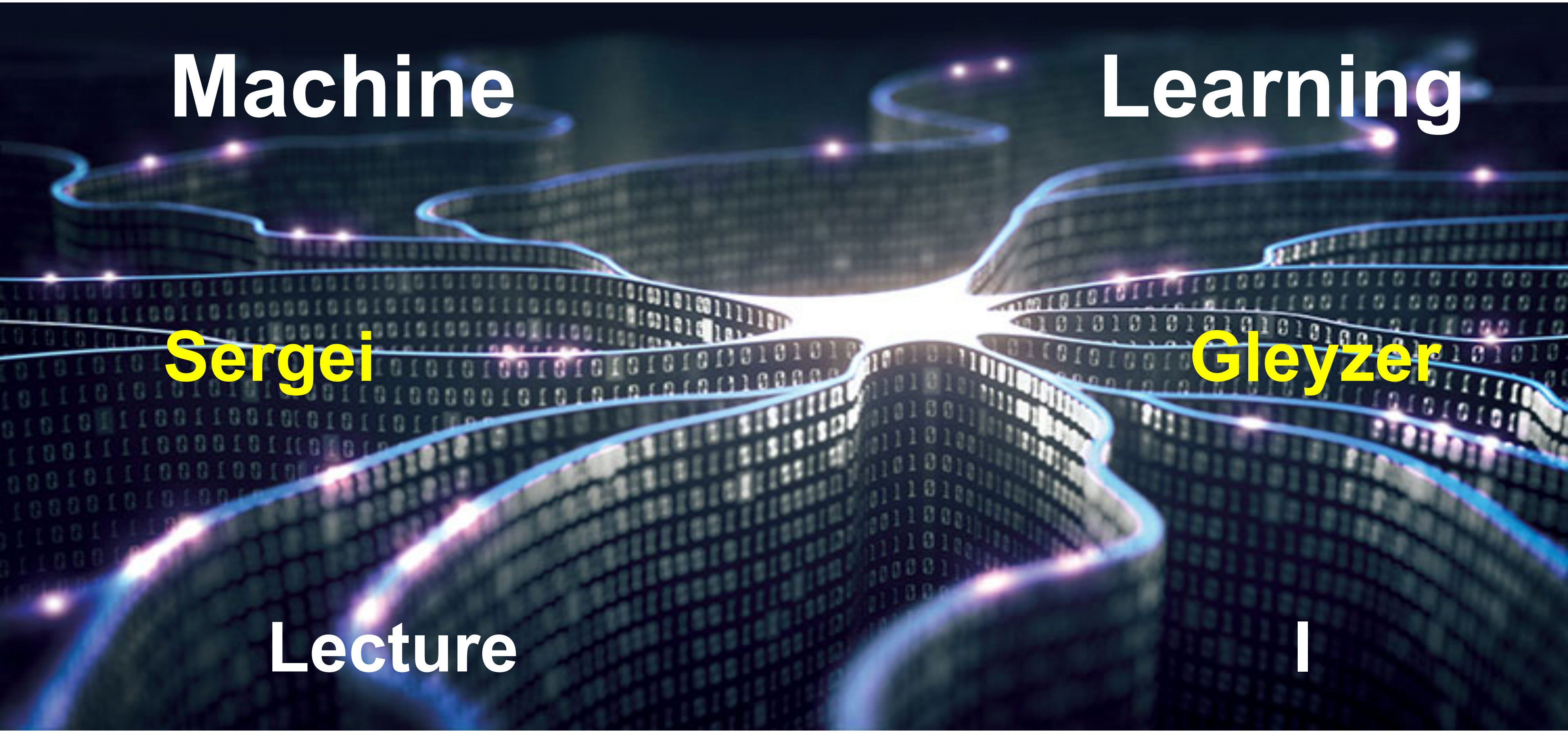


Machine Learning

Sergei Gleyzer

Lecture I

INFN Paestum Lectures

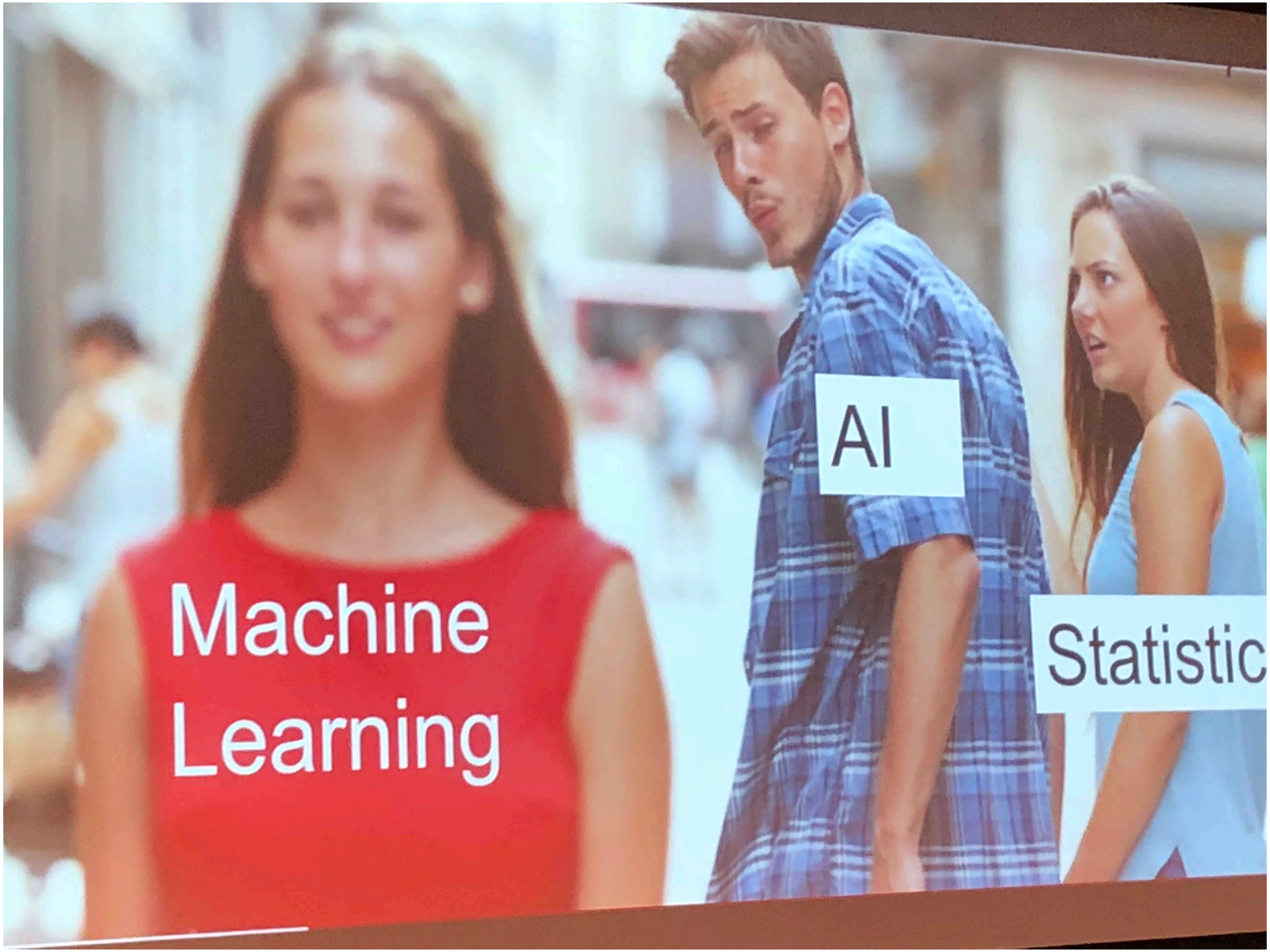
June 6, 2019

Today's Outline

Machine Learning

- **in Theory**
- **in Practice**

Machine Learning Basics


Machine Learning
AI
Statistic

Machine Learning

What is Machine Learning?

- Study of algorithms that
  improve their <u>performance</u> **P**
  for a given <u>task</u> **T**
  with more <u>experience</u> **E**

Sample tasks: identifying cats, particles

Machine Learning

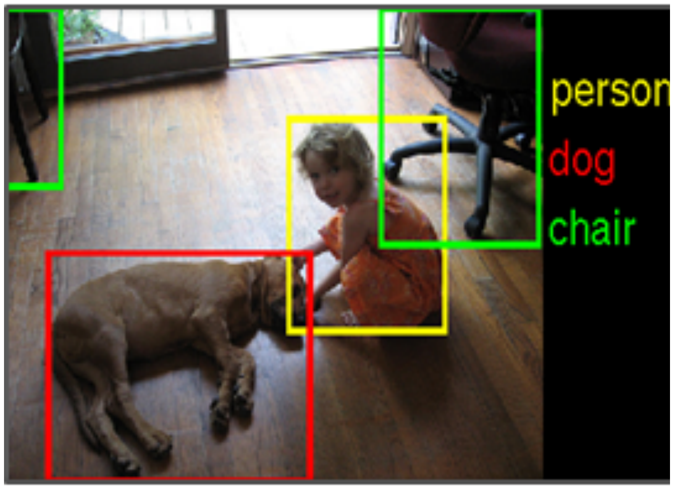
Approach:

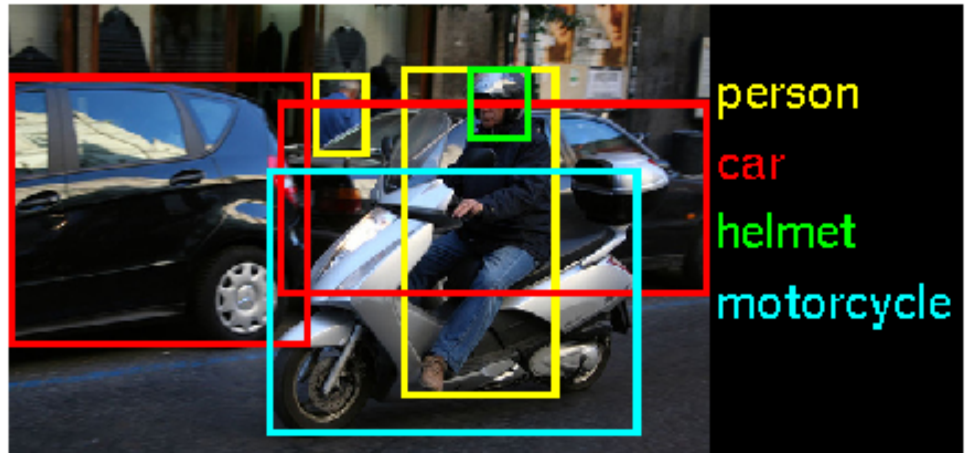
Training data $T_D = \{y, \mathbf{x}\} = (y,x)_1 \ldots (y,x)_N$,

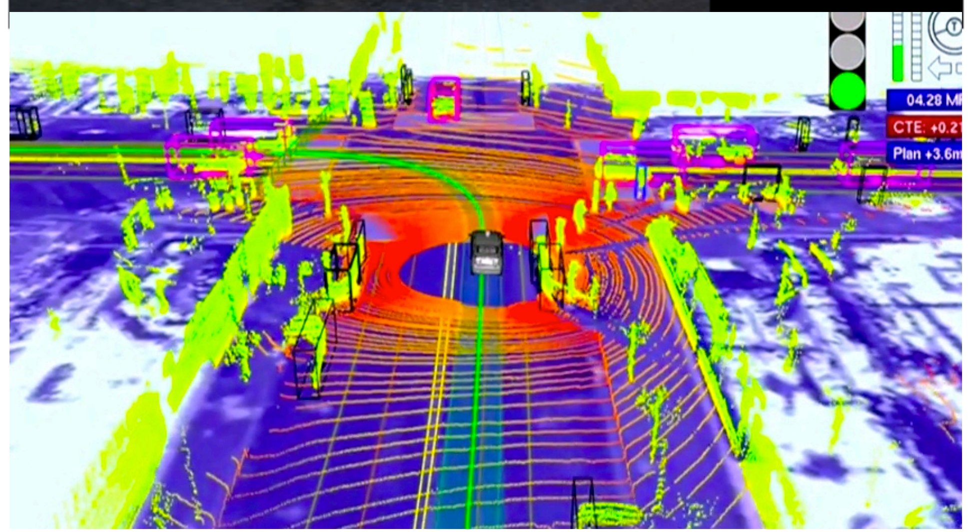
Function space {f} and a **constraint** on these functions

Learn the **mapping** $y = f(x)$

Examples







Machine Learning

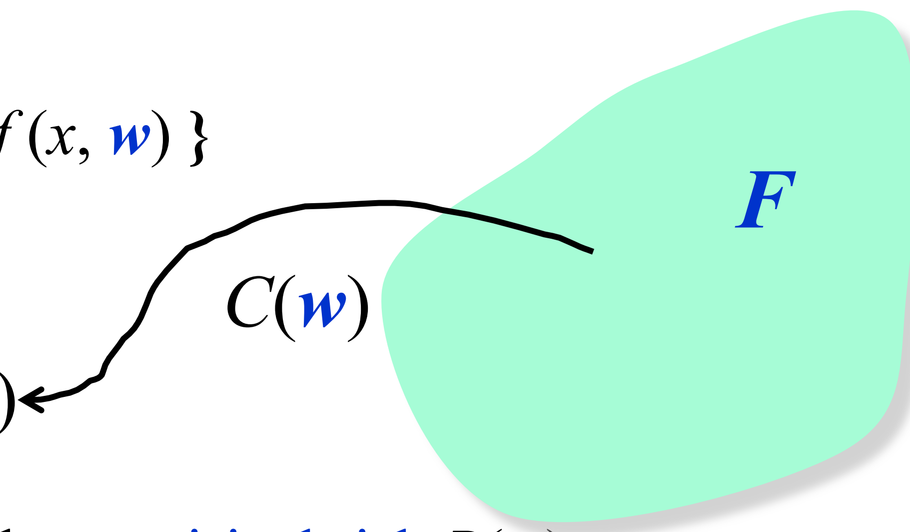
Choose

Function space $\quad F = \{ f(x, w) \}$

Constraint $\quad C$

Loss function* $\quad L$

$$f(x, w^*)$$



Method

Find $f(x)$ by minimizing the empirical risk $R(w)$

$$R[f_w] = \frac{1}{N}\sum_{i=1}^{N} L(y_i, f(x_i, w))$$

subject to the constraint $C(w)$

*The loss function measures the cost of choosing badly

Machine Learning

Many methods (e.g., neural networks, boosted decision trees, rule-based systems, random forests,…) use the

quadratic loss

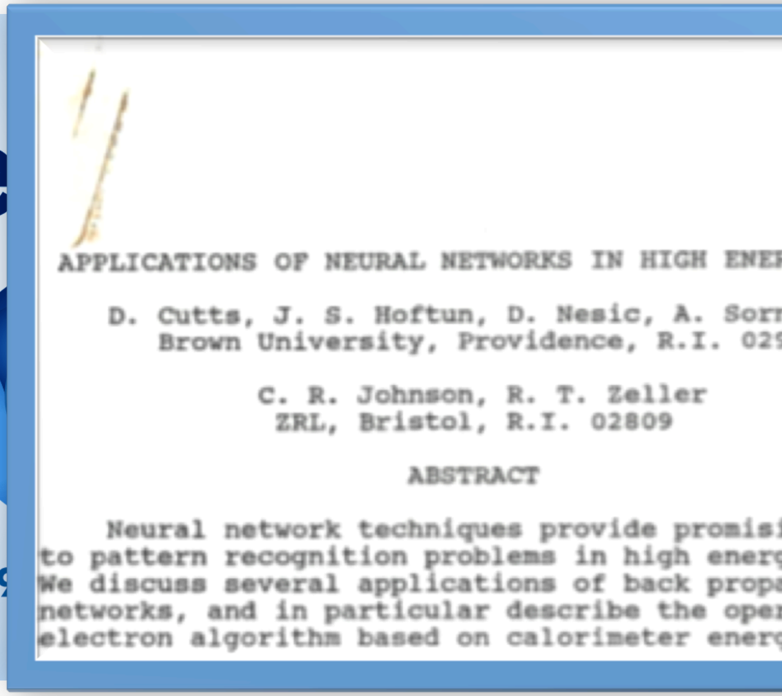
$$L(y, f(x, w)) = [y - f(x, w)]^2$$

and choose $f(x, w^*)$ by minimizing the

constrained mean square empirical risk

$$R[f_w] = \frac{1}{N}\sum_{i=1}^{N}[y_i - f(x_i, w)]^2 + C(w)$$

History


APPLICATIONS OF NEURAL NETWORKS IN HIGH ENER

D. Cutts, J. S. Hoftun, D. Nesic, A. Sorn
Brown University, Providence, R.I. 029

C. R. Johnson, R. T. Zeller
ZRL, Bristol, R.I. 02809

ABSTRACT

Neural network techniques provide promisi
to pattern recognition problems in high energ
We discuss several applications of back propa
networks, and in particular describe the oper
electron algorithm based on calorimeter energ

