions

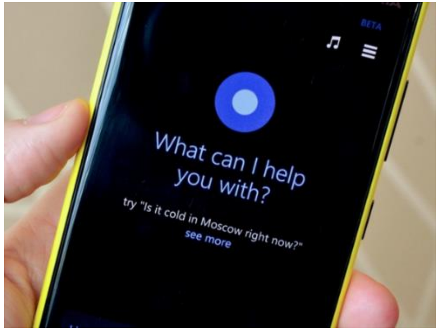


Cure Diseases
Eliminate Fraud
Unlock Dark Data

UNLEASH
Scientific Discovery

Explore Deep Sea/Space
Solve Particle Physics
Decode the Brain

Augment
Human Capability



Personalized Guidance
Enhance Decisions
Prevent Crime

Automate
Risky/Tedious Tasks

Automate Driving
Search & Rescue
No More Chores

Source: Intel

Intel Strategy: Intel® Nervana™ Portfolio

Machine Learning Framework Optimizations

- Spark MLlib
- Intel ® Distribution for Python

Deep Learning Framework Optimizations

neon TensorFlow Caffe theano

Low Level Software Primitives

- Nervana Graph
- Intel® DAAL
- Intel® MKL
- Intel® MKL-DNN

Intel® Silicon

intel XEON inside | intel XEON PHI inside | intel XEON inside + ALTERA Arria 10 FPGA • SoC | intel XEON inside + LAKE CREST

+ Storage, Network

Xeon Phi: Scalable, Larger Memory Footprint & Great Performance

UP TO **400GB** DIRECT MEMORY ACCESS

vs 16GB with a GPU[1]

NEAR LINEAR SCALING

31X REDUCTION IN TIME TO TRAIN

when scaling to 32 nodes[2]

AVAILABLE 2017

Knights Mill

Next-Gen Intel Xeon Phi

4X Deep learning Performance

vs current gen[3]

