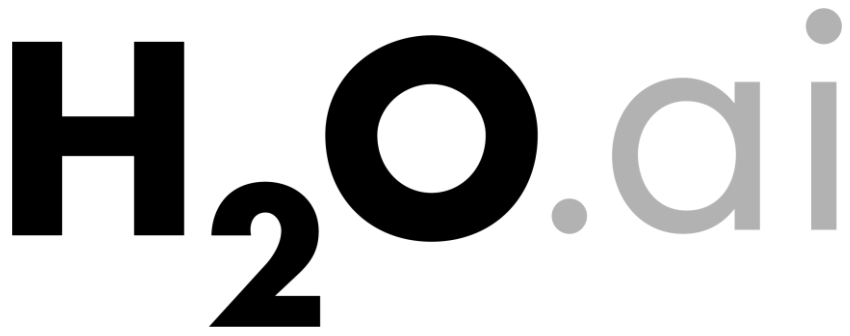
H_2O Deep Water

Making Deep Learning Accessible to Everyone



Jo-fai (Joe) Chow

Data Scientist

joe@h2o.ai

@matlabulous

SF Big Data Science at Metis

23rd February, 2017

About Me

- Civil (Water) Engineer
  - 2010 – 2015
  - Consultant (UK)
    - Utilities
    - Asset Management
    - Constrained Optimization
  - Industrial PhD (UK)
    - Infrastructure Design Optimization
    - Machine Learning + Water Engineering
    - Discovered H_2O in 2014
- Data Scientist
  - From 2015
  - Virgin Media (UK)
  - Domino Data Lab (Silicon Valley)
  - H_2O.ai (Silicon Valley)

Figure 1. Magic Quadrant for Data Science Platforms

Source: Gartner (February 2017)

H2O.ai recognized for completeness of vision and ability to execute

We are thrilled to be named a Visionary among the 16 vendors included in Gartner's 2017 Magic Quadrant for Data Science Platforms. As a Visionary we believe we are positioned highest in Ability to Execute for companies of our size and scale.

Since 2011, our mission has been to democratize data science through open source AI and deep learning. Today, H2O.ai is focused on bringing AI to enterprises with a growing community of more than 8,500 organizations that depend on H2O for mission critical applications. H2O.ai was recently named CB Insights AI 100 and is used by 107 of the Fortune 500 companies.

Disclaimer: This graphic was published by Gartner, Inc. as part of a larger research document and should be evaluated in the context of the entire document. The Gartner document is available upon request from H2O.ai.

http://www.h2o.ai/gartner-magic-quadrant/

H2O.ai

Agenda

- About H_2O.ai
  - Company
  - Machine Learning Platform
- Deep Learning Tools
  - TensorFlow, MXNet, Caffe, H_2O
- Deep Water
  - Motivation
  - Benefits
  - Interface
  - Learning Resources
- Conclusions

About H_2O.ai

Company Overview

Founded	2011 Venture-backed, debuted in 2012
Products	• H_2O Open Source In-Memory AI Prediction Engine • Sparkling Water • Steam
Mission	Operationalize Data Science, and provide a platform for users to build beautiful data products
Team	70 employees • Distributed Systems Engineers doing Machine Learning • World-class visualization designers
Headquarters	Mountain View, CA




H_2O Machine Learning Platform

Algorithms Overview

Supervised Learning

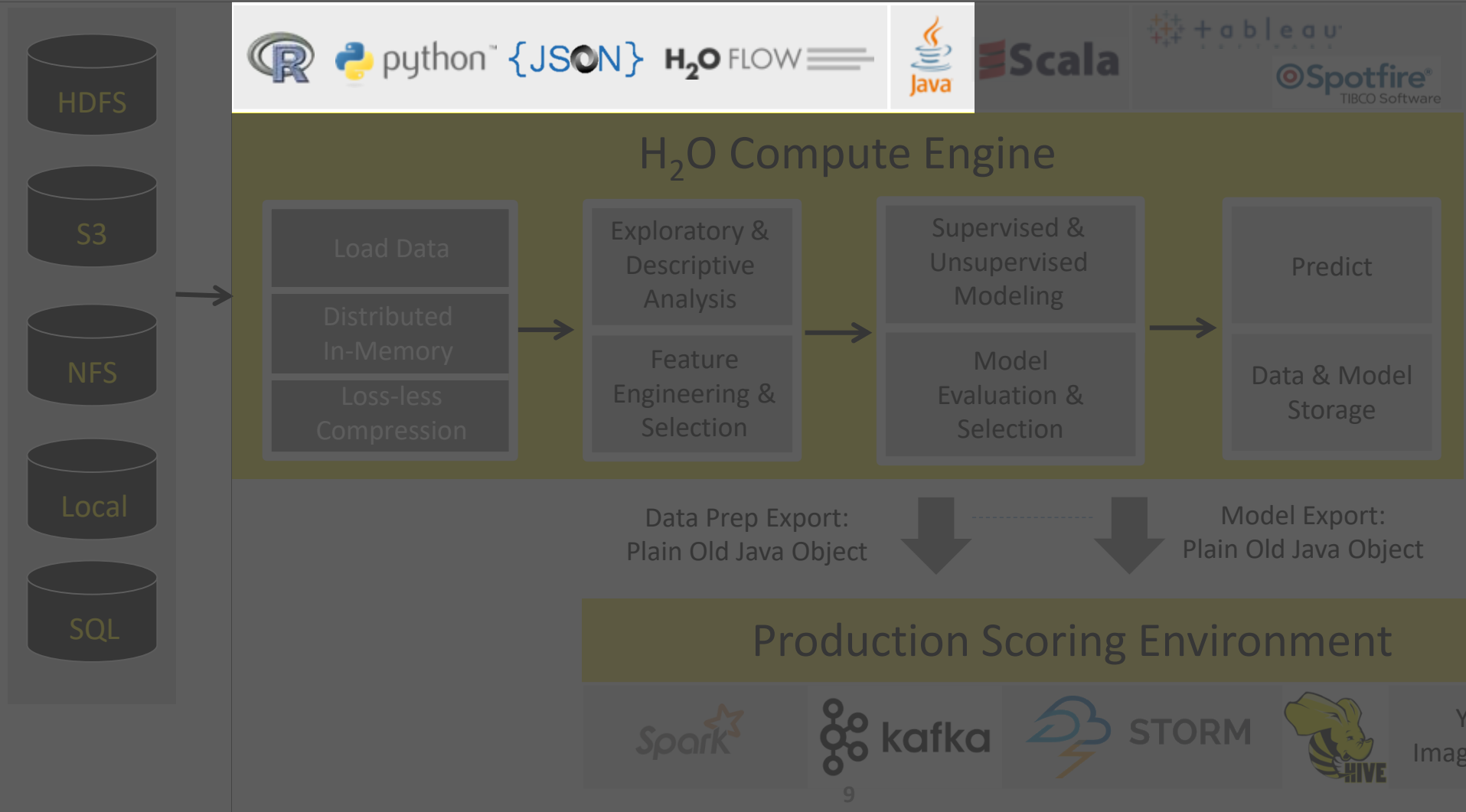
Statistical Analysis	• **Generalized Linear Models**: Binomial, Gaussian, Gamma, Poisson and Tweedie • **Naïve Bayes**
Ensembles	• **Distributed Random Forest**: Classification or regression models • **Gradient Boosting Machine**: Produces an ensemble of decision trees with increasing refined approximations
Deep Neural Networks	• **Deep learning**: Create multi-layer feed forward neural networks starting with an input layer followed by multiple layers of nonlinear transformations

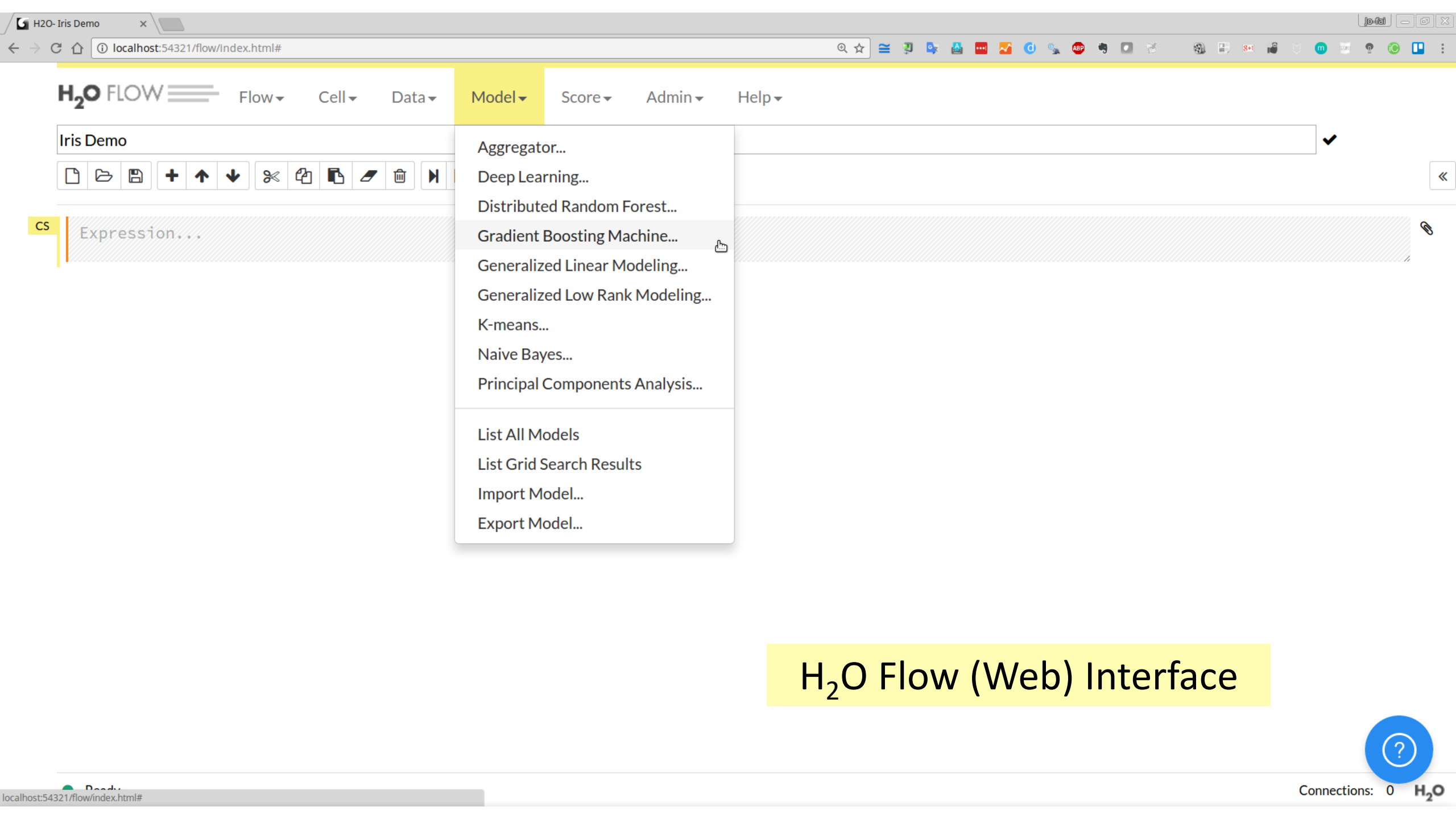
Unsupervised Learning

Clustering	• **K-means**: Partitions observations into k clusters/groups of the same spatial size. Automatically detect optimal k
Dimensionality Reduction	• **Principal Component Analysis**: Linearly transforms correlated variables to independent components • **Generalized Low Rank Models:** extend the idea of PCA to handle arbitrary data consisting of numerical, Boolean, categorical, and missing data
Anomaly Detection	• **Autoencoders**: Find outliers using a nonlinear dimensionality reduction using deep learning

