COMP GI13/M050
Deep Learning Lecture 1

Thore Graepel & Guest Lecturers from DeepMind

Overview

- Team and Structure of the Course
- DeepMind approach to AI
- Why Deep Learning?
- Deep Reinforcement Learning at work
  - Learning to Play Atari Games with Deep RL
  - AlphaGo - Learning to master level Go
- Guest Lectures and Lecturers (with Hado & Joseph)
- Assignment 1: TensorFlow and MNIST
- Brief intro to TensorFlow

The DeepMind/UCL Team

Koray Kavukcuoglu
(Co-Lead DL)

Hado van
Hasselt (Co-Lead
RL)

Joseph Modayil
(Co-Lead RL)

Diana Borsa
(TA)

Marie Mulville
(PgM)

Format, Coursework and Exam Papers

- Format: Two streams, both streams mandatory
  - Tuesdays: Deep Learning taught by a selection of fantastic guest lecturers from DeepMind
  - Thursdays: Reinforcement Learning taught by Hado & Joseph (also DeepMind)
  - Check timetable at https://timetable.ucl.ac.uk/
- Programming Coursework (50%)
  - Programming assignments, mostly using TensorFlow
  - Assignment 1: “TensorFlow and MNIST”, due 31st January (10%)
  - Assignment 2: “Sequence Generation”, due 28th February (20%)
  - Assignment 3: “Deep RL”, due 28th March (20%)
- Exam Paper (50%)
  - Topics from Deep Learning and Reinforcement Learning
  - Four questions for each topic
  - Need to complete five questions
- Course requires a lot of work - rewarded by learning about cutting edge AI/ML

DeepMind founded 2010 (joined Google 2014)

Mission: “Solve Intelligence”

An Apollo Programme for AI (150+ scientists)

A new approach to organizing science

General-Purpose Learning Algorithms

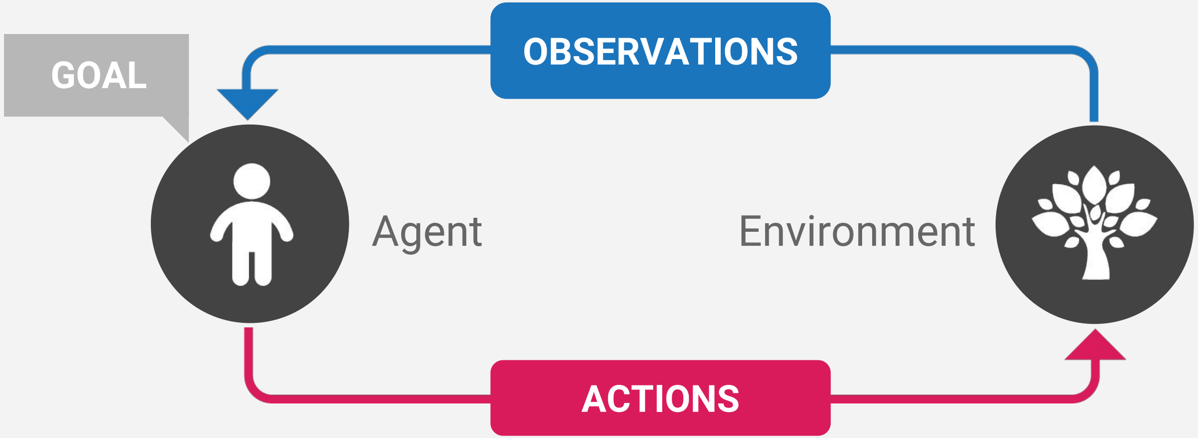
Learn automatically from raw inputs - not pre-programmed

General - same system can operate across a wide range of tasks

Artificial 'General' Intelligence (AGI) – flexible, adaptive, inventive

'Narrow' AI – hand-crafted, special-cased, brittle

Reinforcement Learning



- General Purpose Framework for AI
- Agent interacts with the environment
- Select actions to maximise long-term reward
- Encompasses supervised and unsupervised learning as special cases

What is intelligence?

Intelligence measures an agent's ability to achieve goals in a wide range of environments

$$\underbrace{\Upsilon(\pi)}_{\text{Measure of Intelligence}} := \underbrace{\sum_{\mu \in E}}_{\text{Sum over environments}} \underbrace{2^{-K(\mu)}}_{\text{Complexity penalty}} \underbrace{V_{\mu}^{\pi}}_{\text{Value achieved}} .$$

Universal Intelligence: A Definition of Machine Intelligence, Legg & Hutter 2007

Grounded Cognition

A true thinking machine has to be grounded in a rich sensorimotor reality

Games are the perfect platform for developing and testing AI algorithms

Unlimited training data, no testing bias, parallel testing, measurable progress

‘End-to-end’ learning agents: from pixels to actions

Thanks to Koray for DL slides

Why Deep Learning?

- Enables End-To-End Training
  - Optimise for the end loss
  - Don't engineer your inputs
  - Learn good representations
- Versatile: Can be applied to images, text, audio, video
- Modular design of systems (modular backprop)
- Represent weak prior knowledge (e.g., convolutions)
- Now computationally feasible at scale (GPUs)

Supervised Learning

- Convolutional Networks on **MNIST**

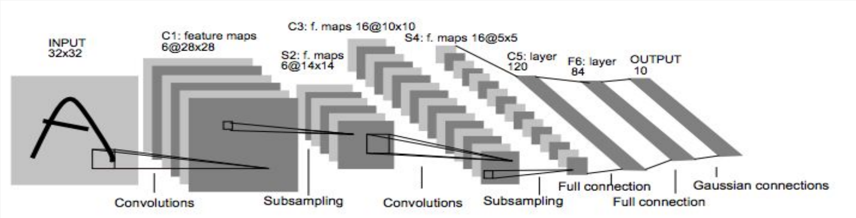
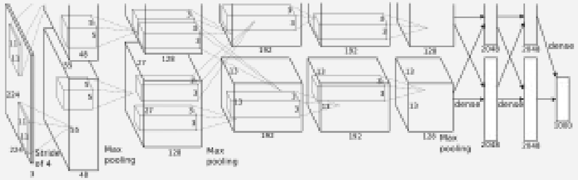
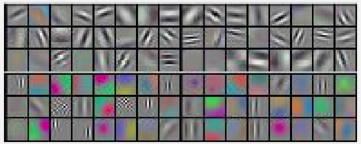


Fig. 2. Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical.

[Lecun, et. al]

- Convolutional Networks on **ImageNet**

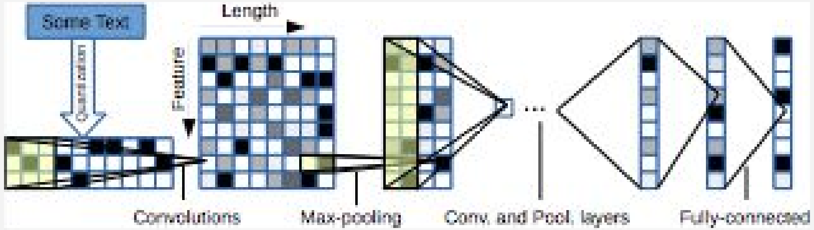




[Krizhevsky, et. al]

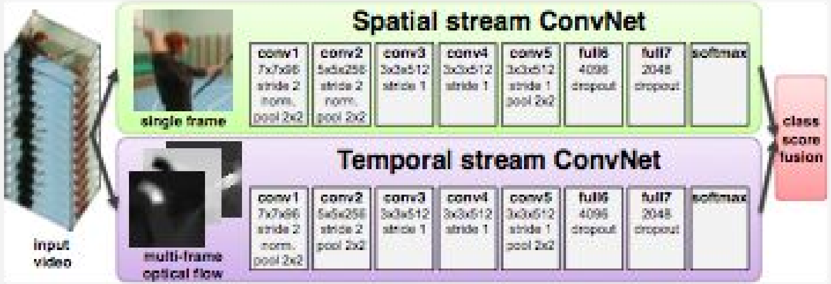
Supervised Learning

- Convolutional Networks on **Text**

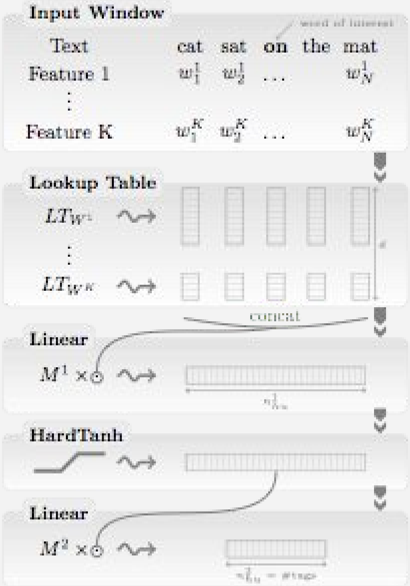


[Zhang, et. al]

- Convolutional Networks on **Video**



[Simonyan, et. al]



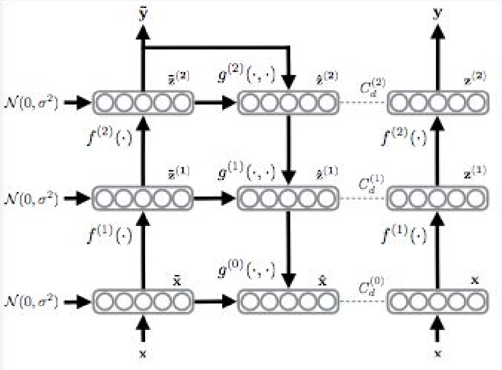
[Collobert, et. al]

Supervised Learning

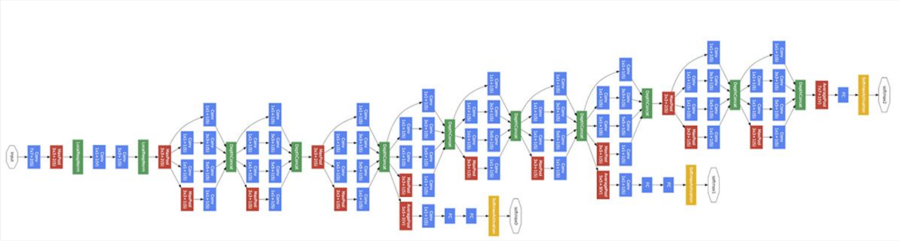
- End-to-End Training
- **Optimize** for the **end loss**
- **No engineered** inputs
- With enough data, **learn a big non-linear function**
- Learn **good representations** of data
  - Rich enough supervised labeling is enough to train transferrable representations
  - Best feature extractor
  - Karpathy, Razavian et al, Yosinski et al, Donahue et al
- Large labeled dataset + big/deep neural network + GPUs