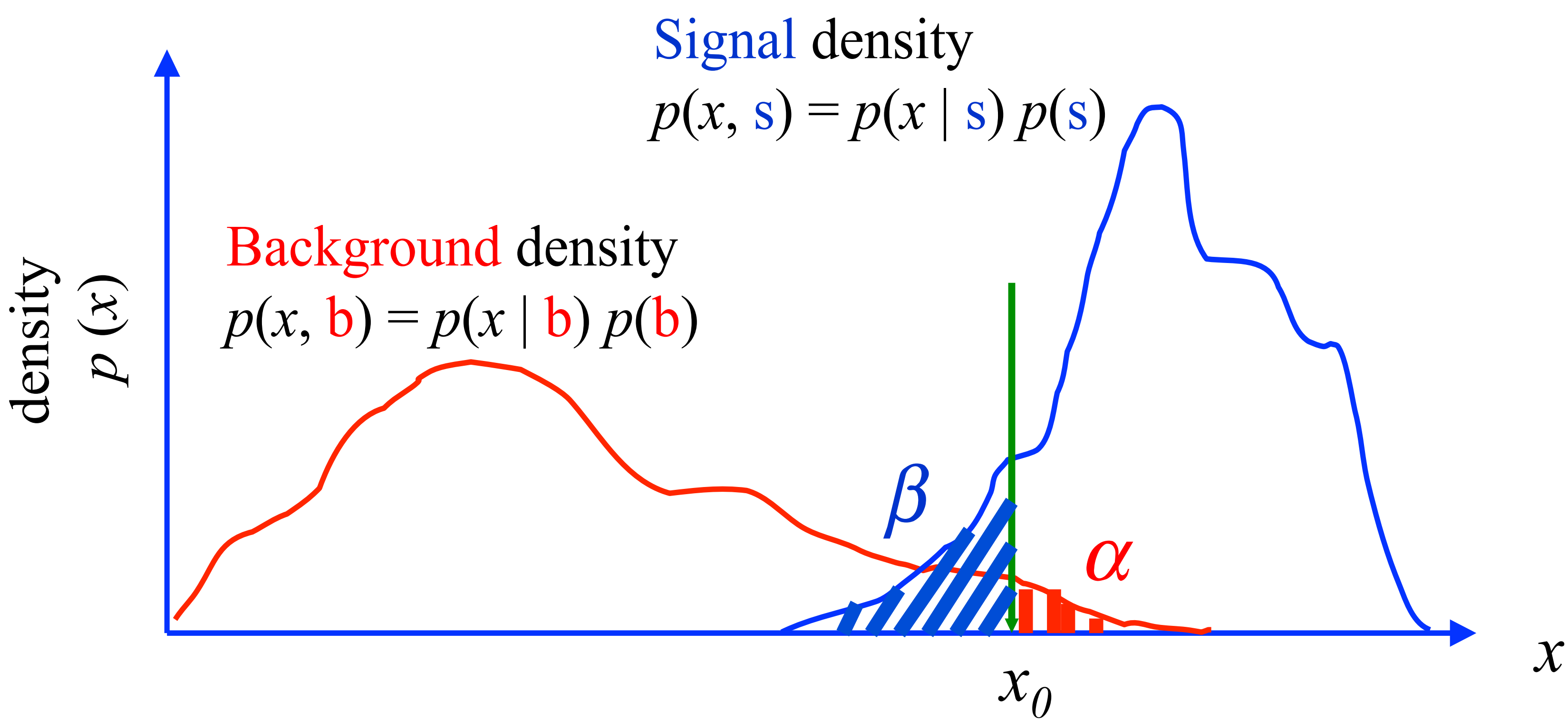
Machine Learning in HEP is at least 30

World Wide Web is 30!

Long history

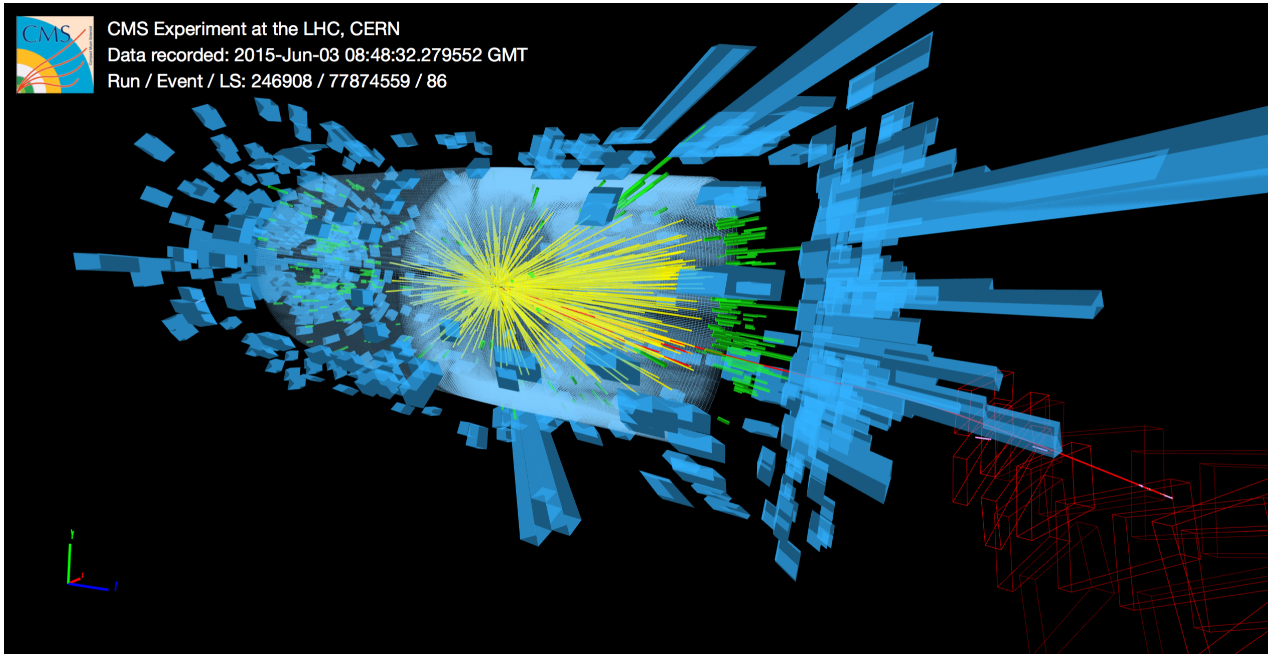
- First methods used in late 80s
- **Tevatron:** pioneering ML efforts in the 90s
- **LHC Run 1:** wide use shallow machine learning (2009-2013)
- Explosion of applications in all frontiers - **energy**, **intensity**, **cosmic**, achieving state of the art performance in various tasks (2014 – present)

Classification Theory



Optimality criterion: minimize the error rate, $\alpha + \beta$

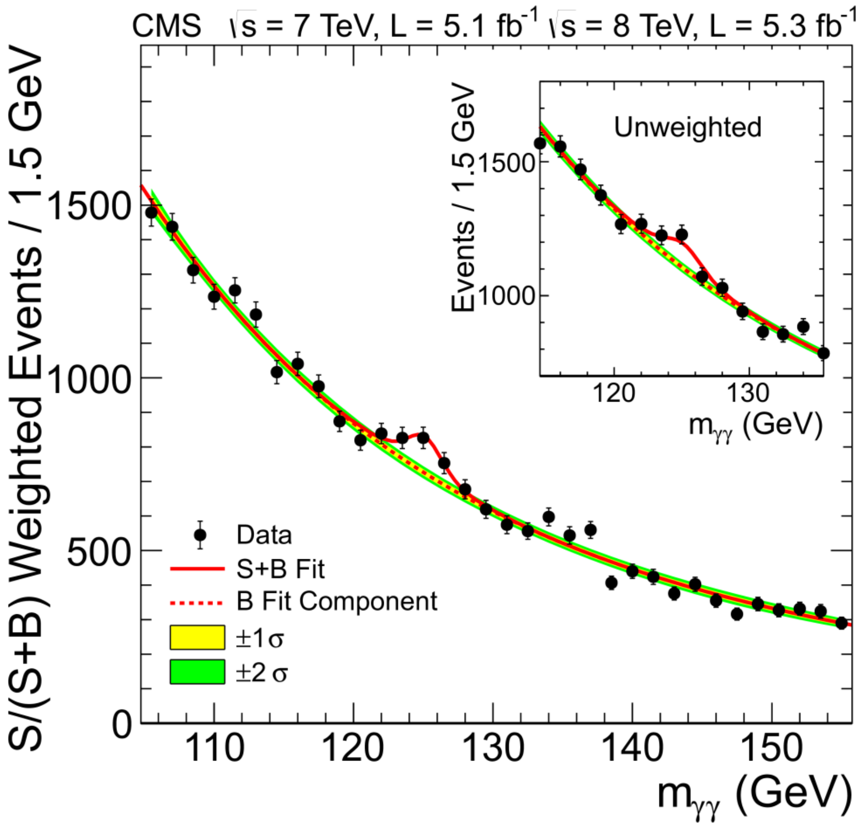
Classification in Practice



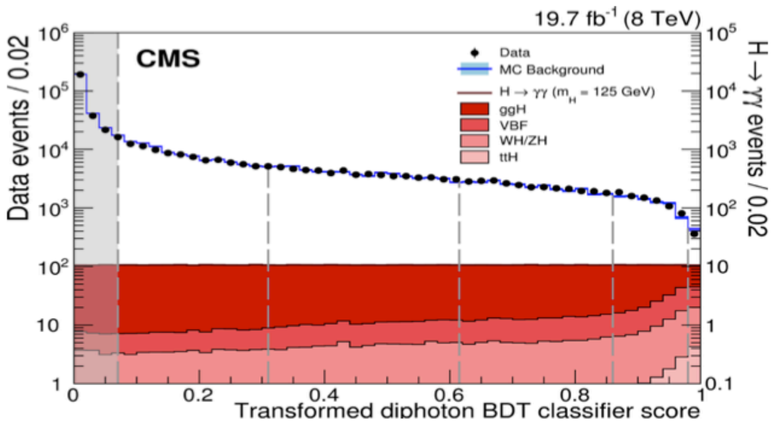
In Particle Physics

Higgs Boson Discovery

in Higgs Discovery



- Particle Identification
- Identification of interactions
- Energy regression
- Event selection



Improvement from all areas

HEP Applications

Primarily Classification

- Particle Level:
  - **Particle identification**

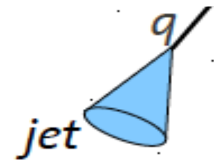
    Photon or a jet?

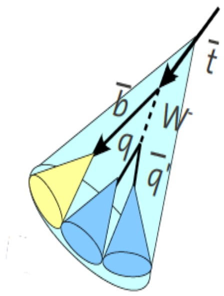
  - **Pattern recognition**

    Tracks, vertices

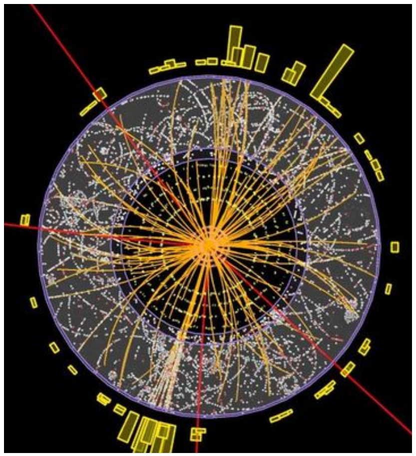
- Event Level:
  - **New Physics searches**

    New Physics event or background?