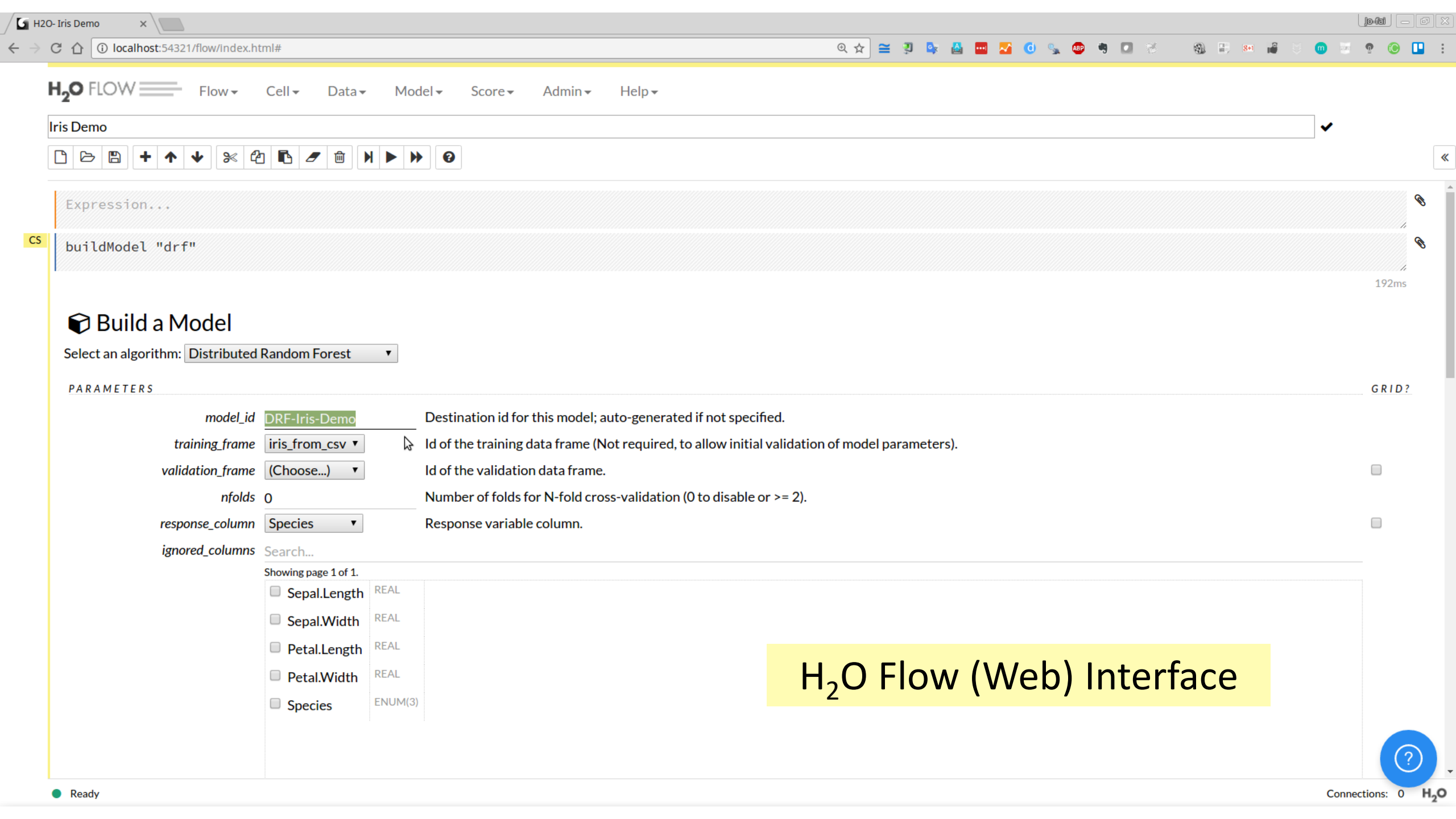
High Level Architecture

Flow (Web), R, Python API
Java for computation





H_2O Flow (Web) Interface


H2O- Iris Demo
localhost:54321/flow/index.html#
H2O FLOW
Flow
Cell
Data
Model
Score
Admin
Help
Iris Demo
Expression...
CS
buildModel "drf"
192ms
Build a Model
Select an algorithm: Distributed Random Forest
PARAMETERS
GRID?
model_id
DRF-Iris-Demo
Destination id for this model; auto-generated if not specified.
training_frame
iris_from_csv
Id of the training data frame (Not required, to allow initial validation of model parameters).
validation_frame
(Choose...)
Id of the validation data frame.
nfolds
0
Number of folds for N-fold cross-validation (0 to disable or >= 2).
response_column
Species
Response variable column.
ignored_columns
Search...
Showing page 1 of 1.
Sepal.Length REAL
Sepal.Width REAL
Petal.Length REAL
Petal.Width REAL
Species ENUM(3)
H2O Flow (Web) Interface
Ready
Connections: 0

H_2O + R

h2o_iris_demo.R

Source on Save | Run | Source

```r
# ------------------------------------------------------------------------------
# Build a simple classification model using iris dataset
# ------------------------------------------------------------------------------

# Start and connect to a local H2O cluster
library(h2o)
h2o.init(nthreads = -1)

# Import data from a R data frame
data(iris)
d_iris <- as.h2o(iris)

# Define Targets and Features
target <- "Species"
features <- setdiff(colnames(d_iris), c("Species"))

# ------------------------------------------------------------------------------
# Train a H2O Model
# ------------------------------------------------------------------------------

# Train three basic H2O models
model_drf <- h2o.randomForest(x = features,
                              y = target,
                              model_id = "iris_random_forest",
                              training_frame = d_iris)

model_gbm <- h2o.gbm(x = features,
                     y = target,
                     model_id = "iris_gbm",
                     training_frame = d_iris)

model_dnn <- h2o.deeplearning(x = features,
                              y = target,
                              model_id = "iris_deep_learning",
                              training_frame = d_iris)
```

docs.h2o.ai

Languages

R

- Quick Start Video - R
- R Package Docs
- R Booklet
- Examples and Demos
- R FAQ
- Ensemble R Package Readme
- RSparkling Readme
- Migrating from H2O-2

Python

- Quick Start Video - Python
- Python Module Docs
- Python Booklet
- Examples and Demos
- Python FAQ
- PySparkling Readme 2.0 | 1.6
- skutil Docs

Java

- POJO and MOJO Model Javadoc
- H2O Core Javadoc
- H2O Algorithms Javadoc

Scala

Sparkling Water API	2.0	1.6
Sparkling Water Scaladoc	2.0	1.6
H2O Scaladoc	2.11	2.10

Tutorials, Examples, & Presentations

Tutorials and Blogs

- H2O Tutorials HTML | PDF
- H2O Blogs
- H2O University

Use Case Examples

Chicago crime prediction	R	Python	ScalaSW	PySW
Airlines delays prediction	R	Python	ScalaSW	PySW
Lending Club loan prediction	R	Python	ScalaSW	PySW
Ham or Spam	R	Python	ScalaSW	PySW
Prediction with prostate dataset	R	Python	ScalaSW	PySW

Presentations

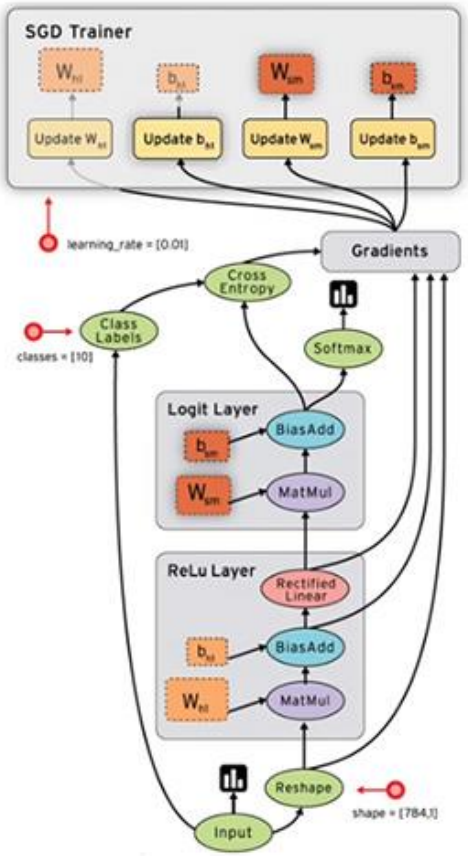
- H2O Meetups
- H2O World 2014 Videos
- H2O World 2015 Videos
- Open Tour Chicago Videos
- Open Tour NYC Videos
- Open Tour Dallas Videos

Deep Learning Tools

TensorFlow, mxnet, Caffe and H_2O Deep Learning

TensorFlow

- Open source machine learning framework by Google
- Python / C++ API
- TensorBoard
  - Data Flow Graph Visualization
- Multi CPU / GPU
  - v0.8+ distributed machines support
- Multi devices support
  - desktop, server and Android devices
- Image, audio and NLP applications
- **HUGE** Community
- Support for Spark, Windows …



https://github.com/tensorflow/tensorflow

TensorFlow Wrappers

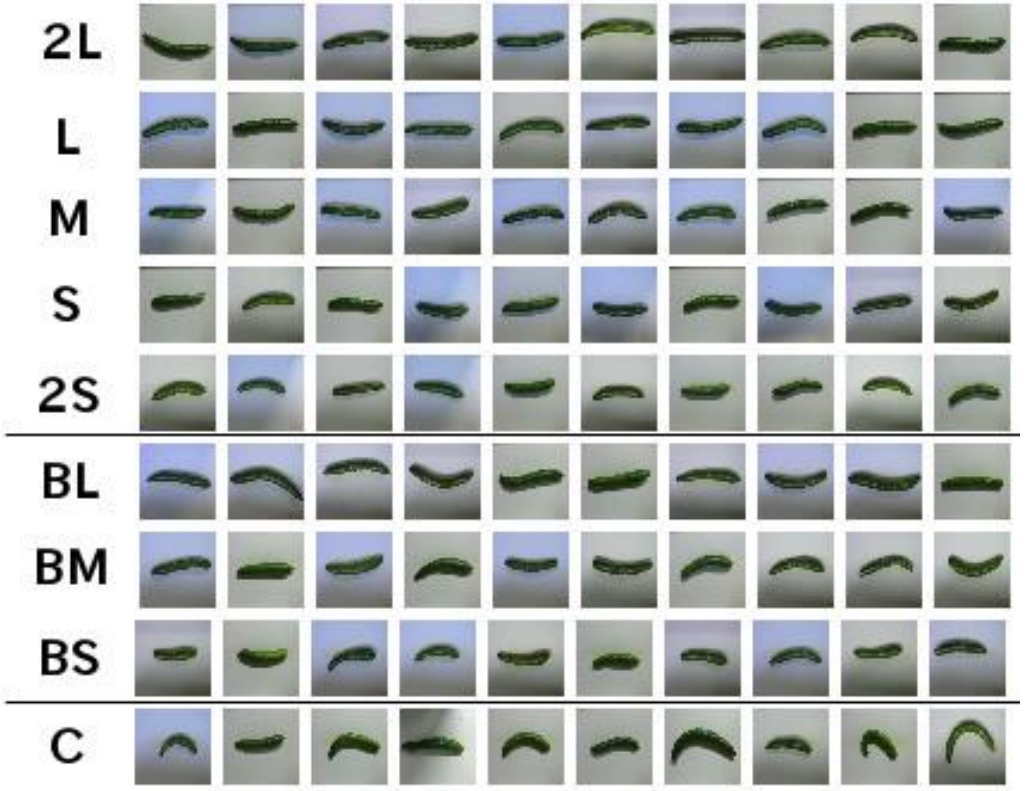
- TFLearn – Simplified interface
- keras – TensorFlow + Theano
- tensorflow.rb – Ruby wrapper
- TensorFlow.jl – Julia wrapper
- TensorFlow for R – R wrapper

- … and many more!
- See: github.com/jtoy/awesome-tensorflow

Sorting Cucumbers

- Problem
  - Sorting cucumbers is a laborious process.
  - In a Japanese farm, the farmer's wife can spend up to eight hours a day sorting cucumbers during peak harvesting period.
- Solution
  - Farmer's son (Makoto Koike) used TensorFlow, Arduino and Raspberry Pi to create an automatic cucumber sorting system.