Supervised Learning

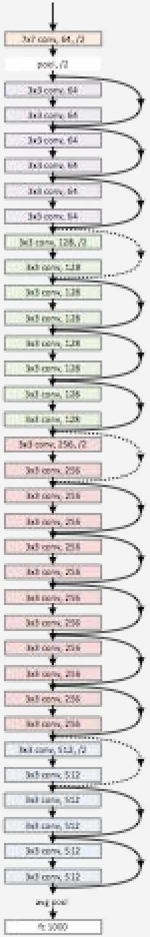
- Innovation continues
  - Inception
  - Ladder Nets
  - Residual Connections
  - ...
- Performance is continuously improving
- Architectures for easier optimization
  - Batchnorm



[Rasmus, et. al]



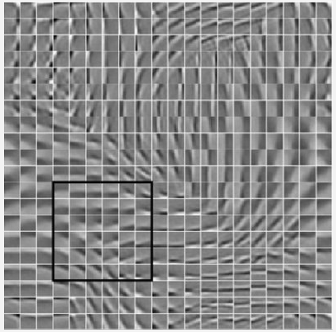
[Szegedy, et. al]

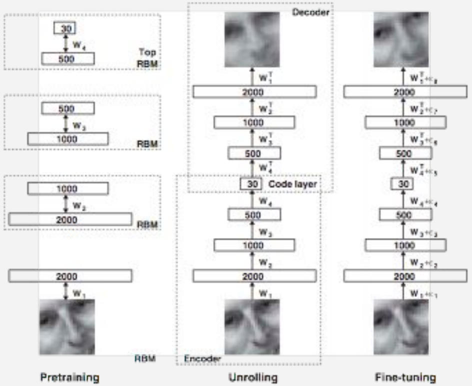


[He, et. al]

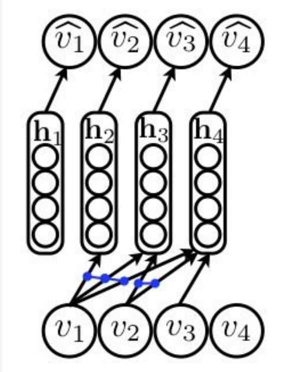
Unsupervised Learning

- Unsupervised Learning/Generative Models
  - RBM
  - Auto-encoders
  - PCA, ICA, Sparse Coding
  - VAE
  - NADE - and all variants
  - GANs
- How to evaluate/rank different algorithms?
- Quantitative approach or visual quality?
  - How can we trust if the input domain itself is not interpretable?
- How can unsupervised learning help a task?





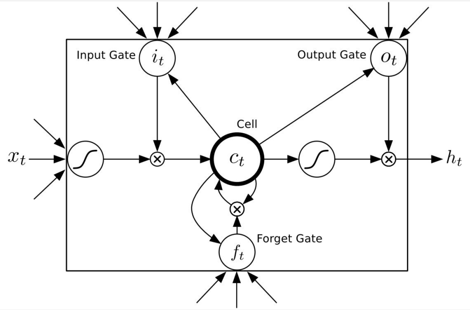
[Hinton, et. al]



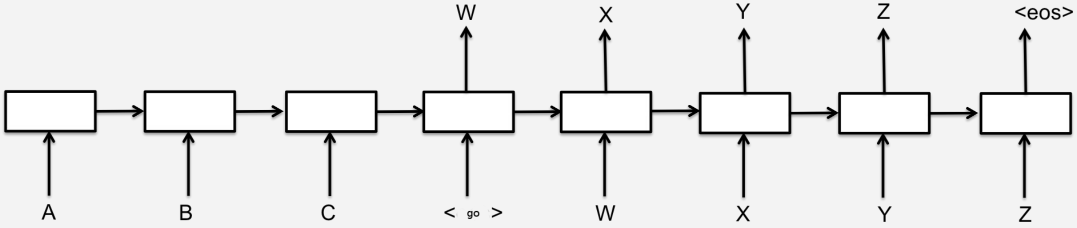
[Larochelle, Murray]

Sequence Modeling

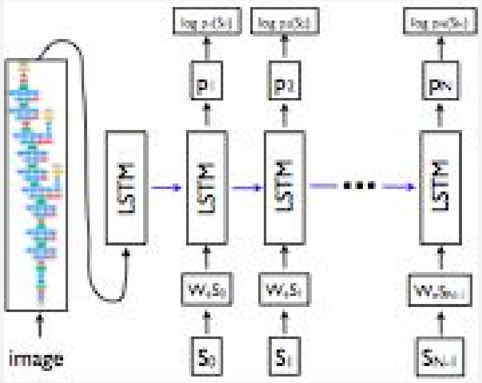
- Almost all data are sequence
  - Text
  - Video
  - Audio
  - Image [nade, pixelrnn]
  - Multi-modal (caption → image, image → caption)



[Hochreiter and Schmidhuber]



[Sutskever, et. al]



[Vinyals, et. al]

Human-level control through deep reinforcement learning

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, Demis Hassabis

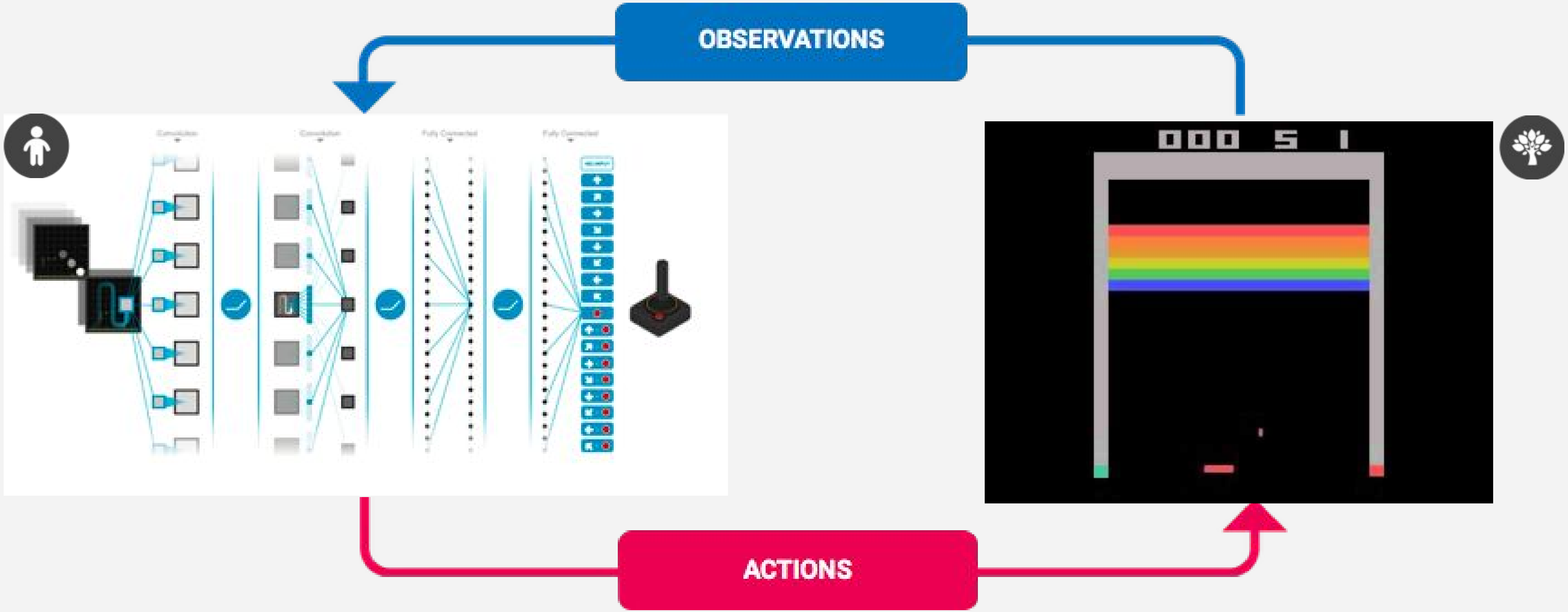
Google DeepMind

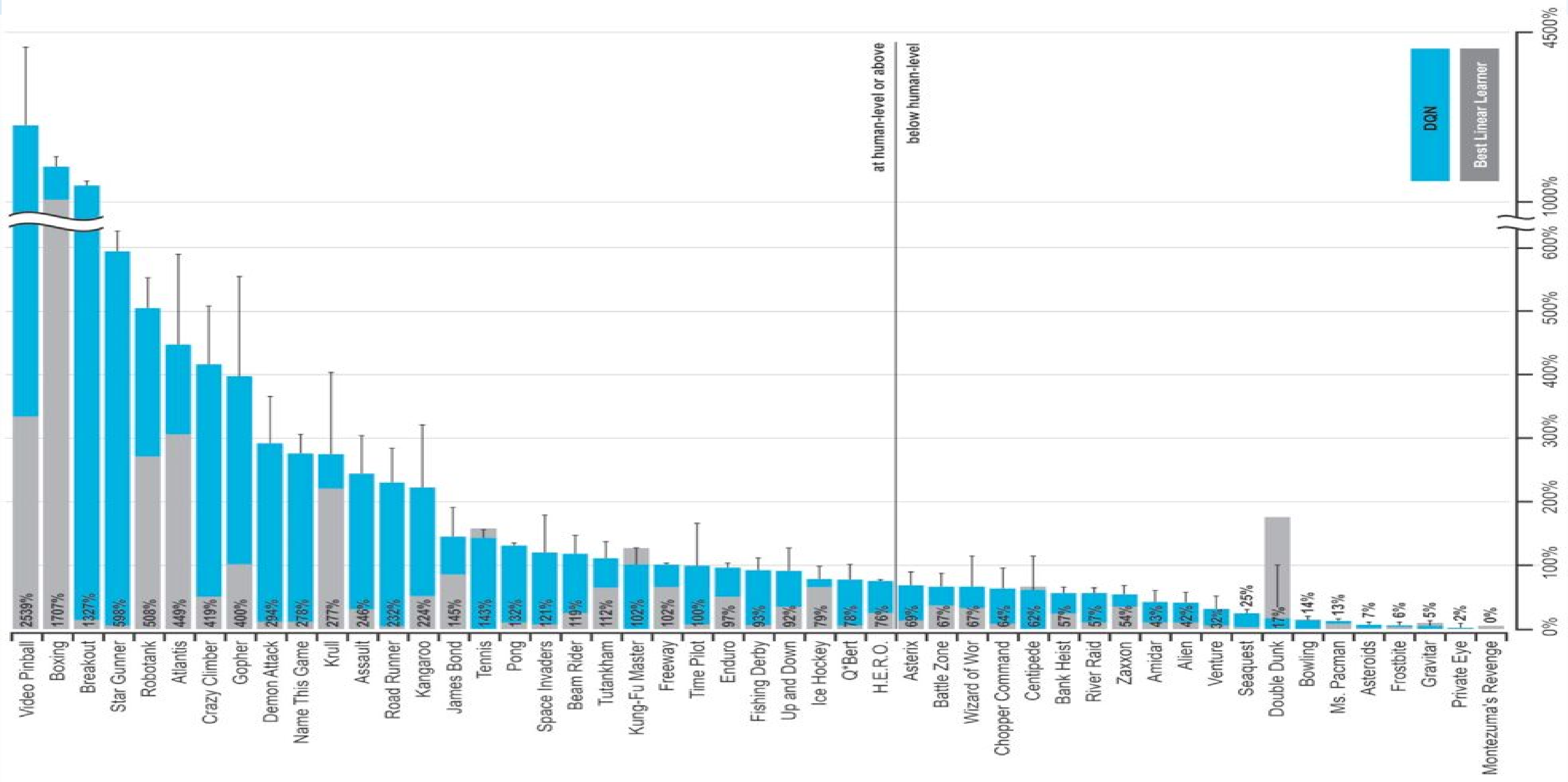
(Mnih et al. Nature 2015)

