Machine Learning in Particle Physics

Harrison B. Prosper
Florida State University

ENHEP 19, Ain Shams University, Egypt

30 January, 2019

Topics

- Introduction
- Decision Trees
- Boosted Decision Trees
- Deep Neural Networks
- The Future of Machine Learning

INTRODUCTION

What is Machine Learning?

The use of computer-based algorithms for constructing useful *models* of data.

Machine learning algorithms fall into five broad categories:

1. Supervised Learning
2. Semi-supervised Learning
3. Unsupervised Learning
4. Reinforcement Learning
5. Generative Learning

Machine Learning

Method

Choose $f\,(x, \boldsymbol{w}^*)$ from $\boldsymbol{F}$ by minimizing the *average loss* (or *empirical risk*)

$$R(\boldsymbol{w}) = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{L}(\boldsymbol{y_i}, \boldsymbol{f_i}) + C(\boldsymbol{w}),$$

$\boldsymbol{F}$ = Function class

where

$T = \{(y_i\,,\, x_i)\}$	are training data,
f_i	$f\,(x, \boldsymbol{w})$ evaluated at x_i, and
$\boldsymbol{L}(\boldsymbol{y_i}, \boldsymbol{f_i})$,	the *loss function*, is a measure of the quality of the choice of function.

$C(\boldsymbol{w})$ is a constraint that guides the choice of $f\,(x, \boldsymbol{w})$.

Minimizing the Average Loss

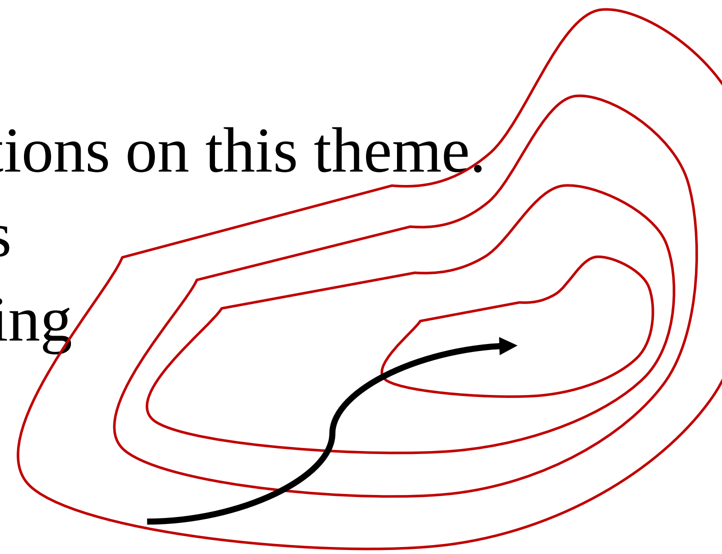
The average loss function defines a "landscape" in the *space of functions*, or, equivalently, the space of parameters.

The goal is to find the lowest point in that landscape, by moving in the direction of the *negative* gradient:

$$w_i \leftarrow w_i - \rho \frac{\partial R(w)}{\partial w_i}$$

Most minimization algorithms are variations on this theme.

Stochastic **G**radient **D**escent (SGD) uses random subsets (*batches*) of the training data to provide *noisy* estimates of the gradient.



Minimizing the Average Loss

Consider $R(w)$ in the limit $N \to \infty$

$$R(w) = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{L}(\boldsymbol{y}_i, \boldsymbol{f}_i) + C(w)$$
$$\to \int dx \int dy\, \boldsymbol{L}(\boldsymbol{y}, \boldsymbol{f})\, p(y, x)$$

Since $p(y|x) = p(y, x)/p(x)$ we can write

$$= \int dx\, p(x) \left[\int dy\, \boldsymbol{L}(\boldsymbol{y}, \boldsymbol{f})\, p(y|x)\right]$$

We have assumed the influence of the constraint to be negligible in this limit.

Minimizing the Average Loss

Now, consider the *quadratic* loss $\boldsymbol{L}(\boldsymbol{y}, \boldsymbol{f}) = (\boldsymbol{y} - \boldsymbol{f})^2$

$$R(w) = \int dx\, p(x) \left[\int dy\, \boldsymbol{L}(\boldsymbol{y}, \boldsymbol{f})\, p(y|x) \right]$$

$$= \int dx\, p(x) \left[\int dy\, (\boldsymbol{y} - \boldsymbol{f})^2\, p(y|x) \right]$$

and its minimization with respect to the choice of function $\boldsymbol{f}$.