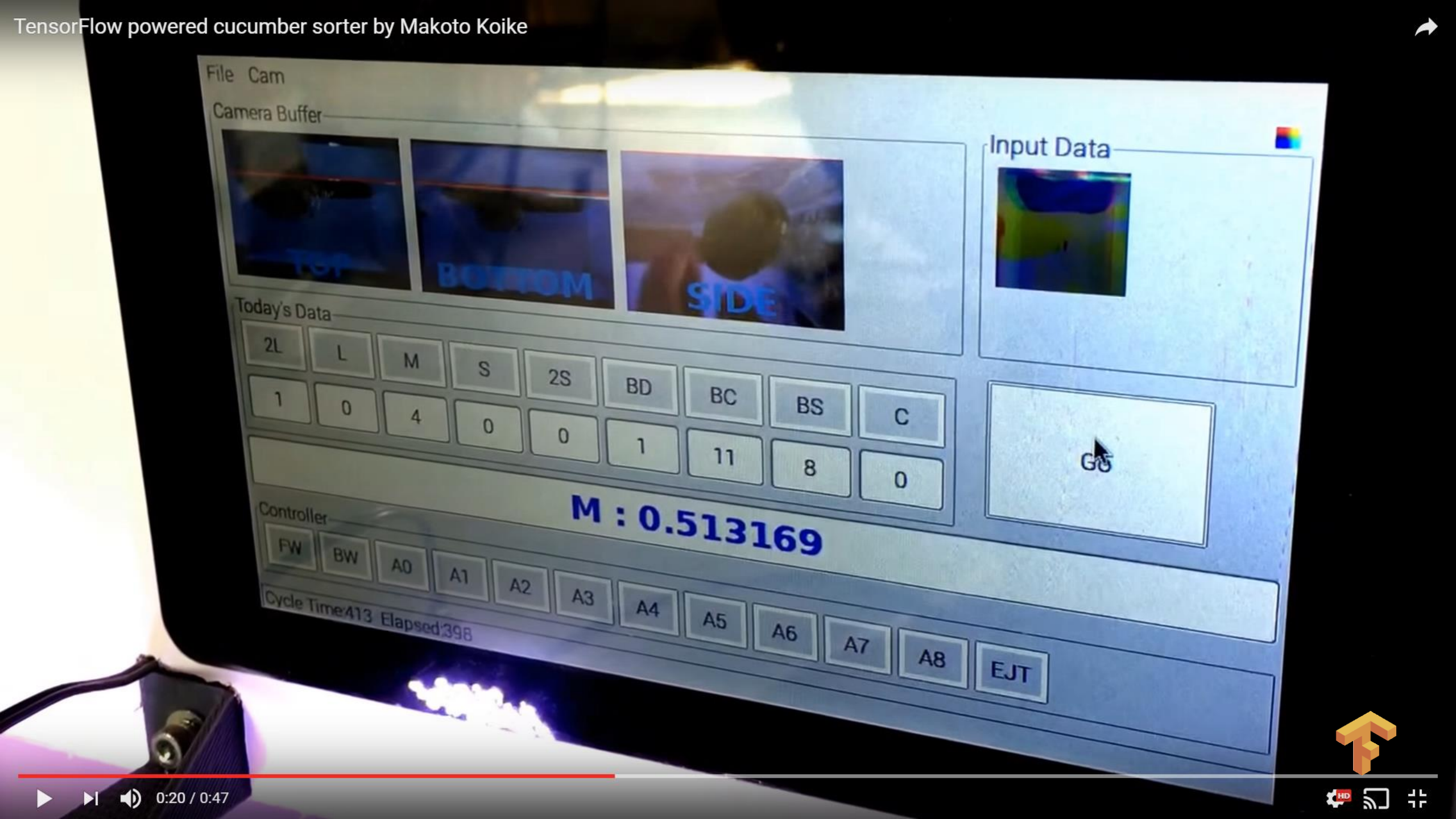
Sorting Cucumbers



- Classification Problem
  - Input: cucumber photos (side, top, bottom)
  - Output: one of nine classes

- Google's Blog Post [Link]
- YouTube Video [Link]


TensorFlow powered cucumber sorter by Makoto Koike
0:02 / 0:47


TensorFlow powered cucumber sorter by Makoto Koike
File Cam
Camera Buffer
TOP
BOTTOM
SIDE
Input Data
Today's Data
2L L M S 2S BD BC BS C
1 0 4 0 0 1 11 8 0
GO
M : 0.513169
Controller
FW BW A0 A1 A2 A3 A4 A5 A6 A7 A8 EJT
Cycle Time:413 Elapsed:398
0:20 / 0:47


TensorFlow powered cucumber sorter by Makoto Koike
宅急便
Bou
0:10 / 0:47

TensorKart – Self-driving Mario Kart


Kevin Hughes, the developer behind TensorKart

"The idea of exploring AI techniques in video games was not new, but what motivated me to do this project was to showcase complete pipeline of a machine learning system. I wanted to pick a popular game because I thought it might interest more people and expose them to how machine learning works," reveals Hughes.

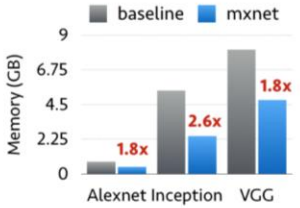
https://github.com/kevinhughes27/TensorKart
https://opensourceforu.com/2017/01/tensorflow-brings-self-driving-to-mario-kart/

dmlc mxnet for Deep Learning

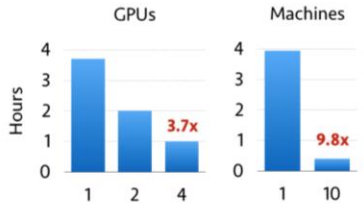
build passing | docs latest | license Apache 2.0

Portable

Efficient



Scalable



MXNet is a deep learning framework designed for both *efficiency* and *flexibility*. It allows you to ***mix*** the flavours of symbolic programming and imperative programming to ***maximize*** efficiency and productivity. In its core, a dynamic dependency scheduler that automatically parallelizes both symbolic and imperative operations on the fly. A graph optimization layer on top of that makes symbolic execution fast and memory efficient. The library is portable and lightweight, and it scales to multiple GPUs and multiple machines.

MXNet is also more than a deep learning project. It is also a collection of blue prints and guidelines for building deep learning system, and interesting insights of DL systems for hackers.

MXNet now chosen by Amazon as Deep Learning Framework

By Geneva Clark | 2016-11-24

19 0

Share this magazine

Amazon has announced that it has chosen MXNet as its deep learning framework of choice for its web services(AWS). Amazon extensively uses machine learning in areas like fraud detection, abusive review detection, and book classification. Amazon also uses it in application areas such as text and speech recognition, autonomous drones etc...

https://github.com/dmlc/mxnet
https://www.zeolearn.com/magazine/amazon-to-use-mxnet-as-deep-learning-framework

H2O.ai

Caffe

- **C**onvolution **A**rchitecture **F**or **F**eature **E**xtraction (CAFFE)
- Pure C++ / CUDA architecture for deep learning
- Command line, Python and MATLAB interface
- Model Zoo
  - Open collection of models

https://docs.google.com/presentation/d/1UeKXVgRvvxg9OUdh_UiC5G71UMscNPlvArsWER41PsU/

H2O.ai

H_2O Deep Learning

Supervised Learning

Statistical Analysis	• **Generalized Linear Models**: Binomial, Gaussian, Gamma, Poisson and Tweedie • **Naïve Bayes**
Ensembles	• **Distributed Random Forest**: Classification or regression models • **Gradient Boosting Machine**: Produces an ensemble of decision trees with increasing refined approximations
Deep Neural Networks	• **Deep learning**: Create multi-layer feed forward neural networks starting with an input layer followed by multiple layers of nonlinear transformations

Unsupervised Learning

Clustering	• **K-means**: Partitions observations into k clusters/groups of the same spatial size. Automatically detect optimal k
Dimensionality Reduction	• **Principal Component Analysis**: Linearly transforms correlated variables to independent components • **Generalized Low Rank Models:** extend the idea of PCA to handle arbitrary data consisting of numerical, Boolean, categorical, and missing data
Anomaly Detection	• **Autoencoders**: Find outliers using a nonlinear dimensionality reduction using deep learning

H_2O Deep Learning

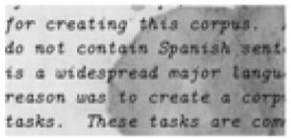
CIFAR-10 Competition Winners: Interviews with Dr. Ben Graham, Phil Culliton, & Zygmunt Zając

Triskelion | 01.02.2015

READ MORE

"I did really like H2O's deep learning implementation in R, though - the interface was great, the back end extremely easy to understand, and it was scalable and flexible. Definitely a tool I'll be going back to."

Kaggle challenge 2nd place winner Colin Priest



Completed • Knowledge • 161 teams

Denoising Dirty Documents

Mon 1 Jun 2015 – Mon 5 Oct 2015 (3 months ago)

READ MORE

"For my final competition submission I used an ensemble of models, including 3 deep learning models built with R and h2o."

H_2O.ai

Both TensorFlow and H_2O are widely used

The usage of Hadoop/Big Data tools grew to 39%, up from 29% in 2015 (and 17% in 2014), driven by Apache Spark, MLlib (Spark Machine Learning Library) and H2O.
See also

- KDnuggets interview with Spark Creator Matei Zaharia
- KDnuggets interview with Arno Candel, H2O.ai on How to Quick Start Deep Learning with H2O

http://www.kdnuggets.com

H2O and TensorFlow are tied

usage of Deep Learning tools in past year

Deep Learning Tools & Platforms: Caffe (63)	0%
Cuda-convnet (22)	0%
Convnet.js (8)	0%
carch (2)	0%
Deeplearning4j (46)	0%
H2O (190)	1%
MATLAB Deep Learning Toolbox (57)	0%
Microsoft CNTK (25)	0%
mxnet (16)	0%
Nervana (2)	0%
Tensorflow (190)	1%
Theano ecosystem including Keras, Lasagne, Pylearn2 (140)	1%
Torch (28)	0%
Veles (2)	0%
Other Deep Learning Tools (104)	1%

H_2O.ai

TensorFlow , MXNet, Caffe and H_2O democratize the power of deep learning.

H_2O platform democratizes artificial intelligence & big data science.

There are other open source deep learning libraries like Theano and Torch too.

Let's have a party, this will be fun!