Intel Xeon Phi Results

Nov'16 Top500 List

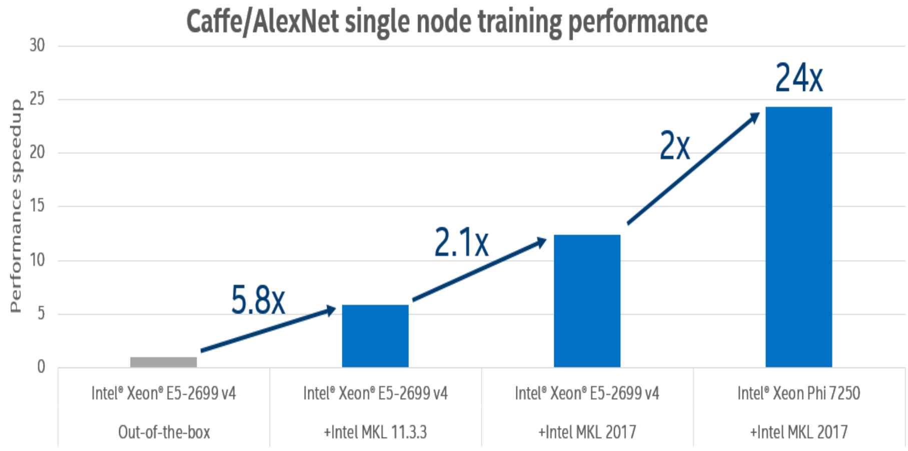
+45 PFLOPS

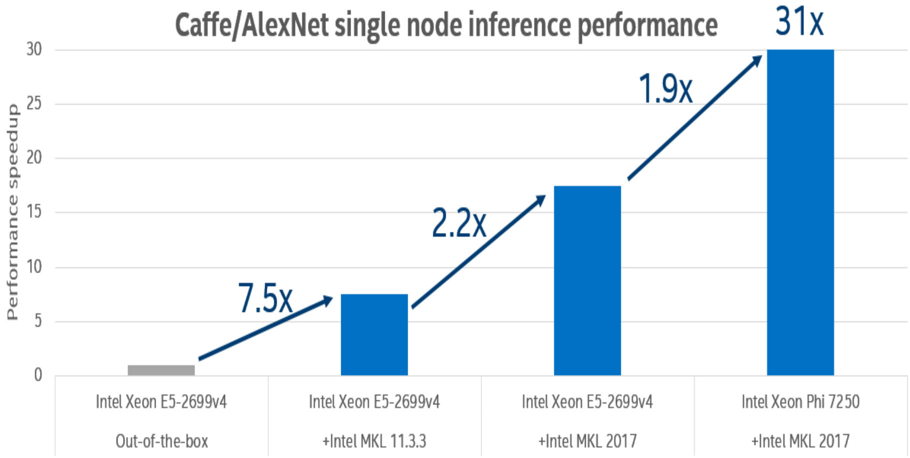
80% NEW System Accelerator Flops

Caffe + Intel® MKL 2017

Intel Caffe https://github.com/intelcaffe/caffe

- The fork is aimed at improving Caffe performance on Intel® Xeon® CPUs.





Configurations:

- 2 socket system with Intel® Xeon® Processor E5-2699 v4 (22 Cores, 2.2 GHz,), 128 GB memory, Red Hat* Enterprise Linux 6.7, BVLC Caffe, Intel Optimized Caffe framework, Intel® MKL 11.3.3, Intel® MKL 2017
- Intel® Xeon Phi™ Processor 7250 (68 Cores, 1.4 GHz, 16GB MCDRAM), 128 GB memory, Red Hat* Enterprise Linux 6.7, Intel® Optimized Caffe framework, Intel® MKL 2017

All numbers measured without taking data manipulation into account.

Reinforcement Learning on IA

Experiments on Xeon/Xeon Phi
Team: deepsense.io, Intel
Platform: PLGRID Prometheus

Reinforcement Learning

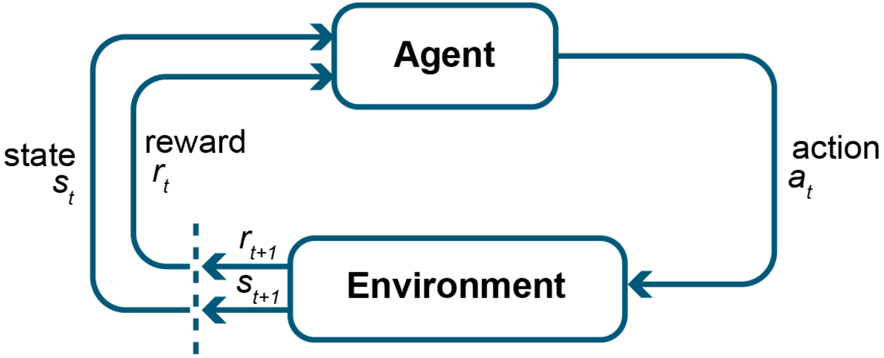
Agent learns from interaction with an **Environment**.

Very general problem

Examples:

- Robot learning to move items in real environment - a reward is given when item is moved from A to B.
- Robot learning the same task in a simulator.
- An agent playing a board game like Chess - reward for winning a game.
- An agent playing a video game – rewards like in the actual game.

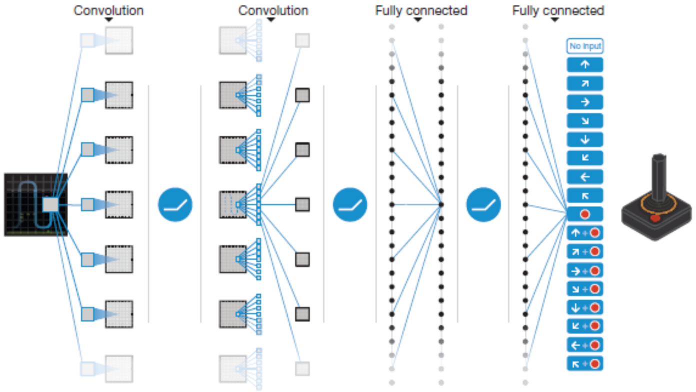
an



Reinforcement Learning

The task: Atari games on CPU

- Train agents for playing **Atari games** from pixel information
- The agent should maximize their score in the game
- Async A2C as an RL algorithm
- 4-layer ConvNet for processing input images