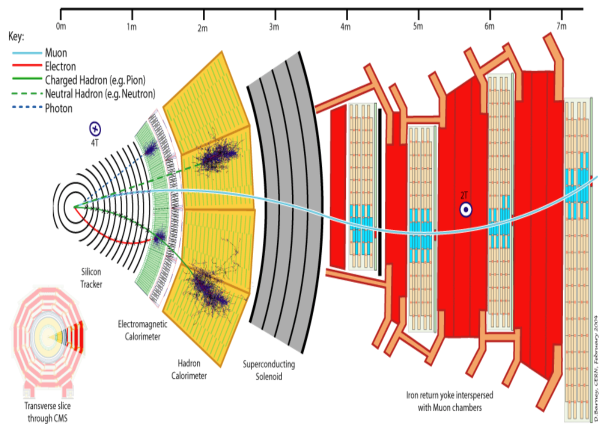




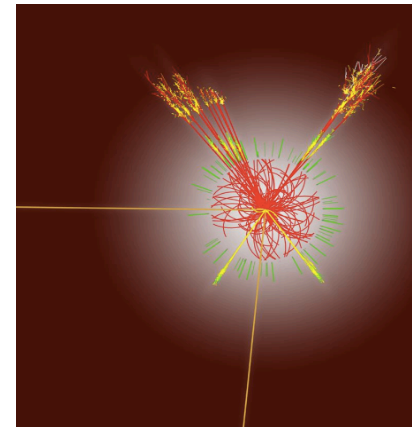
Relevant areas



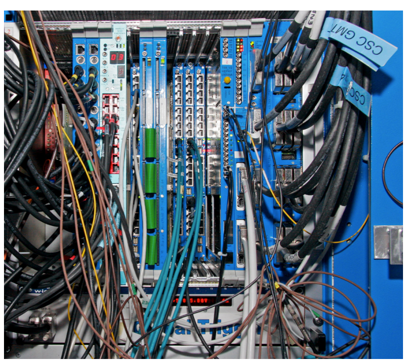
Tracking



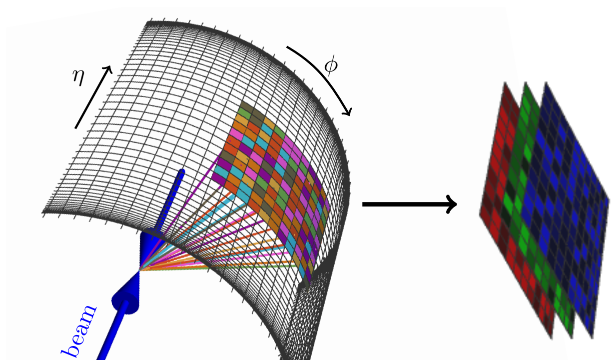
Object Identification



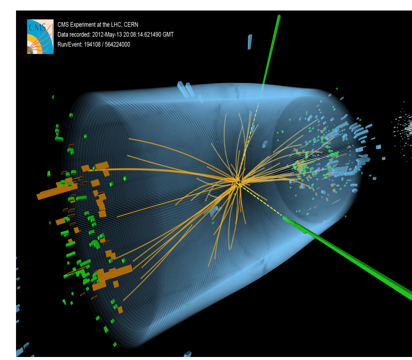
Fast Simulation



Trigger



Imaging Techniques

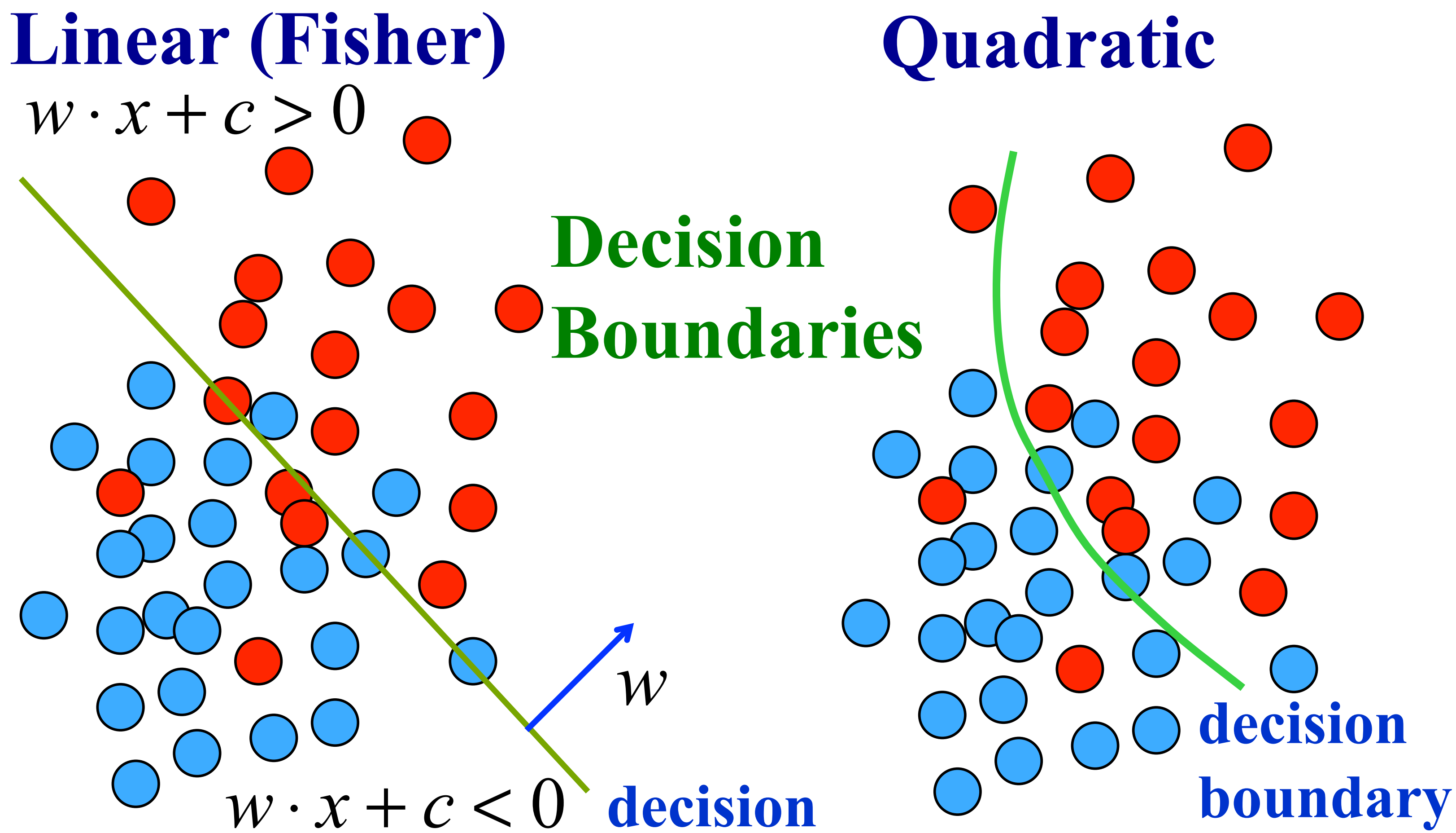


Event Level ID

ML Algorithms

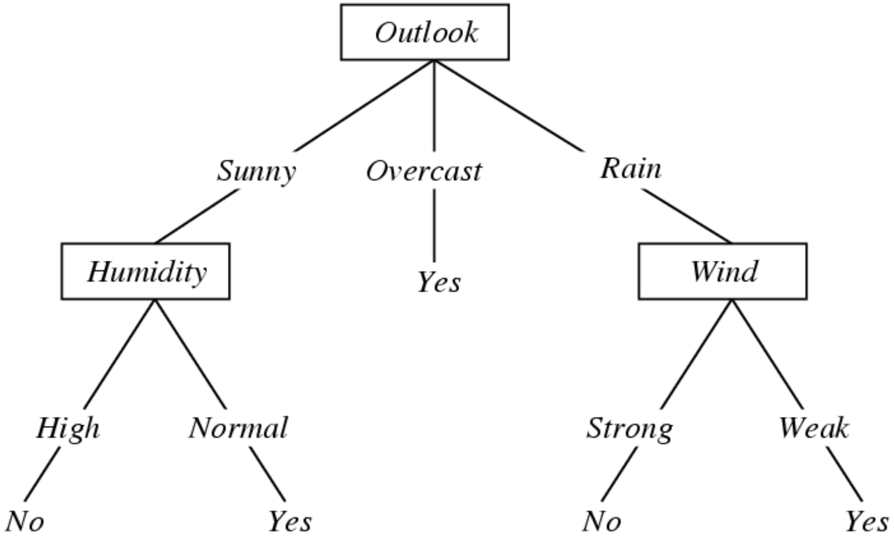
- Fisher, Quadratic
- Naïve Bayes (Likelihood)
- Kernel Density Estimation
- Random Grid Search
- Rule ensembles
- Boosted decision trees
- Random forests
- Support vector machines
- Genetic algorithms
- Deep learning neural networks

Linear and Quadratic

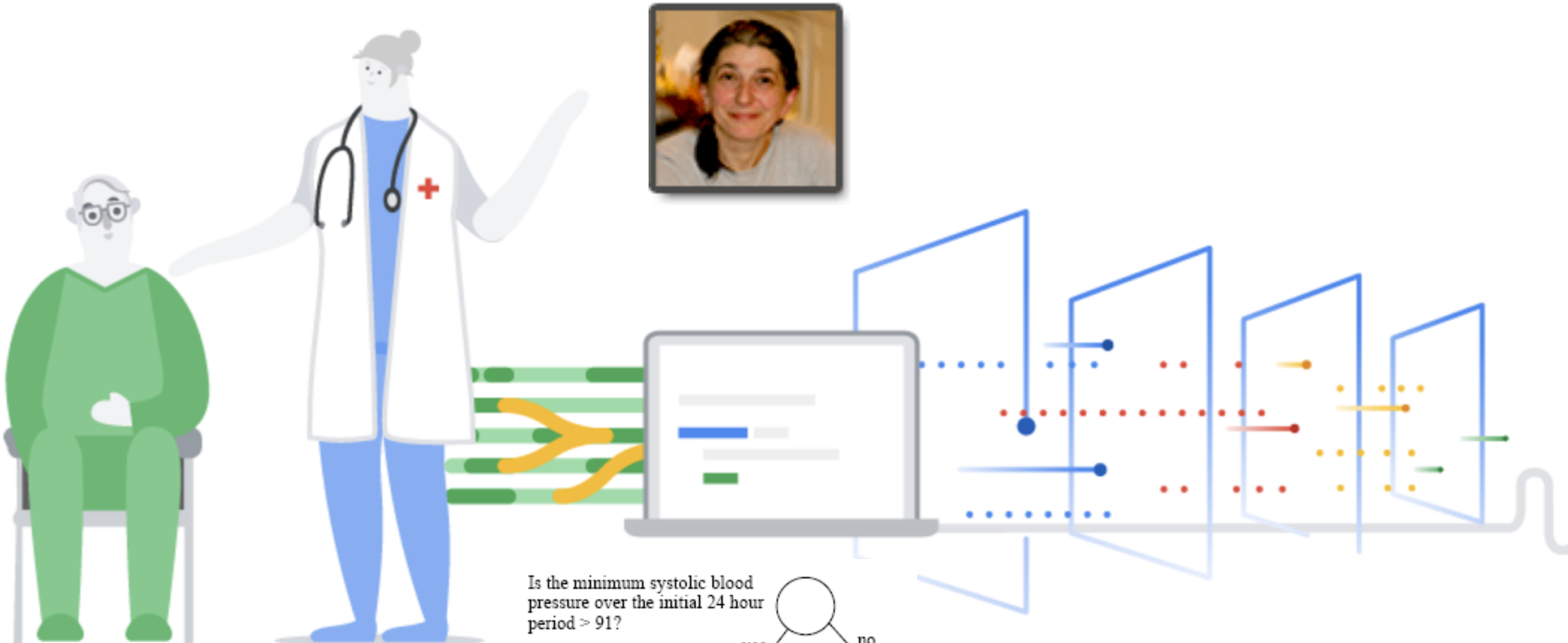


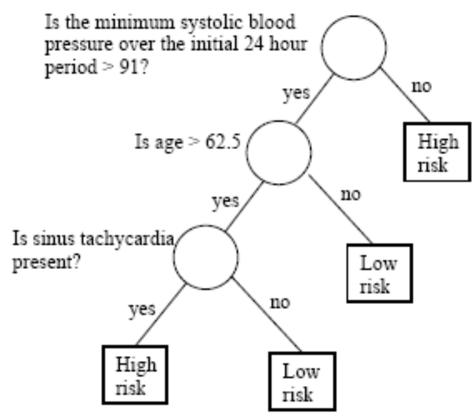
Decision Trees

- **Decision trees** are recursively constructed **multidimensional histograms**
  - Each leaf associated to the value (**class**) of f(x) to be approximated



Doctors: Decision Trees





Ensemble Methods

Suppose you have a **collection** of discriminants $f(x, w_k)$, which, individually, perform only **marginally** better than random guessing.

$$f(x) = a_0 + \sum_{k=1}^{K} a_k f(x, w_k)$$

From such discriminants, **weak learners**, it is possible to build highly effective ones by averaging over them:

Jerome Friedman & Bogdan Popescu (2008)

Adaptive Boosting

AdaBoost (Freund & Shapire 1997)

- **Train in stages:** adaptive weights
- **Misclassified** events get a larger weight going into the next training stage
  - Classify with a majority vote from all trees

- **Works** very well to improve classification power of “greedy” decision trees

Adaptive Boosting

Repeat *K* times:

1. Create a decision tree $f(x, \boldsymbol{w})$
2. Compute its error rate $\boldsymbol{\varepsilon}$ on the ***weighted*** training set
3. Compute $\boldsymbol{\alpha} = \ln (1 - \boldsymbol{\varepsilon}) / \boldsymbol{\varepsilon}$
4. Modify training set: ***increase weight*** of ***incorrectly classified examples*** relative to the weights of those that are correctly classified

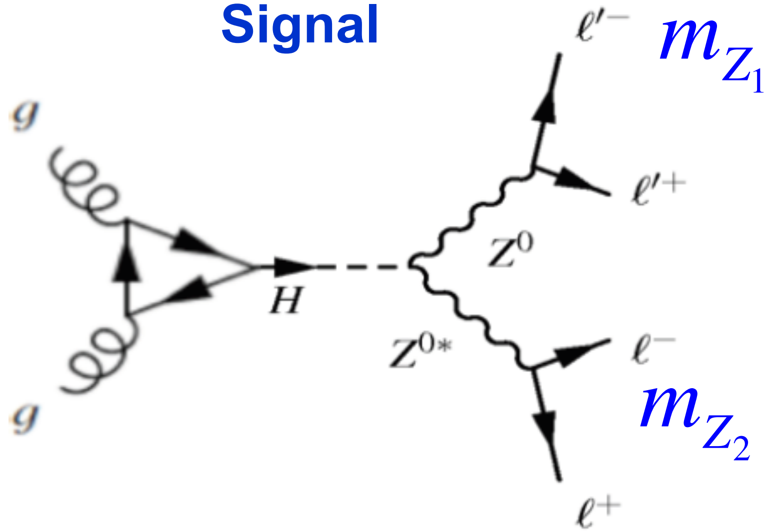
Then compute weighted average $f(x) = \sum \alpha_k \; f(x, w_k)$

Y. Freund and R.E. Schapire (1997)

Illustrative Example

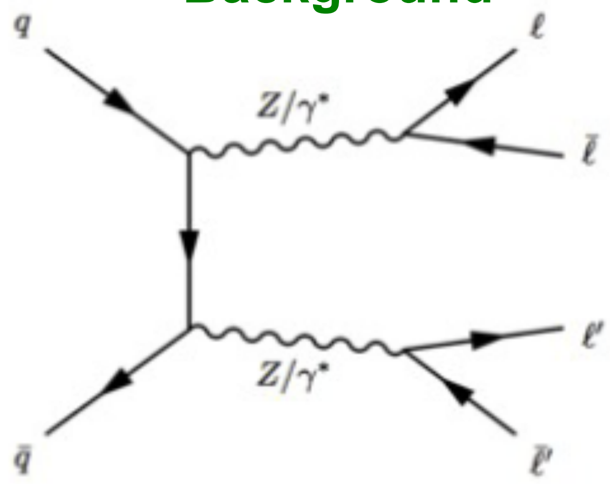
$H \rightarrow ZZ^* \rightarrow 4$ leptons