H_2O.ai


H2O.ai
Products
Industries
Customers
Community
Documentation
Download
Deep Water
H2O.ai
Caffe
mxnet
TensorFlow

Deep Water

Next-Gen Distributed Deep Learning with H_2O

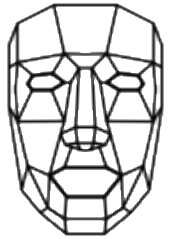
One Interface - GPU Enabled - Significant Performance Gains

Inherits All H_2O Properties in Scalability, Ease of Use and Deployment

TensorFlow Caffe H_2O.ai mxnet

Deep Water

H_2O integrates with existing GPU backends for significant performance gains



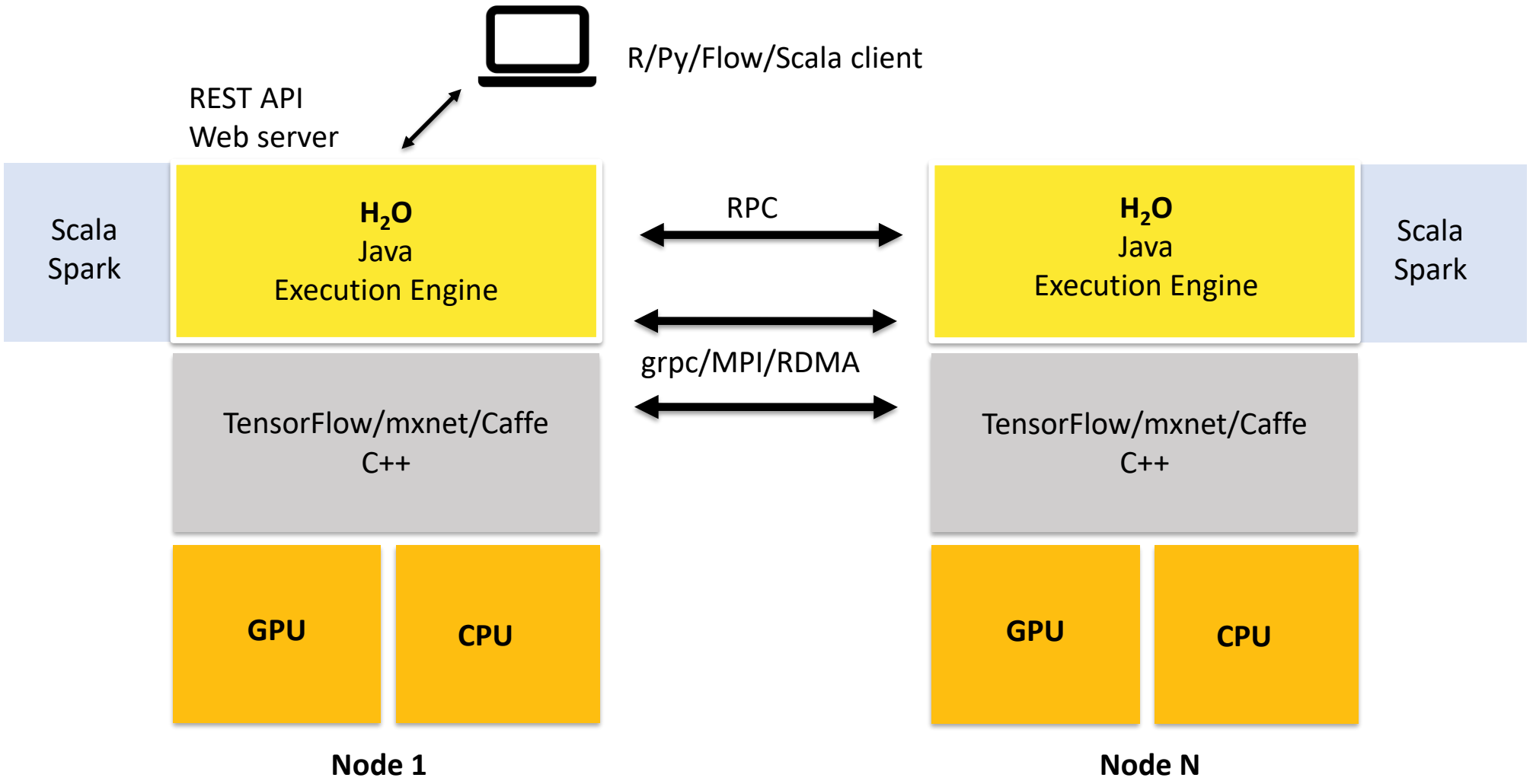
Convolutional Neural Networks enabling **Image, video, speech recognition**

Recurrent Neural Networks enabling **natural language processing, sequences, time series,** and more

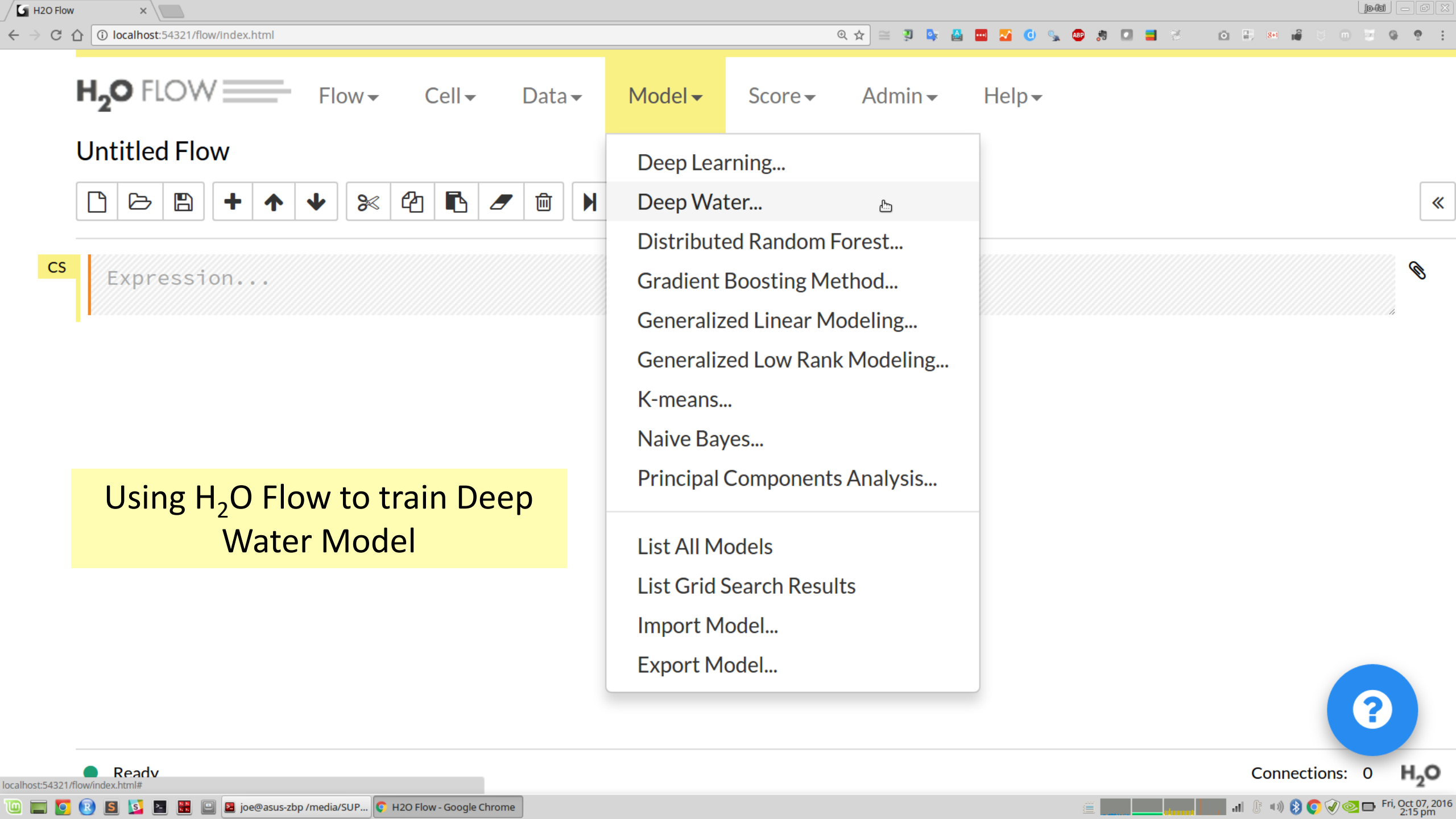
Hybrid Neural Network Architectures enabling **speech to text translation, image captioning, scene parsing** and more

H_2O.ai

Deep Water Architecture

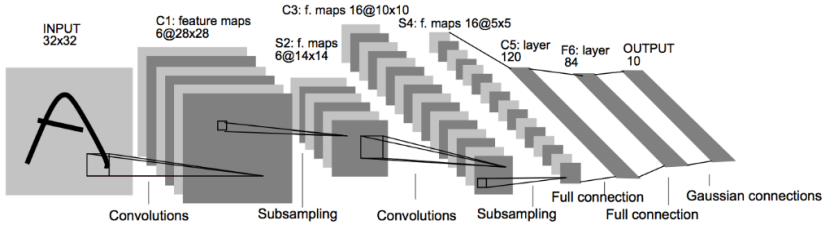


H2O.ai

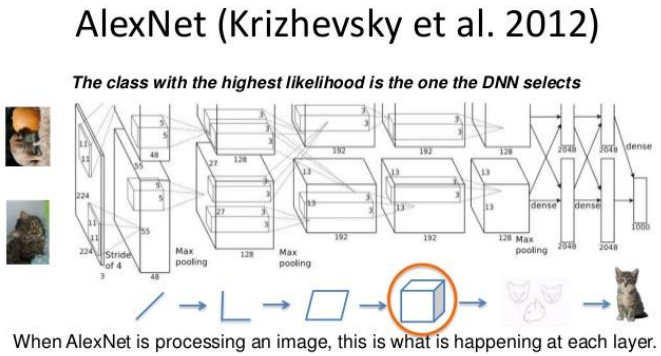

H2O Flow
localhost:54321/flow/index.html
H2O FLOW
Flow
Cell
Data
Model
Score
Admin
Help
Untitled Flow
CS
Expression...
Deep Learning...
Deep Water...
Distributed Random Forest...
Gradient Boosting Method...
Generalized Linear Modeling...
Generalized Low Rank Modeling...
K-means...
Naive Bayes...
Principal Components Analysis...
List All Models
List Grid Search Results
Import Model...
Export Model...
Using H2O Flow to train Deep Water Model
Ready
Connections: 0
H2O
localhost:54321/flow/index.html#
joe@asus-zbp /media/SUP...
H2O Flow - Google Chrome
Fri, Oct 07, 2016
2:15 pm

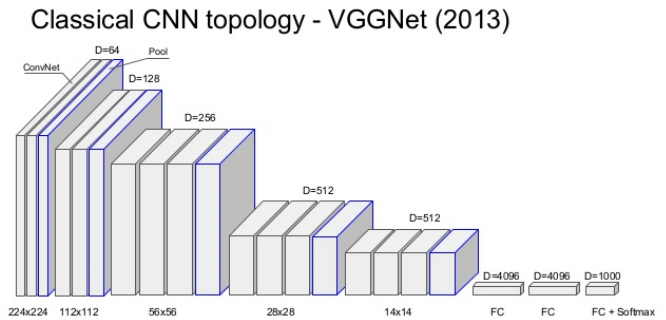
Available Networks in Deep Water

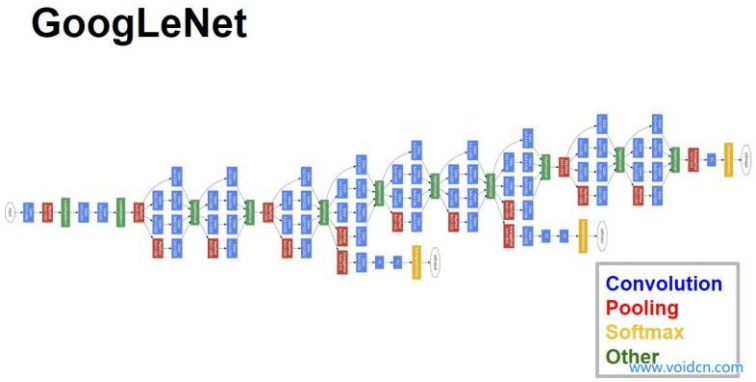
- LeNet
- AlexNet
- VGGNet
- Inception (GoogLeNet)
- ResNet (Deep Residual Learning)
- Build Your Own

