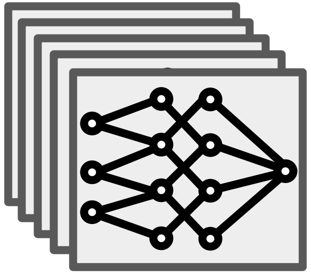
HYPR.AI

Cloud-based AutoML

Ryan Reece
Insight AI Fellow

Also with Jiangming “Jimmy” Hu
Previous Insight Fellow

Intro

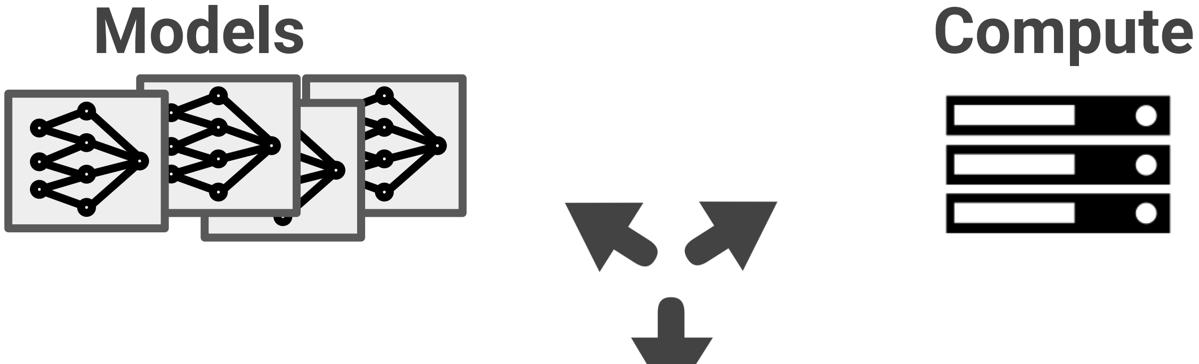


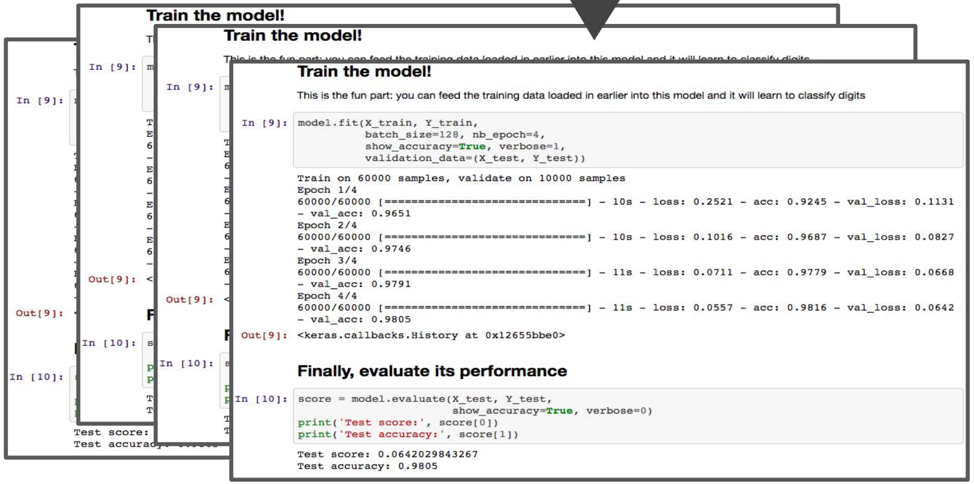
- AI is an *experimental* science
- Search & test many models: *“hyperparameters”*
- Better tools can streamline this process
- Leads to discovering and organizing better models
- Jeff Dean: tools for “ML 2.0” or “AutoML”

Google Cloud Platform

H_2O.ai

Typical ML workflow today





Train the model!

This is the fun part: you can feed the training data loaded in earlier into this model and it will learn to classify digits

```
In [9]:  model.fit(X_train, Y_train,
                   batch_size=128, nb_epoch=4,
                   show_accuracy=True, verbose=1,
                   validation_data=(X_test, Y_test))

         Train on 60000 samples, validate on 10000 samples
         Epoch 1/4
         60000/60000 [==============================] - 10s - loss: 0.2521 - acc: 0.9245 - val_loss: 0.1131
         - val_acc: 0.9651
         Epoch 2/4
         60000/60000 [==============================] - 10s - loss: 0.1016 - acc: 0.9687 - val_loss: 0.0827
         - val_acc: 0.9746
         Epoch 3/4
         60000/60000 [==============================] - 11s - loss: 0.0711 - acc: 0.9779 - val_loss: 0.0668
         - val_acc: 0.9791
         Epoch 4/4
         60000/60000 [==============================] - 11s - loss: 0.0557 - acc: 0.9816 - val_loss: 0.0642
         - val_acc: 0.9805

Out[9]:  <keras.callbacks.History at 0x12655bbe0>
```

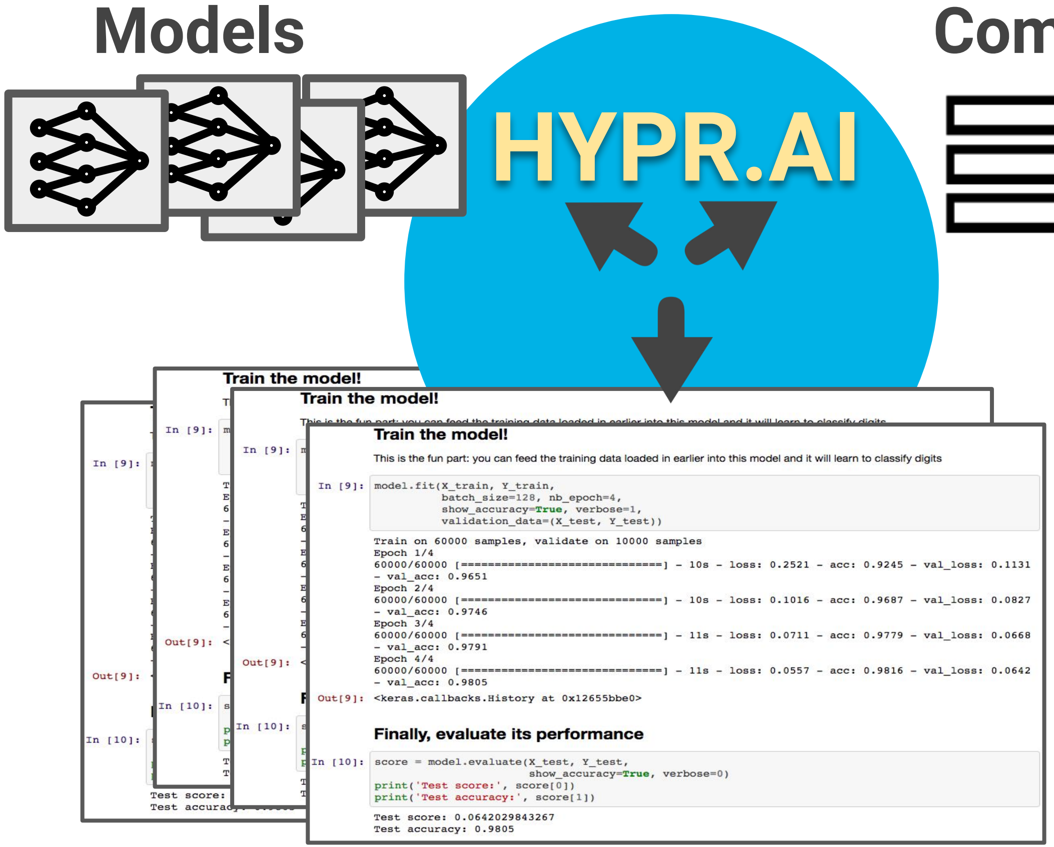
Finally, evaluate its performance

```
In [10]: score = model.evaluate(X_test, Y_test,
                                show_accuracy=True, verbose=0)
         print('Test score:', score[0])
         print('Test accuracy:', score[1])

         Test score: 0.0642029843267
         Test accuracy: 0.9805
```

- **Workflow**
- **Visualization**
- **Iterate**
- **Bookkeeping?**

Typical ML workflow today



Train the model!