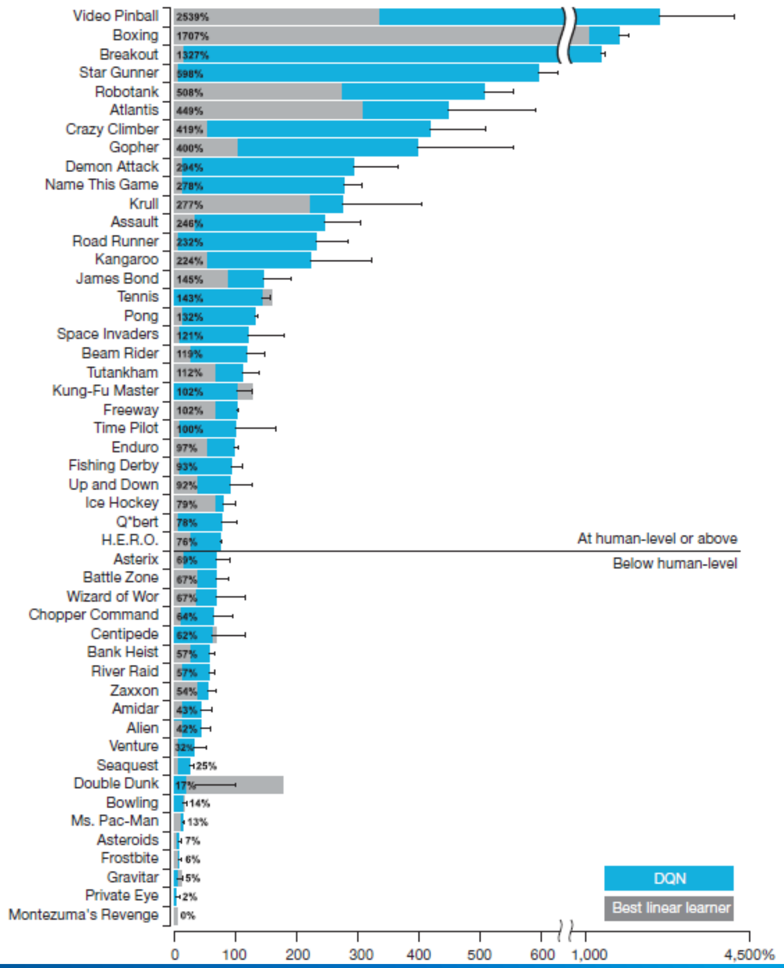
Reinforcement Learning

Benchmark games: Atari 2600 classics

Environment: Open AI Gym

Input: game screens, 210 x 160 pixels, 3 color channels

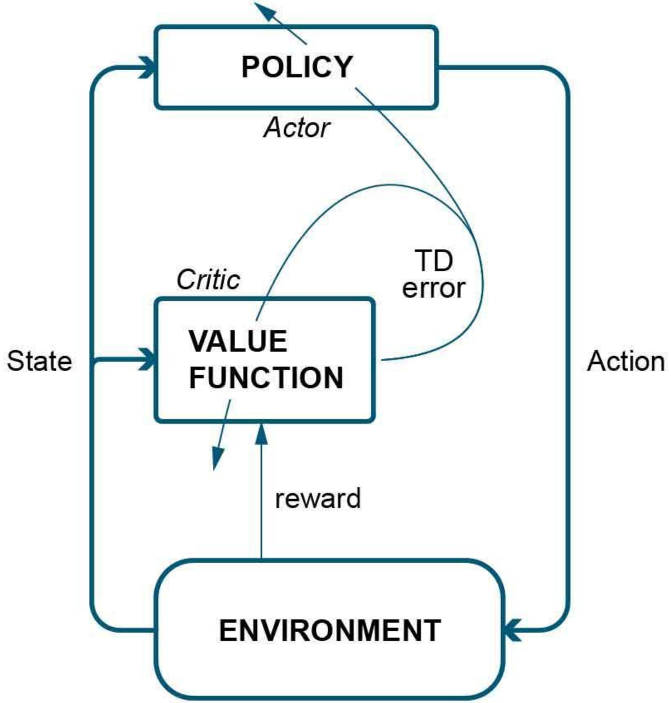
Output: one of 18 controller actions



Reinforcement Learning

Features of the Batch Asynchronous Advantage Actor-Critic Algorithm (BA3C):