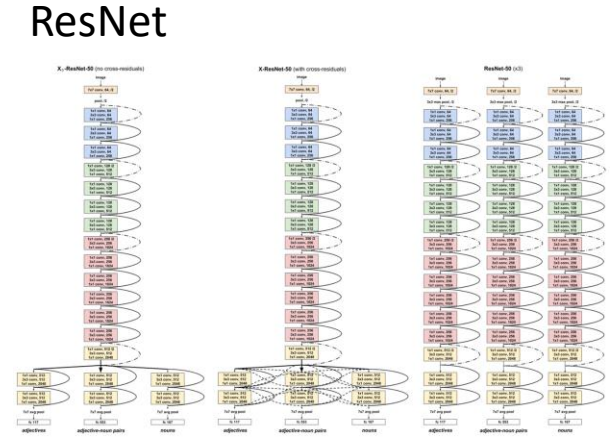


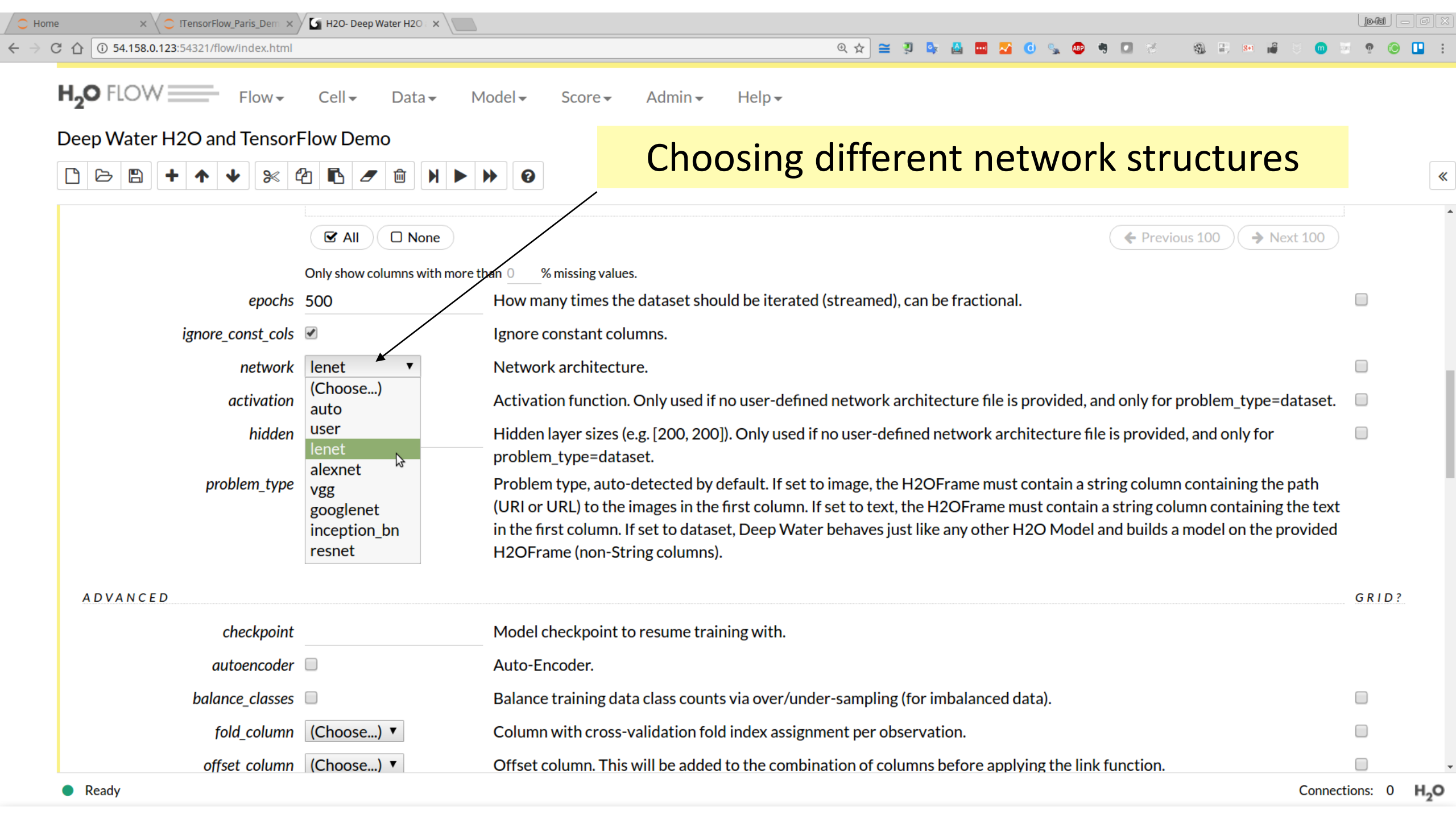
CNN called LeNet by Yann LeCun (1998)

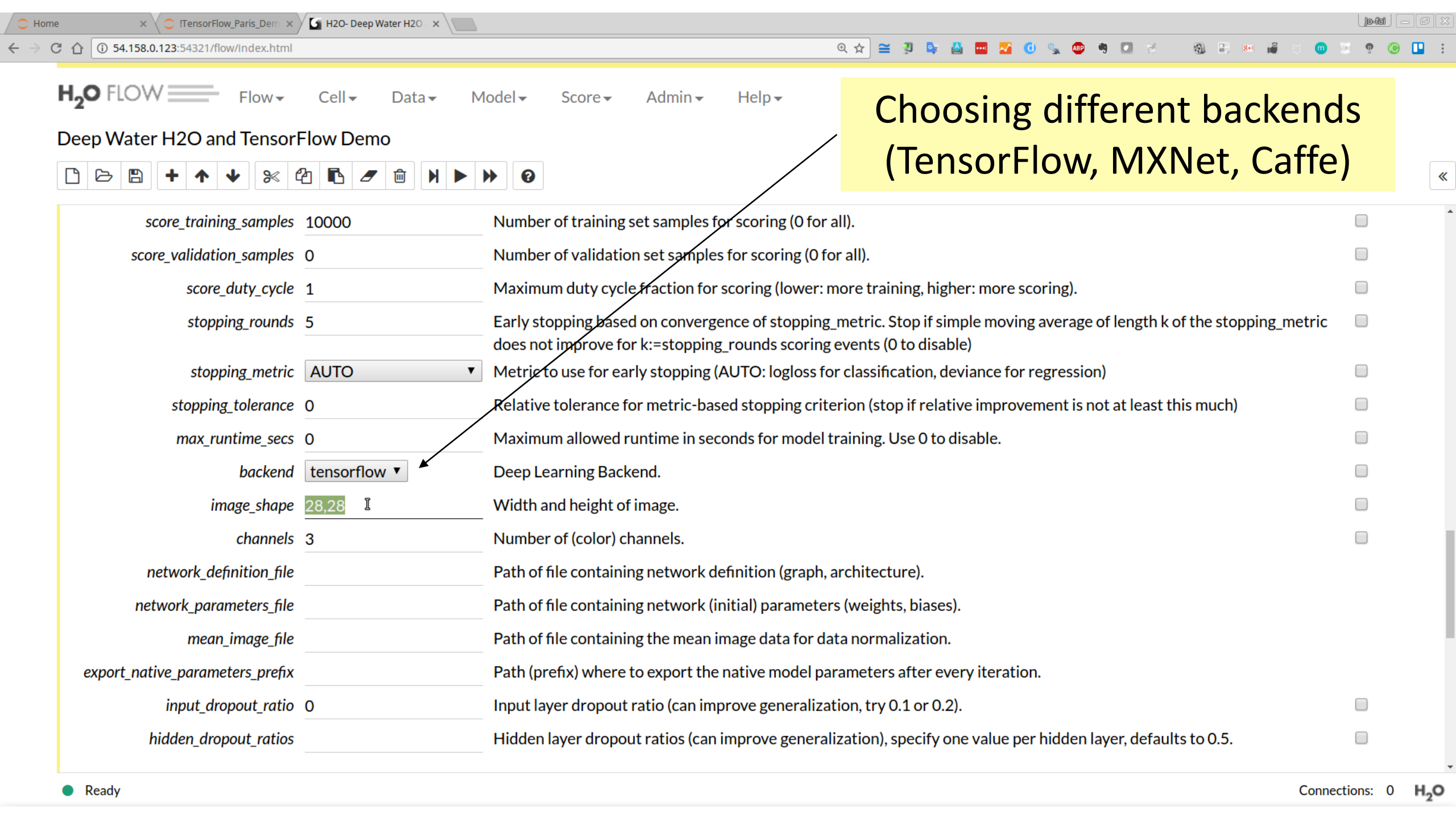









Home
ITensorFlow_Paris_Dem
H2O- Deep Water H2O
54.158.0.123:54321/flow/index.html
H2O FLOW
Flow
Cell
Data
Model
Score
Admin
Help
Deep Water H2O and TensorFlow Demo
Choosing different network structures
All
None
Previous 100
Next 100
Only show columns with more than 0 % missing values.
epochs
500
How many times the dataset should be iterated (streamed), can be fractional.
ignore_const_cols
Ignore constant columns.
network
lenet
Network architecture.
activation
Activation function. Only used if no user-defined network architecture file is provided, and only for problem_type=dataset.
hidden
Hidden layer sizes (e.g. [200, 200]). Only used if no user-defined network architecture file is provided, and only for problem_type=dataset.
problem_type
Problem type, auto-detected by default. If set to image, the H2OFrame must contain a string column containing the path (URI or URL) to the images in the first column. If set to text, the H2OFrame must contain a string column containing the text in the first column. If set to dataset, Deep Water behaves just like any other H2O Model and builds a model on the provided H2OFrame (non-String columns).
(Choose...)
auto
user
lenet
alexnet
vgg
googlenet
inception_bn
resnet
ADVANCED
GRID?
checkpoint
Model checkpoint to resume training with.
autoencoder
Auto-Encoder.
balance_classes
Balance training data class counts via over/under-sampling (for imbalanced data).
fold_column
(Choose...)
Column with cross-validation fold index assignment per observation.
offset_column
(Choose...)
Offset column. This will be added to the combination of columns before applying the link function.
Ready
Connections: 0


Home
ITensorFlow_Paris_Dem
H2O- Deep Water H2O
54.158.0.123:54321/flow/index.html
H2O FLOW
Flow
Cell
Data
Model
Score
Admin
Help
Deep Water H2O and TensorFlow Demo
Choosing different backends
(TensorFlow, MXNet, Caffe)
score_training_samples 10000 Number of training set samples for scoring (0 for all).
score_validation_samples 0 Number of validation set samples for scoring (0 for all).
score_duty_cycle 1 Maximum duty cycle fraction for scoring (lower: more training, higher: more scoring).
stopping_rounds 5 Early stopping based on convergence of stopping_metric. Stop if simple moving average of length k of the stopping_metric does not improve for k:=stopping_rounds scoring events (0 to disable)
stopping_metric AUTO Metric to use for early stopping (AUTO: logloss for classification, deviance for regression)
stopping_tolerance 0 Relative tolerance for metric-based stopping criterion (stop if relative improvement is not at least this much)
max_runtime_secs 0 Maximum allowed runtime in seconds for model training. Use 0 to disable.
backend tensorflow Deep Learning Backend.
image_shape 28,28 Width and height of image.
channels 3 Number of (color) channels.
network_definition_file Path of file containing network definition (graph, architecture).
network_parameters_file Path of file containing network (initial) parameters (weights, biases).
mean_image_file Path of file containing the mean image data for data normalization.
export_native_parameters_prefix Path (prefix) where to export the native model parameters after every iteration.
input_dropout_ratio 0 Input layer dropout ratio (can improve generalization, try 0.1 or 0.2).
hidden_dropout_ratios Hidden layer dropout ratios (can improve generalization), specify one value per hidden layer, defaults to 0.5.
Ready
Connections: 0
H2O

Unified Interface for TF, MXNet and Caffe

Choosing different network structures

```
model = H2ODeepWaterEstimator(epochs      = 500,
                              network     = "lenet",
                              image_shape = [28,28],  ## provide image size
                              channels    = 3,
                              backend     = "tensorflow",
                              model_id    = "deepwater_tf_simple")

model.train(x = [0], # file path e.g. xxx/xxx/xxx.jpg
            y = 1, # label cat/dog/mouse
            training_frame = frame)

model.show()
```

```
deepwater Model Build progress: |████████████████████████████████████████| 100%
Model Details
=============
H2ODeepWaterEstimator :  Deep Water
Model Key:  deepwater_tf_simple
```

Change backend to “mxnet”, “caffe” or “auto”

Easy Stacking with other H_2O Models

Model Stacking

Now we have three different models, we are ready to carry out model stacking.