This is the fun part: you can feed the training data loaded in earlier into this model and it will learn to classify digits

```
In [9]: model.fit(X_train, Y_train,
                  batch_size=128, nb_epoch=4,
                  show_accuracy=True, verbose=1,
                  validation_data=(X_test, Y_test))

Train on 60000 samples, validate on 10000 samples
Epoch 1/4
60000/60000 [==============================] - 10s - loss: 0.2521 - acc: 0.9245 - val_loss: 0.1131
- val_acc: 0.9651
Epoch 2/4
60000/60000 [==============================] - 10s - loss: 0.1016 - acc: 0.9687 - val_loss: 0.0827
- val_acc: 0.9746
Epoch 3/4
60000/60000 [==============================] - 11s - loss: 0.0711 - acc: 0.9779 - val_loss: 0.0668
- val_acc: 0.9791
Epoch 4/4
60000/60000 [==============================] - 11s - loss: 0.0557 - acc: 0.9816 - val_loss: 0.0642
- val_acc: 0.9805

Out[9]: <keras.callbacks.History at 0x12655bbe0>
```

Finally, evaluate its performance

```
In [10]: score = model.evaluate(X_test, Y_test,
                                show_accuracy=True, verbose=0)
         print('Test score:', score[0])
         print('Test accuracy:', score[1])

Test score: 0.0642029843267
Test accuracy: 0.9805
```