- Hundreds of game simulators are running in parallel on a single machine
- The simulators use a shared model to evaluate actions
- The model can batch predictions from multiple simulators to increase efficiency
- The games played by the simulators are also batched and used for training the model

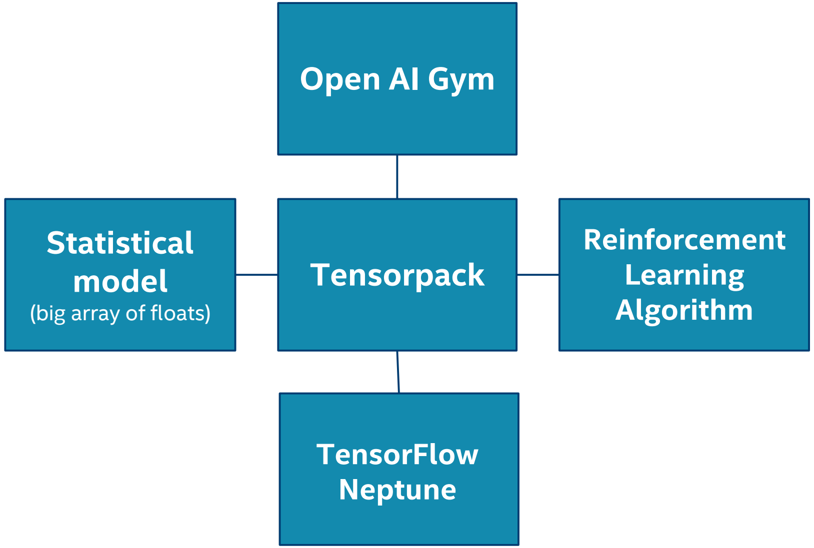


Source: Ben Lau, Using Keras...

Reinforcement Learning

Software and hardware stack:

- **Tensorpack** Framework implementing selected learning algorithms in TensorFlow (Yuxin Wu). Provides an efficient implementation of Async A2C algorithm
- **TensorFlow** General framework for machine learning
- **OpenAI Gym** Framework providing standard environments for reinforcement learning
- **Neptune** Tool for monitoring and managing experiments (deepsense.io)
- **Prometheus** and Xeon Phi server (KNL)



DNN functions from Math Kernel Library

- We discovered that some TF convolutions were significantly slowing down the training.
- We used **MKL** (version 2017.0.098) for better performance
- We forked TensorFlow and provided alternative implementation of convolution using MKL primitives

MKL convolution - backpropagation

Input shape	Kernel shape	Default TF time (Xeon) [ms]	MKL TF time (Xeon) [ms]	Default TF time (Xeon Phi) [ms]	MKL TF time (Xeon Phi) [ms]
128x84x84x16,	5x5x16x32	368.18	**29.63**	1,236.98	**8.97**
128x40x40x32,	5x5x32x32	114.72	**19.55**	343.73	**6.33**
128x18x18x32	5x5x32x64	28.82	6.07	36.74	2.52
128x7x7x64	3x3x64x64	5.57	3.18	7.38	2.31