In [47]:
```
# Create a list to include all the models for stacking
models <- list(model_dw, model_gbm, model_drf)
```

In [48]:
```
# Define a metalearner (one of the H2O supervised machine learning algorithms)
metalearner <- "h2o.glm.wrapper"
```

In [49]:
```
# Use h2o.stack() to carry out metalearning
stack <- h2o.stack(models = models,
                   response_frame = h_train$medv,
                   metalearner = metalearner)
```

```
[1] "Metalearning"
```

In [50]:
```
# Finally, we evaluate the predictive performance on the ensemble as well as individual models.
h2o.ensemble_performance(stack, newdata = h_test)
```

```
Base learner performance, sorted by specified metric:
        learner      MSE
1 h2o_deepwater 8.377644
2       h2o_gbm 8.106541
3       h2o_drf 7.443517


H2O Ensemble Performance on <newdata>:
----------------
Family: gaussian

Ensemble performance (MSE): 5.80436983051916
```

Ensemble of Deep Water, Gradient Boosting Machine & Random Forest models

H2O, Sparkling Water, Steam, & Deep Water Documentation

Getting Started | Data Science Algorithms | Languages | Tutorials, Examples, & Presentations | For Developers | For the Enterprise

docs.h2o.ai

Getting Started

H2O

What is H2O?
H2O User Guide
H2O Book (O'Reilly)
Recent Changes
Open Source License (Apache V2)

Quick Start Video - Flow Web UI
Quick Start Video - R
Quick Start Video - Python

Download H2O

Sparkling Water

What is Sparkling Water?
Sparkling Water Booklet
PySparkling Readme 2.0 | 1.6
RSparkling Readme
Open Source License (Apache V2)

Quick Start Video - Scala
Quick Start Video - Python

Download Sparkling Water

Steam

What is Steam?
Steam User Guide
Recent Changes
Open Source License (AGPL)

Download Steam

Deep Water (preview)

Deep Water Readme
Deep Water AMI Guide
Open Source License (Apache V2)

Launch Deep Water AMI
(choose g2.2xlarge)

Q & A

FAQ
Community Forum
h2ostream Google Group
Issue Tracking (JIRA)
Gitter
Stack Overflow
Cross Validated

For Supported Enterprise Customers

Enterprise Support Web | Email

Branch: master ▾ h2o-3 / examples / deeplearning / notebooks / Create new file | Upload files | Find file | History

mstensmo changing the name of deeplearning_credit_card_default_risk_prediction... ... Latest commit 5568350 11 days ago

..		
images	Add cat/dog/mouse lenet example.	3 months ago
README.md	Update README.md	2 months ago
deeplearning_anomaly_detection.ip...	Update notebooks, introduce local paths to ~/h2o-3/	3 months ago
deeplearning_benchmark_mnist.ipynb	Update lenet test to remove all. Update MNIST benchmark with comments.	3 months ago
deeplearning_cat_dog_mouse_incep...	Add credit card default risk model, update other notebooks.	
deeplearning_cat_dog_mouse_lenet....	Add credit card default risk model, update other notebooks.	
deeplearning_cat_dog_mouse_lenet...	Add back model.plot() and scoring history.	
deeplearning_cifar10_vgg.ipynb	Rename notebooks.	
deeplearning_credit_card_default_ri...	changing the name of deeplearning_credit_card_default_risk_prediction...	
deeplearning_ensemble_boston_ho...	Ensemble demo using GBM, DRF and Deep Water (#676)	17 days ago
deeplearning_grid_iris.ipynb	Add two new notebooks: Lenet for R and iris grid for python	3 months ago
deeplearning_grid_iris_R.ipynb	Update R py notebook.	3 months ago
deeplearning_image_reconstruction...	Update notebooks, introduce local paths to ~/h2o-3/	3 months ago
deeplearning_mnist_convnet.ipynb	Update notebooks, introduce local paths to ~/h2o-3/	3 months ago
deeplearning_mnist_introduction.ip...	Add missing file.	3 months ago
deeplearning_tensorflow_cat_dog_...	Add tensorflow example (#529)	2 months ago
deeplearning_tensorflow_mnist.ipynb	Added MNIST example for TensorFlow	a month ago

Deep Water Example notebooks

https://github.com/h2oai/h2o-3/tree/master/examples/deeplearning/notebooks

H2O.ai

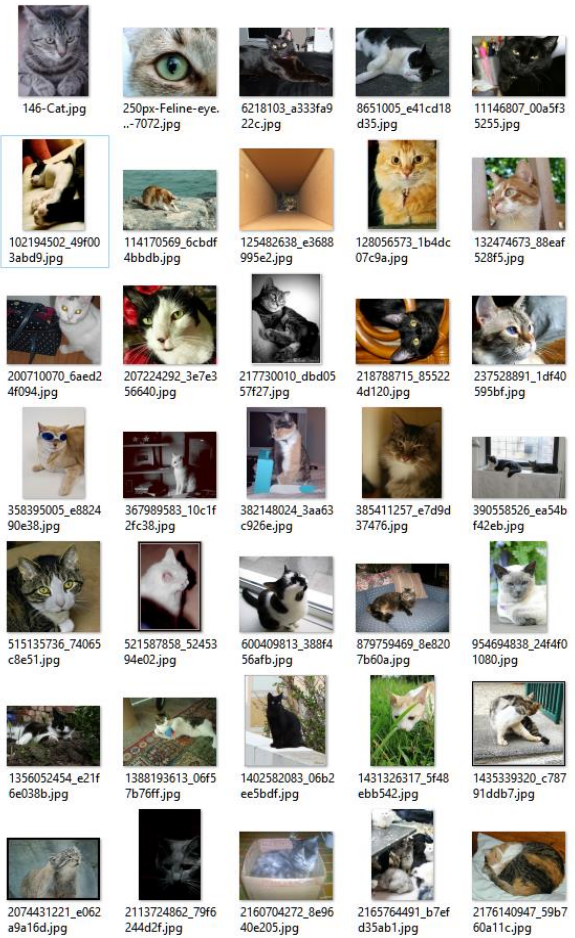
Deep Water Cat/Dog/Mouse Demo

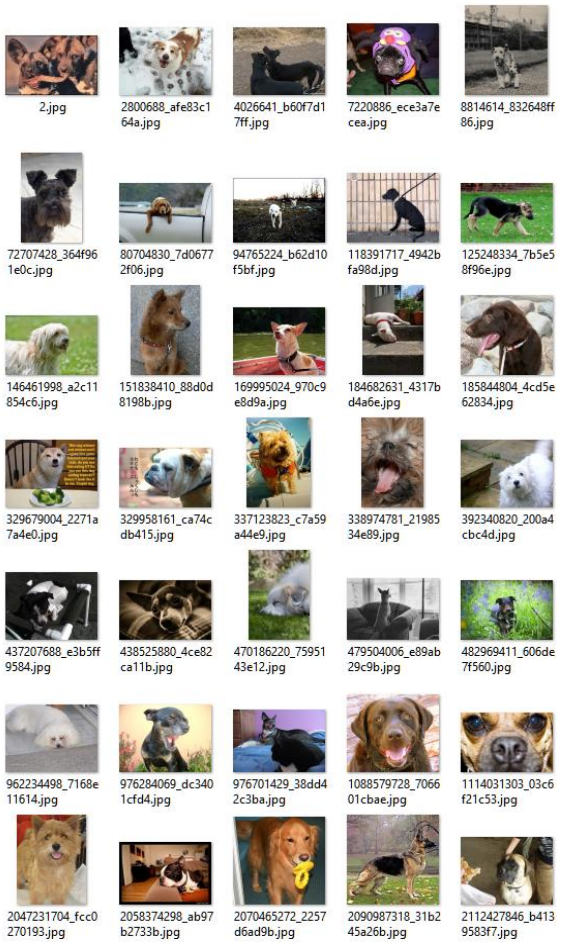
Deep Water H_2O + TensorFlow Demo

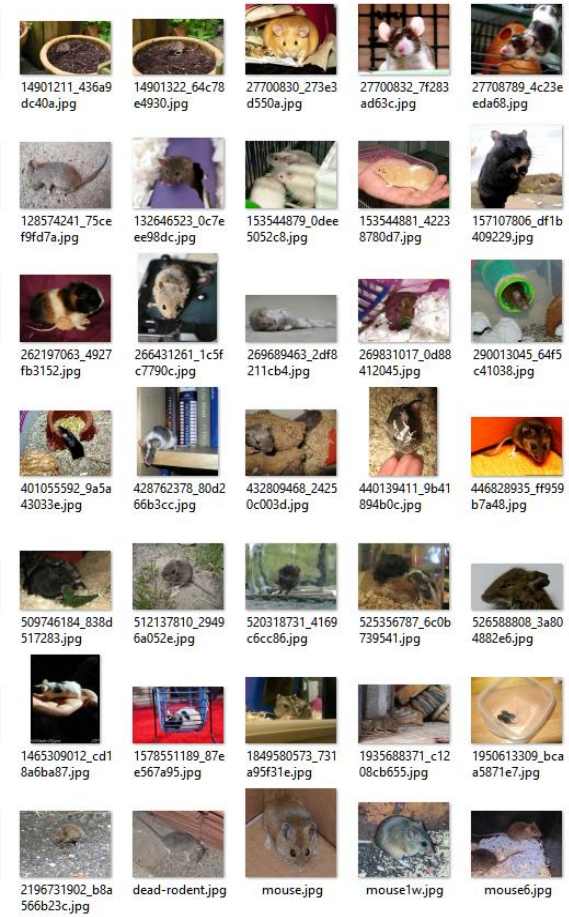
- H_2O + TensorFlow
  - Dataset – Cat/Dog/Mouse
  - TensorFlow as GPU backend
  - Train a LeNet (CNN) model
  - Interfaces
    - Python (Jupyter Notebook)
    - Web (H_2O Flow)

- Code and Data
  - github.com/h2oai/deepwater

Data – Cat/Dog/Mouse Images







H2O.ai

Data – CSV

	A	B
1	bigdata/laptop/deepwater/imagenet/cat/102194502_49f003abd9.jpg	cat
2	bigdata/laptop/deepwater/imagenet/cat/11146807_00a5f35255.jpg	cat
3	bigdata/laptop/deepwater/imagenet/cat/1140846215_70e326f868.jpg	cat
4	bigdata/laptop/deepwater/imagenet/cat/114170569_6cbdf4bbdb.jpg	cat
5	bigdata/laptop/deepwater/imagenet/cat/1217664848_de4c7fc296.jpg	cat
6	bigdata/laptop/deepwater/imagenet/cat/1241603780_5e8c8f1ced.jpg	cat
7	bigdata/laptop/deepwater/imagenet/cat/1241612072_27ececbdef.jpg	cat
8	bigdata/laptop/deepwater/imagenet/cat/1241613138_ef1d82973f.jpg	cat
9	bigdata/laptop/deepwater/imagenet/cat/1244562192_35becd66bd.jpg	cat
10	bigdata/laptop/deepwater/imagenet/cat/125482638_e3688995e2.jpg	cat
11	bigdata/laptop/deepwater/imagenet/cat/128056573_1b4dc07c9a.jpg	cat
12	bigdata/laptop/deepwater/imagenet/cat/12945197_75e607e355.jpg	cat
13	bigdata/laptop/deepwater/imagenet/cat/132474673_88eaf528f5.jpg	cat
14	bigdata/laptop/deepwater/imagenet/cat/1350530984_ecf3039cf0.jpg	cat
15	bigdata/laptop/deepwater/imagenet/cat/1351606235_c9fbebf634.jpg	cat
16	bigdata/laptop/deepwater/imagenet/cat/1356052454_e21f6e038b.jpg	cat
17	bigdata/laptop/deepwater/imagenet/cat/1388193613_06f57b76ff.jpg	cat

Using Tensorflow with H2O

This notebook shows how to use the tensorflow backend to tackle a simple image classification problem.