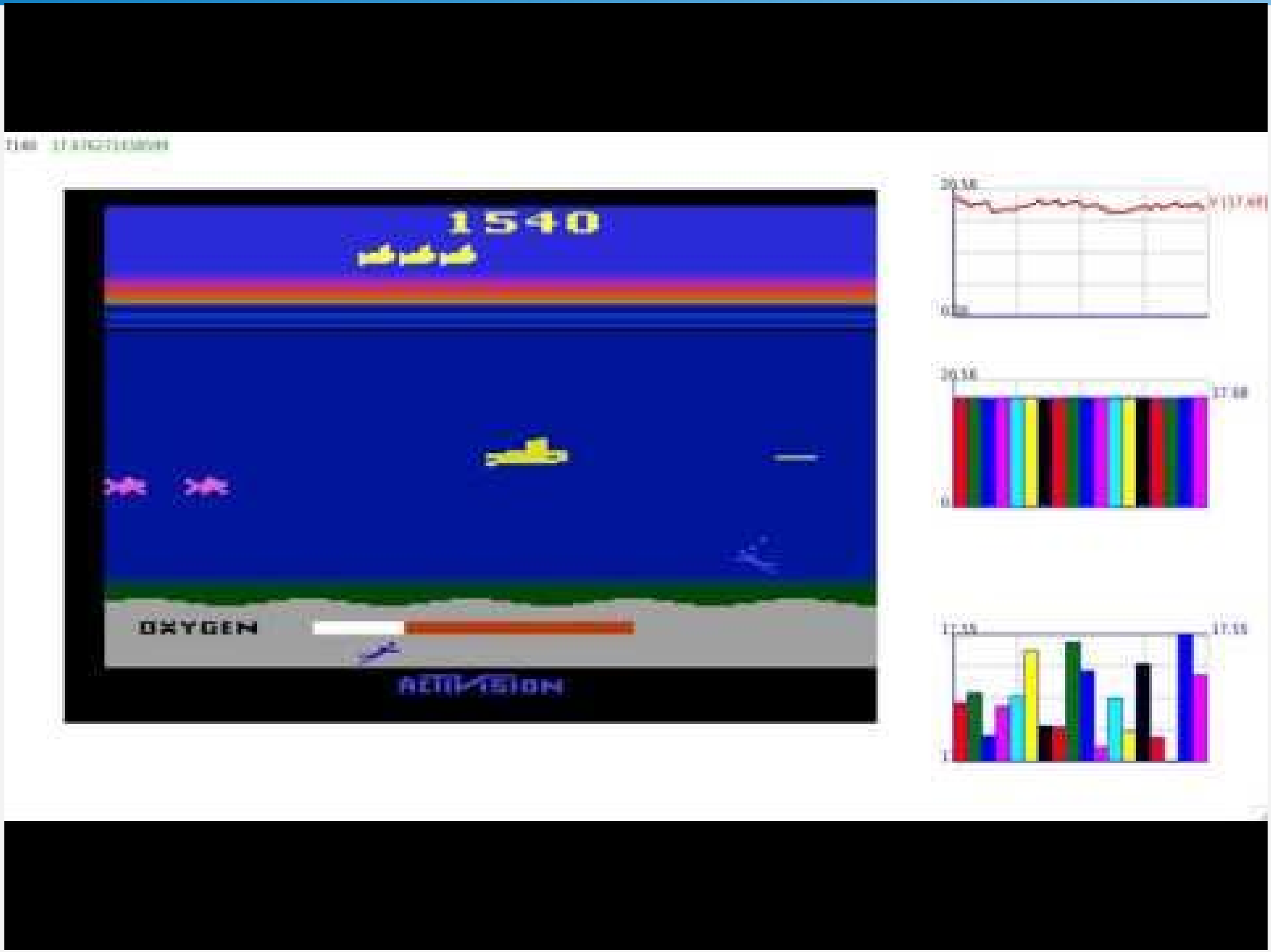
ATARI Games I ♥

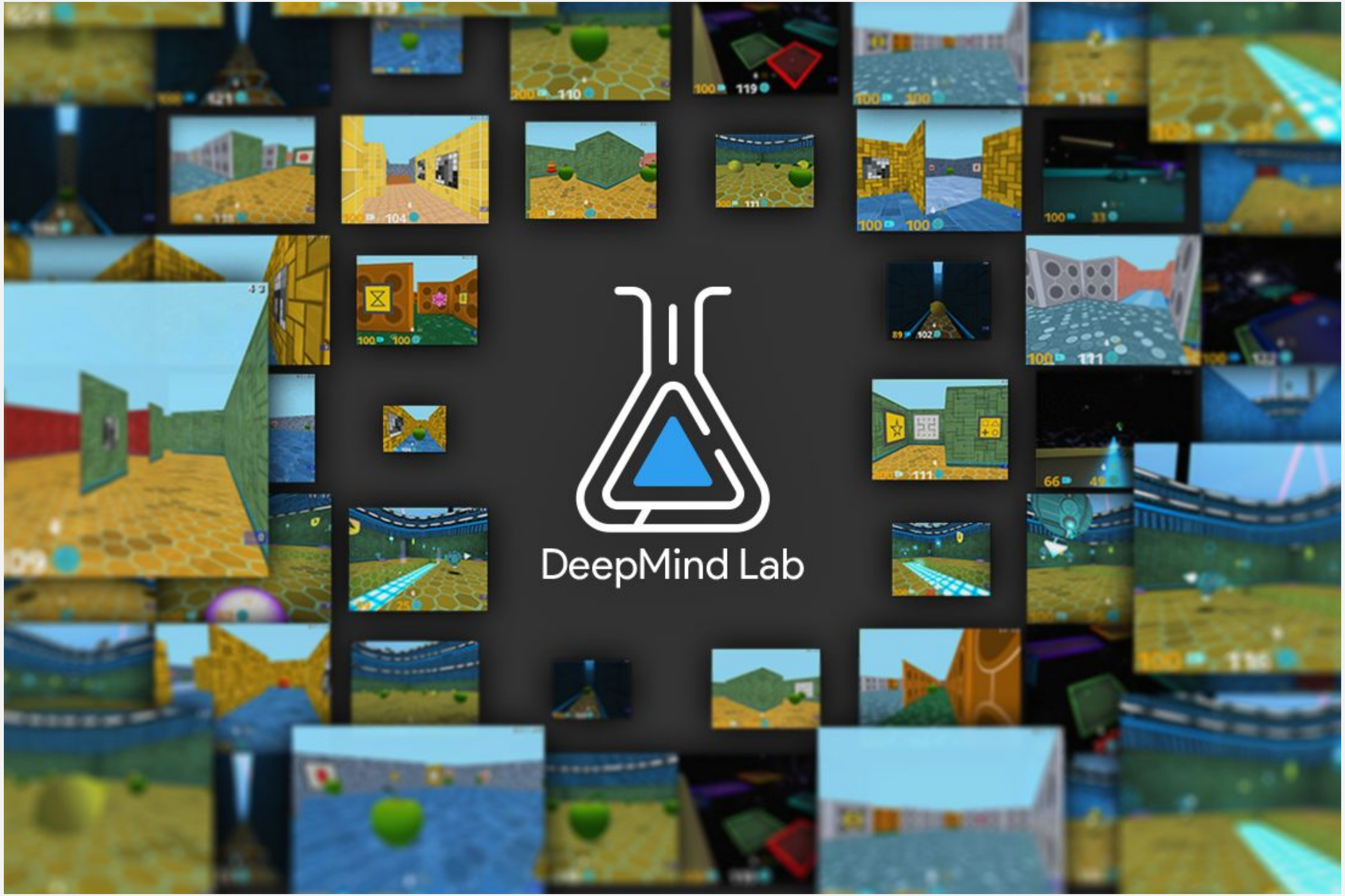
- Designed to be **challenging** and interesting for humans
- Provides a **good platform** for sequential decision making
- Widely adopted **RL benchmark** for evaluating agents (Bellemare'13)
- Many **different games** emphasize control, strategy, …
- Provide a rich visual domain

End-to-End Reinforcement Learning



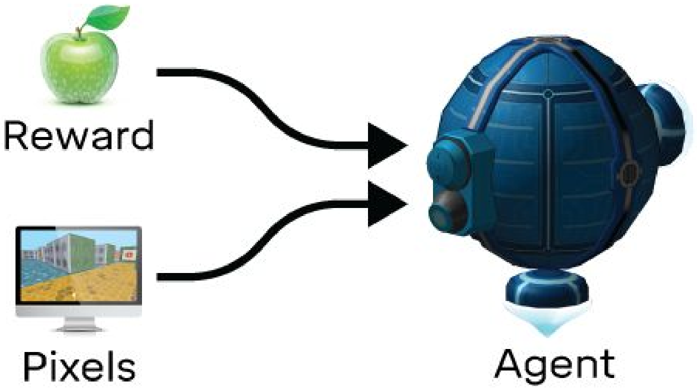

DQN
Best Linear Learner
at human-level or above
below human-level
4500%
1000%
600%
500%
400%
300%
200%
100%
0%
Video Pinball 2539%
Boxing 1707%
Breakout 1327%
Star Gunner 598%
Robotank 508%
Atlantis 449%
Crazy Climber 419%
Gopher 400%
Demon Attack 294%
Name This Game 278%
Krull 277%
Assault 246%
Road Runner 232%
Kangaroo 224%
James Bond 145%
Tennis 143%
Pong 132%
Space Invaders 121%
Beam Rider 119%
Tutankham 112%
Kung-Fu Master 102%
Freeway 102%
Time Pilot 100%
Enduro 97%
Fishing Derby 93%
Up and Down 92%
Ice Hockey 79%
Q*Bert 78%
H.E.R.O. 76%
Asterix 69%
Battle Zone 67%
Wizard of Wor 67%
Chopper Command 64%
Centipede 62%
Bank Heist 57%
River Raid 57%
Zaxxon 54%
Amidar 43%
Alien 42%
Venture 32%
Seaquest 25%
Double Dunk 17%
Bowling 14%
Ms. Pacman 13%
Asteroids 7%
Frostbite 6%
Gravitar 5%
Private Eye 2%
Montezuma's Revenge 0%