Signal



$$pp \rightarrow H \rightarrow ZZ \rightarrow \ell^+\ell^-\ell'^+\ell'^-$$

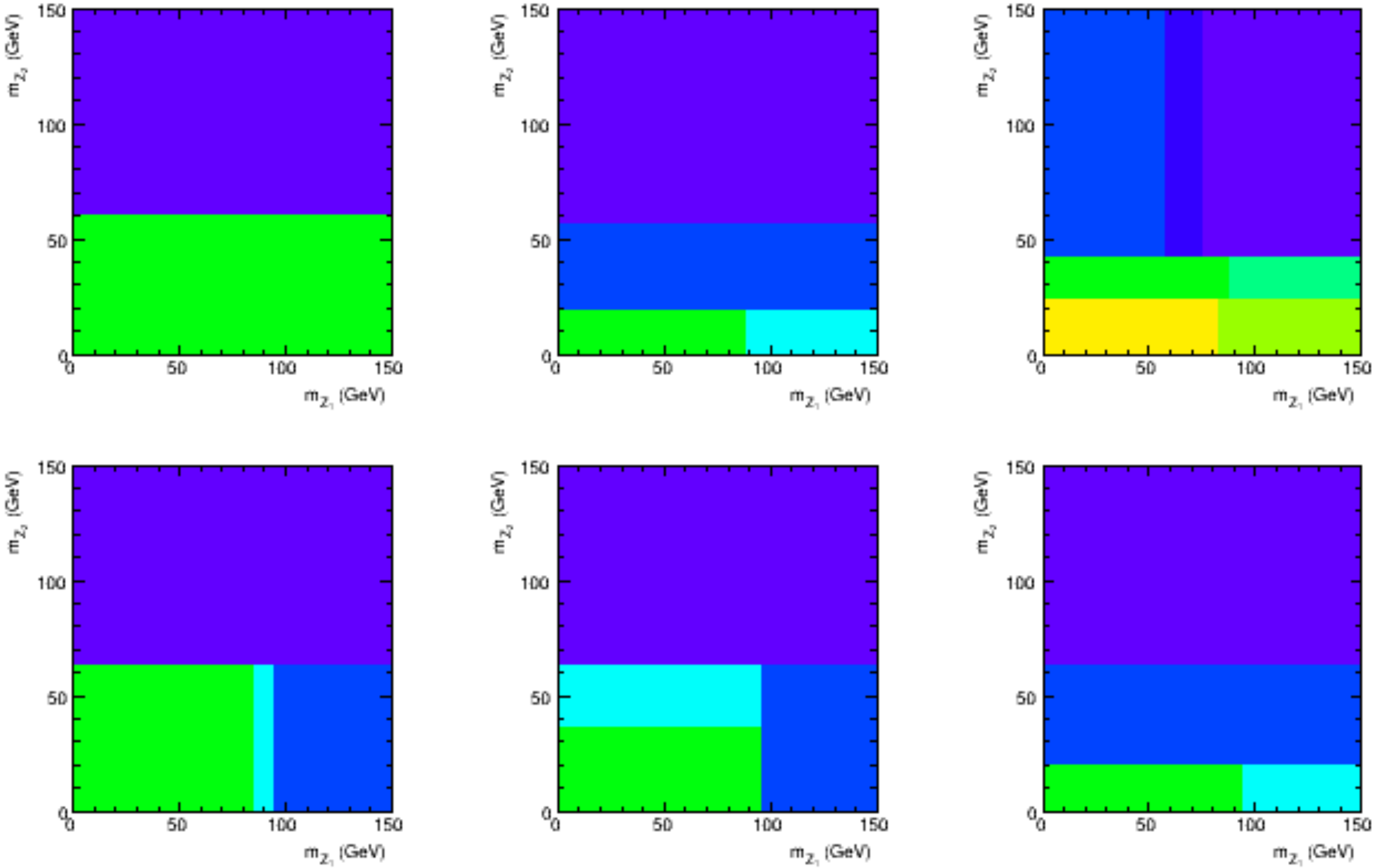
Background



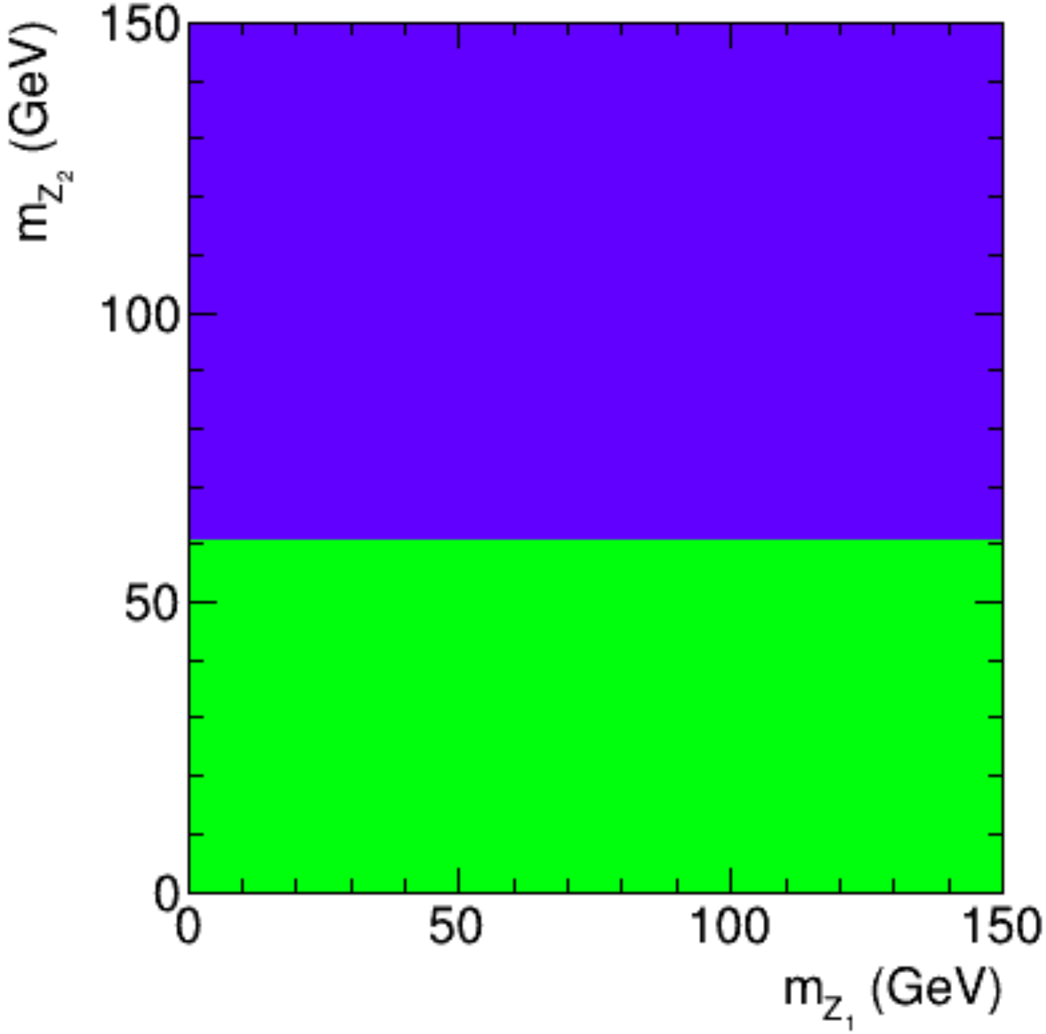
$$pp \rightarrow ZZ \rightarrow \ell^+\ell^-\ell'^+\ell'^-$$

$x = (m_{Z1}, m_{Z2})$

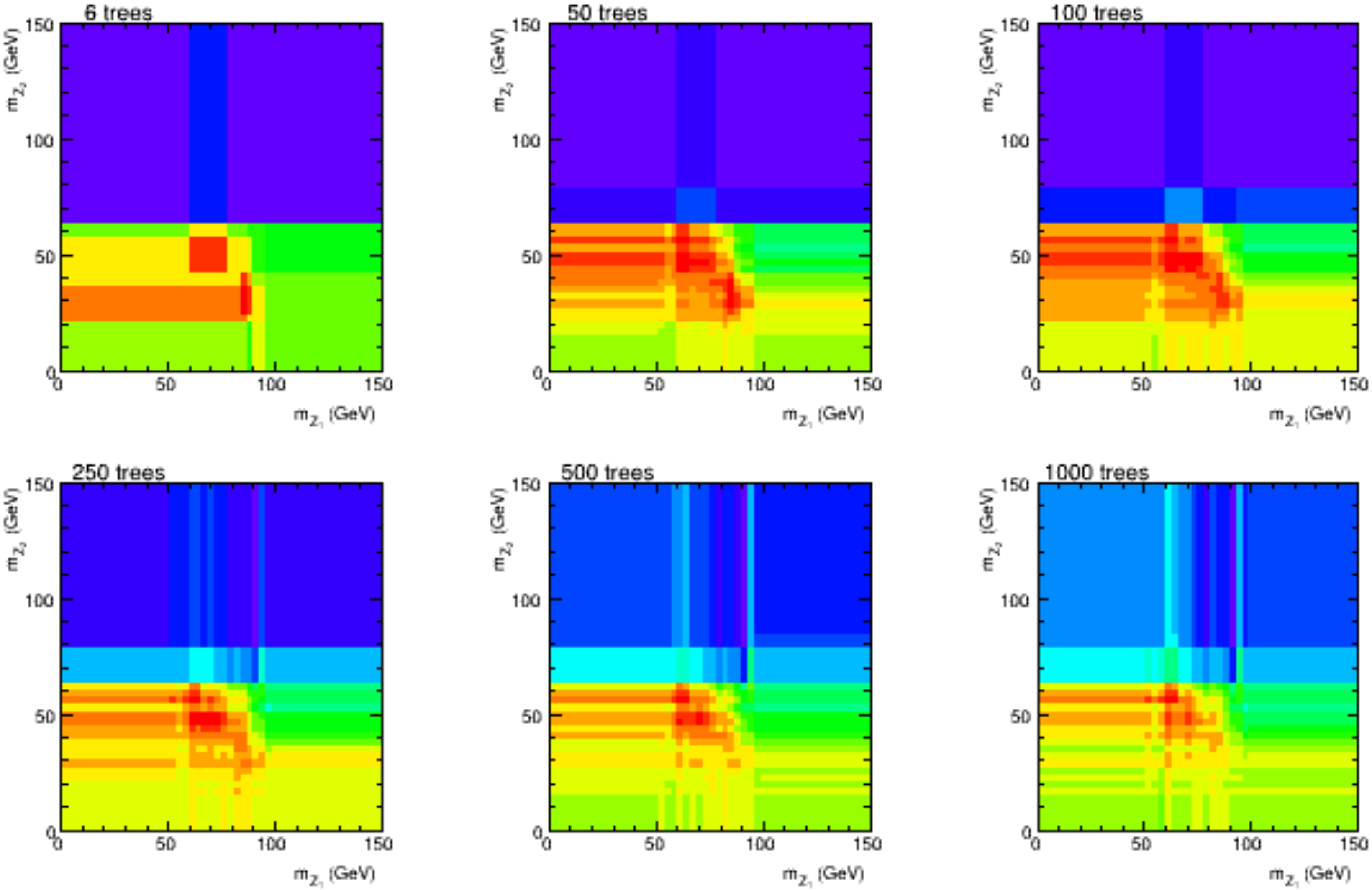
First 6 Decision Trees



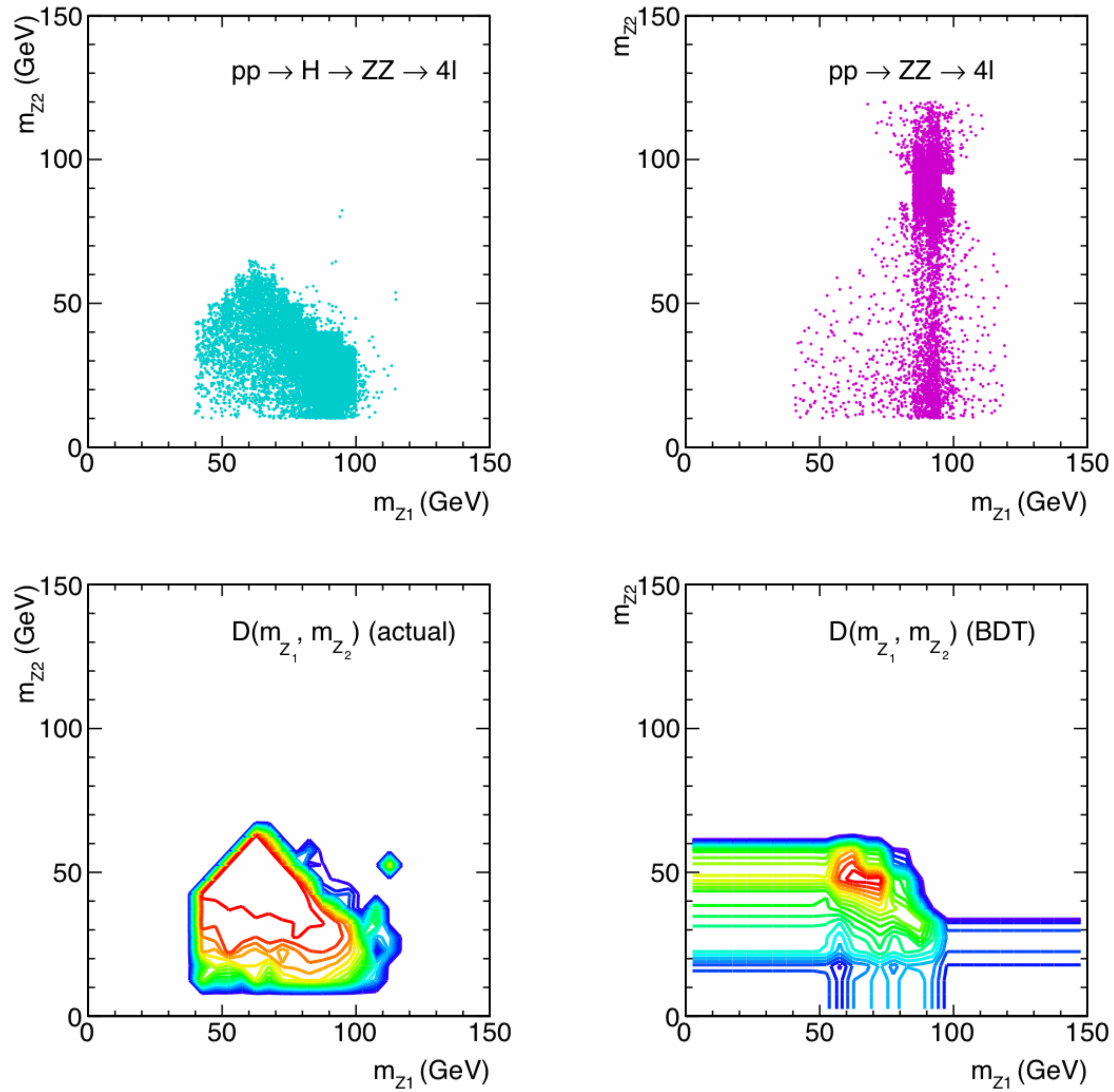
First 100 Decision Trees



Averaging over a Forest



H to ZZ to 4Leptons



HiggsML Challenge

First HEP-ML Challenge

Kaggle platform

Classification

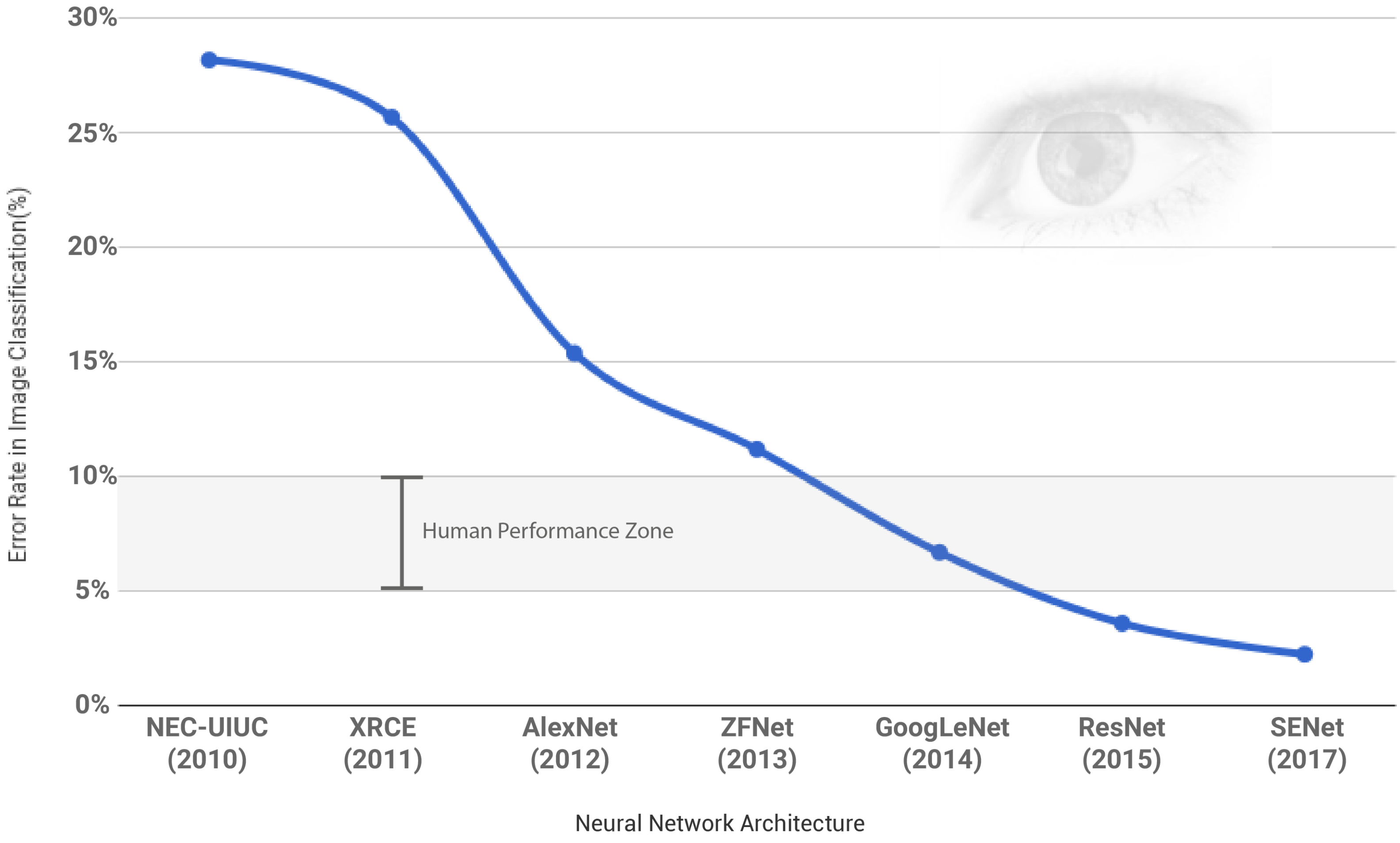
1785 teams

35772 solutions

Tons of excitement

Winners: non-HEP!