- **Workflow**
- **Visualization**
- **Iterate**
- **Bookkeeping?**

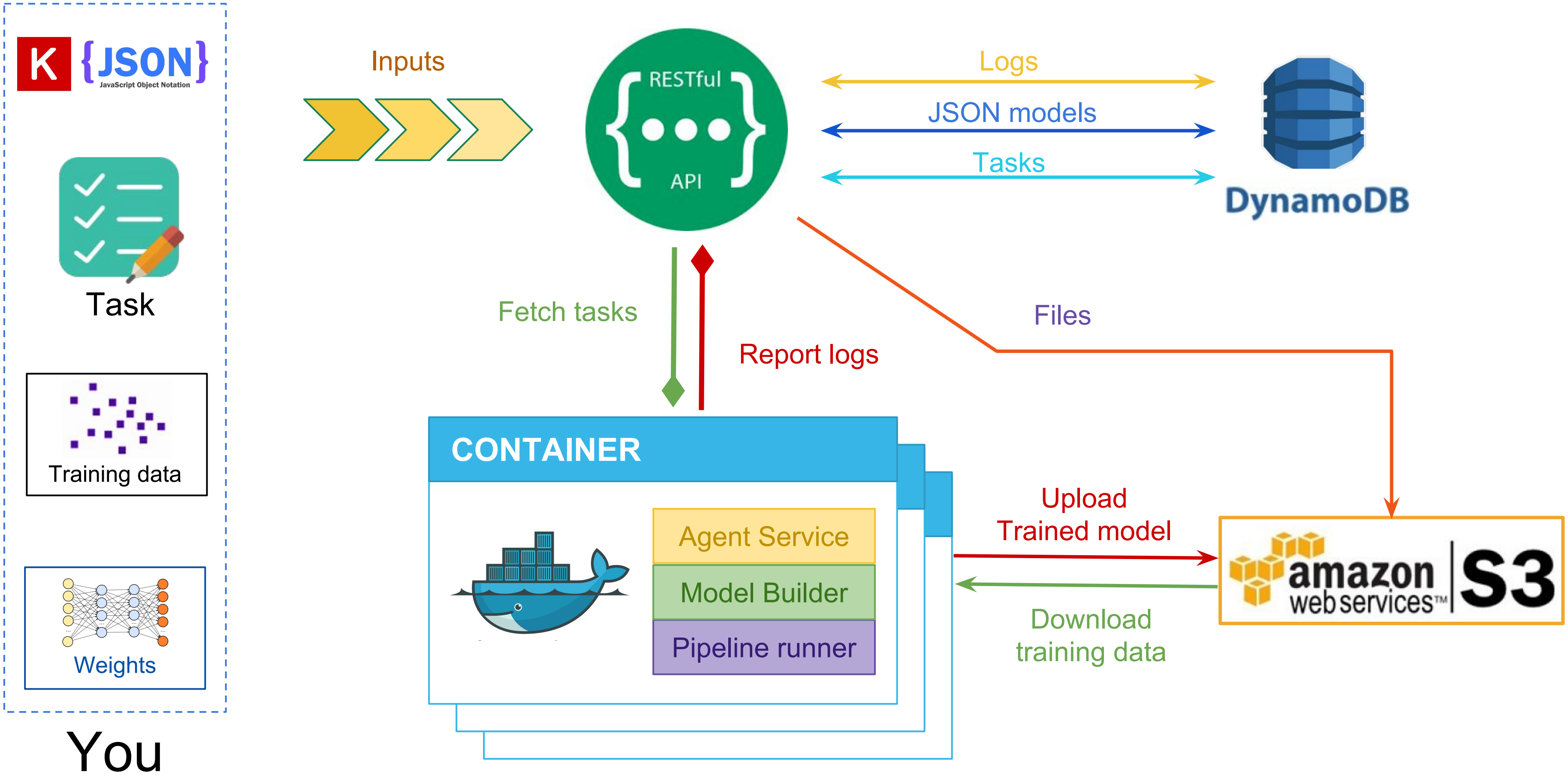
HYPR.AI provides

Website UI: construct or upload keras models

Database for storing models and results

Task queue for training many models

Compute back-ends for AWS & Paperspace



K {JSON} JavaScript Object Notation

Task

Training data

Weights

You

Inputs

RESTful API

Logs

JSON models

Tasks

DynamoDB

Fetch tasks

Report logs

Files

CONTAINER

Agent Service

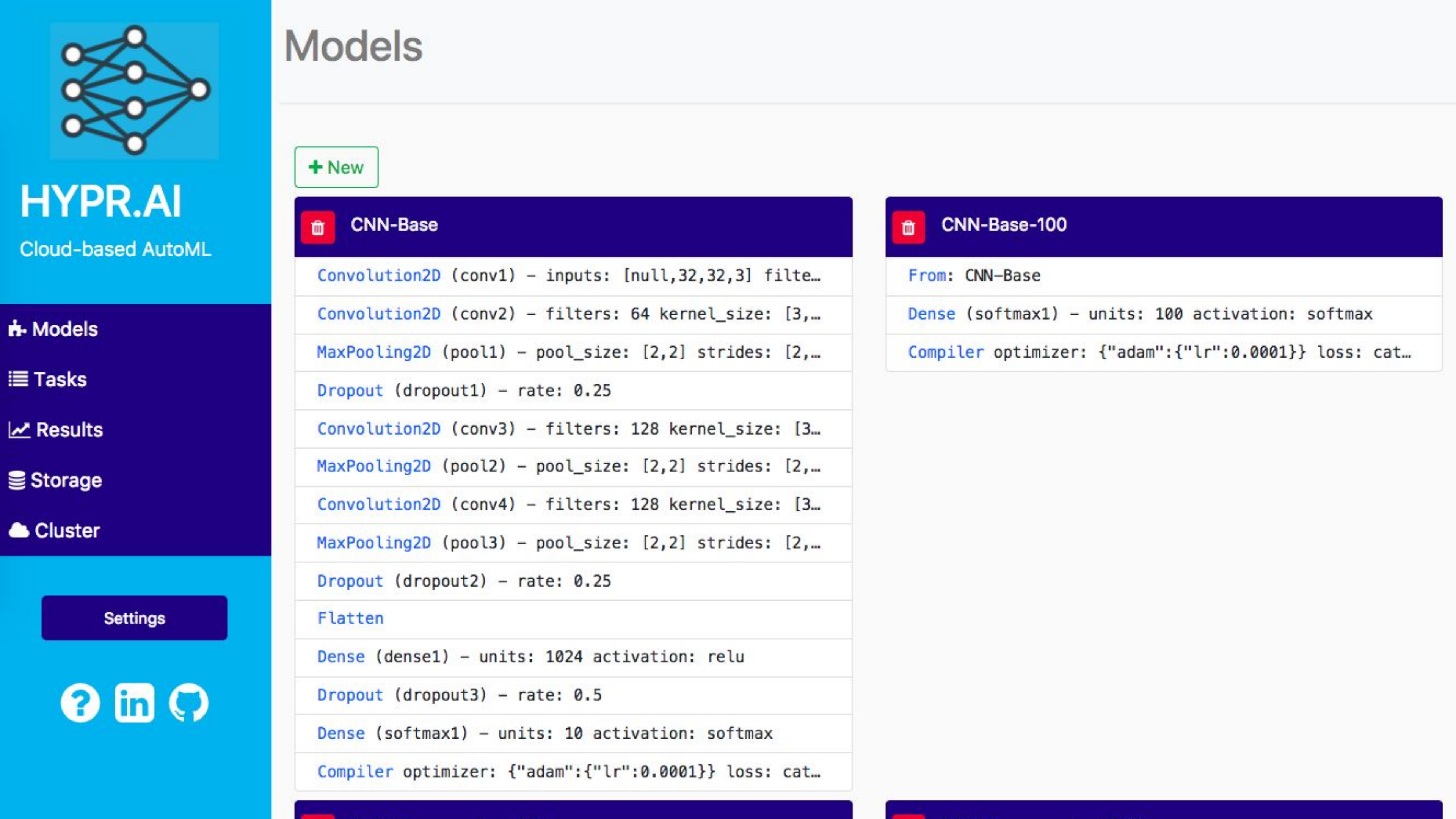
Model Builder

Pipeline runner

Upload
Trained model

Download
training data

amazon web services™ S3



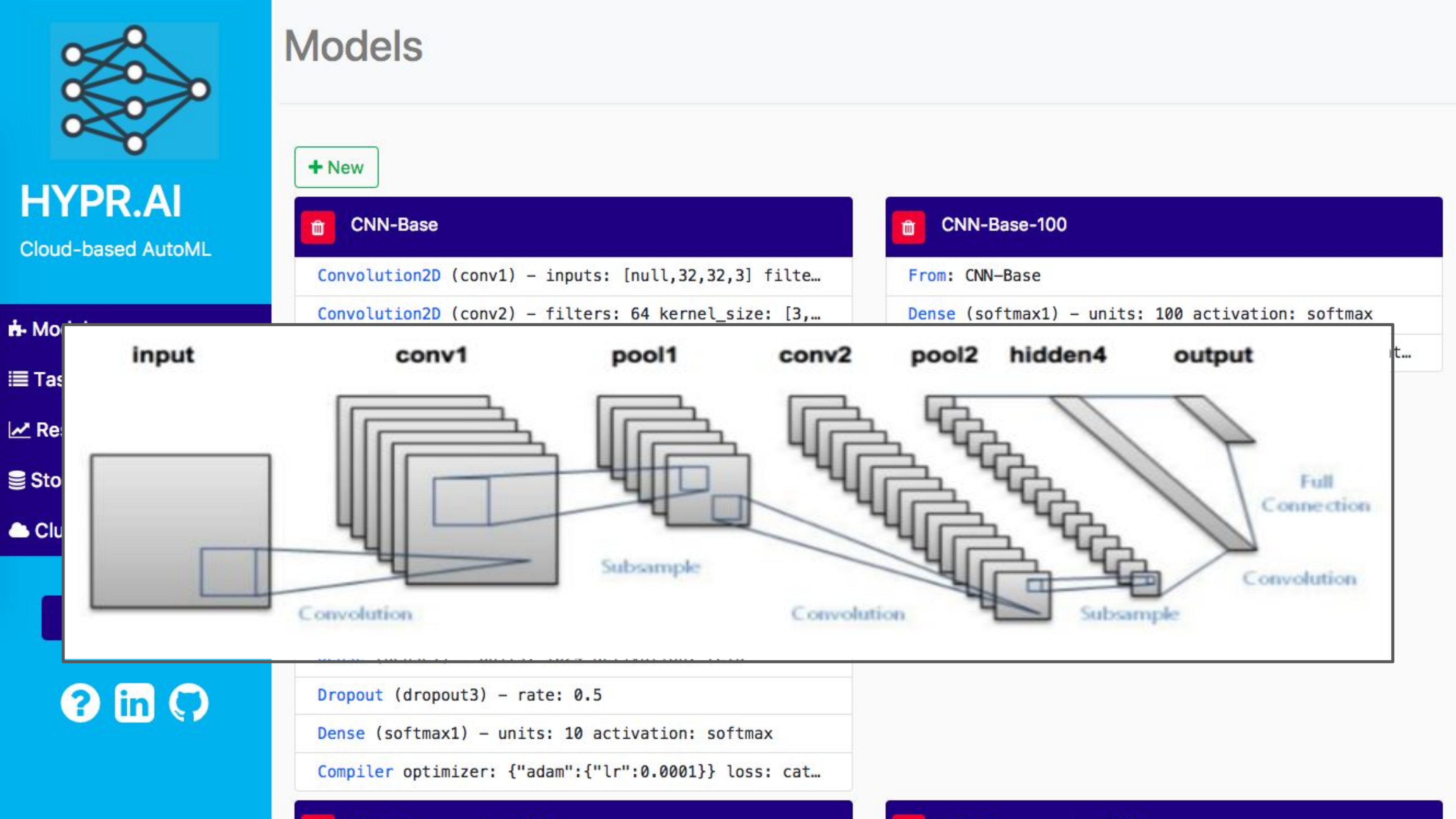
Models

CNN-Base

```
Convolution2D (conv1) - inputs: [null,32,32,3] filte…
Convolution2D (conv2) - filters: 64 kernel_size: [3,…
MaxPooling2D (pool1) - pool_size: [2,2] strides: [2,…
Dropout (dropout1) - rate: 0.25
Convolution2D (conv3) - filters: 128 kernel_size: [3…
MaxPooling2D (pool2) - pool_size: [2,2] strides: [2,…
Convolution2D (conv4) - filters: 128 kernel_size: [3…
MaxPooling2D (pool3) - pool_size: [2,2] strides: [2,…
Dropout (dropout2) - rate: 0.25
Flatten
Dense (dense1) - units: 1024 activation: relu
Dropout (dropout3) - rate: 0.5
Dense (softmax1) - units: 10 activation: softmax
Compiler optimizer: {"adam":{"lr":0.0001}} loss: cat…
```

CNN-Base-100

```
From: CNN-Base
Dense (softmax1) - units: 100 activation: softmax
Compiler optimizer: {"adam":{"lr":0.0001}} loss: cat…
```



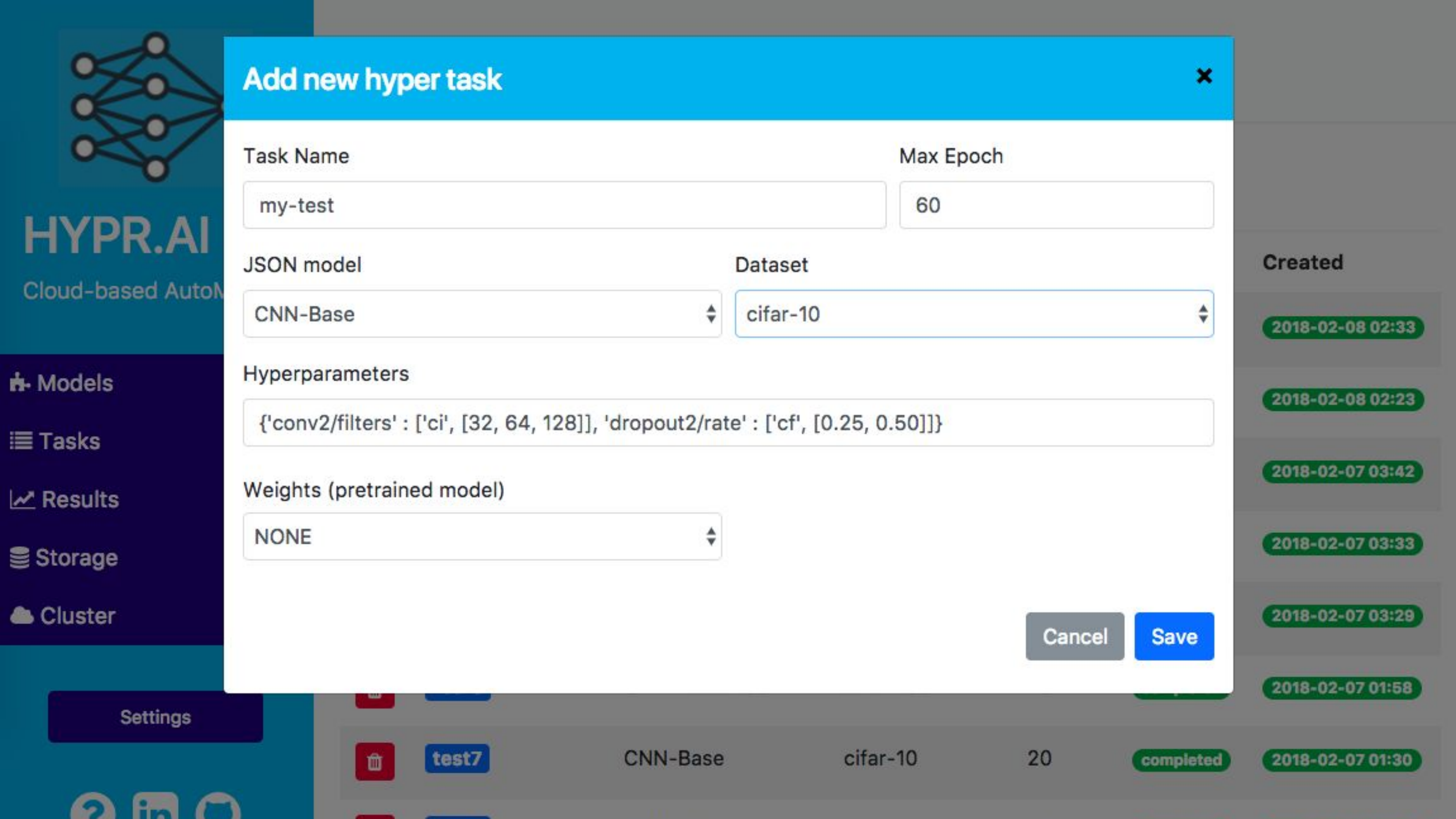
Models

CNN-Base

```
Convolution2D (conv1) - inputs: [null,32,32,3] filte…
Convolution2D (conv2) - filters: 64 kernel_size: [3,…
Dropout (dropout3) - rate: 0.5
Dense (softmax1) - units: 10 activation: softmax
Compiler optimizer: {"adam":{"lr":0.0001}} loss: cat…
```

CNN-Base-100

```
From: CNN-Base
Dense (softmax1) - units: 100 activation: softmax
t…
```



Add new hyper task

Task Name: my-test

Max Epoch: 60

JSON model: CNN-Base

Dataset: cifar-10

Hyperparameters: {'conv2/filters' : ['ci', [32, 64, 128]], 'dropout2/rate' : ['cf', [0.25, 0.50]]}