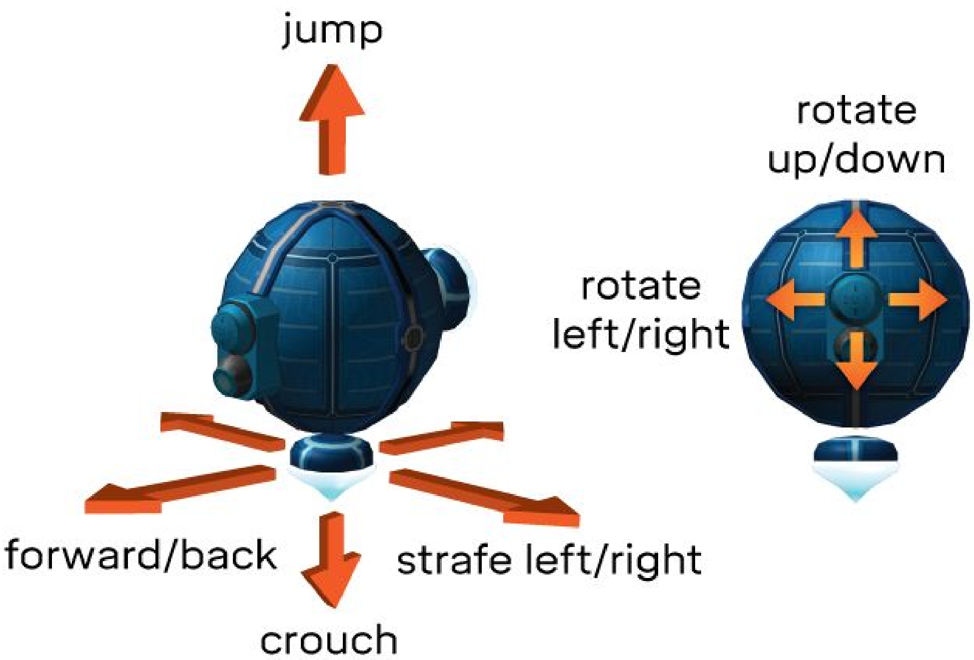

1540
OXYGEN
ACTIVISION


DeepMind Lab

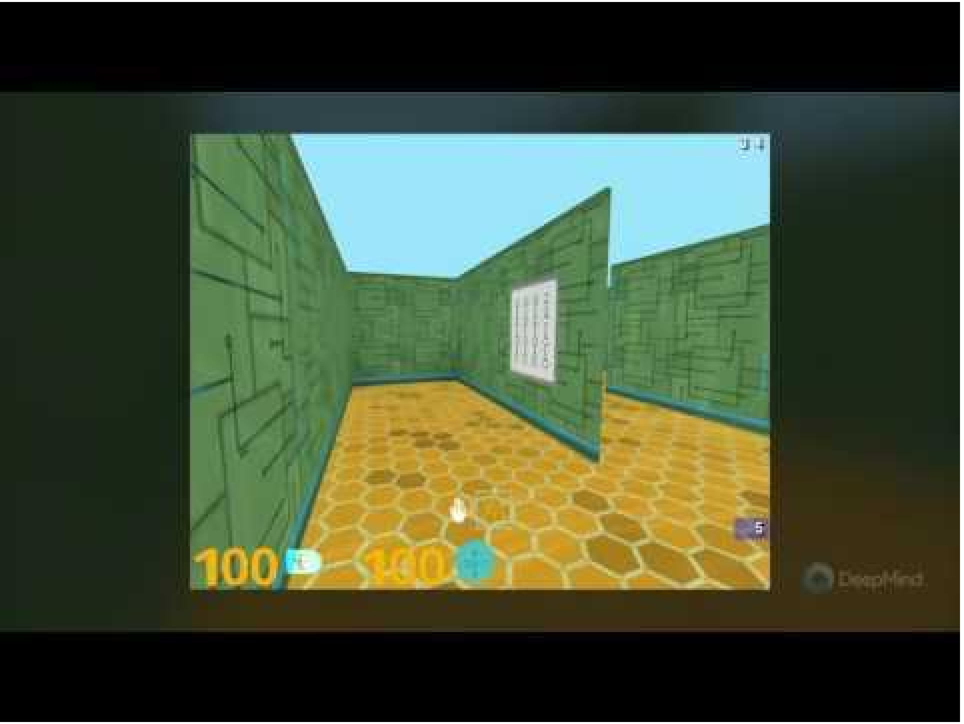
Observations

Actions





DeepMind Lab - Challenging RL Problems in 3D



Mastering the game of Go with deep neural networks and tree search

David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel & Demis Hassabis

Google DeepMind

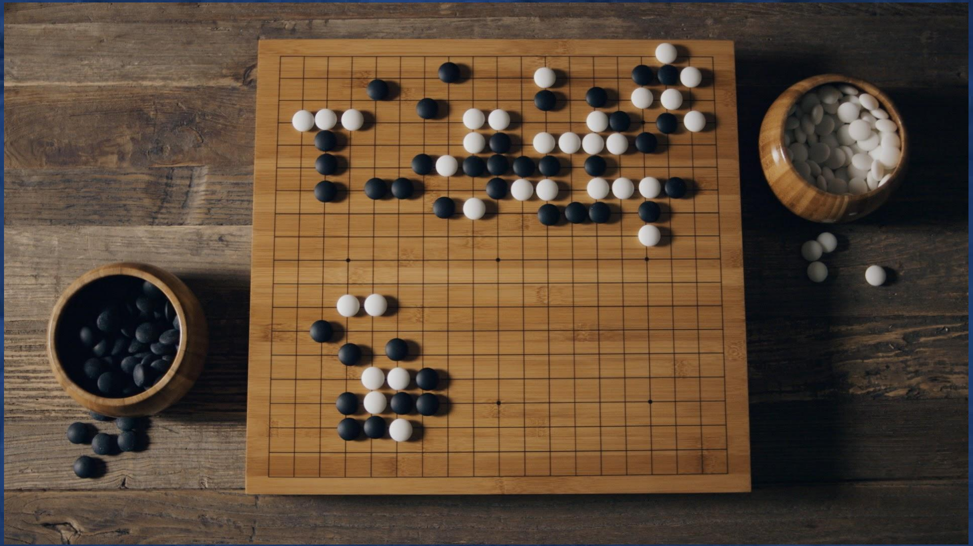
(Silver, Huang, et al 2016)
#3 most downloaded academic paper this month

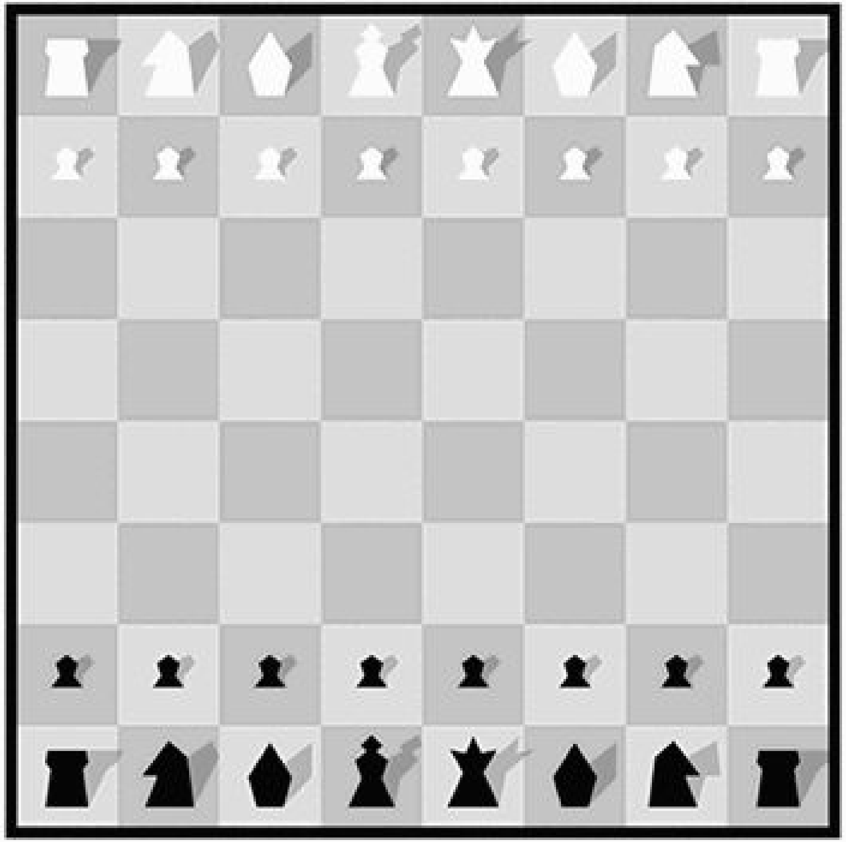
Why is Go hard for computers to play?