Diving Deeper



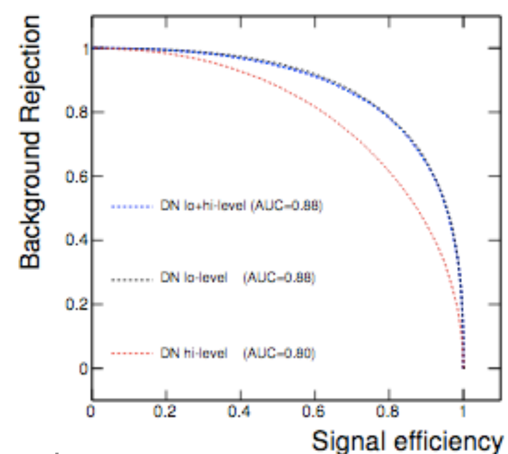
First DNN paper in HEP

Searching for Exotic Particles in High-Energy Physics with Deep Learning

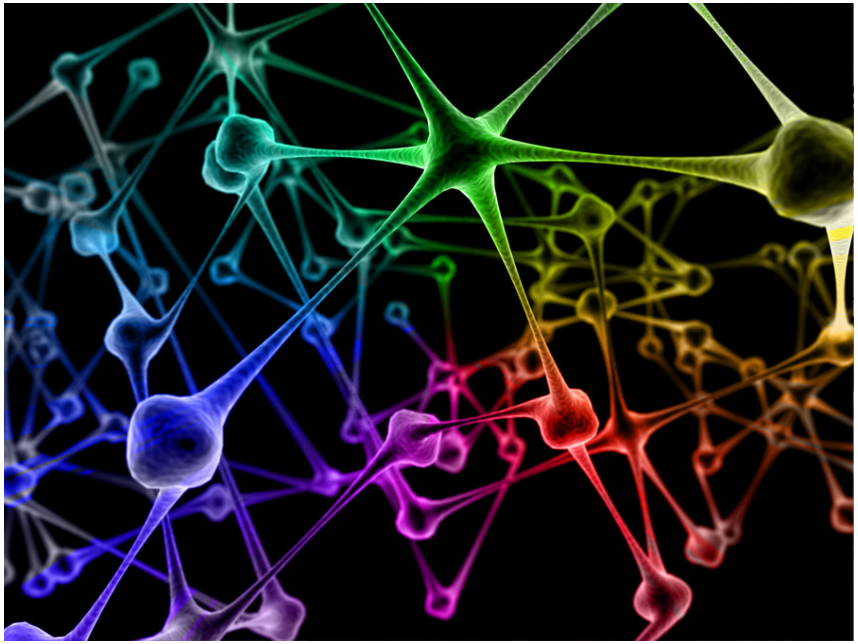
P. Baldi,[1] P. Sadowski,[1] and D. Whiteson[2]

[1]*Dept. of Computer Science, UC Irvine, Irvine, CA 92617*[*]
[2]*Dept. of Physics and Astronomy, UC Irvine, Irvine, CA 92617*[†]

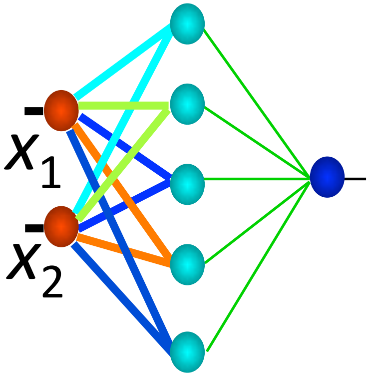
Collisions at high-energy particle colliders are a traditionally fruitful source of exotic particle discoveries. Finding these rare particles requires solving difficult signal-versus-background classification problems, hence machine learning approaches are often used. Standard approaches have relied on 'shallow' machine learning models that have a limited capacity to learn complex non-linear functions of the inputs, and rely on a pain-staking search through manually constructed non-linear features. Progress on this problem has slowed, as a variety of techniques have shown equivalent performance. Recent advances in the field of deep learning make it possible to learn more complex functions and better discriminate between signal and background classes. Using benchmark datasets, we show that deep learning methods need no manually constructed inputs and
metric by as much as 8% over the best current approaches. This dem
approaches can improve the power of collider searches for exotic parti

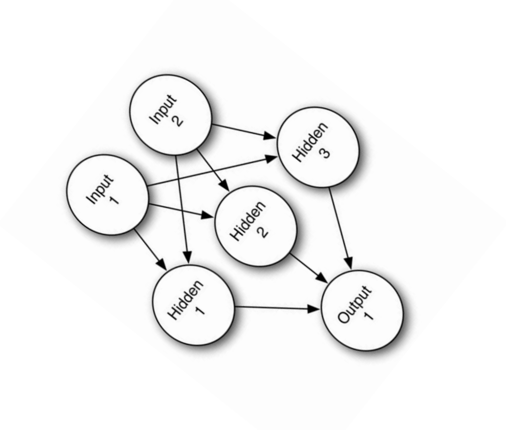


Baldi, Sadowski, & Whiteson, 2014

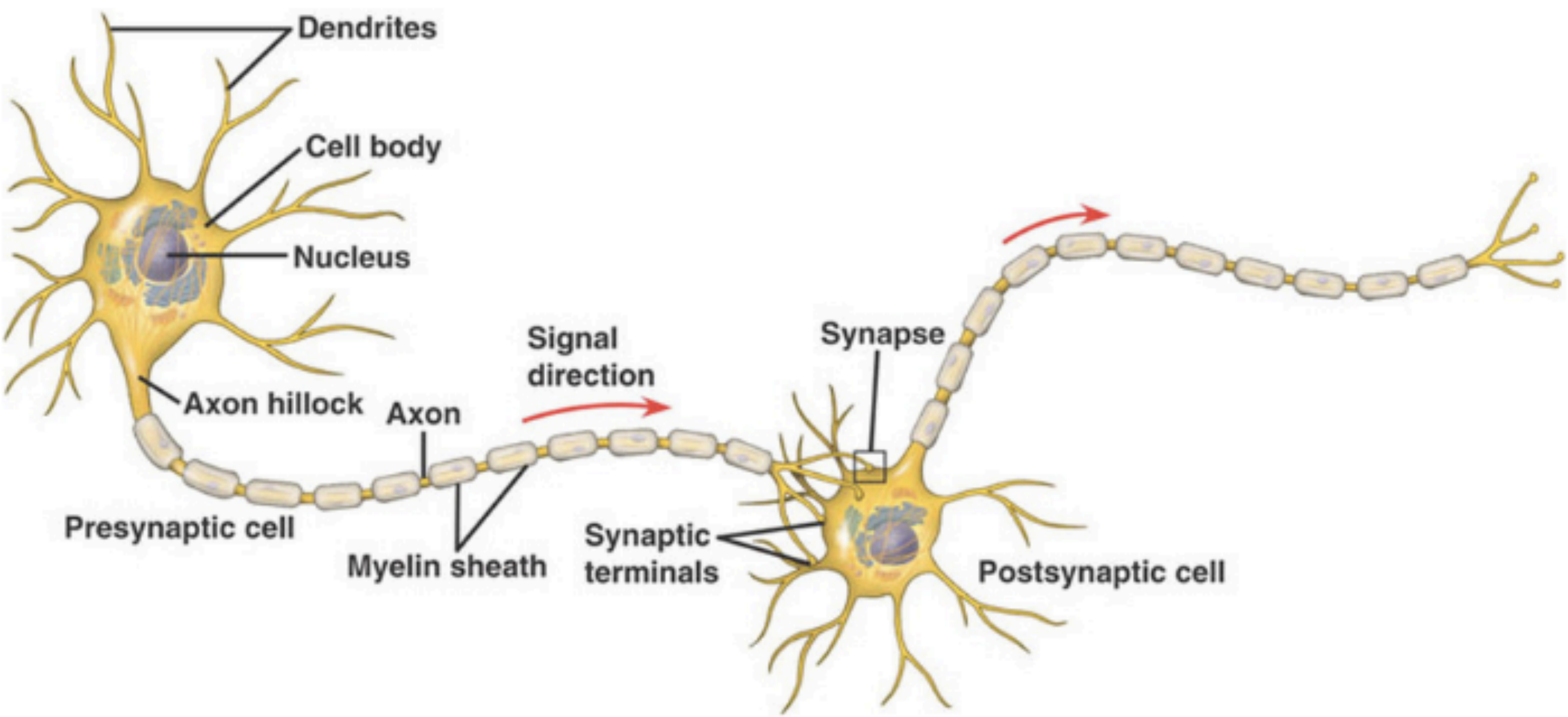


Artificial Neural Networks

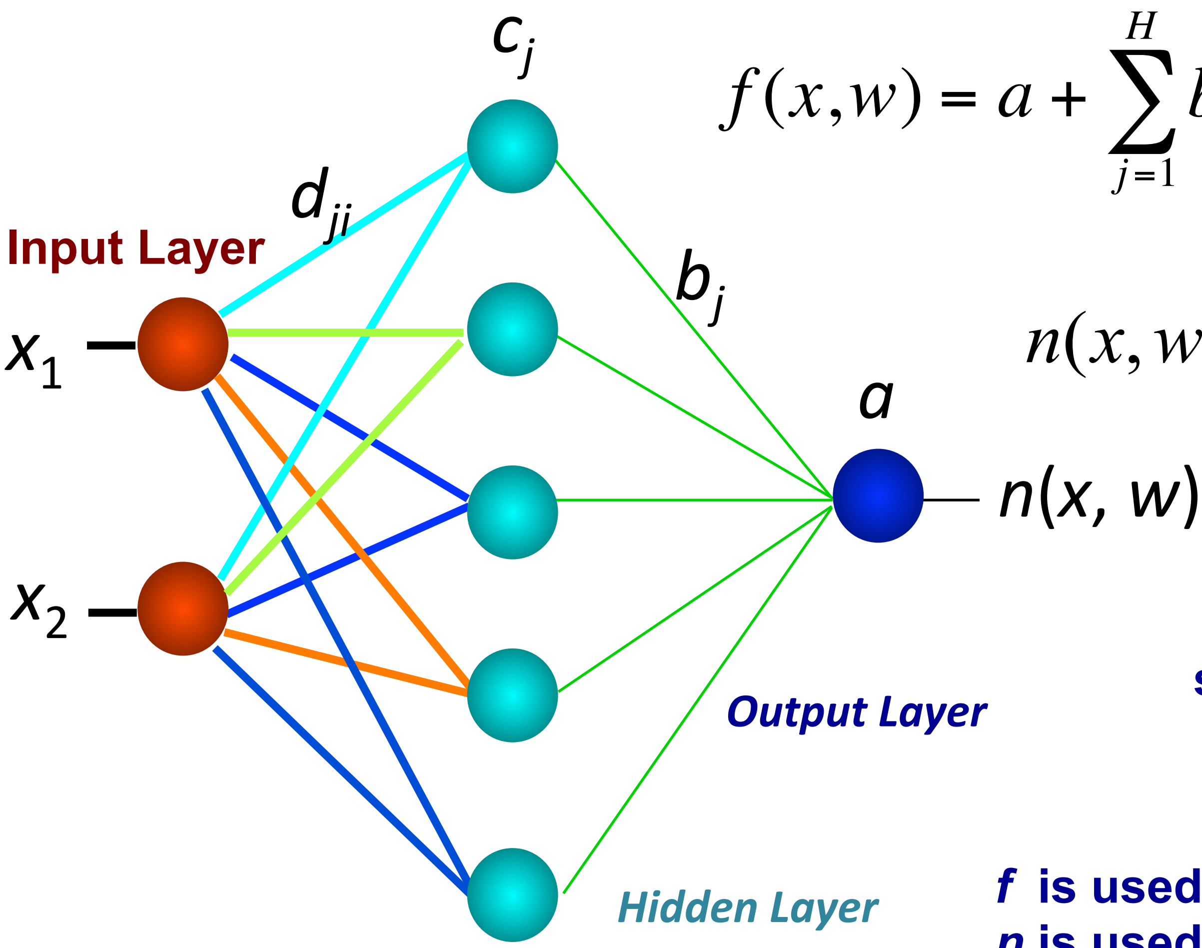




A Real Neuron



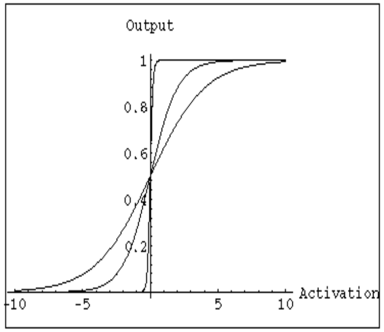
Graphical Representation



$$f(x,w) = a + \sum_{j=1}^{H} b_j \tanh\left[c_j + \sum_{i=1}^{I} d_{ji} x_i\right]$$

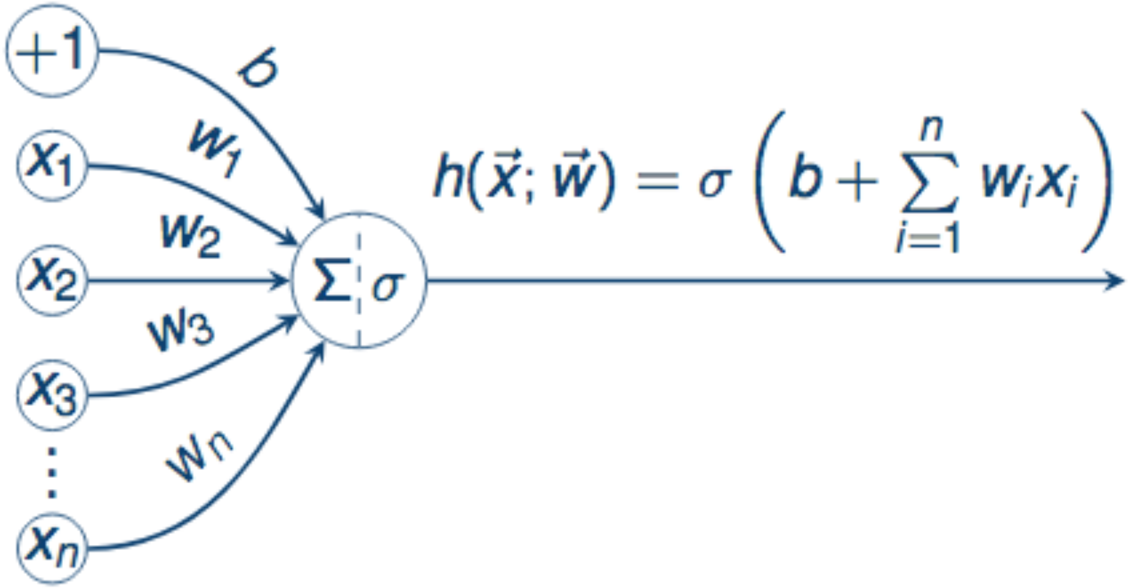
$$n(x,w) = \frac{1}{1+\exp[-f(x,w)]}$$

sigmoid



f **is used for regression**
n **is used for classification**
w = a, b, c, d

Artificial Neuron



Identity: σ(X) = X

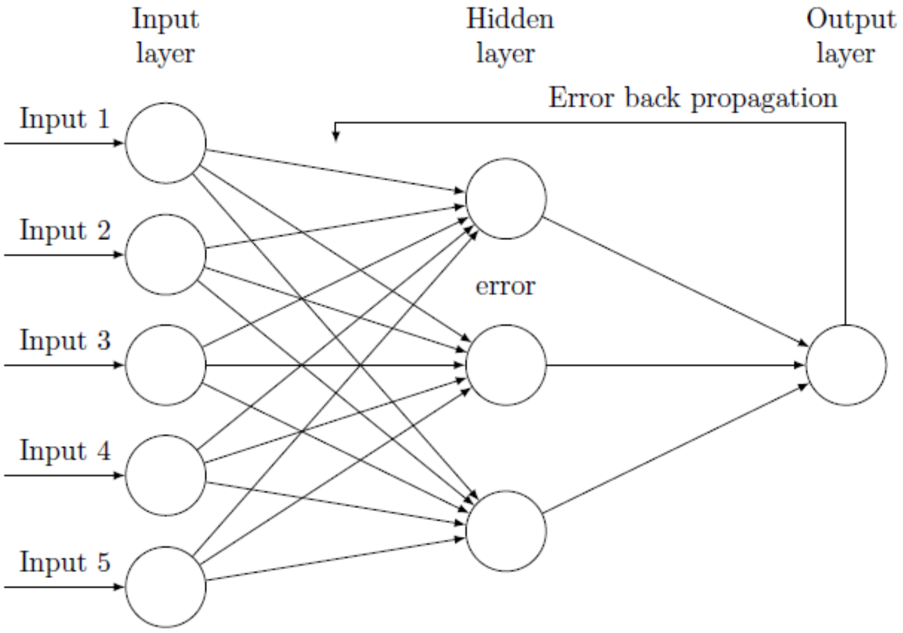
ReLU: **σ(X) = max (0, x)**

Sigmoidal: σ(X) = [1+exp(-x)]$^{-1}$, σ(X) = tanhx

Adjustable Weights

Compute network weights with

- Error gradients



Inputs forward

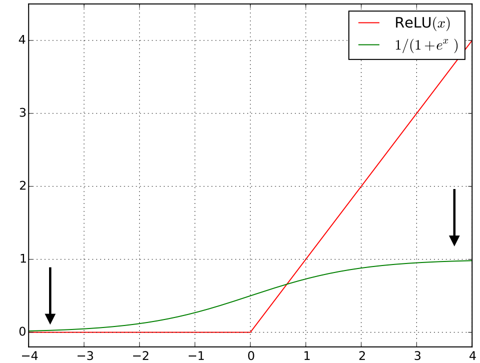
Errors go backward!