Weights (pretrained model): NONE

Created

2018-02-08 02:33

2018-02-08 02:23

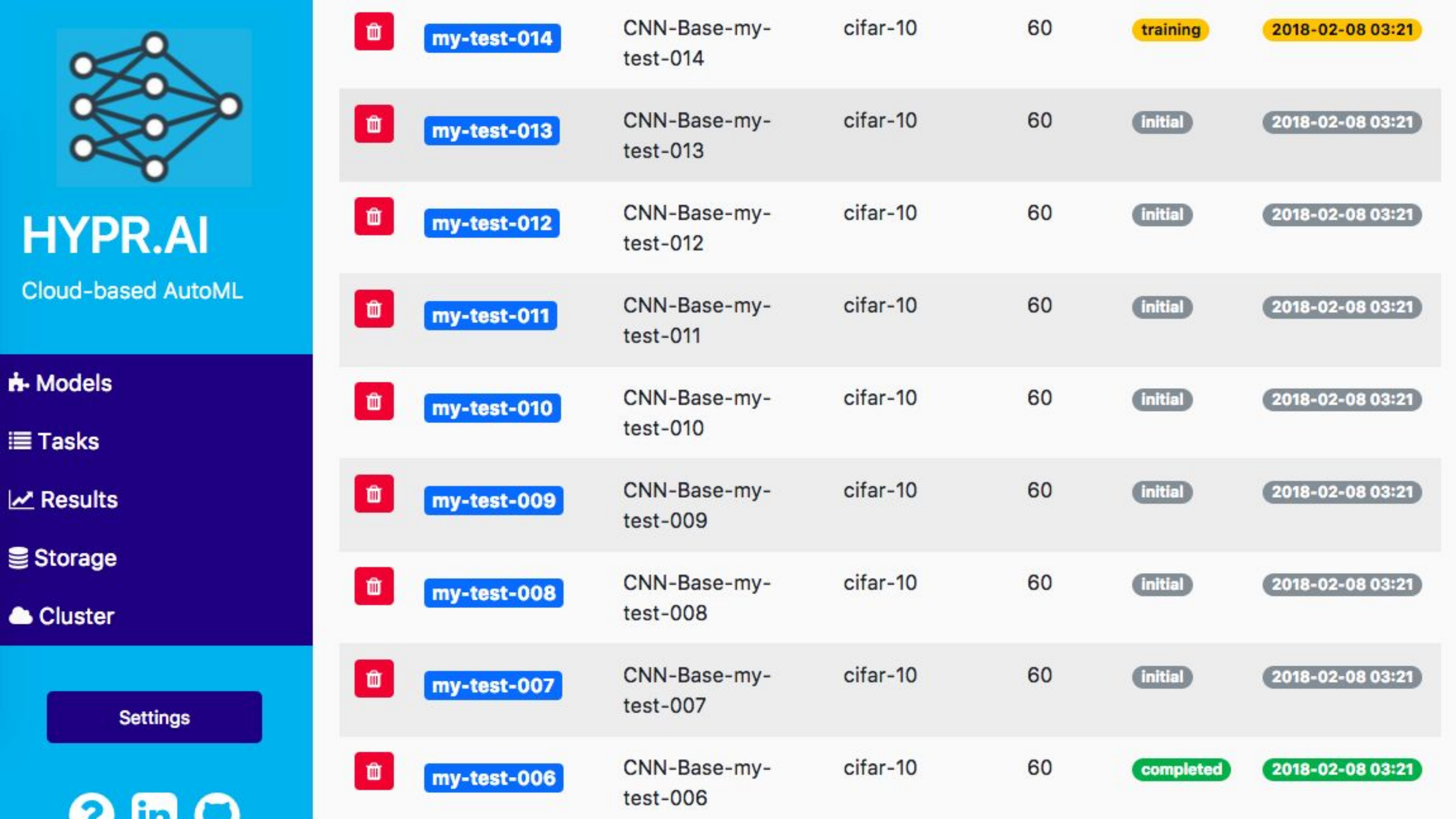
2018-02-07 03:42

2018-02-07 03:33

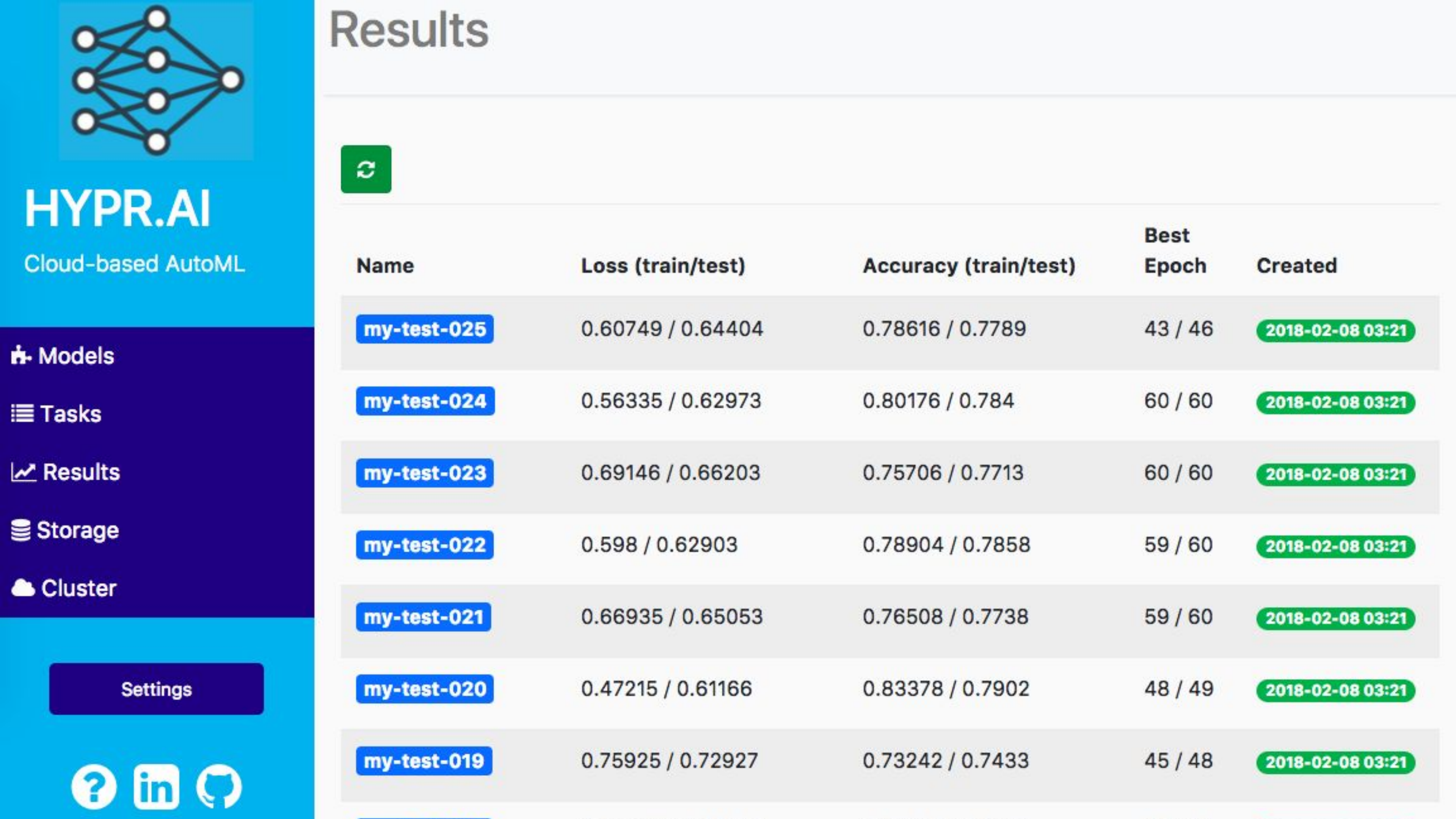
2018-02-07 03:29

2018-02-07 01:58

test7 CNN-Base cifar-10 20 completed 2018-02-07 01:30

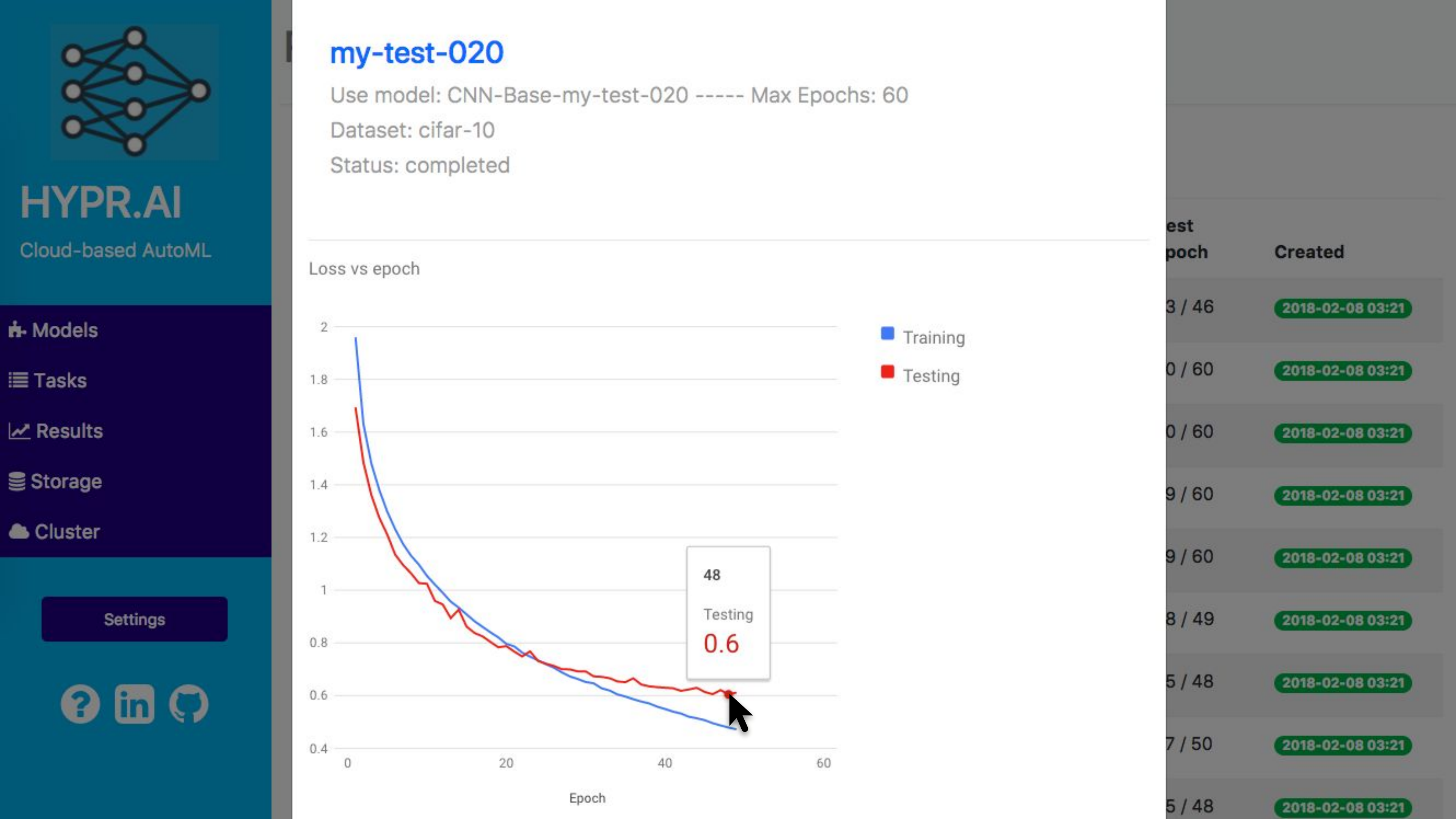


🗑	my-test-014	CNN-Base-my-test-014	cifar-10	60	training	2018-02-08 03:21
🗑	my-test-013	CNN-Base-my-test-013	cifar-10	60	initial	2018-02-08 03:21
🗑	my-test-012	CNN-Base-my-test-012	cifar-10	60	initial	2018-02-08 03:21
🗑	my-test-011	CNN-Base-my-test-011	cifar-10	60	initial	2018-02-08 03:21
🗑	my-test-010	CNN-Base-my-test-010	cifar-10	60	initial	2018-02-08 03:21
🗑	my-test-009	CNN-Base-my-test-009	cifar-10	60	initial	2018-02-08 03:21
🗑	my-test-008	CNN-Base-my-test-008	cifar-10	60	initial	2018-02-08 03:21
🗑	my-test-007	CNN-Base-my-test-007	cifar-10	60	initial	2018-02-08 03:21
🗑	my-test-006	CNN-Base-my-test-006	cifar-10	60	completed	2018-02-08 03:21



Results

Name	Loss (train/test)	Accuracy (train/test)	Best Epoch	Created
my-test-025	0.60749 / 0.64404	0.78616 / 0.7789	43 / 46	2018-02-08 03:21
my-test-024	0.56335 / 0.62973	0.80176 / 0.784	60 / 60	2018-02-08 03:21