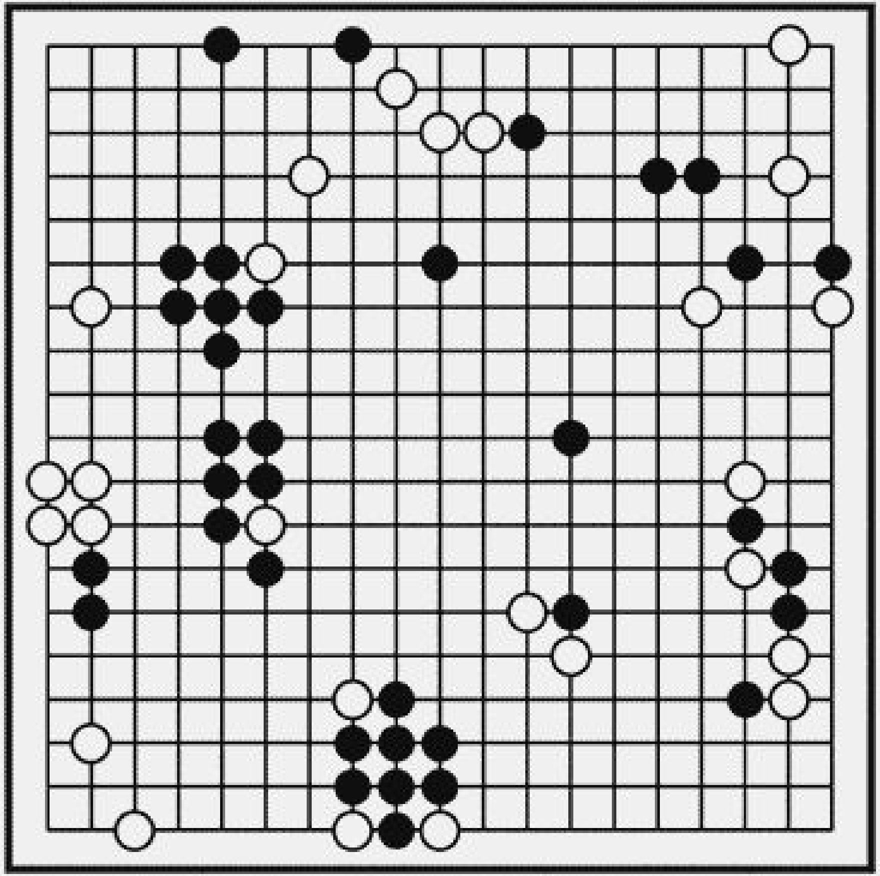
Game tree complexity = b^d

Brute force search intractable:

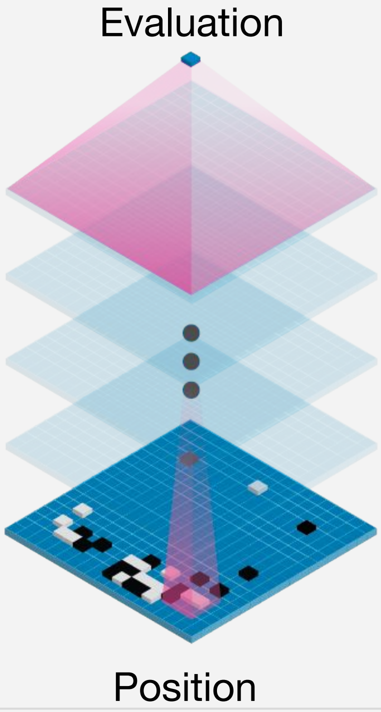
1. Search space is huge
2. “Impossible” for computers to evaluate who is winning