Deep Learning

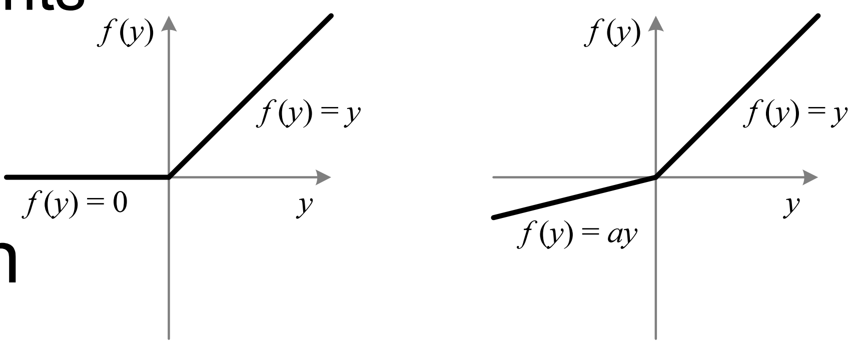
- **Training more complex models**
  - Increased Depth
  - Novel activation functions: ReLU
  - Specialized Architectures: Convolutional, Recurrent, Graphs
- **Effective strategies to avoid over-fitting (regularization)**
  - Data Augmentation, DropOut

ReLU

Rectified Linear Unit (ReLU)



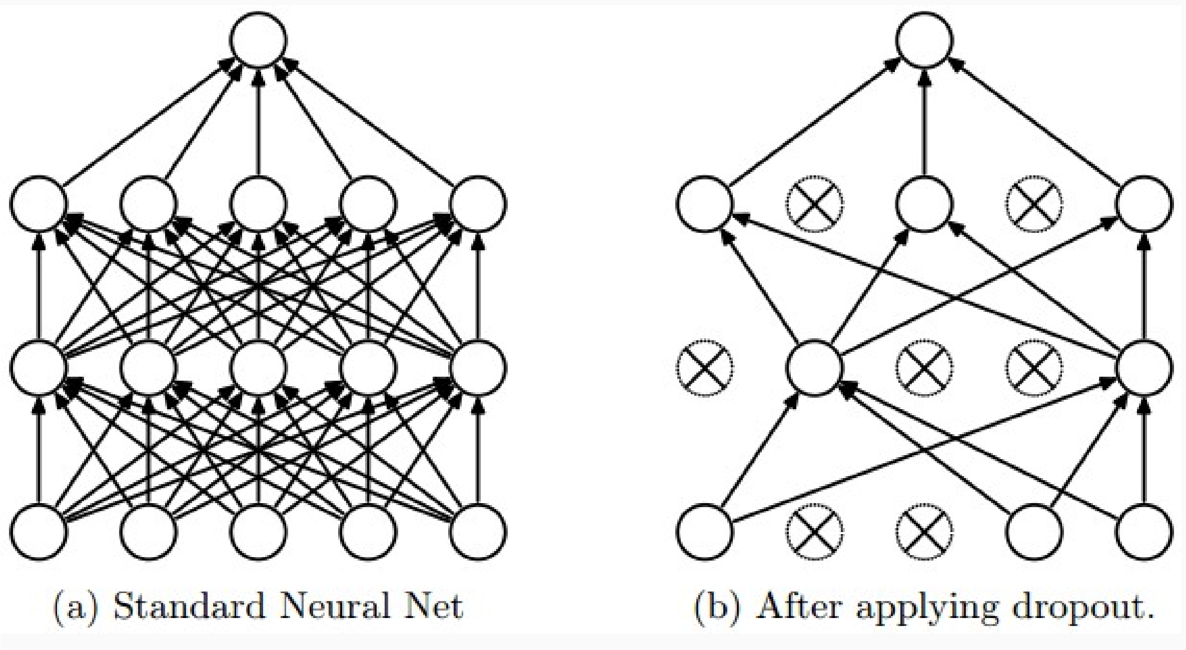
- Rectified neuron
- Faster training convergence
  - Better solutions than sigmoids
    - Vanishing gradients
  - Trained by back-propagation



ReLU and Parametric PReLU

Regularization

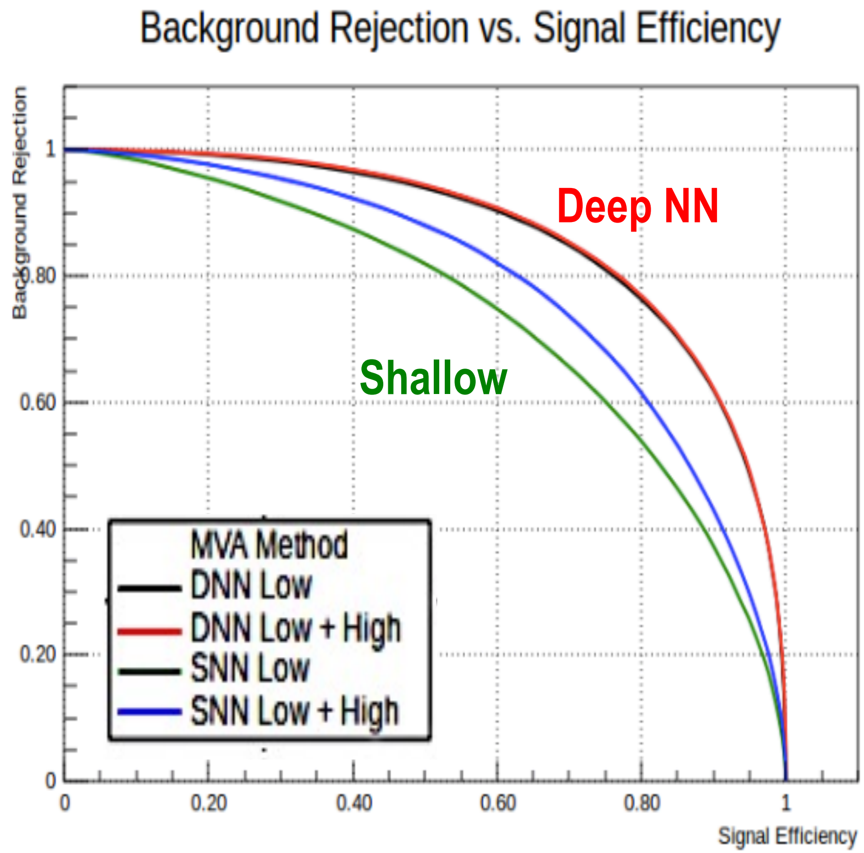
- i.e. Drop-Out



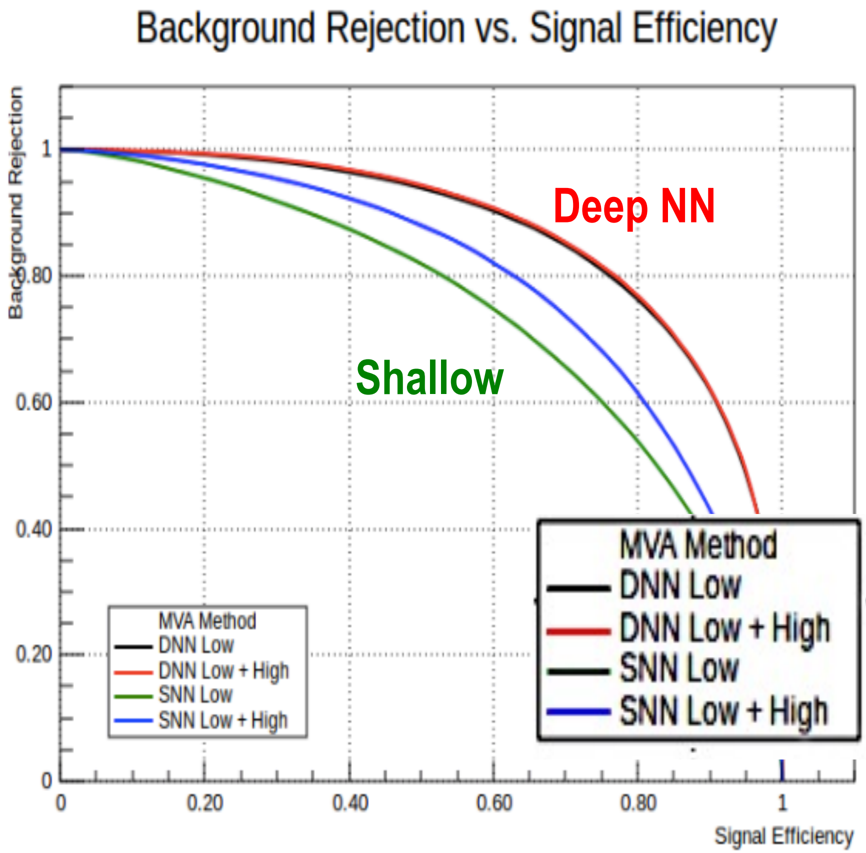
Feature Engineering

- **Historically, features had to be manually extracted and provided as inputs to shallow models**
- **Deep learning paradigm shift: networks learn best features from raw data**
  - Automatic feature extraction

Feature Extraction



Feature Extraction

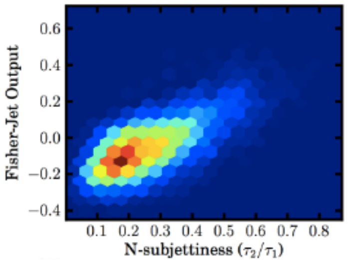


Inherent Feature Extraction

Baldi et al., 2014

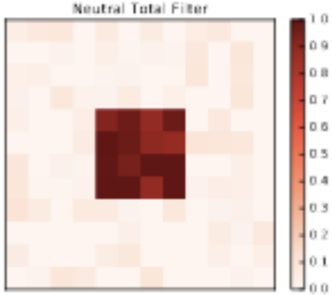
What is learned?