my-test-023	0.69146 / 0.66203	0.75706 / 0.7713	60 / 60	2018-02-08 03:21
my-test-022	0.598 / 0.62903	0.78904 / 0.7858	59 / 60	2018-02-08 03:21
my-test-021	0.66935 / 0.65053	0.76508 / 0.7738	59 / 60	2018-02-08 03:21
my-test-020	0.47215 / 0.61166	0.83378 / 0.7902	48 / 49	2018-02-08 03:21
my-test-019	0.75925 / 0.72927	0.73242 / 0.7433	45 / 48	2018-02-08 03:21



my-test-020

Use model: CNN-Base-my-test-020 ----- Max Epochs: 60

Dataset: cifar-10

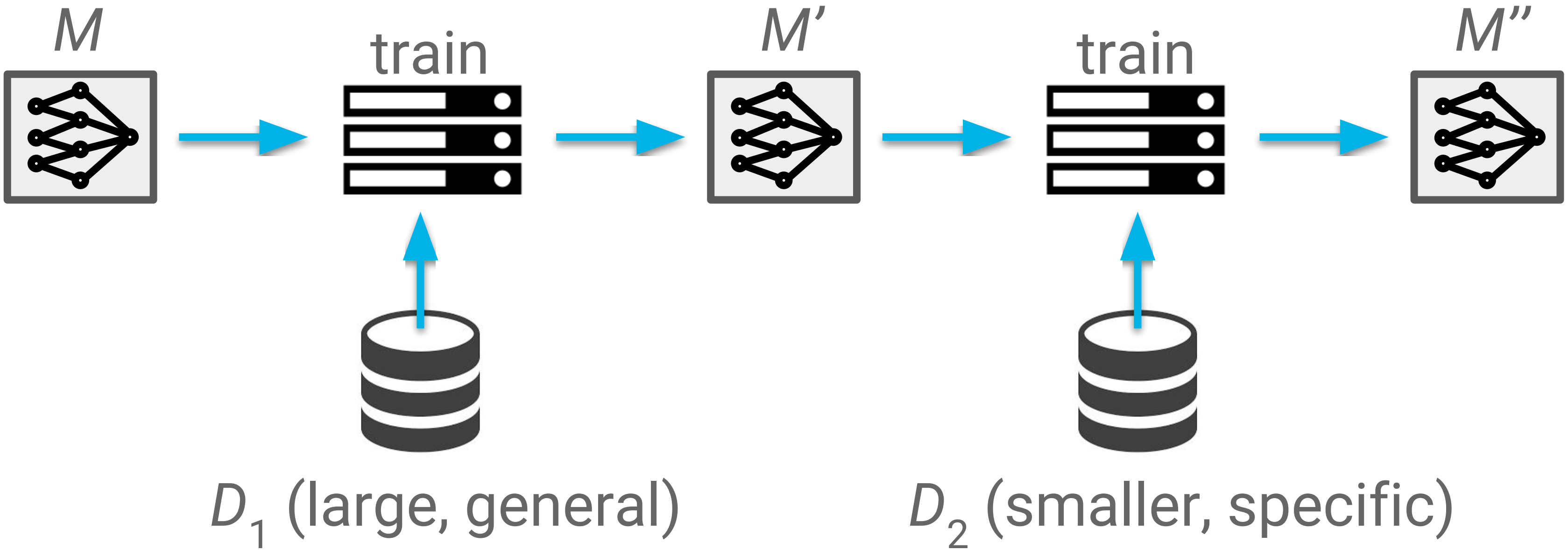
Status: completed

Loss vs epoch

est poch	Created
3 / 46	2018-02-08 03:21
0 / 60	2018-02-08 03:21
0 / 60	2018-02-08 03:21
9 / 60	2018-02-08 03:21
9 / 60	2018-02-08 03:21
8 / 49	2018-02-08 03:21
5 / 48	2018-02-08 03:21
7 / 50	2018-02-08 03:21
5 / 48	2018-02-08 03:21

Transfer learning

Train a model on one dataset, and then more on another.

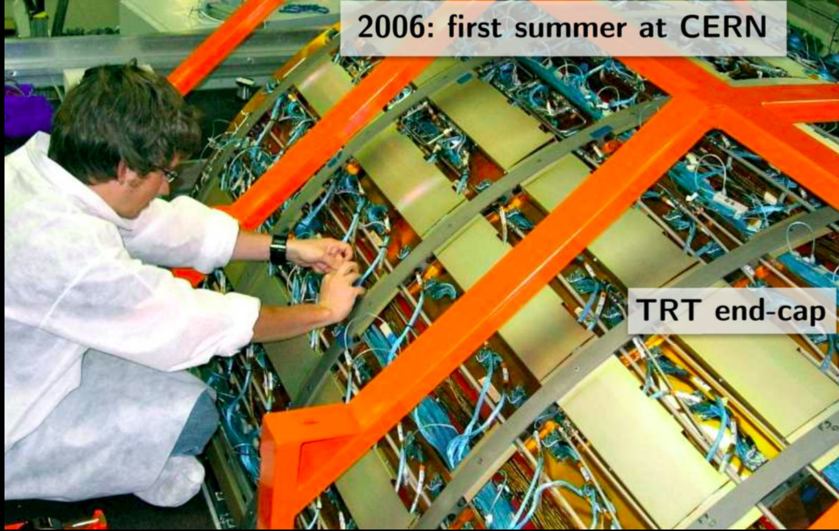


Advantages of using HYPR.AI

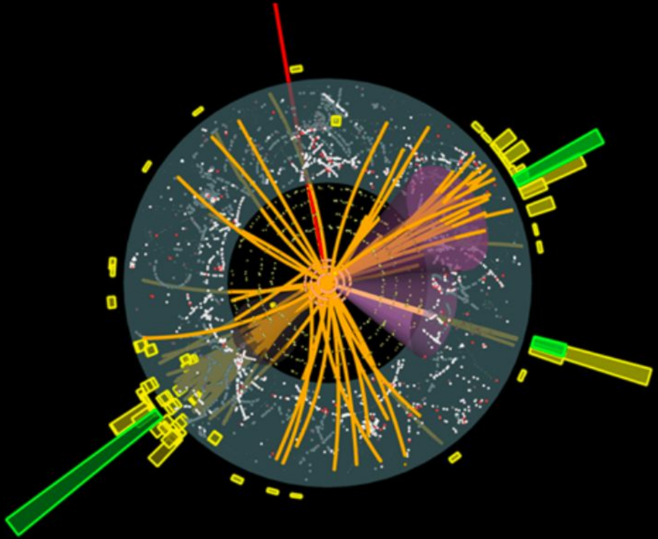
- Bookkeeping datasets/models/weights in the cloud
- RESTful API allows back-end independence
- Modular models facilitate transfer learning
- Hyperparameter optimization: currently random search or scan. Possible future plugins: e.g. Bayesian (SigOpt), Hyperopt
- Try out the running example: http://hypr.umx.io