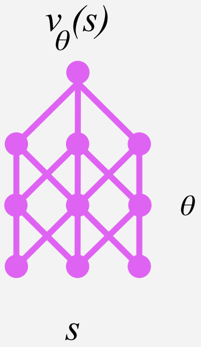






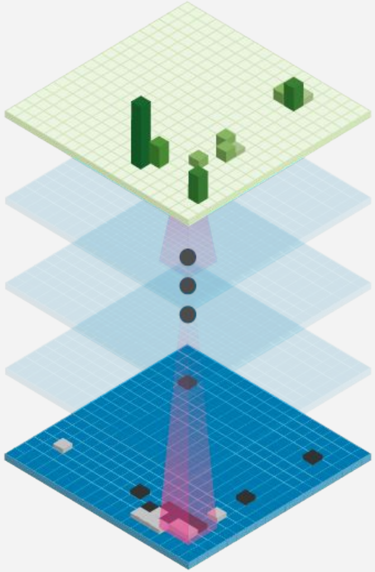
Value network





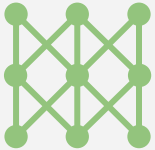
Policy network

Move probabilities



Position

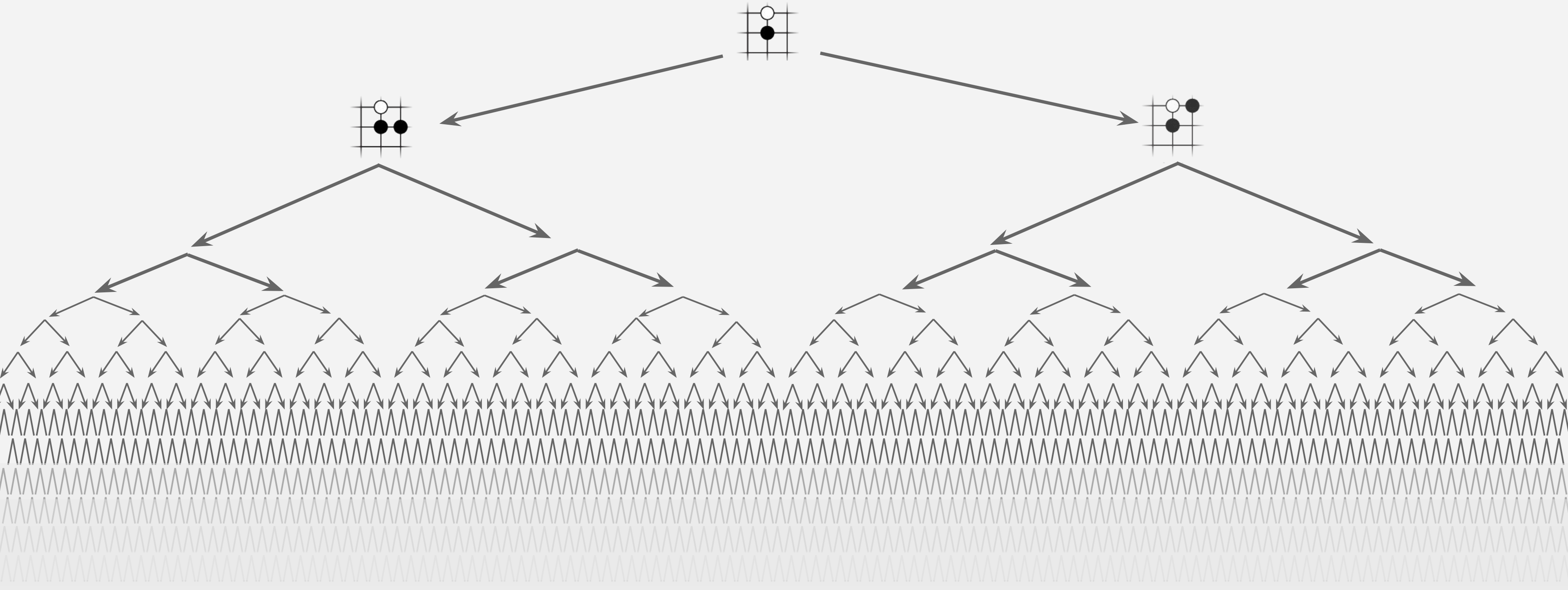
$p_\sigma(a|s)$



σ

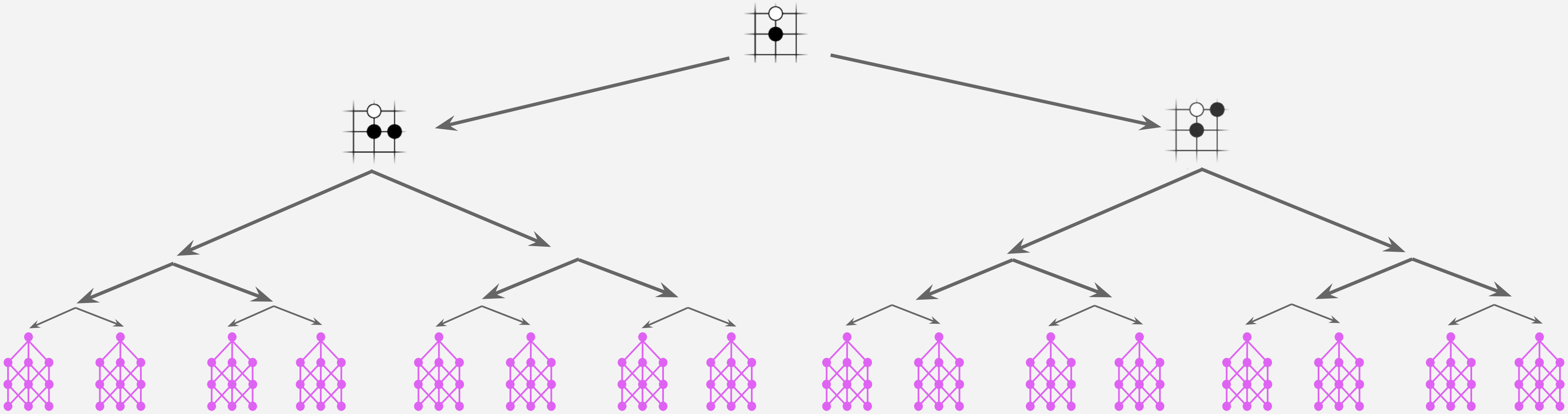
s

Reducing depth with value network

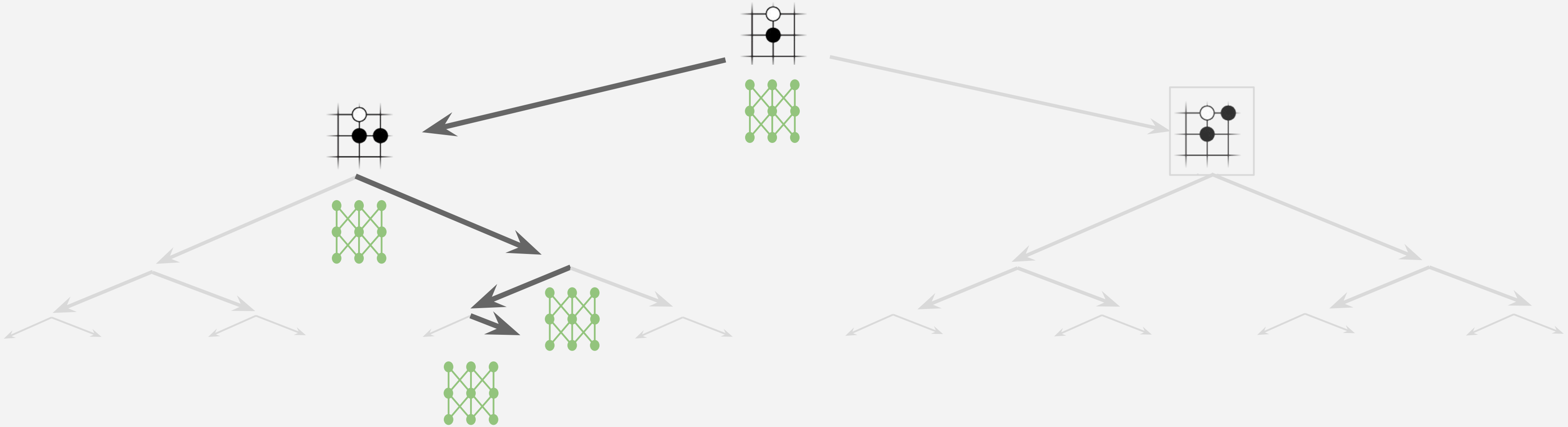


Google DeepMind

Reducing depth with value network



Reducing breadth with policy network

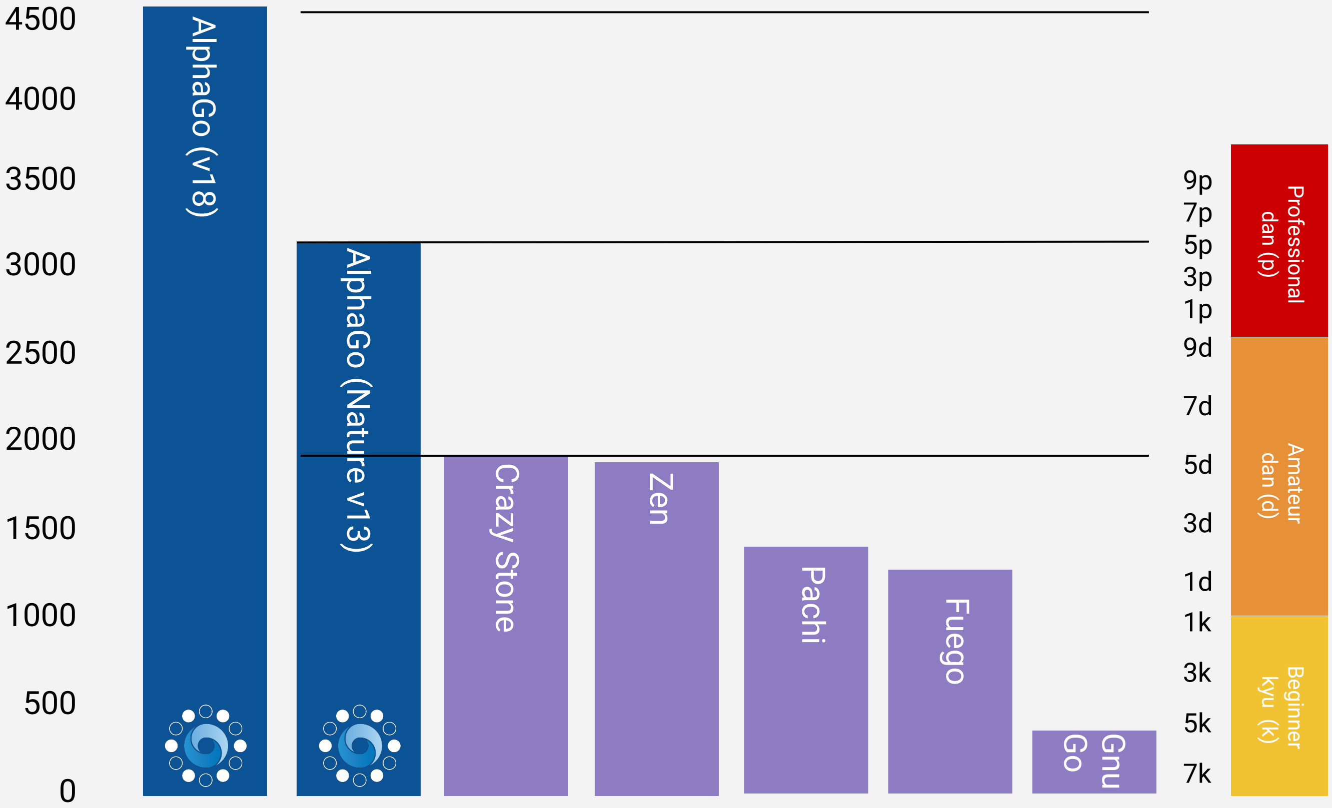


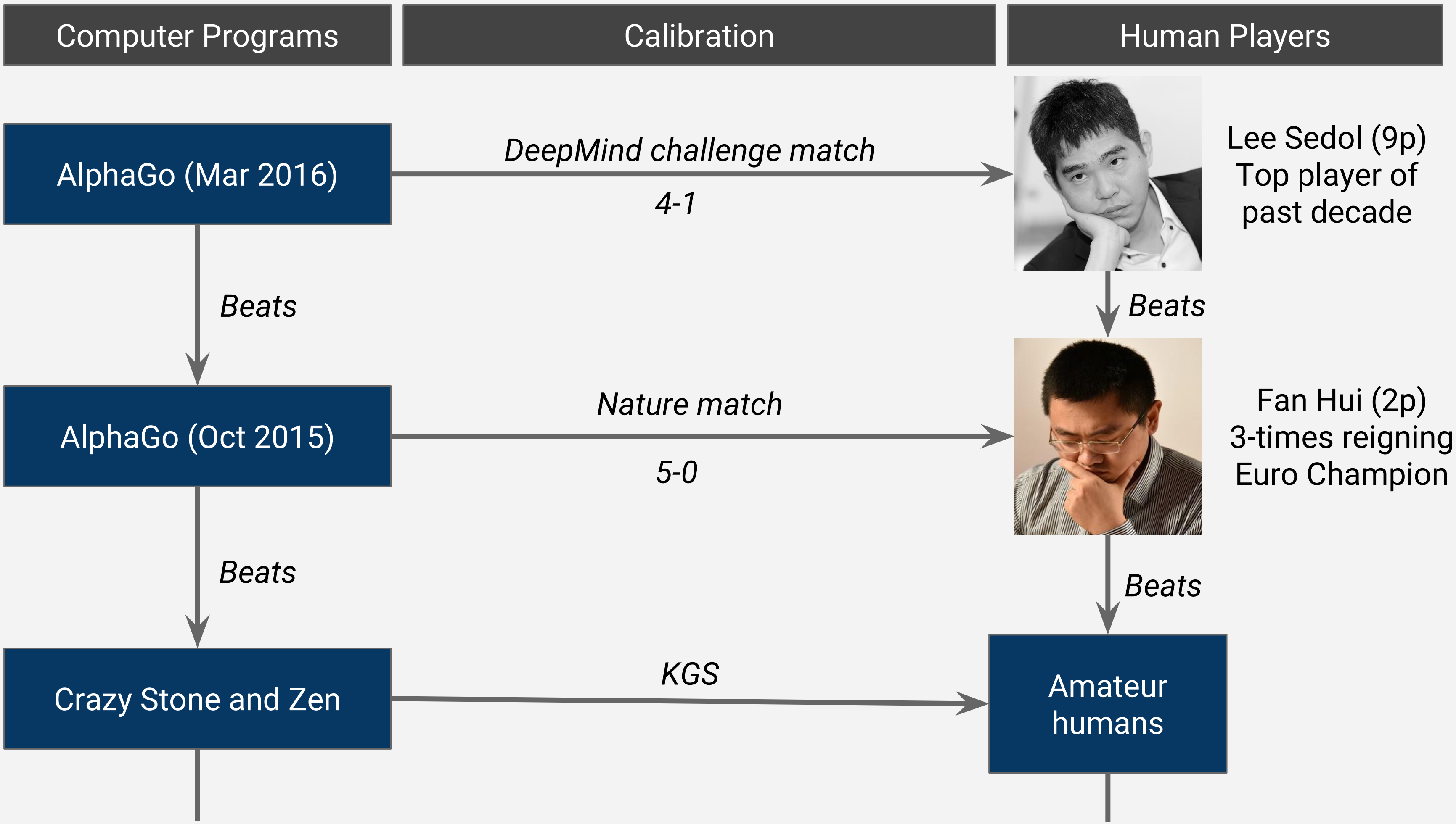
Evaluating current AlphaGo against computers

V13 scores 494/495 against computer opponents

V18 beats V13 3 to 4 stones handicap

CAUTION: ratings based on self-play results





Computer Programs
Calibration
Human Players
AlphaGo (Mar 2016)
DeepMind challenge match
4-1
Lee Sedol (9p)
Top player of
past decade
Beats
Beats
AlphaGo (Oct 2015)
Nature match
5-0
Fan Hui (2p)
3-times reigning
Euro Champion
Beats
Beats
Crazy Stone and Zen
KGS
Amateur
humans

DeepMind Guest Lecturers

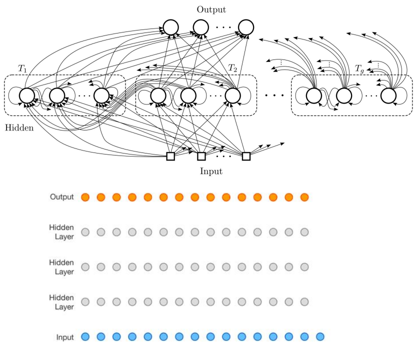
Neural Nets, Backprop, Automatic Differentiation

- Lecture topics:
  - Neural nets
  - Multi-class classification and softmax loss
  - Modular backprop
  - Automatic differentiation
- Guest Lecturer: **Simon Osindero**
  - Joined DeepMind in 2016.
  - Undergrad/Masters in Natural Sciences/Physics at University of Cambridge.
  - PhD in Computational Neuroscience from UCL (2004). Supervisor: Peter Dayan.
  - Postdoc at University of Toronto with Geoff Hinton. (Deep belief nets, 2006).
  - Started an A.I. company, LookFlow, in 2009. Sold to Yahoo in 2013.
  - Current research topics: deep learning, RL agent architectures and algorithms, memory, continual learning.

Convolutional Neural Networks

- Lecture topics:
  - Convolutional networks
  - Large-scale image recognition
  - ImageNet models
- Guest Lecturer: **Karen Simonyan**
  - Joined DeepMind in 2014
  - DPhil (2013) and Postdoc (2014) at the University of Oxford with Andrew Zisserman
  - Research topics: deep learning, computer vision
    - VGGNets, two-stream ConvNets, ConvNet visualisation, etc.
    - https://scholar.google.co.uk/citations?user=L7lMQkQAAAAJ

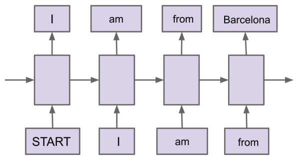
Temporal Hierarchies



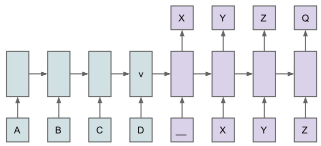
Recurrent Nets and Sequence Generation

- Lecture topics:
  - Recurrent Neural Networks
  - Long-Short Term Memory (LSTM)
  - (Conditional) Sequence Generation
- Guest Lecturer: **Oriol Vinyals**
  - Joined DeepMind in 2016.
  - Worked in Google Brain from 2013 to 2016.
  - PhD in Artificial Intelligence from UC Berkeley (2009-13). Supervisor: Darrell / Morgan.
  - Current research topics: deep learning, sequence modeling, generative models, distillation, RL/Starcraft, one shot learning.