Minimizing the Average Loss

If we change the function $\boldsymbol{f}$ by a small *arbitrary* function $\boldsymbol{\delta f}$ a small change

$$\boldsymbol{\delta R} = 2\int dx\, p(x)\boldsymbol{\delta f}\left[\int dy(y - f)p(y|x)\right]$$

will be induced in R. In general, $\boldsymbol{\delta R} \neq 0$.

However, if the function $\boldsymbol{f}$ is flexible enough then we shall be able to reach the minimum of R, where $\boldsymbol{\delta R} = 0$.

But, in order to guarantee that $\boldsymbol{\delta R} = 0$ *for all* $\boldsymbol{\delta f}$ and *for all* $\boldsymbol{x}$ the quantity in brackets must be *zero*. This implies:

$$\boxed{f(x, w^*) = \int y\, p(y \mid x)\, dy}$$

Classification

According to *Bayes' theorem*

$$p(y|x) = \frac{p(x\,|y)\,p(y)}{\int p(x\,|y)\,p(y)dy}$$

Let's assign the *target* value $\boldsymbol{y = 1}$ to objects of class ***S*** and the target value $\boldsymbol{y = 0}$ to objects of class ***B***.

Then

$$\boxed{\begin{aligned} f(x, w^*) &= \int y\, p(y\,|\,x)\, dx = p(1|x) \\ &\equiv p(\boldsymbol{S}|x) \end{aligned}}$$

That is, the function $f(x, w^*)$ equals the *class probability*.

Classification

1. In summary, the result

$$f(x, w^*) = p(S|x) = \frac{p(x|S)p(S)}{p(x|S)p(S) + p(x|B)p(B)}$$

depends *only* on the *form* of the loss function provided that:

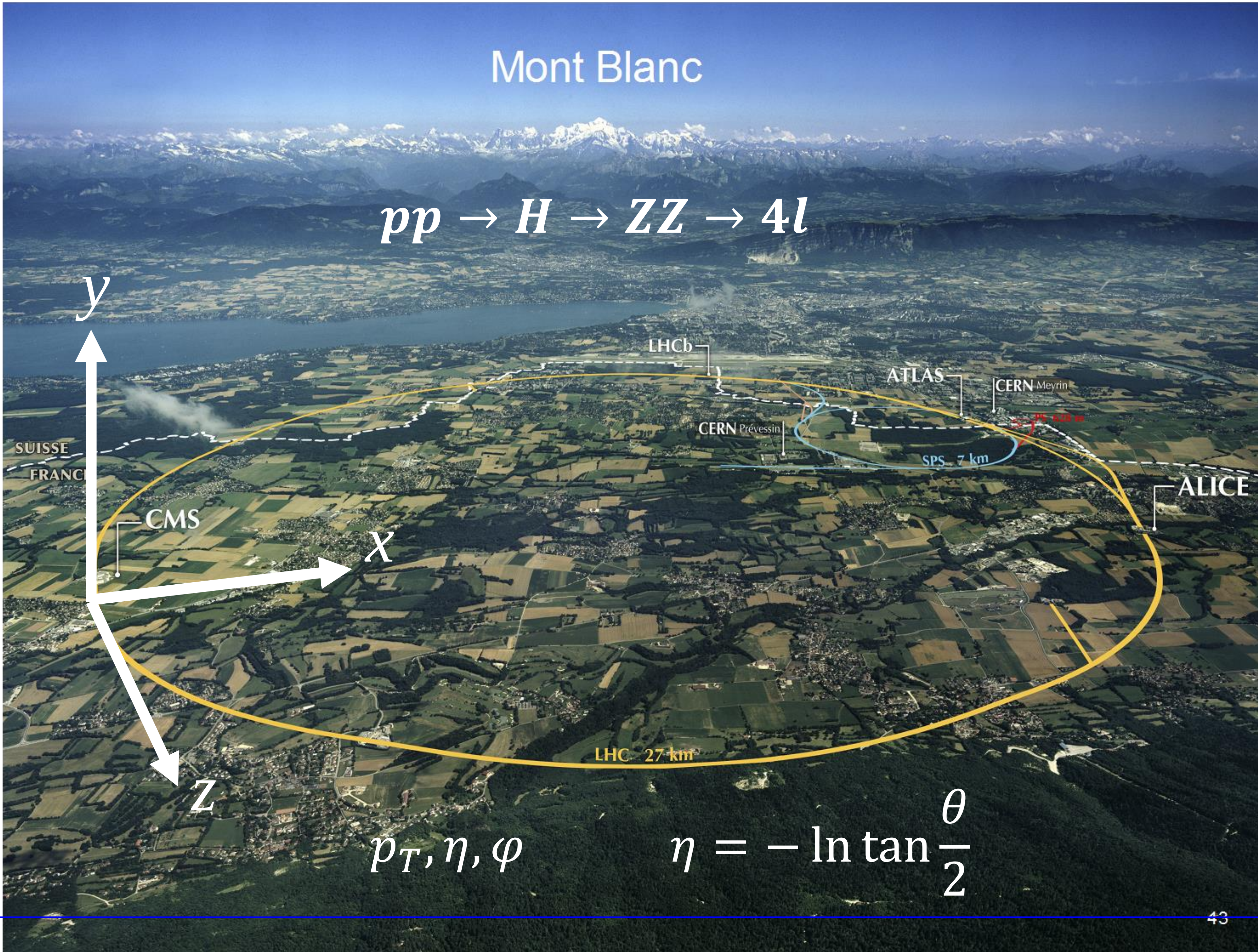
   1. the training data are sufficiently numerous,
   2. the function $f(x, \mathbf{w})$ is sufficiently flexible, and
   3. the minimum of the average loss, R, can be found.