Ryan Reece, Ph.D.
Data scientist, AI/ML/Stats

Previously a particle physicist with the ATLAS experiment at the Large Hadron Collider

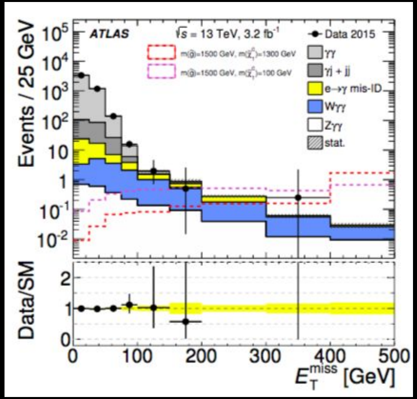


electronics



CERN

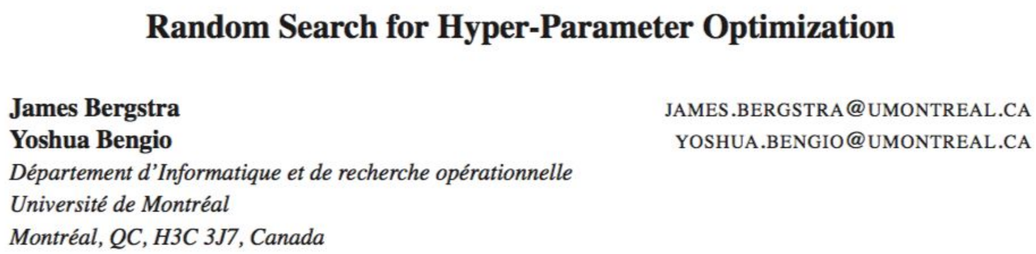
statistics



climbing

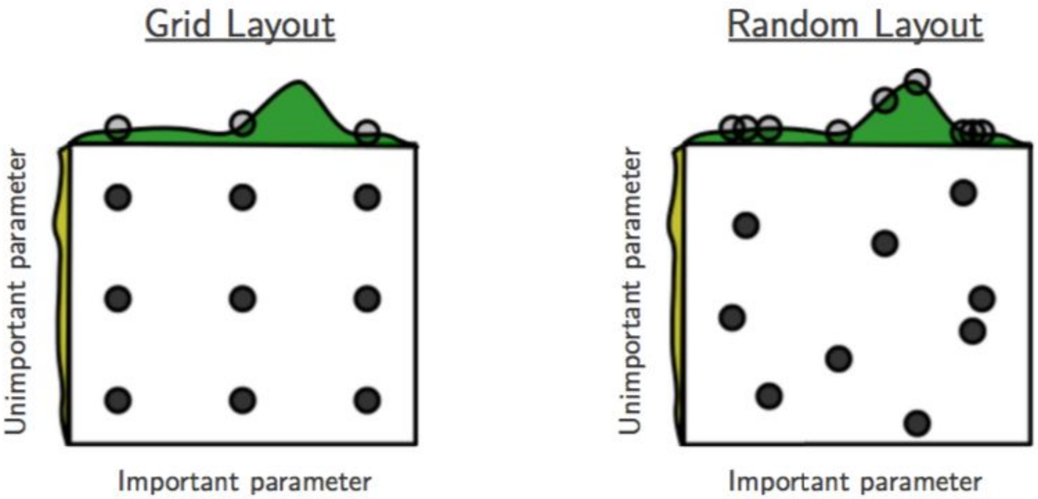
Backup slides

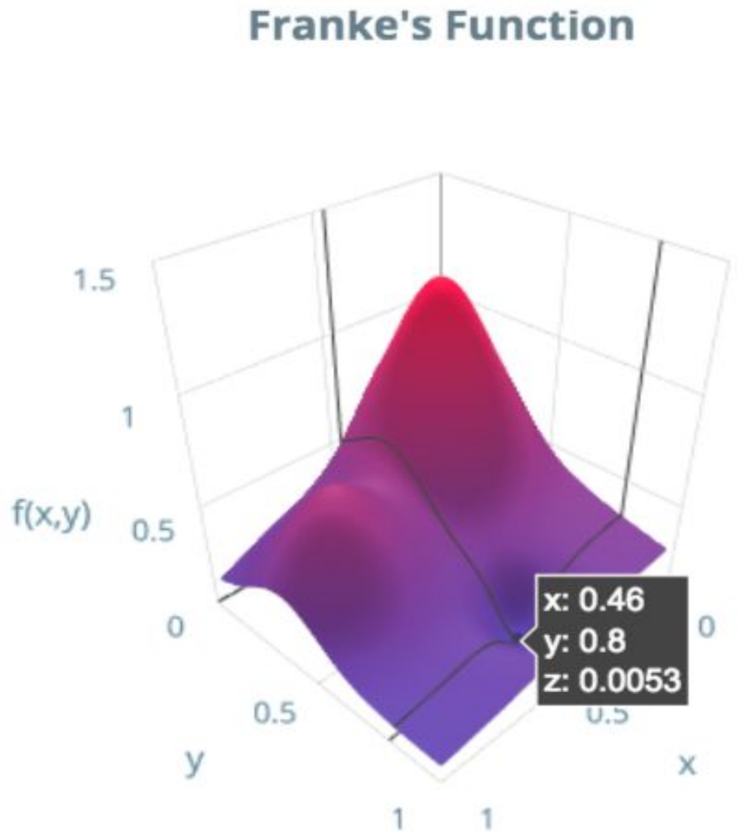
HYPR.AI



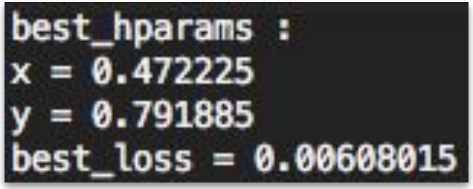
Random Search for Hyper-Parameter Optimization

James Bergstra JAMES.BERGSTRA@UMONTREAL.CA
Yoshua Bengio YOSHUA.BENGIO@UMONTREAL.CA
Département d'Informatique et de recherche opérationnelle
Université de Montréal
Montréal, QC, H3C 3J7, Canada