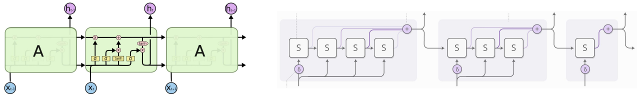
Sequence Prediction



Seq2Seq



Recurrent Architectures



End-To-End and Energy-Based Learning

- Lecture topics:
  - End-to-end learning
  - Energy based learning
  - Ranking
  - Embeddings
  - Triplet loss
- Guest Lecturer: **Raia Hadsell**
  - PhD From NYU, postdoc at CMU’s Robotics Institute
  - Senior Scientist and Tech Manager at SRI International
  - Now leading a research team at DeepMind
  - Research in Deep Learning, Robotics, Navigation, Life-Long Learning

Optimisation

- Lecture topics:
    - First-order methods
    - Second-order methods
    - Stochastic methods
    - Some convergence theory
- Guest Lecturer: **James Martens**
    - Joined DeepMind in Sept 2016
    - PhD from University of Toronto under Geoff Hinton & Rich Zemel in 2015
    - Undergrad from Waterloo in Math and Computer Science
    - Working on: second-order optimization for neural nets, characterizing expressive power/efficiency of neural nets, generative models / unsupervised learning

Attention and Memory Models

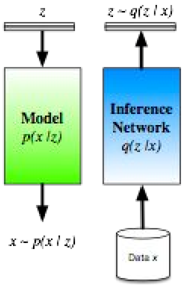
- Lecture topics:
  - Neural attention models
  - Recurrent neural networks with external memory
  - Neural Turing Machines / Differentiable Neural Computers
- Guest Lecturer: **Alex Graves**
  - Joined Deepmind 2013
  - Undergrad Theoretical Physics, Univ. of Edinburgh
  - Masters Mathematics and Theoretical Physics, Univ. of Cambridge
  - PhD Artificial Intelligence TU Munich, supervisor Jürgen Schmidhuber
  - CIFAR Junior fellow with Geoff Hinton, Univ. of Toronto
  - Research focuses on sequence learning with recurrent neural networks: memory, attention, sequence generation, model compression

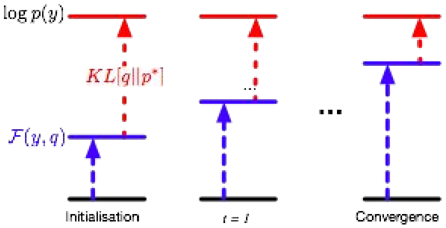
Deep Learning for Natural Language Processing

- Lecture topics:
  - Deep Learning for Natural Language Processing
  - Neural word embeddings
  - Neural machine translation
- Guest Lecturer: **Ed Grefenstette**
  - DPhil from Oxford
  - Co-Founder of Dark Blue Labs (acquired by DeepMind)
  - Research in Machine Learning, Computational Linguistics

Unsupervised Learning and Deep Generative Models

- Lecture topics:
  - Density estimation and unsupervised learning.
  - Deep Generative Models: latent variable and implicit models.
  - Approximate inference and variational inference.
  - Stochastic optimisation





- Guest Lecturer: **Shakir Mohamed**
  - Joined DeepMind in 2013.
  - PhD in Statistical Machine Learning, St John's College, University of Cambridge. Supervisor: Zoubin Ghahramani.
  - CIFAR Junior Research Fellow at the University of British Columbia with Nando de Freitas.
  - Topics in Probabilistic thinking, approximate Bayesian inference, unsupervised learning and density estimation, deep Learning, reinforcement learning.
  - Undergrad in electrical engineering. From Johannesburg, South Africa.

Integral problem

$$p(y) = \int p(y|z)p(z)dz$$

Reinforcement Learning Stream (Hado & Joseph)

- Introduction to Reinforcement Learning
- Markov Decision Processes
- Planning by Dynamic Programming
- Model-Free Prediction
- Model-Free Control

- Value Function Approximation (Deep RL)
- Policy Gradient Methods
- Integrating Learning and Planning
- Exploration and Exploitation
- Case Study: AlphaGo

Hado van Hasselt

Joseph Modayil

Reinforcement learning

A framework for making decisions

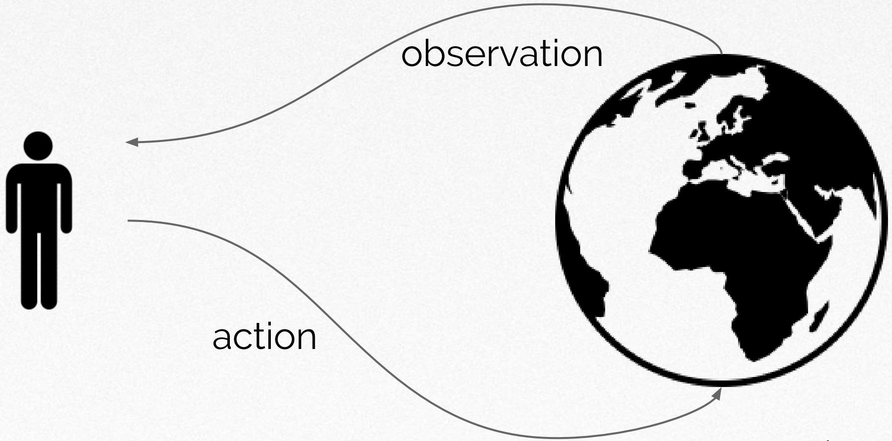


Image credits - AIGA Collection, Martin Vanco

Case Study: AlphaGo

- Lecture topics:
    - The story behind AlphaGo
    - Deep RL applied to Classical Board Games
    - Combining Tree Search and Neural Networks
    - Evaluation against machines and humans
- Guest Lecturer: **David Silver**
    - Computer Science at Cambridge, PhD Alberta
    - Co-Founder/CTO of Elixier Studios
    - Faculty member at UCL (on leave at DeepMind)
    - Joined DeepMind in 2013
    - Research in deep reinforcement learning, integration of learning and planning, games

Assignment 1 - TensorFlow and MNIST

- Goals: Intro to TensorFlow, understand NNs, classify handwritten digits

- Assignment is already on Moodle:
  - Part 1: Playing with TensorFlow
  - Part 2: Your own neural network implementation

- Infrastructure: GitHub(email TA your github account), Moodle

TensorFlow

http://tensorflow.org/

and

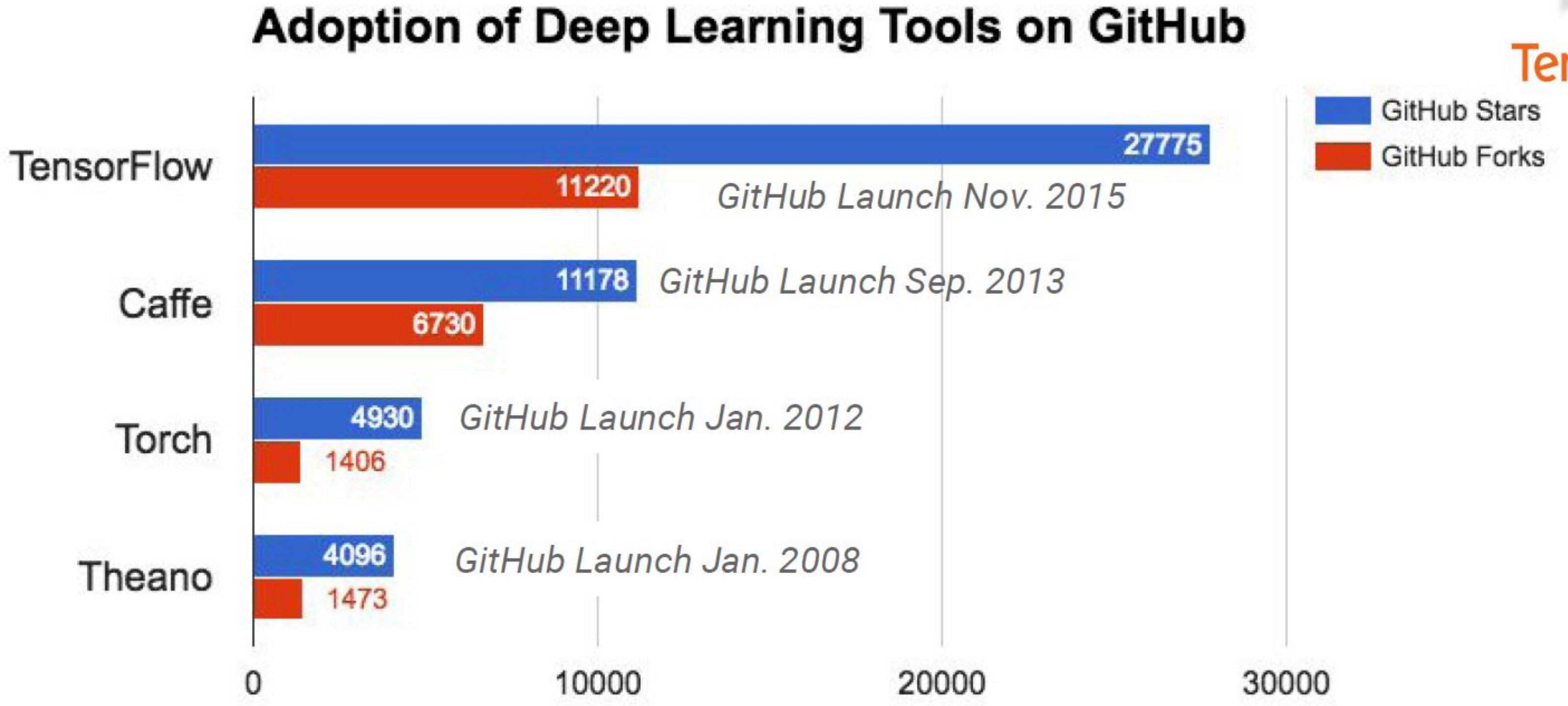
https://github.com/tensorflow/tensorflow