2. Note, if $p(S) = p(B)$, we arrive at the *discriminant*

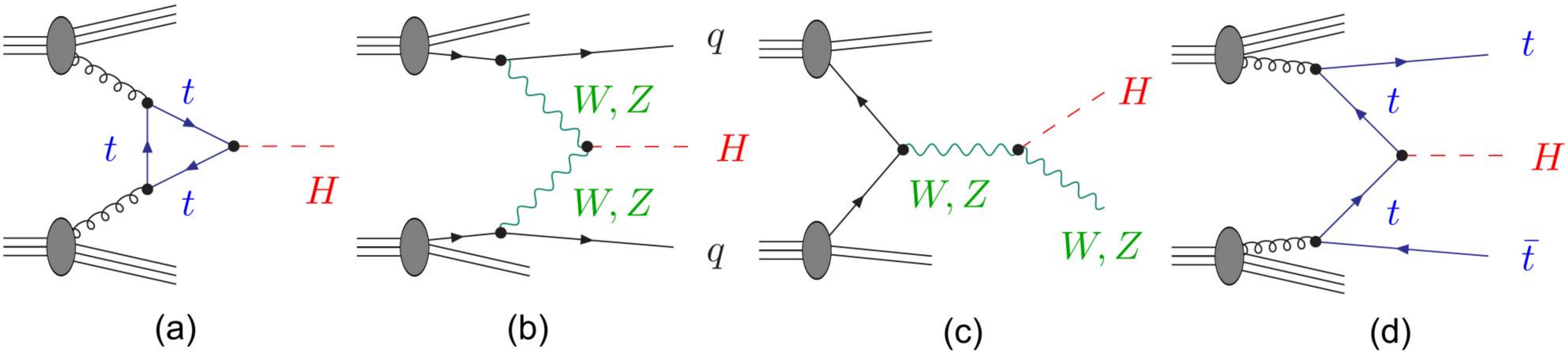
$$D(x) = \frac{p(x|S)}{p(x|S) + p(x|B)} \equiv \frac{s(x)}{s(x) + b(x)}$$

DECISION TREES

$$pp \to H \to ZZ \to 4l$$


Mont Blanc
$pp \to H \to ZZ \to 4l$
y
x
z
SUISSE
FRANCE
LHCb
ATLAS
CERN Meyrin
CERN Prévessin
PS 628 m
SPS 7 km
ALICE
CMS
LHC 27 km
p_T, η, φ
$\eta = -\ln \tan \frac{\theta}{2}$