- **Projections**



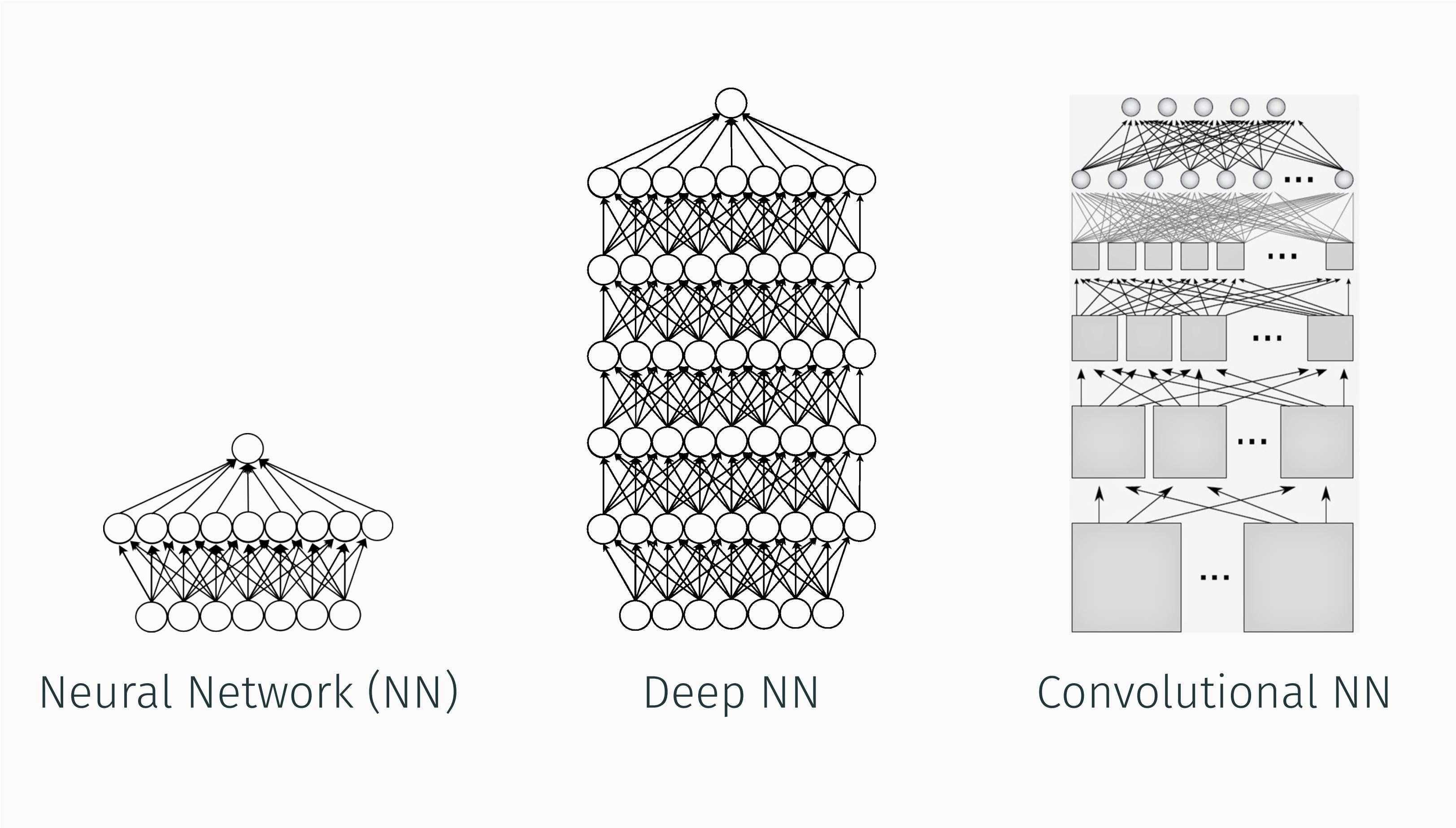
Cogan et al., 2014

- **Filters**

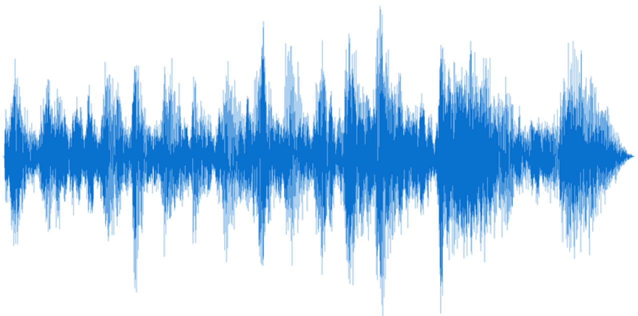


Komiske et al., 2018

Convolutional NN



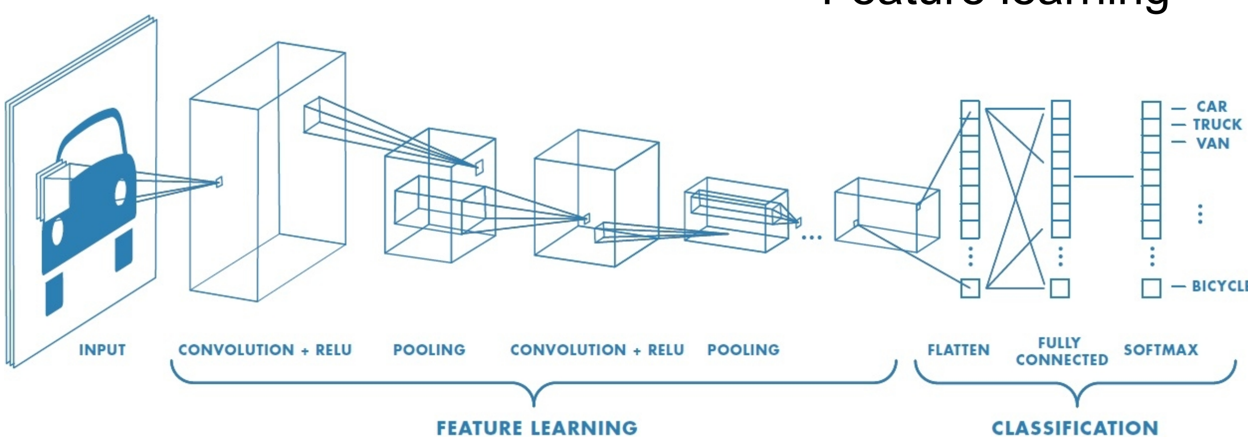
Convolutional Networks



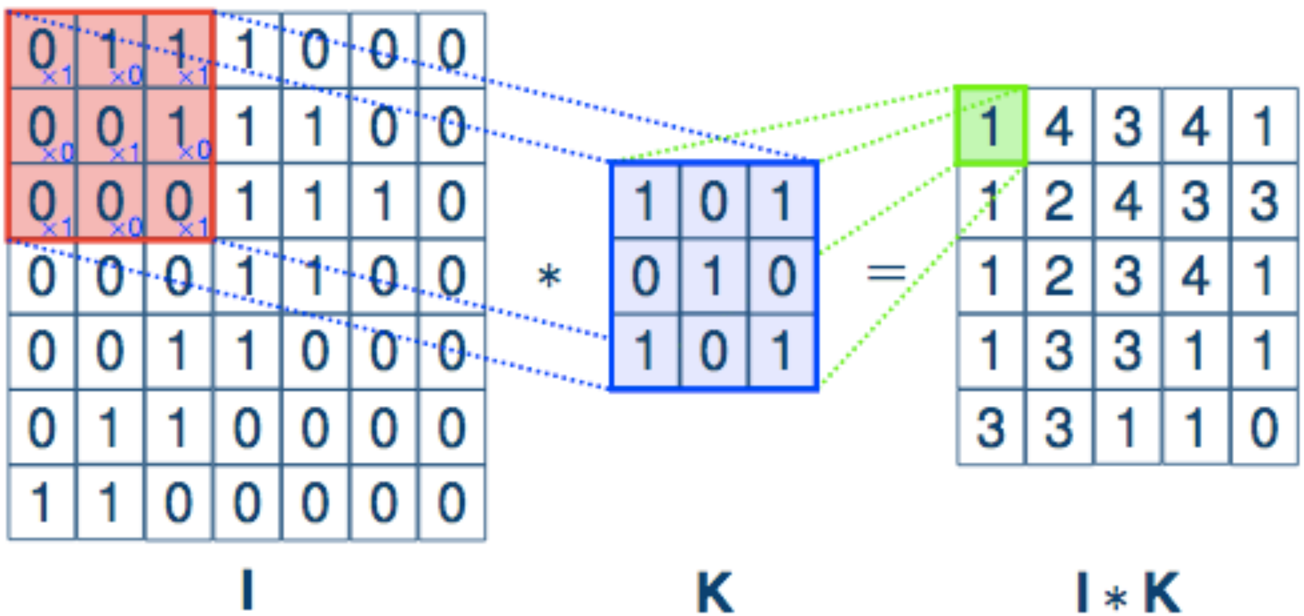
$[x^1\ x^2\ x^3 \ldots x^i]^T$
waveform heights

$[x^{11}\ x^{12} \ldots x^{1n}\ x^{21}\ x^{22} \ldots]^T$
pixel intensities

Feature learning



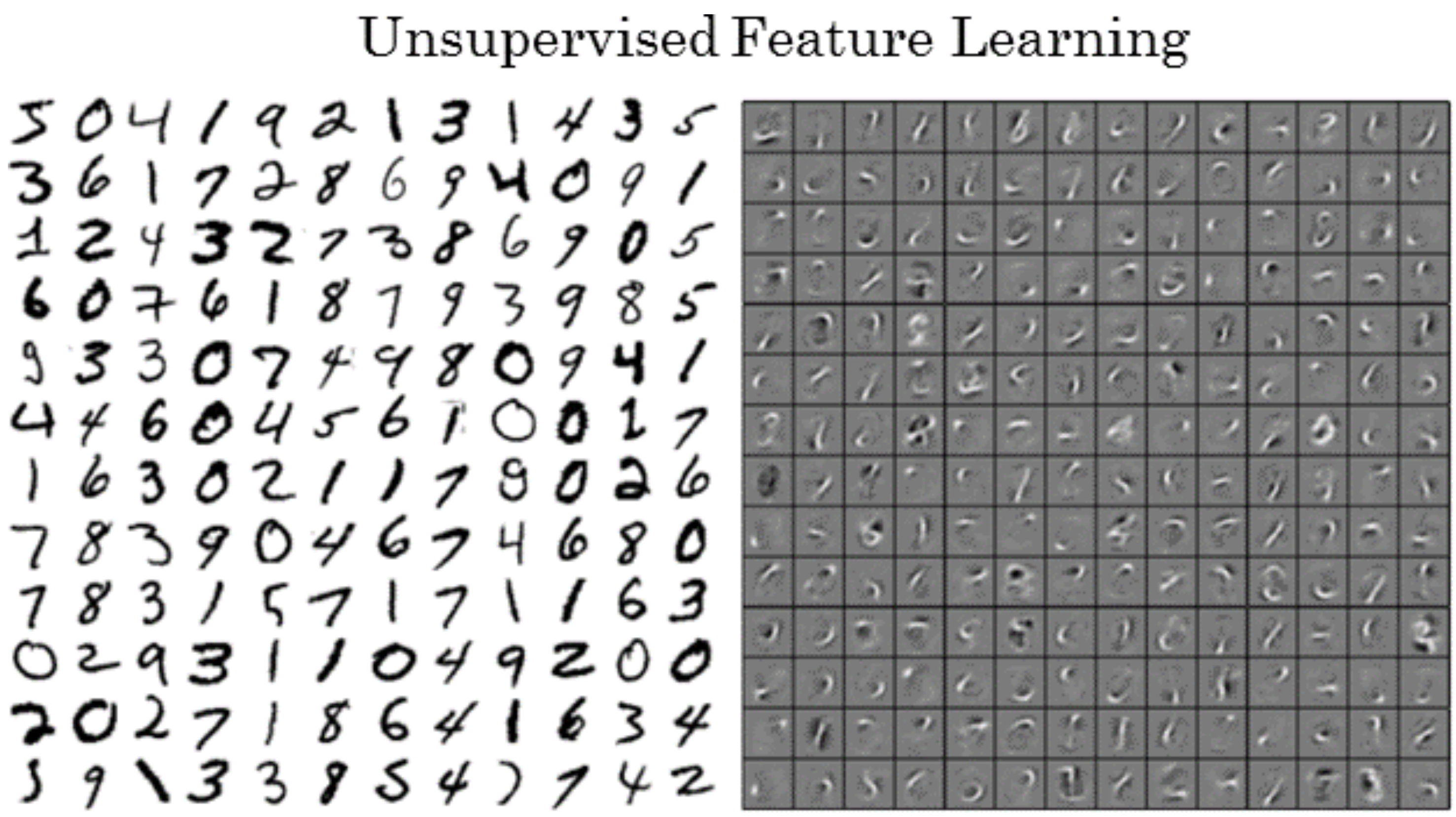
Convolution Example



Exploit structure, neighboring pixel dependence

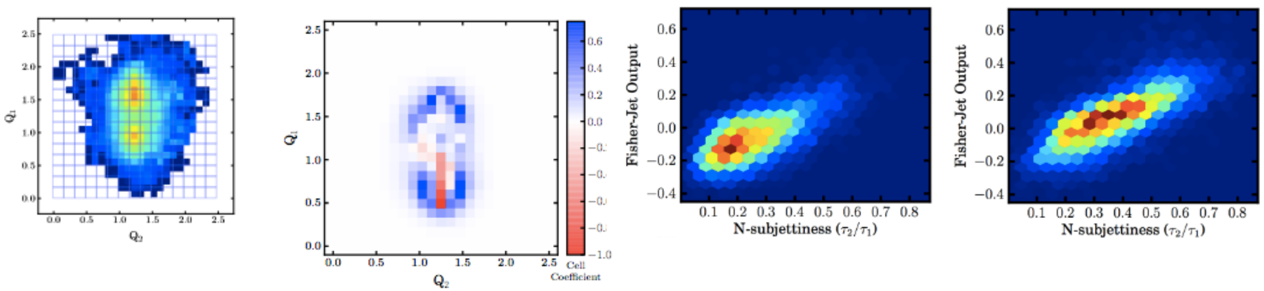
Filters

Convolutional Neural Networks:



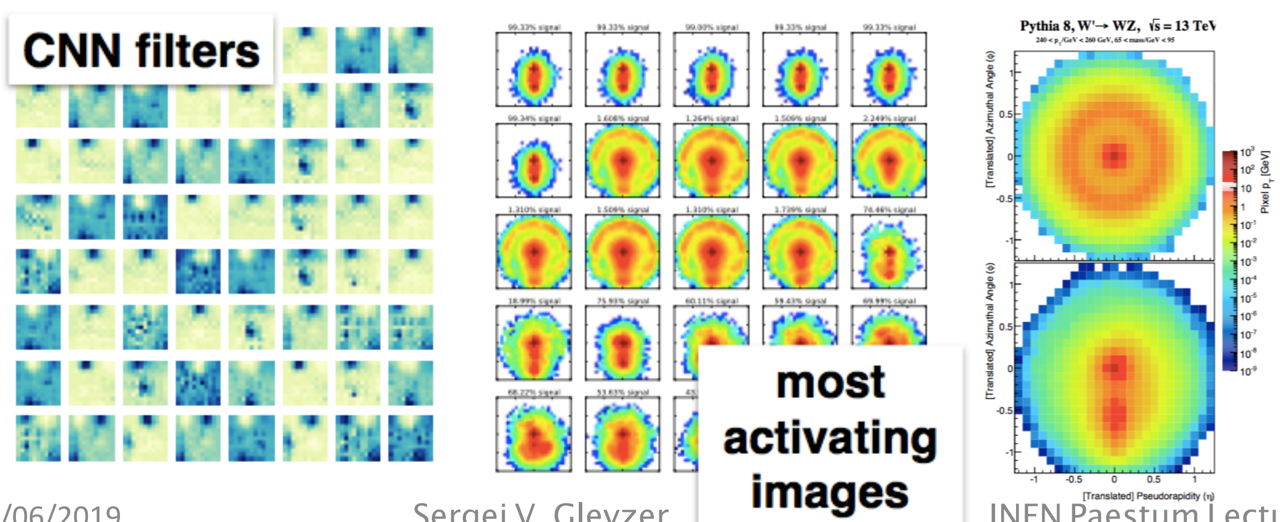
Jet Images

Link to Computer Vision



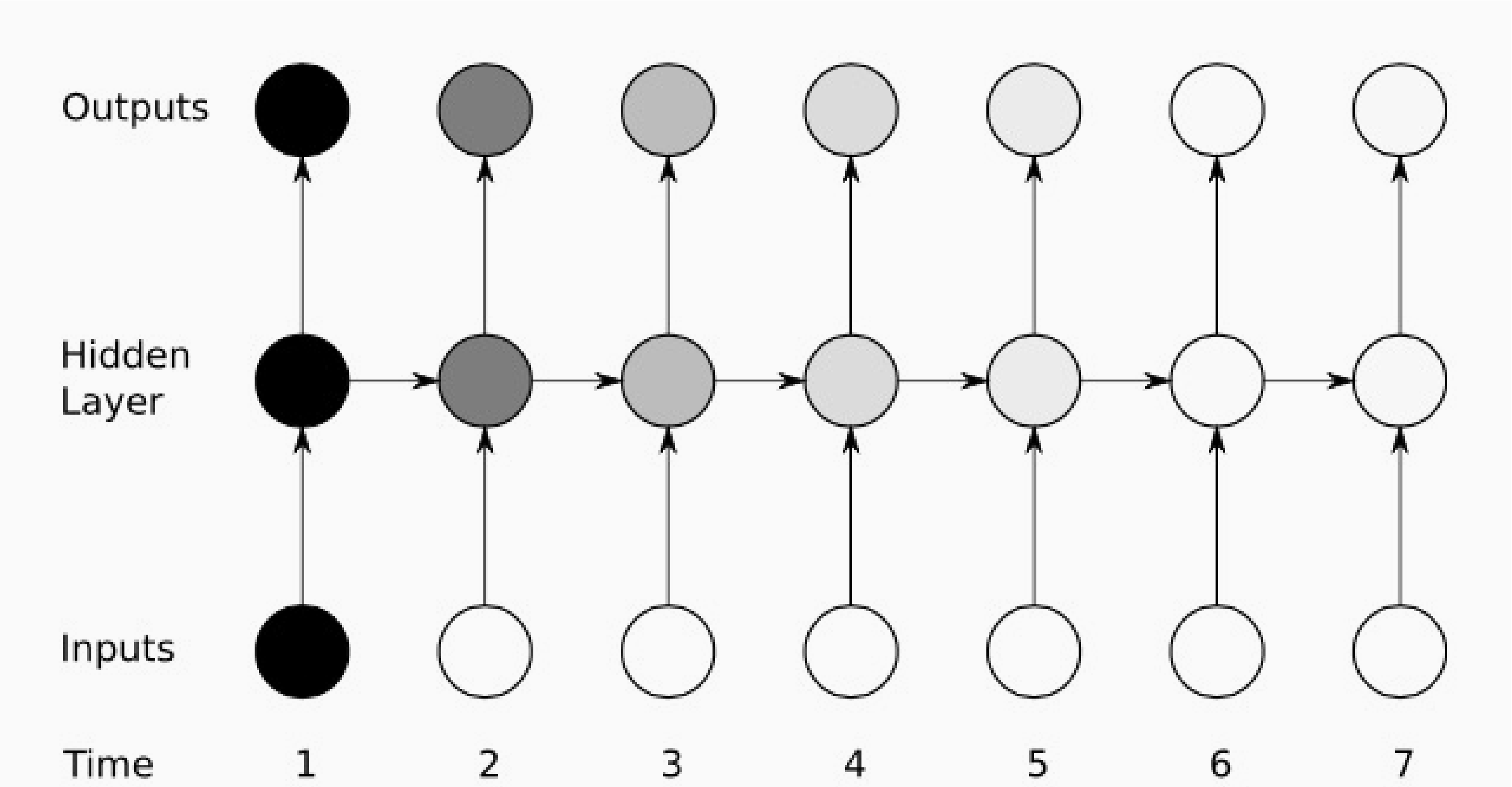
Cogan et al., 2014

Convolutional Neural Networks

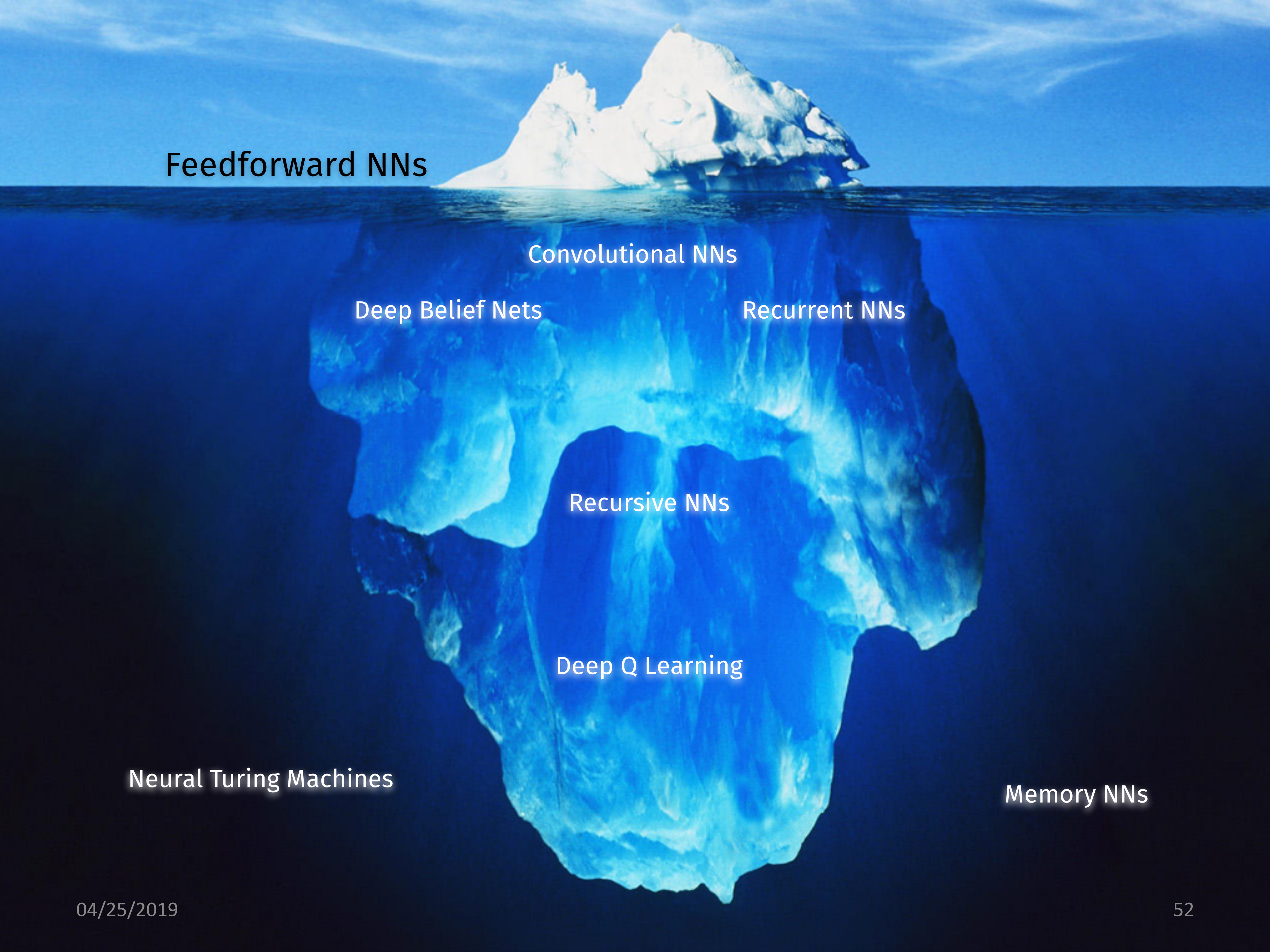


L. de Oliveira et al., 2015

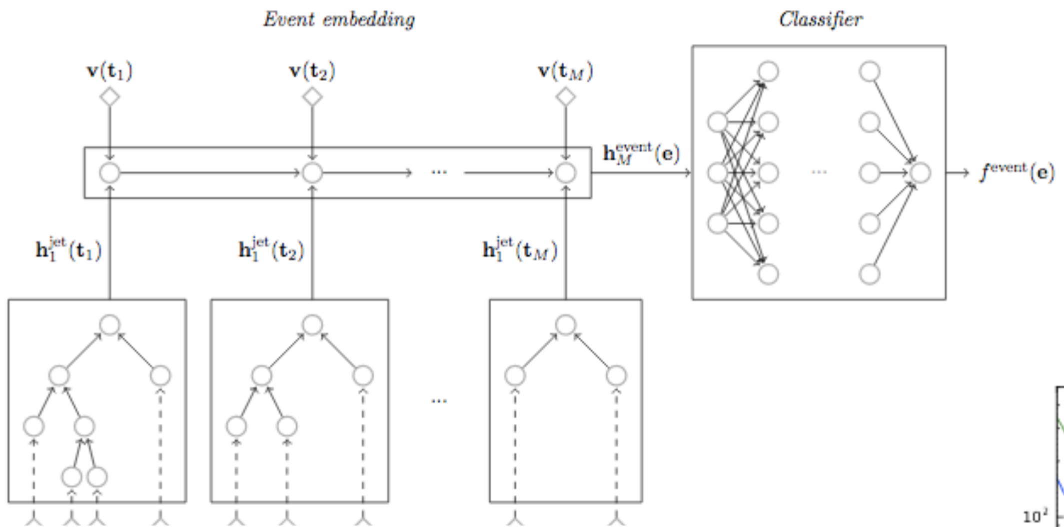
Recurrent NN



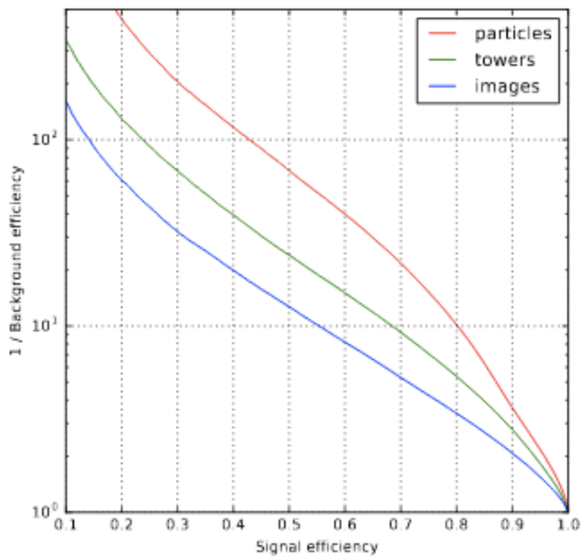
Cycles

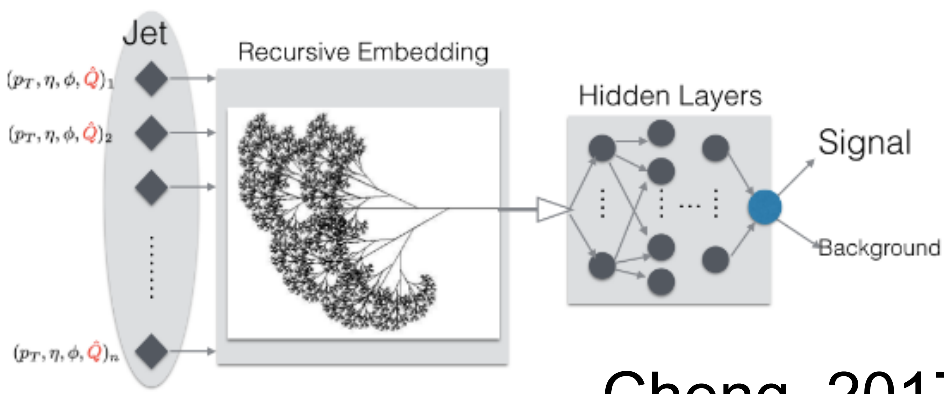

Feedforward NNs
Convolutional NNs
Deep Belief Nets
Recurrent NNs
Recursive NNs
Deep Q Learning
Neural Turing Machines
Memory NNs

Recursive-NNs

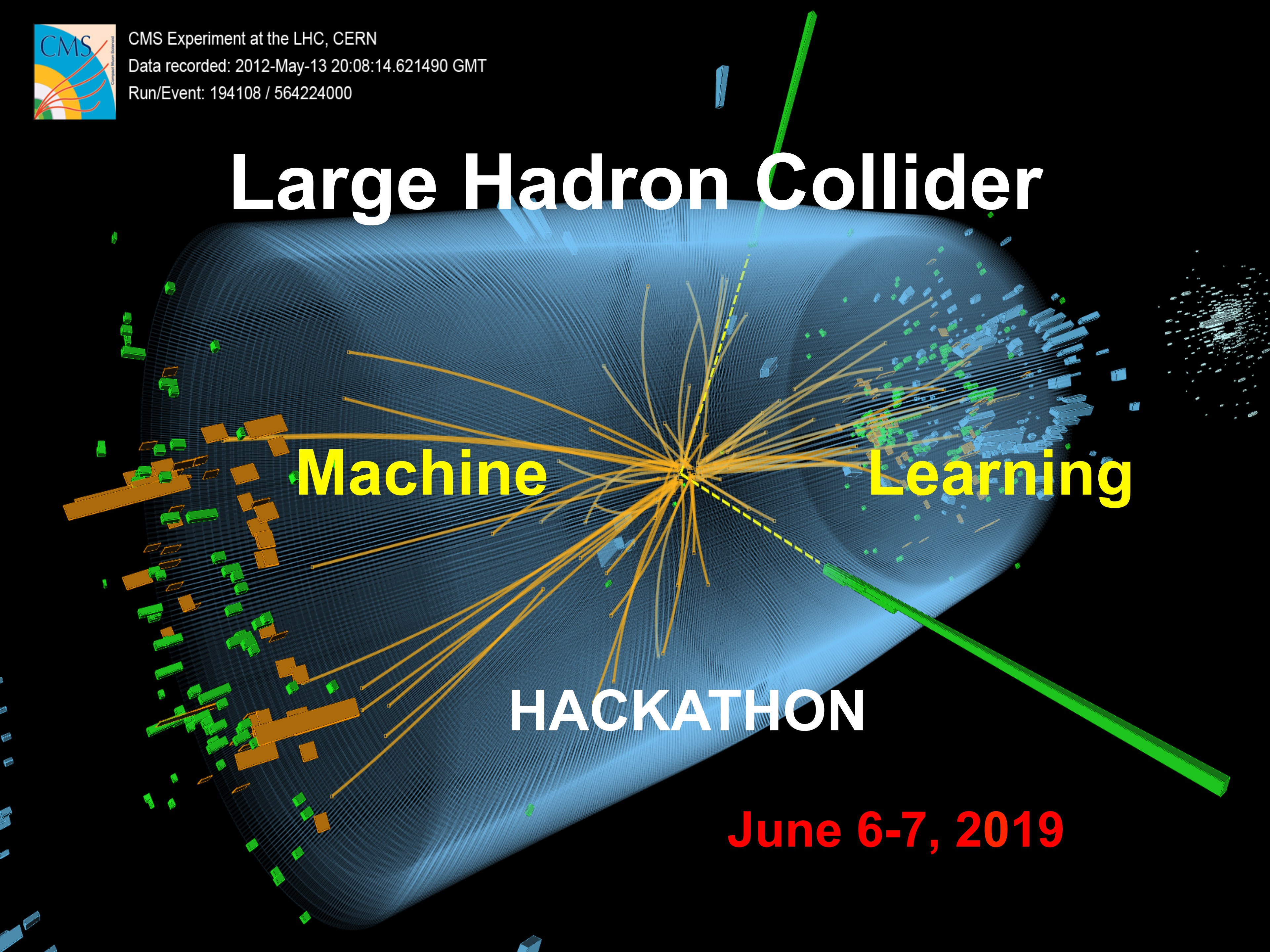


Louppe et al., 2017





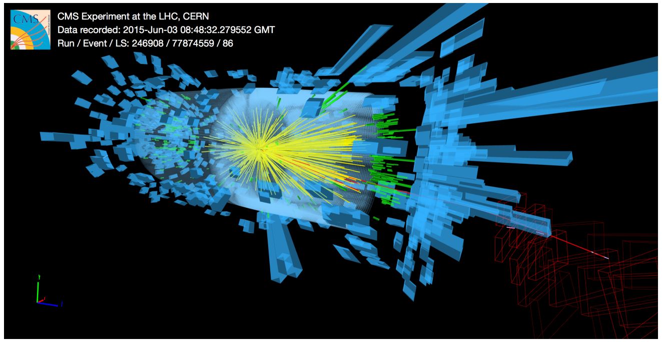
Cheng, 2017


CMS Experiment at the LHC, CERN
Data recorded: 2012-May-13 20:08:14.621490 GMT
Run/Event: 194108 / 564224000
Large Hadron Collider
Machine
Learning
HACKATHON
June 6-7, 2019

LHC ML Hackathon

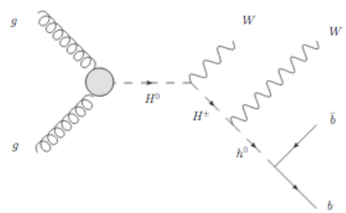
Challenge is to identify particles and events at the Large Hadron Collider

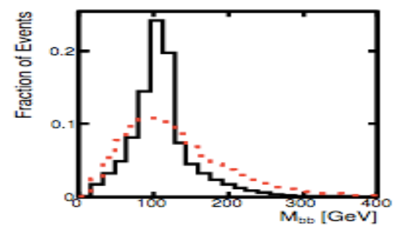
- Top accuracy wins
- Can use any algorithm or approach
- New machine learning ideas welcome
- Can form teams



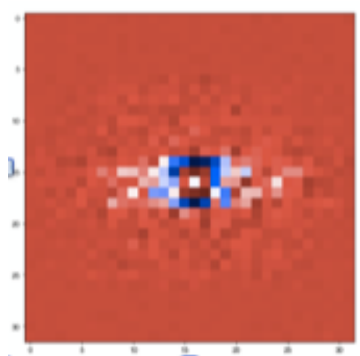
Two competitions

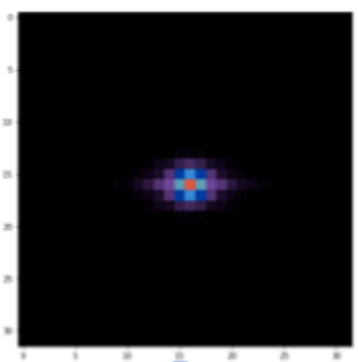
1) Identifying the Higgs Boson





2) Classifying Particle Images

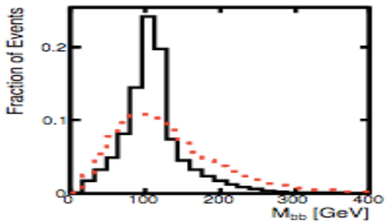




1) Higgs Challenge

Dataset:

- https://archive.ics.uci.edu/ml/datasets/HIGGS

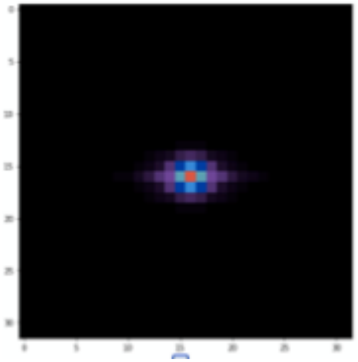


Paper with detailed description

- https://arxiv.org/pdf/1402.4735.pdf

2) Particle Images

Dataset:



- Download from git
- 32x32 Energy matrices

Identify electrons from photons:

- With any algorithm (i.e. for example neural networks)

Starter Kit

Download Starter Kit from git:

- https://github.com/iml-wg/lhcmlhackathon

Some Jupyter notebook examples in python on data visualization and basic classification

Evaluation Criteria

Duration:

- **Today – Friday 12pm**

Please provide solution

- **Code or Jupyter notebook**

Best ROC curve wins

- **Maximum area under the ROC curve**

Practical

- Can use any algorithm, tool or resource
- Train using your own systems
- We provide basic cpus with Jupyter kernel
- A small starter kit with examples on how to visualize data and run some benchmark algorithms

Jupyter Hub

- http://swan.cern.ch

- Temporary logins if needed

More questions?

- Slack channel:
  - http://bit.ly/2SxUV2C
- Email:
  - lhc-mlhackathon@cern.ch

Winners

- **Announced tomorrow afternoon**
- **Good luck!**