Open, standard software for general machine learning

Great for Deep Learning in particular

First released Nov 2015

Apache 2.0 license

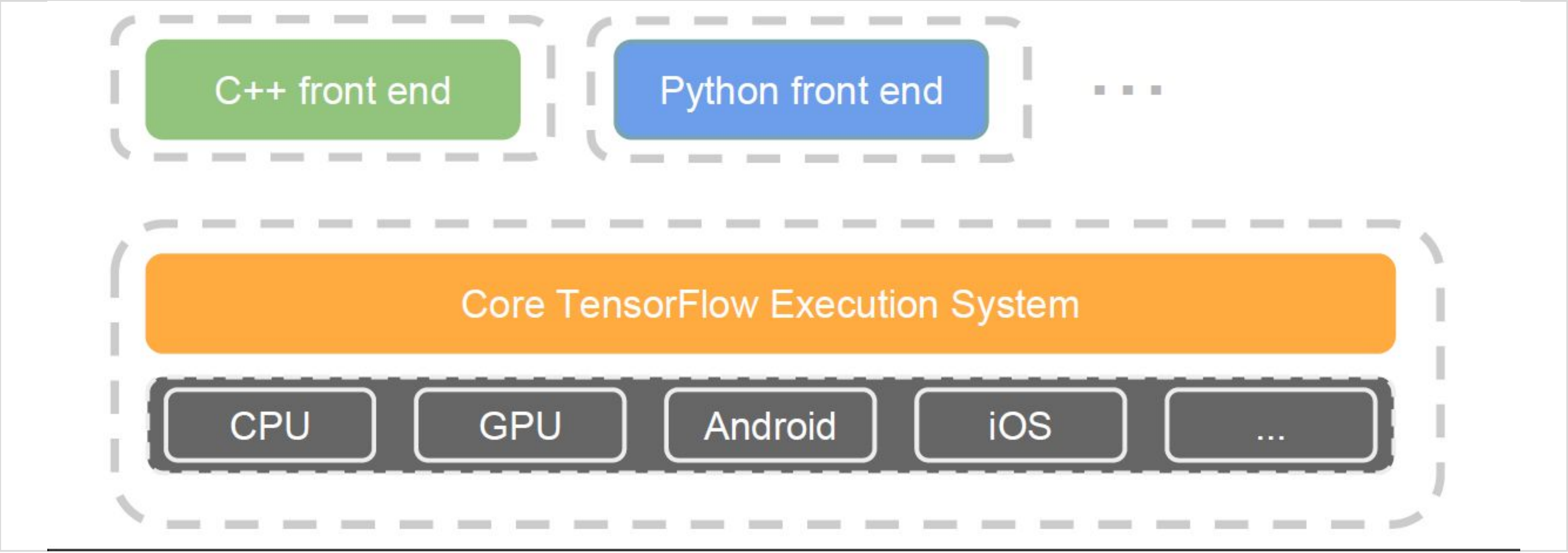


50,000+ binary installs in 72 hours, 500,000+ since November, 2015

Most forked new repo on GitHub in 2015 (despite only being available in Nov, '15)

TensorFlow: High level

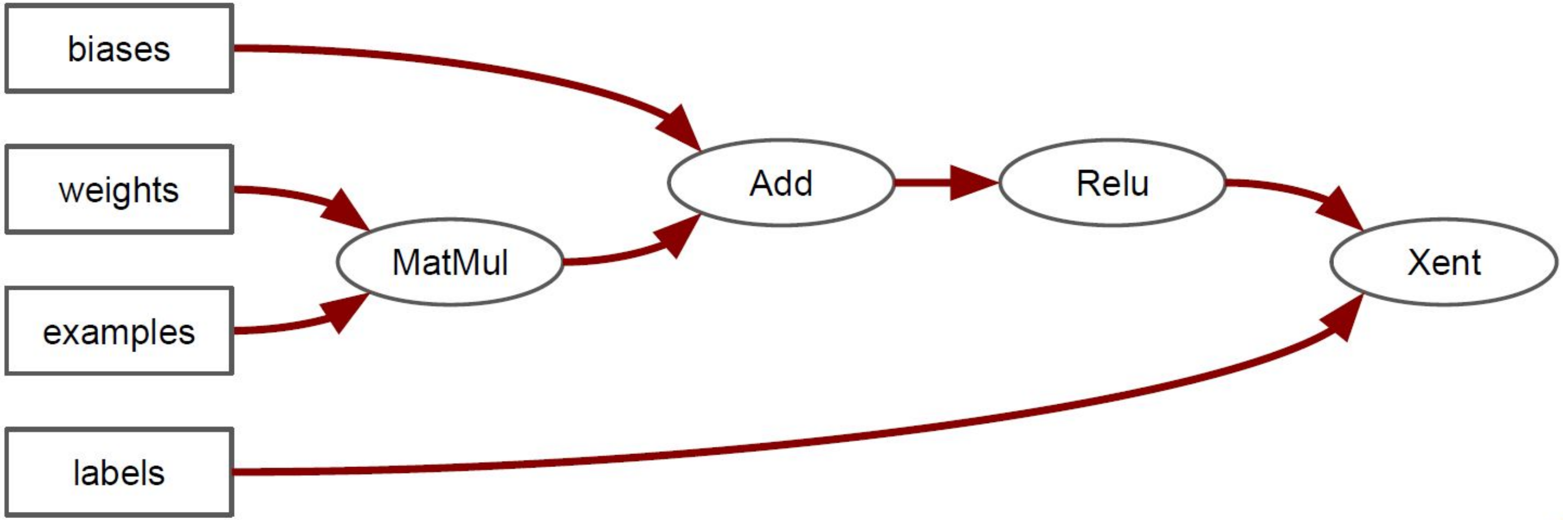
- Core: C++ (Computational engine, cross-platform support)
- Different front-ends for specifying/driving the computation (Python)



(Credit for diagram: Jeff Dean)

TensorFlow:
Computational Graph

- Tensors (multi-dimensional arrays)
- Operations on tensors



(Credit for diagram: Jeff Dean)

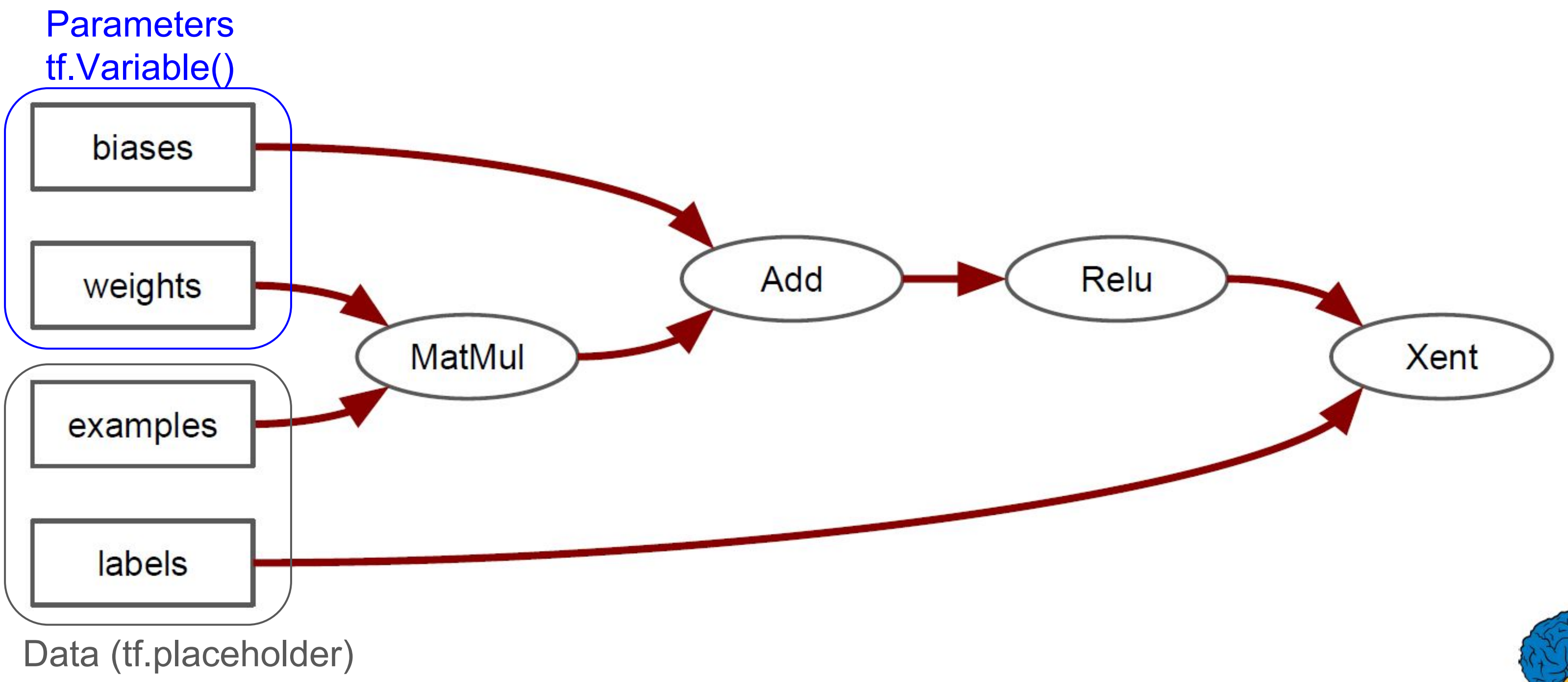
TensorFlow: Example

- Step 1: Build graph
  - Placeholders
  - Variables
  - Tensors

```python

import tensorflow as tf

# building the computation graph
x = tf.placeholder("float", shape=[None, 784])
W = tf.Variable(tf.zeros([784,10]))
b = tf.Variable(tf.zeros([10]))
y = tf.nn.softmax(tf.matmul(x, W) + b)

```

TensorFlow:
Computational Graph

- Tensors (multi-dim arrays)
- Operations



(Credit for diagram: Jeff Dean)

TensorFlow: Example

- Step 2: Specify training operation
  - Symbolic differentiation (automatically adds these ops to the graph to compute gradients)
  - Optimizer applies gradients

```
y_GT = tf.placeholder(tf.float32, [None, 10])
# define loss function
cross_entropy = -tf.reduce_sum(y_GT * tf.log(y))

# define optimizer
opt = tf.train.GradientDescentOptimizer(0.01)
train_op = opt.minimize(cross_entropy)
```

TensorFlow: Example

- Step 3: Launch a session & bring the model to life!
  - Need to initialize or load variables
  - Operations and updates can only be done within a session.

```
# create operation to inialize variables
init = tf.initialize_all_variables()

# launch a session:
with tf.Session() as sess:
    sess.run(init)
    for i in range(1000):
        sess.run(train_step, feed_dict={x: batch_xs, y_GT: batch_ys})
```

Note: Everything happens within sessions. Outside a session, variables won't retain their values.
To experiment and play around: launch an interactive session (sess = tf.InteractiveSession())

TensorFlow Resources

- Tutorials:
  - https://www.tensorflow.org/versions/master/tutorials/
  - https://codelabs.developers.google.com/codelabs/tensorflow-for-poets/
  - Saving/Loading models (https://www.tensorflow.org/versions/master/how_tos/variables/)
- Tensorboard
  - Visualizing computational graph
  - Tracking errors/accuracy (tf.summary)

More about on TF mechanics and advance usage: Check out Jeff Dean's tutorial and overview @ DLSL
http://videolectures.net/deeplearning2016_dean_deep_learning/