Higgs Boson Production

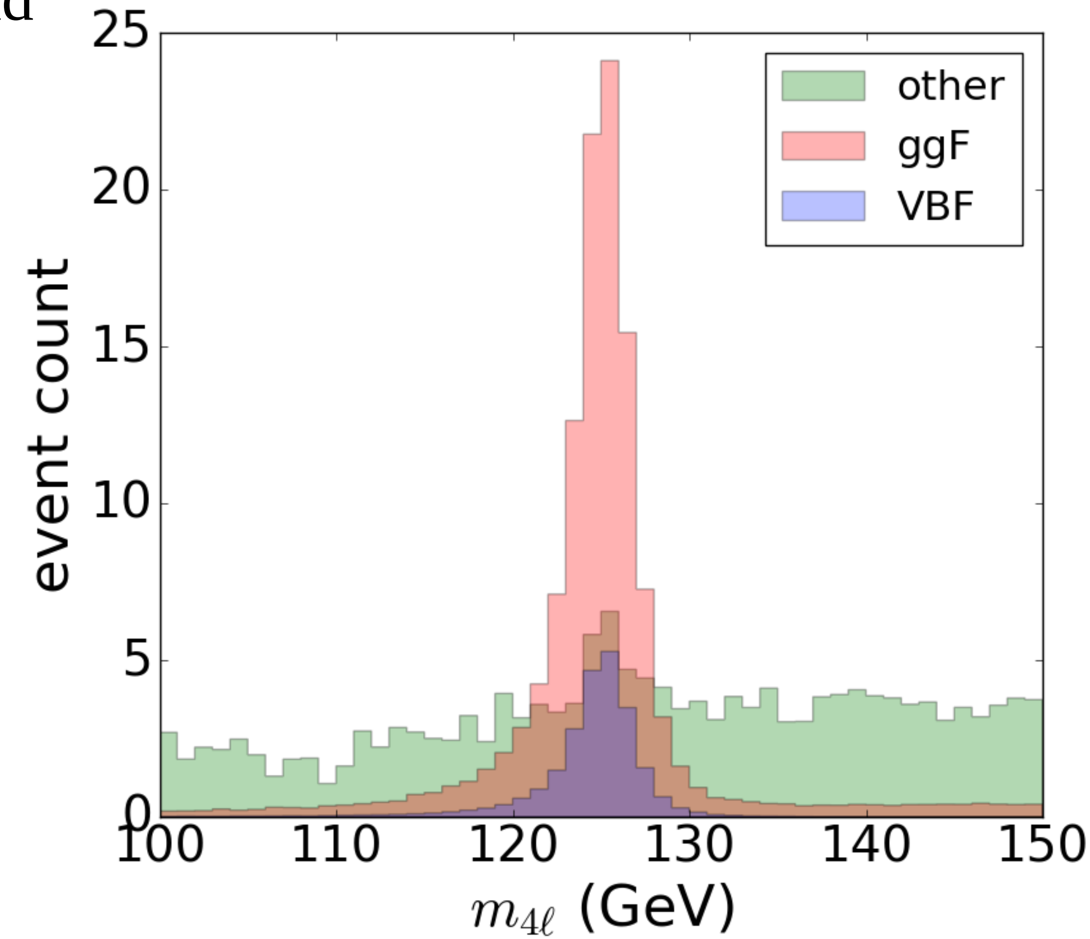


Process		$\boldsymbol{\sigma \times BR}$ **(fb)**
(a) Gluon gluon fusion	(ggF)	12.18
(b) Vector boson fusion	(VBF)	1.044
(c) Associated production	(VH)	1.047
(d) Top anti-top fusion	(ttH)	0.393