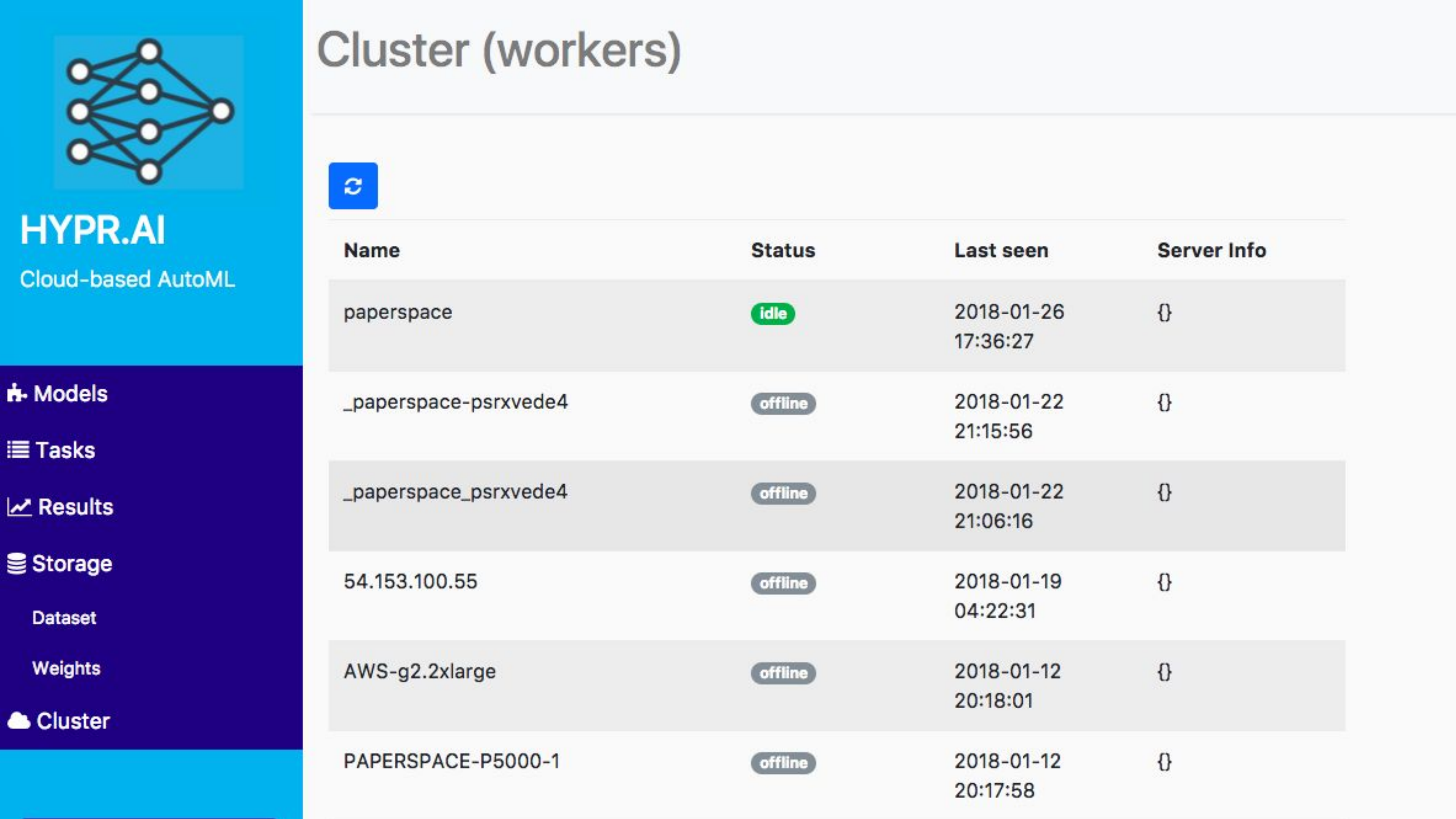




After 100 iterations of my random search alg:



```
best_hparams :
x = 0.472225
y = 0.791885
best_loss = 0.00608015
```

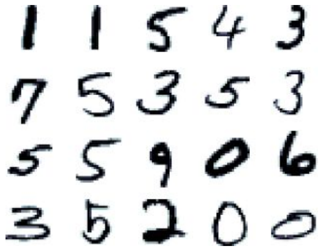


Cluster (workers)

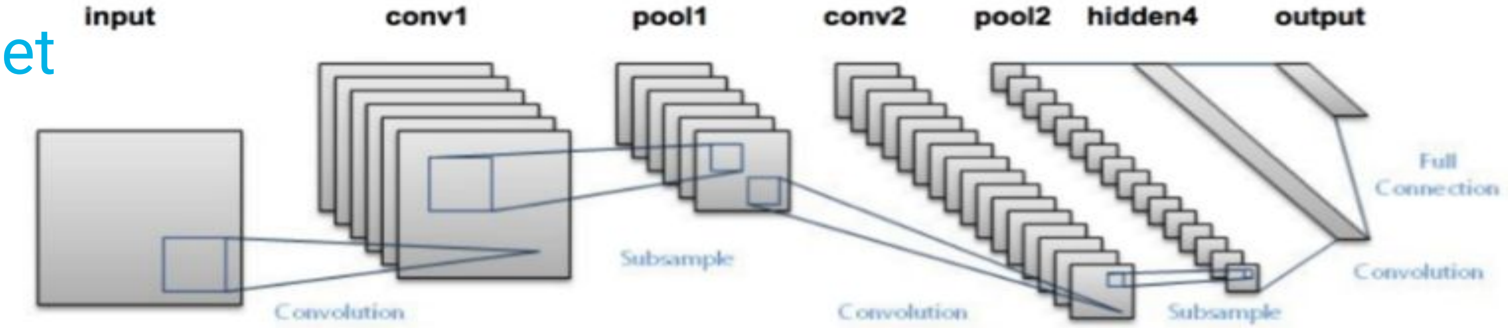
Name	Status	Last seen	Server Info
paperspace	idle	2018-01-26 17:36:27	{}
_paperspace-psrxvede4	offline	2018-01-22 21:15:56	{}
_paperspace_psrxvede4	offline	2018-01-22 21:06:16	{}
54.153.100.55	offline	2018-01-19 04:22:31	{}
AWS-g2.2xlarge	offline	2018-01-12 20:18:01	{}
PAPERSPACE-P5000-1	offline	2018-01-12 20:17:58	{}

Data and models

- As proofs of concept, using standard image classification datasets: CIFAR-10, MNIST



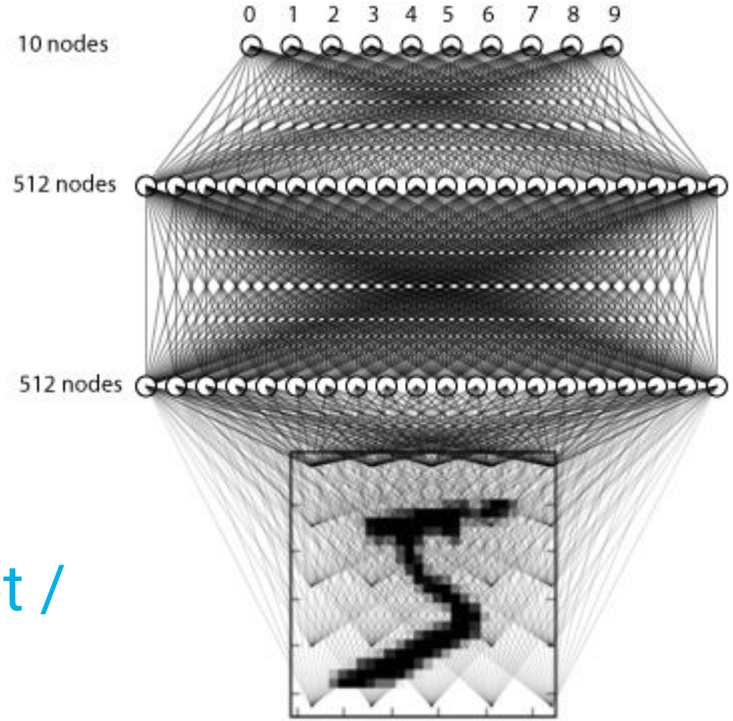
- Successful models use several layers of CNNs + pooling, *e.g.* LeNet



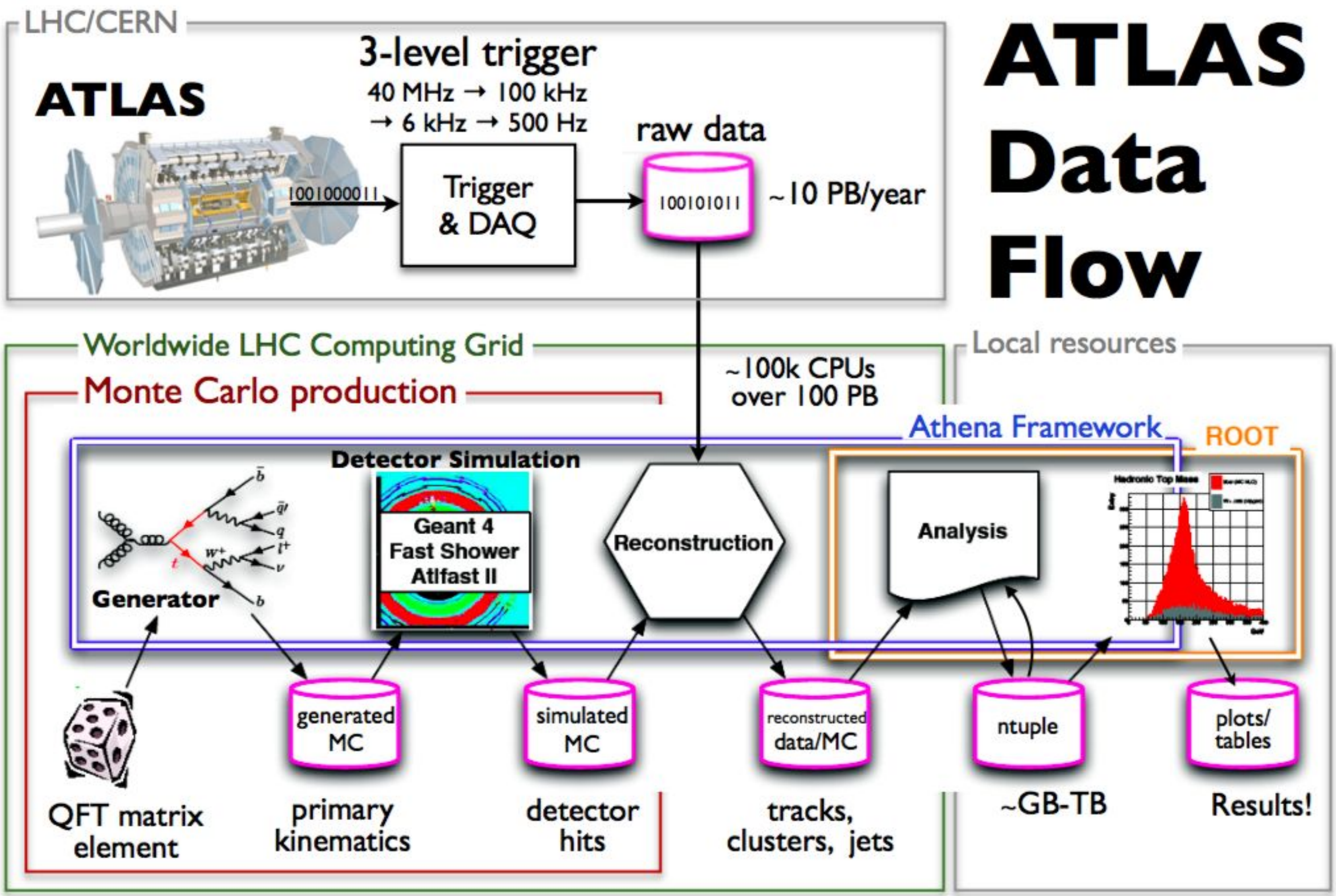
- With this platform, easily upload and test many architectures

Goals

- Demonstrate a successful hyperparameter scan
- Use our system to discover a performant model
- Future upgrades could add better optimization (*e.g.* SigOpt / Hyperopt / custom RL?)