We start by connecting to our h2o cluster:

In [1]:
```
import h2o
h2o.init(port=54321, nthreads=-1)
```

```
Checking whether there is an H2O instance running at http://localhost:54321. connected.
```

H2O cluster uptime:	54 mins 37 secs
H2O cluster version:	3.11.0.99999
H2O cluster version age:	6 days
H2O cluster name:	ubuntu
H2O cluster total nodes:	1
H2O cluster free memory:	8.86 Gb
H2O cluster total cores:	8
H2O cluster allowed cores:	8
H2O cluster status:	locked, healthy
H2O connection url:	http://localhost:54321
H2O connection proxy:	None
Python version:	2.7.12 final

Then we make sure that the H2O cluster has the DeepWater distribution

In [2]:
```
from h2o.estimators.deepwater import H2ODeepWaterEstimator
if not H2ODeepWaterEstimator.available(): exit
```

H2O.ai

Configuration

Set the path to your h2o installation and download the 'bigdata' dataset using `./gradlew syncBigdataLaptop` from the H2O source distribution.

In [5]:
```
H2O_PATH=os.path.expanduser("~/h2o-3/")
```

Image Classification Task

H2O DeepWater allows you to specify a list of URIs (file paths) or URLs (links) to images, together with a response column (either a class membership (enum) or regression target (numeric)).

For this example, we use a small dataset that has a few hundred images, and three classes: cat, dog and mouse.

In [6]:
```
frame = h2o.import_file(H2O_PATH + "/bigdata/laptop/deepwater/imagenet/cat_dog_mouse.csv")
print(frame.dim)
print(frame.head(5))
```

```
Parse progress: |████████████████████████████████████████████████████████| 100%
[267, 2]
```

C1	C2
bigdata/laptop/deepwater/imagenet/cat/102194502_49f003abd9.jpg	cat
bigdata/laptop/deepwater/imagenet/cat/11146807_00a5f35255.jpg	cat
bigdata/laptop/deepwater/imagenet/cat/1140846215_70e326f868.jpg	cat
bigdata/laptop/deepwater/imagenet/cat/114170569_6cbdf4bbdb.jpg	cat
bigdata/laptop/deepwater/imagenet/cat/1217664848_de4c7fc296.jpg	cat

Deep Water Basic Usage

Unified Interface for TF, MXNet and Caffe

Choosing different network structures

```
model = H2ODeepWaterEstimator(epochs      = 500,
                              network     = "lenet",
                              image_shape = [28,28],  ## provide image size
                              channels    = 3,
                              backend     = "tensorflow",
                              model_id    = "deepwater_tf_simple")

model.train(x = [0], # file path e.g. xxx/xxx/xxx.jpg
            y = 1, # label cat/dog/mouse
            training_frame = frame)

model.show()
```

```
deepwater Model Build progress: |████████████████████████████████████████| 100%
Model Details
=============
H2ODeepWaterEstimator :  Deep Water
Model Key:  deepwater_tf_simple
```

Change backend to “mxnet”, “caffe” or “auto”

Home | (Busy) !TensorFlow_Par | 54.158.0.123/notebooks/!TensorFlow_Paris_Demo.ipynb

jupyter !TensorFlow_Paris_Demo Last Checkpoint: 13 minutes ago (u

File Edit View Insert Cell Kernel Help

Markdown CellToolbar

```
print(frame.head(5))
```

Parse progress: |

[267, 2]

C1
bigdata/laptop/deepwater/imagenet/cat/102194502_49f003abd9.jpg
bigdata/laptop/deepwater/imagenet/cat/11146807_00a5f35255.jpg
bigdata/laptop/deepwater/imagenet/cat/1140846215_70e326f868.jpg
bigdata/laptop/deepwater/imagenet/cat/114170569_6cbdf4bbdb.jpg
bigdata/laptop/deepwater/imagenet/cat/1217664848_de4c7fc296.jpg

ubuntu@ip-10-164-48-74: ~

ubuntu@ip-10-164-48-74: ~ 78x9

```
Every 2.0s: gpustat -cp                          Wed Nov 30 09:37:01 2016

ip-10-164-48-74  Wed Nov 30 09:37:01 2016
[0] GRID K520         | 34'C,  76 % |  3806 /  4036 MB | java/1357(3804M)
```

Using GPU for training

ubuntu@ip-10-164-48-74: ~ 78x9

```
  1  [|||||                                  15.2%]   5  [|||||                                  10.8%]
  2  [|||||||||                              26.2%]   6  [|||||                                  11.0%]
  3  [|||||                                  14.0%]   7  [|||||                                  10.5%]
  4  [|||||                                  11.1%]   8  [|||||                                  14.5%]
  Mem[|||||||||||||||||9.09G/14.7G]   Tasks: 50, 122 thr; 2 running
  Swp[                          0K/0K]   Load average: 0.33 0.21 0.15
                                         Uptime: 06:54:06
```

To build a LeNet image classification model in H2O, simply specify `network = "lenet"` and the **Tensorflow** backend to use the tensorflow lenet implementation:

In [*]:

```
model = H2ODeepWaterEstimator(epochs      = 500,
                              network     = "lenet",
                              image_shape = [28,28],  ## provide image size
                              channels    = 3,
                              backend     = "tensorflow",
                              model_id    = "deepwater_tf_simple")

model.train(x = [0], # file path e.g. xxx/xxx/xxx.jpg
            y = 1, # label cat/dog/mouse
            training_frame = frame)

model.show()
```

deepwater Model Build progress: |

H2O.ai

To build a LeNet image classification model in H2O, simply specify `network = "lenet"` and the **Tensorflow** backend to use the tensorflow lenet implementation:

In [12]:
```
model = H2ODeepWaterEstimator(epochs      = 500,
                              network     = "lenet",
                              image_shape = [28,28],  ## provide image size
                              channels    = 3,
                              backend     = "tensorflow",
                              model_id    = "deepwater_tf_simple")

model.train(x = [0], # file path e.g. xxx/xxx/xxx.jpg
            y = 1, # label cat/dog/mouse
            training_frame = frame)

model.show()
```

```
deepwater Model Build progress: |████████████████████████████████████████| 100%
Model Details
=============
H2ODeepWaterEstimator :  Deep Water
Model Key:  deepwater_tf_simple


ModelMetricsMultinomial: deepwater
** Reported on train data. **

MSE: 0.177650603738
RMSE: 0.421486184517
LogLoss: 0.865899719937
Mean Per-Class Error: 0.217708629345
Confusion Matrix: vertical: actual; across: predicted
```

cat	dog	mouse	Error	Rate
85.0	2.0	3.0	0.0555556	5 / 90
18.0	61.0	6.0	0.2823529	24 / 85
27.0	2.0	63.0	0.3152174	29 / 92
130.0	65.0	72.0	0.2172285	58 / 267

Deep Water – Custom Network

If you'd like to build your own Tensorflow network architecture, then this is easy as well. In this example script, we are using the **Tensorflow** backend. Models can easily be imported/exported between H2O and Tensorflow since H2O uses Tensorflow's format for model definition.

In [8]:
```
def simple_model(w, h, channels, classes):
    import json
    import tensorflow as tf
    # always create a new graph inside ipython or
    # the default one will be used and can lead to
    # unexpected behavior
    graph = tf.Graph()
    with graph.as_default():
        size = w * h * channels
        x = tf.placeholder(tf.float32, [None, size])
        W = tf.Variable(tf.zeros([size, classes]))
        b = tf.Variable(tf.zeros([classes]))
        y = tf.matmul(x, W) + b

        # labels
        y_ = tf.placeholder(tf.float32, [None, classes])

        # accuracy
        correct_prediction = tf.equal(tf.argmax(y, 1),
                                      tf.argmax(y_, 1))
        accuracy = tf.reduce_mean(tf.cast(correct_prediction, tf.float32))

        # train
        cross_entropy = tf.reduce_mean(tf.nn.softmax_cross_entropy_with_logits(y, y_))
        train_step = tf.train.GradientDescentOptimizer(0.5).minimize(cross_entropy)

        tf.add_to_collection("train", train_step)
        # this is required by the h2o tensorflow backend
        global_step = tf.Variable(0, name="global_step", trainable=False)

        init = tf.initialize_all_variables()
        tf.add_to_collection("init", init)
        tf.add_to_collection("logits", y)
        saver = tf.train.Saver()
        meta = json.dumps({
                "inputs": {"batch_image_input": x.name, "categorical_labels": y_.name},
                "outputs": {"categorical_logits": y.name},
                "metrics": {"accuracy": accuracy.name, "total_loss": cross_entropy.name},
                "parameters": {"global_step": global_step.name},
        })
        print(meta)
        tf.add_to_collection("meta", meta)
        filename = "/tmp/lenet_tensorflow.meta"
        tf.train.export_meta_graph(filename, saver_def=saver.as_saver_def())
    return filename
```

Saving the custom network structure as a file

H2O.ai

```
In [9]: filename = simple_model(28, 28, 3, classes=3)
```

```
{"metrics": {"total_loss": "Mean_1:0", "accuracy": "Mean:0"}, "inputs": {"categorical_labels": "Placeholder_1:0",
 "batch_image_input": "Placeholder:0"}, "parameters": {"global_step": "global_step:0"}, "outputs": {"categorical_lo
gits": "add:0"}}
```

Creating the custom network structure with size = 28x28 and channels = 3

In [13]:
```
model = H2ODeepWaterEstimator(epochs                 = 500,
                              network_definition_file = filename, ## specify the model
                              image_shape             = [28,28],  ## provide expected image size
                              channels                = 3,
                              backend                 = "tensorflow",
                              model_id                = "deepwater_tf_custom")

model.train(x = [0], # file path e.g. xxx/xxx/xxx.jpg
            y = 1, # label cat/dog/mouse
            training_frame = frame)

model.show()
```

Specifying the custom network structure for training

```
deepwater Model Build progress: |████████████████████████████████████████| 100%
Model Details
=============
H2ODeepWaterEstimator :  Deep Water
Model Key:  deepwater_tf_custom


ModelMetricsMultinomial: deepwater
** Reported on train data. **

MSE: 6.60075876885e+12
RMSE: 2569194.18668
LogLoss: -14.4921790248
Mean Per-Class Error: 0.0
Confusion Matrix: vertical: actual; across: predicted
```

cat	dog	mouse	Error	Rate
90.0	0.0	0.0	0.0	0 / 90
0.0	85.0	0.0	0.0	0 / 85
0.0	0.0	92.0	0.0	0 / 92
90.0	85.0	92.0	0.0	0 / 267