Results

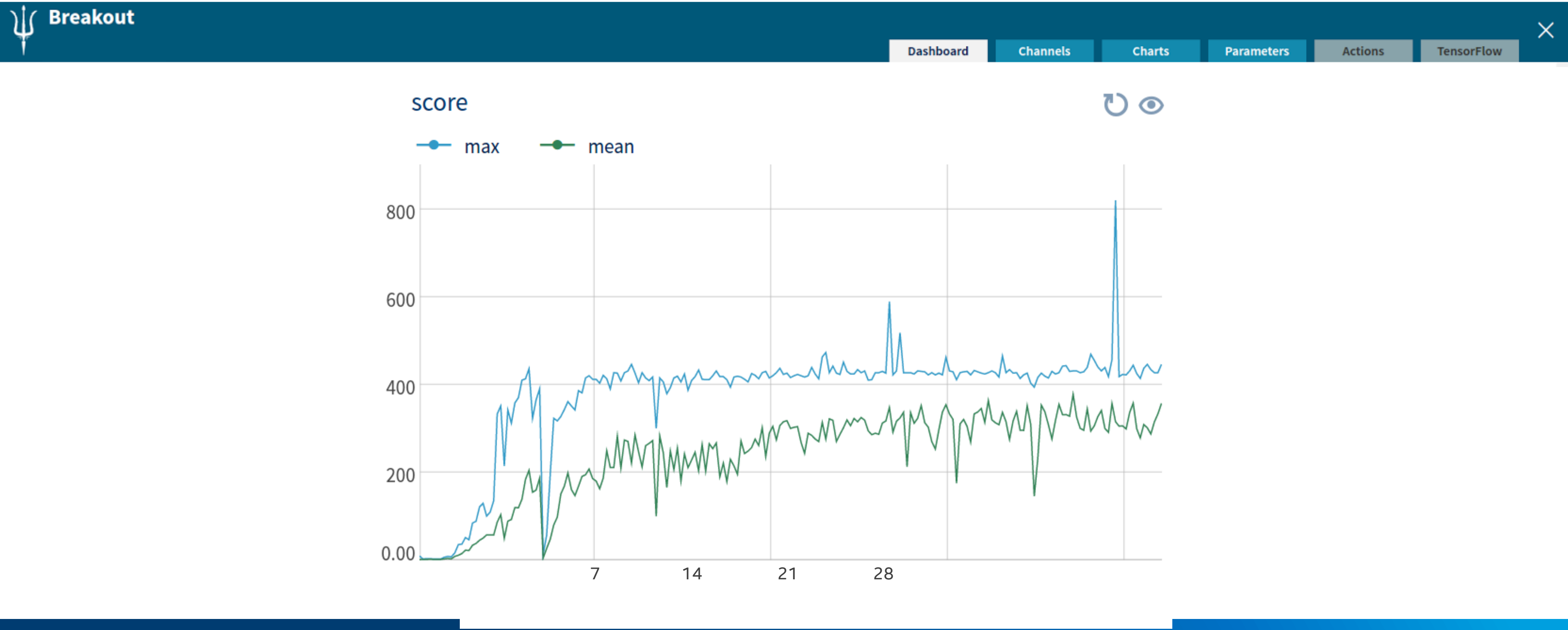
Reinforcement Learning

Top performance in Breakout

Top performance in River Raid

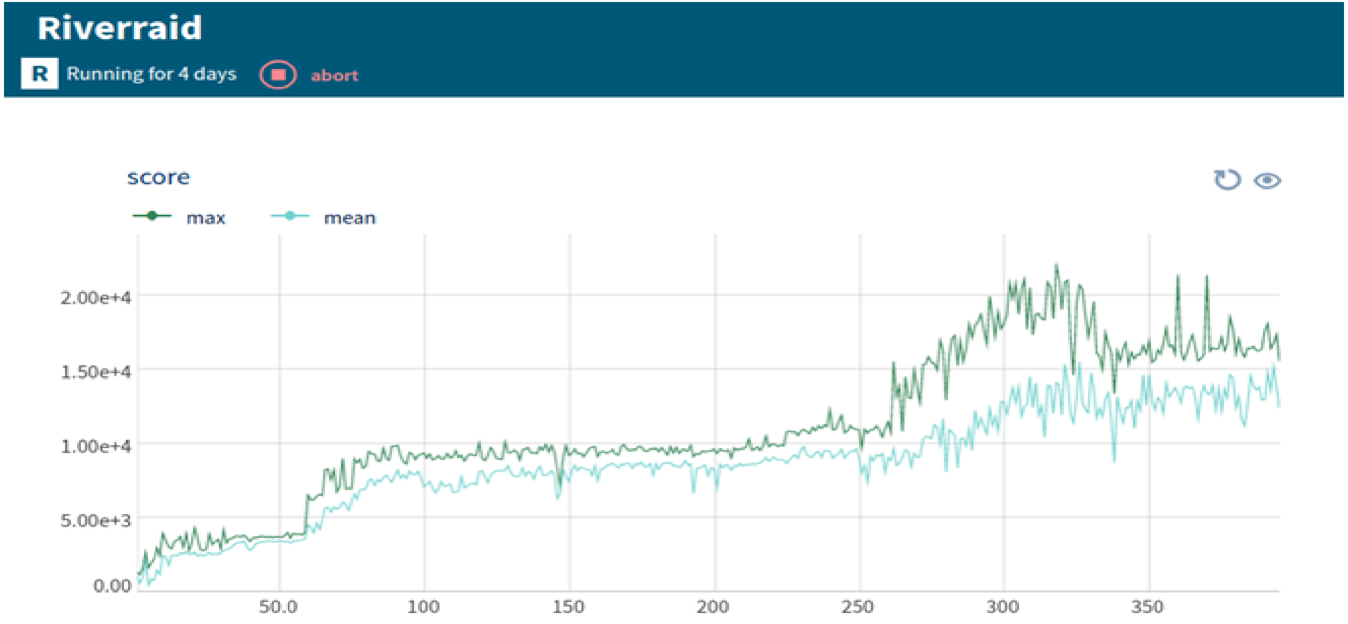
Reinforcement Learning

Monitoring the learning process using the Neptune tool (Breakout on Xeon)

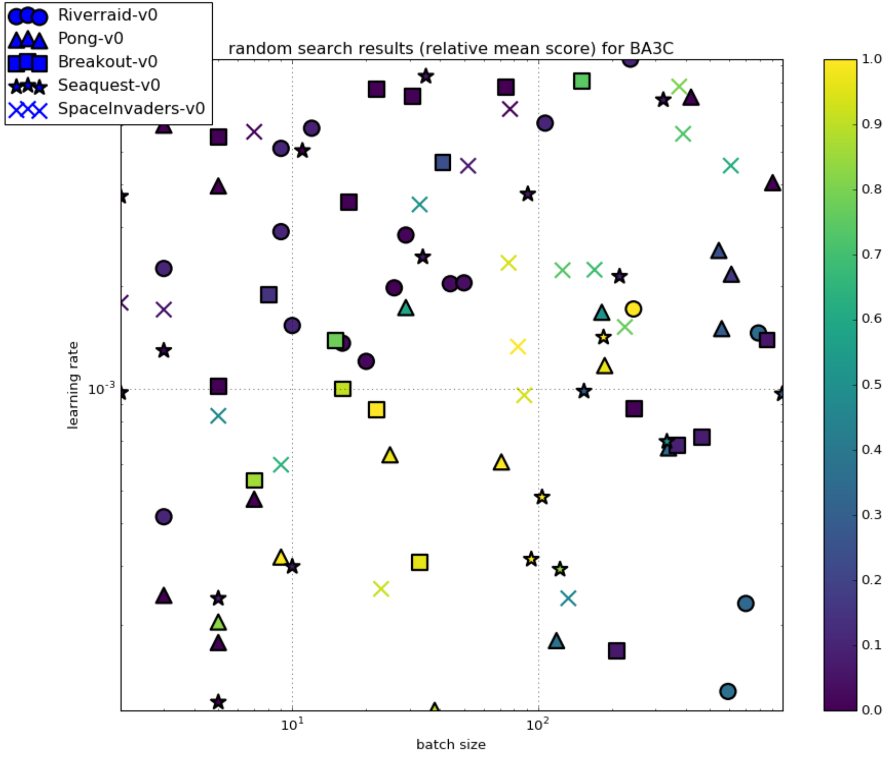


Reinforcement Learning

Monitoring the learning process using the Neptune tool (RiverRaid on Xeon)



Experiments on PLGRID Prometheus – Results



Reinforcement Learning

Summary

- RL agents trained on **CPU** in just a few hours
- 10x performance gain with MKL DNN implementation, 2.5x for convolutions only
- The performance vary drastically depending on the batch size and learning rate

Challenges and future work:

- Multinode impementation of code on optimized TensorFlow