ATLAS Data Flow



LHC/CERN

ATLAS

3-level trigger

40 MHz → 100 kHz

→ 6 kHz → 500 Hz

1001000011

Trigger & DAQ

raw data

100101011

~10 PB/year

Worldwide LHC Computing Grid

~100k CPUs over 100 PB

Local resources

Monte Carlo production

Athena Framework

ROOT

Detector Simulation

Generator

Geant 4
Fast Shower
Atlfast II

Reconstruction

Analysis

generated MC

simulated MC

reconstructed data/MC

ntuple

plots/ tables

QFT matrix element

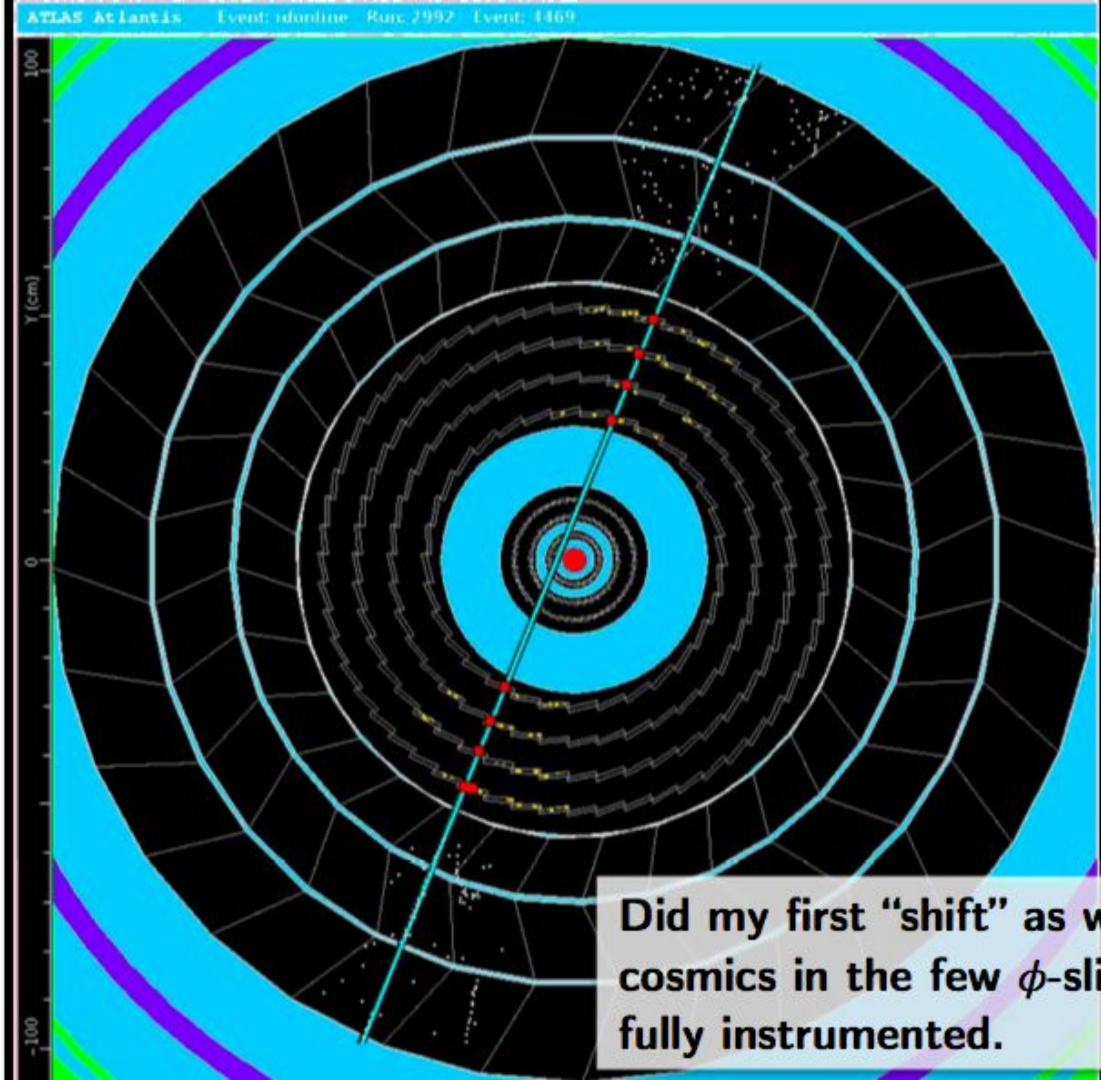
primary kinematics

detector hits

tracks, clusters, jets

~GB-TB

Results!

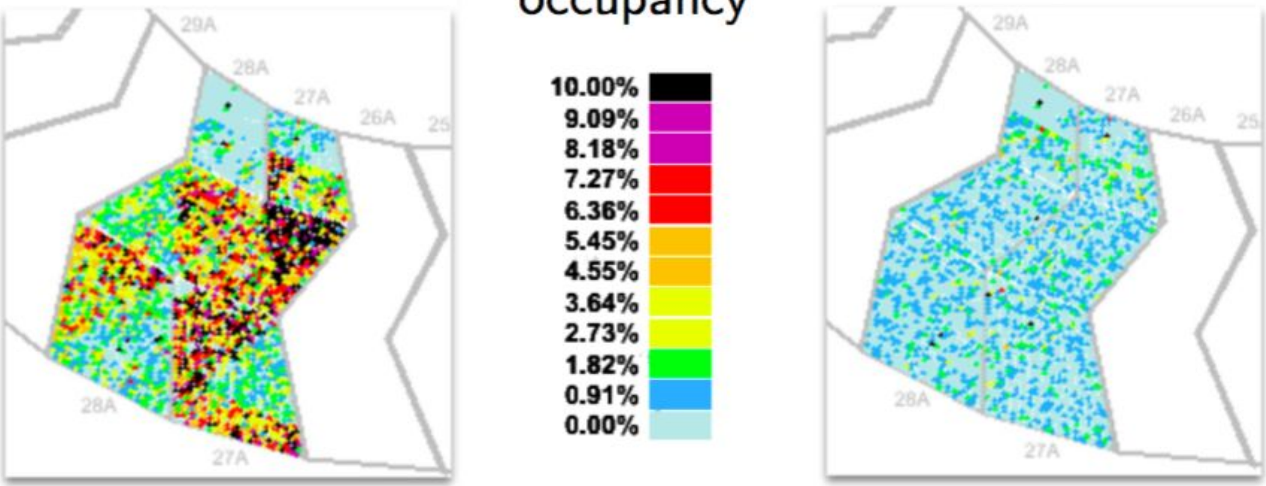


2006

Did my first “shift” as we saw the first cosmics in the few ϕ-slices that were fully instrumented.

TRT threshold calibration

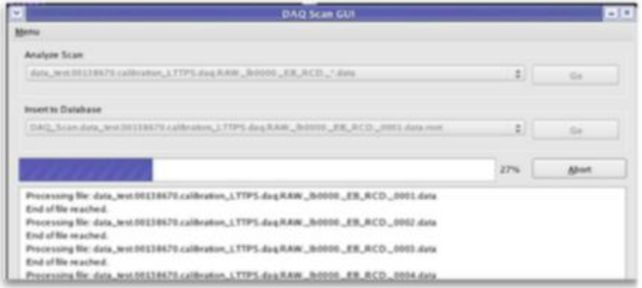
2008


Before

noise occupancy

After

- Developed a GUI making it easy for shifters to archive scans to a database for monitoring long-term detector health.



data_test.00138670.calibration_LTTPS.daq.RAW._lb0000._EB_RCD._*.data

DAQ_Scan.data_test.00138670.calibration_LTTPS.daq.RAW._lb0000._EB_RCD._0001.data.root

```
Processing file: data_test.00138670.calibration_LTTPS.daq.RAW._lb0000._EB_RCD._0001.data
End of file reached.
Processing file: data_test.00138670.calibration_LTTPS.daq.RAW._lb0000._EB_RCD._0002.data
End of file reached.
Processing file: data_test.00138670.calibration_LTTPS.daq.RAW._lb0000._EB_RCD._0003.data
End of file reached.
Processing file: data_test.00138670.calibration_LTTPS.daq.RAW._lb0000._EB_RCD._0004.data
```

- Still used in the regularly scheduled calibration periods between beam fills.

- Supported TRT as part of DAQ on-call team.