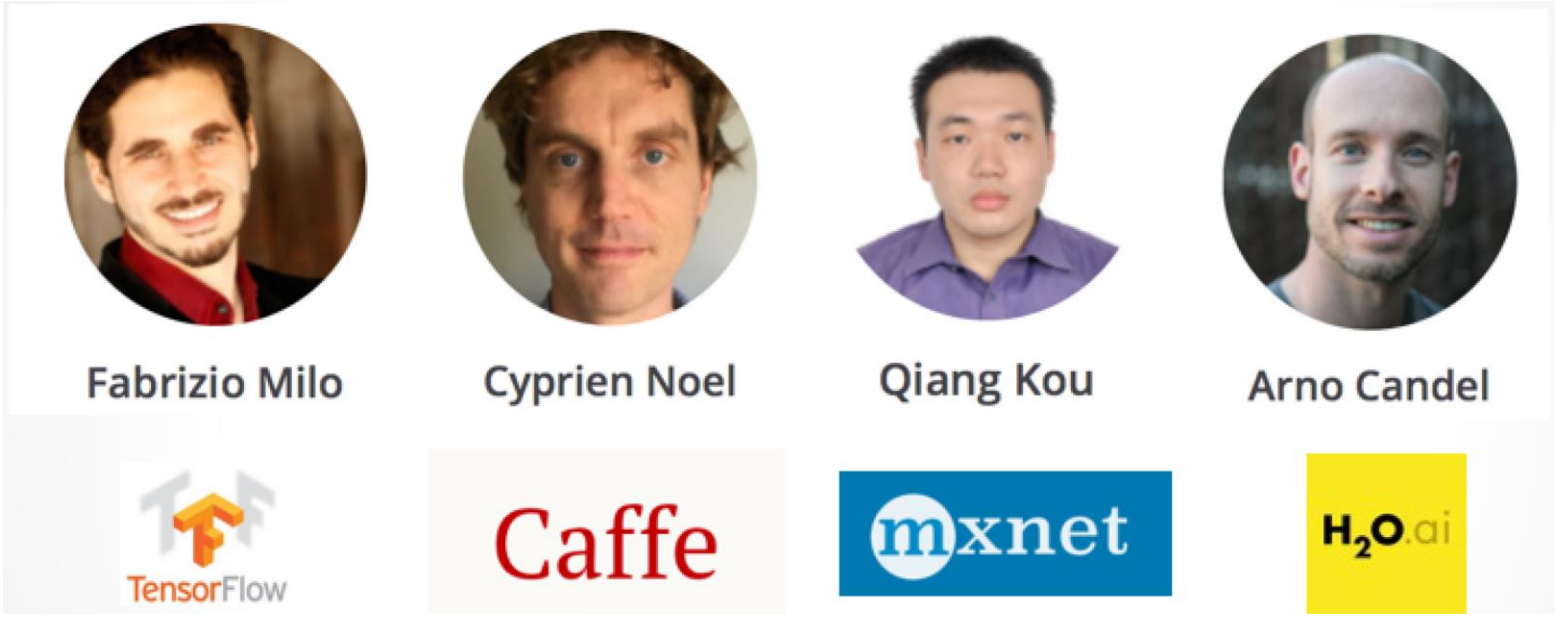
Conclusions

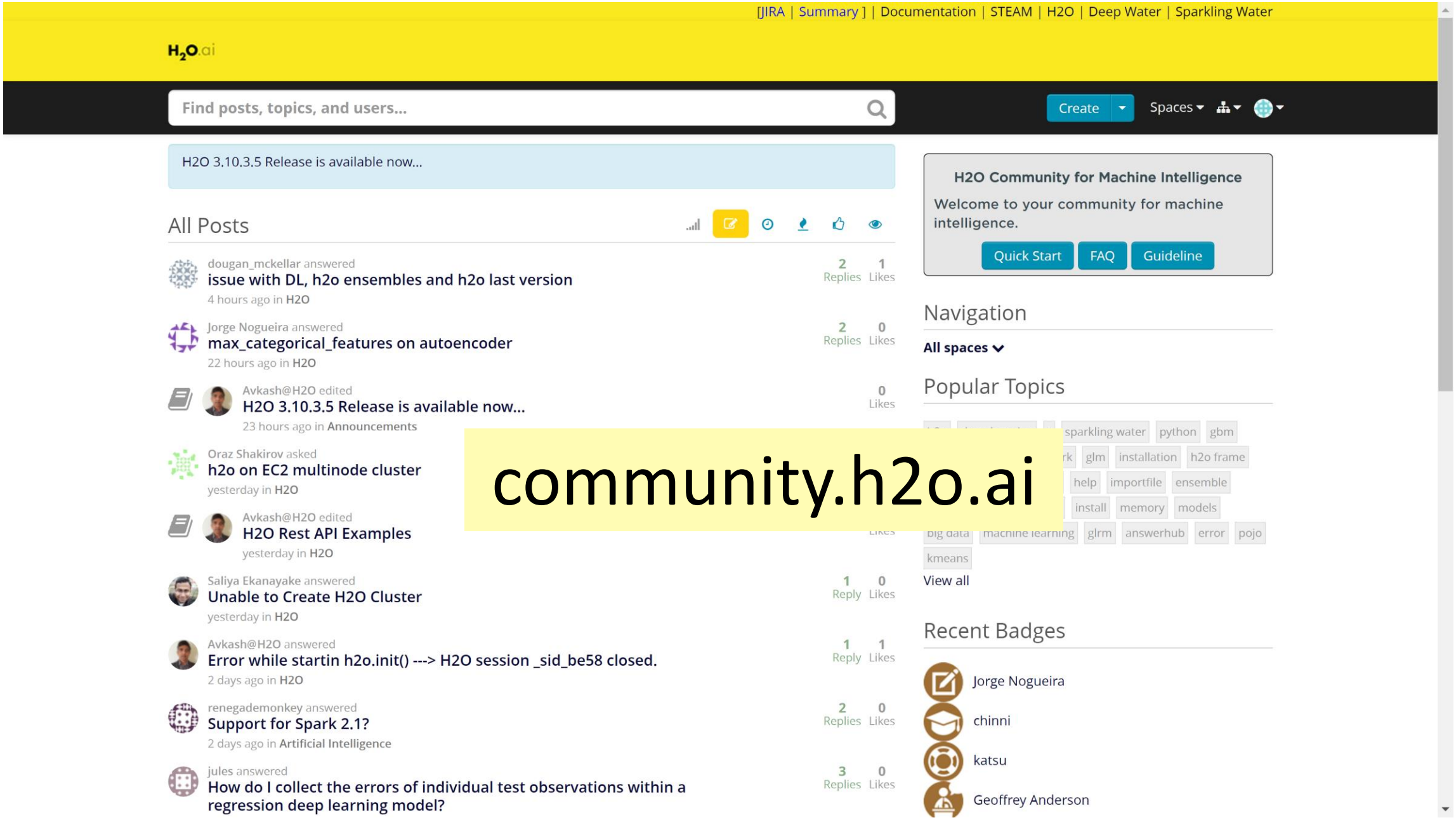
Project “Deep Water”

- H_2O + TF + MXNet + Caffe
  - a powerful combination of widely used open source machine learning libraries.
- All Goodies from H_2O
  - inherits all H_2O properties in scalability, ease of use and deployment.
- Unified Interface
  - allows users to build, stack and deploy deep learning models from different libraries efficiently.

- 100% Open Source
  - the party will get bigger!

Deep Water – Current Contributors





community.h2o.ai

H2O.ai

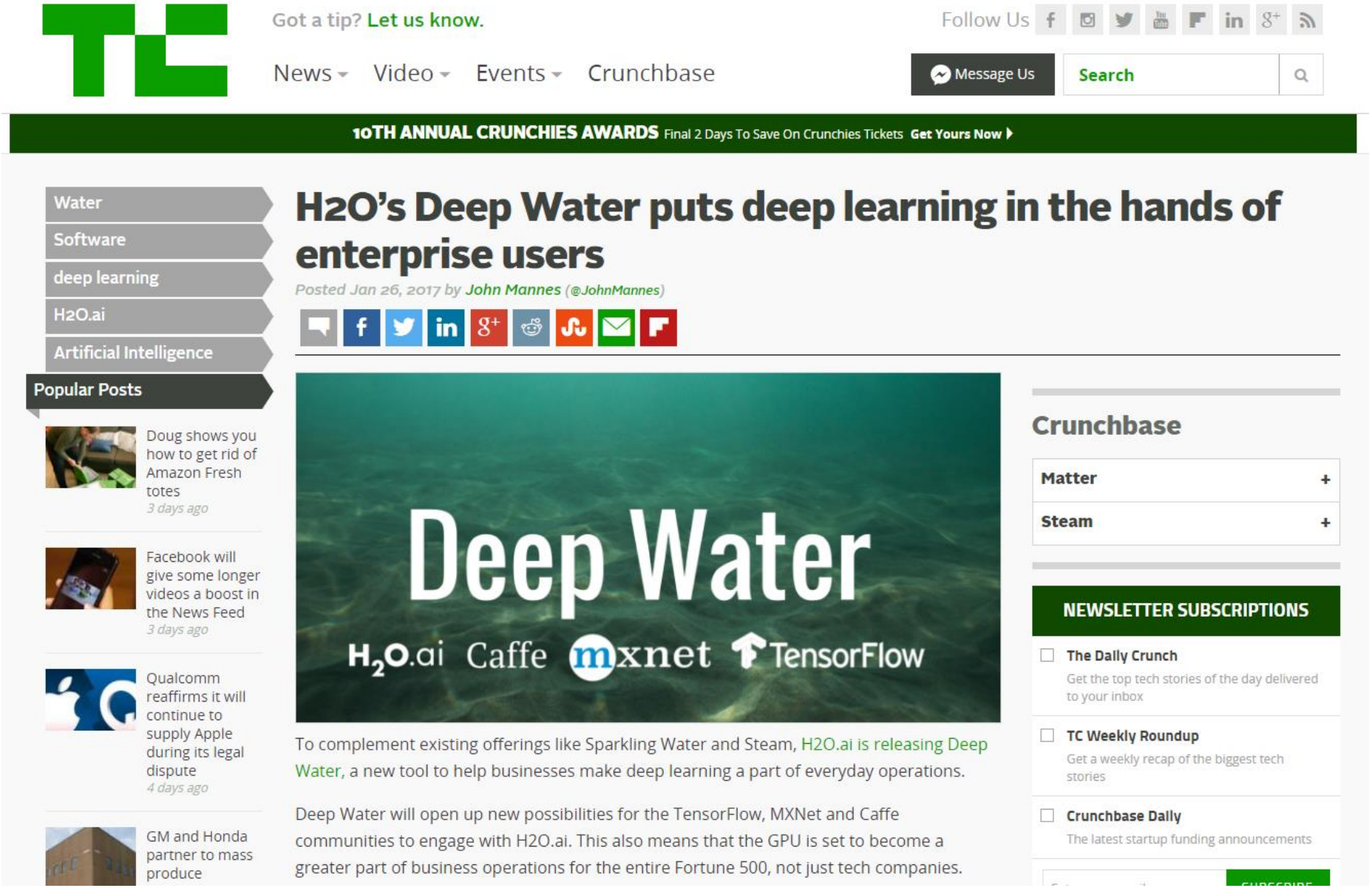
Thanks!

- Organizers & Sponsors
  - Metis & H2O.ai
- Code, Slides & Documents
  - bit.ly/h2o_meetups
  - bit.ly/h2o_deepwater
  - docs.h2o.ai
- Contact
  - joe@h2o.ai
  - @matlabulous
  - github.com/woobe

H2O.ai

Making Machine Learning Accessible to Everyone

Photo credit: Virgin Media



https://techcrunch.com/2017/01/26/h2os-deep-water-puts-deep-learning-in-the-hands-of-enterprise-users/

H2O.ai