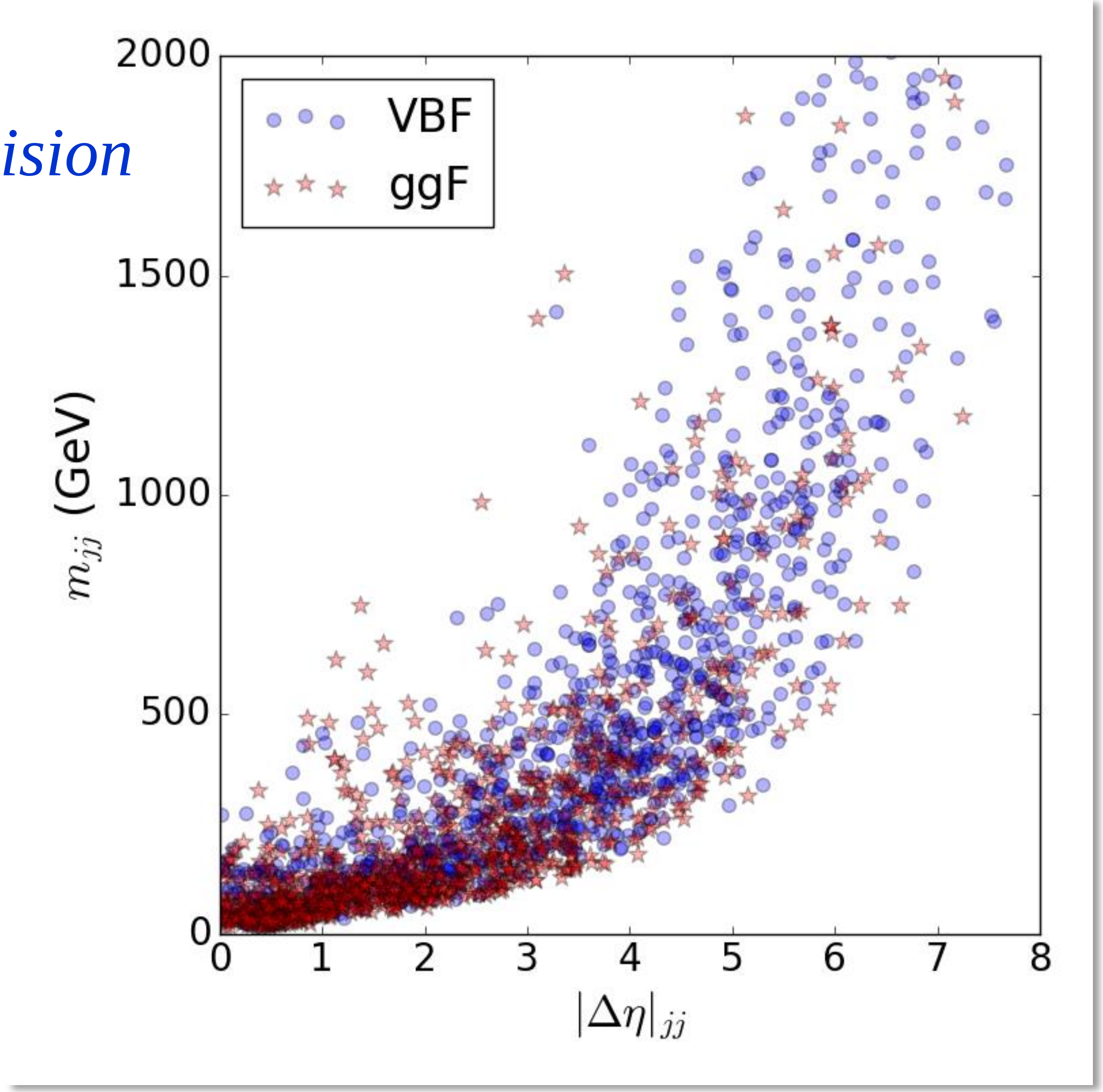
Before event selection, background ~ 1700 times larger!

VBF vs. ggF Higgs Boson Production

The Higgs boson mass is an excellent discriminant between Higgs boson events and other Standard Model events. But, clearly it is not for separating VBF events from ggF events. For that we need other observables.



We shall use *decision trees* with the variables $|\Delta\eta|_{jj}, m_{jj}$ to try to separate $VV \to H$ (VBF) from $gg \to H$ (ggF).



Decision Trees

A decision tree (DT) is a set of **if then else** statements that form a tree-like structure.

Algorithm: recursively partition the space into regions of diminishing *impurity*.

A common impurity measure is the *Gini Index*:

$p\,(1 - p)$, where p is the *purity*

$p = S\,/\,(S + B)$

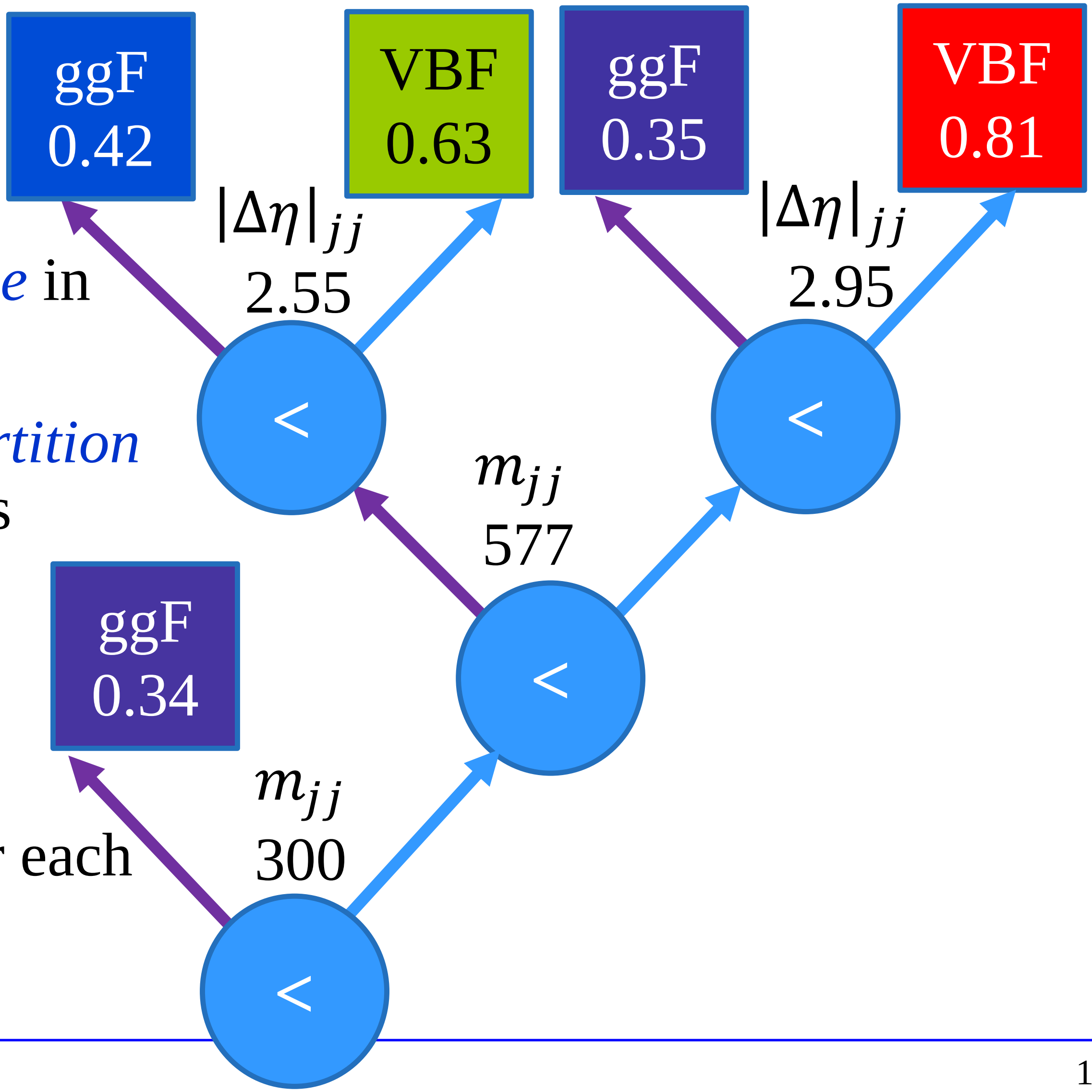
$p = 0$ or 1: maximum purity

$p = 0.5$: maximum impurity

(Corrado Gini, 1884-1965)

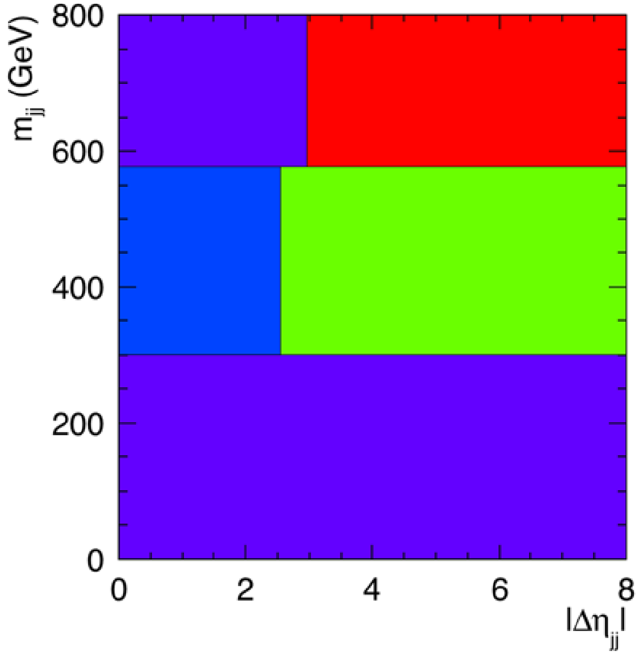
Decision Trees

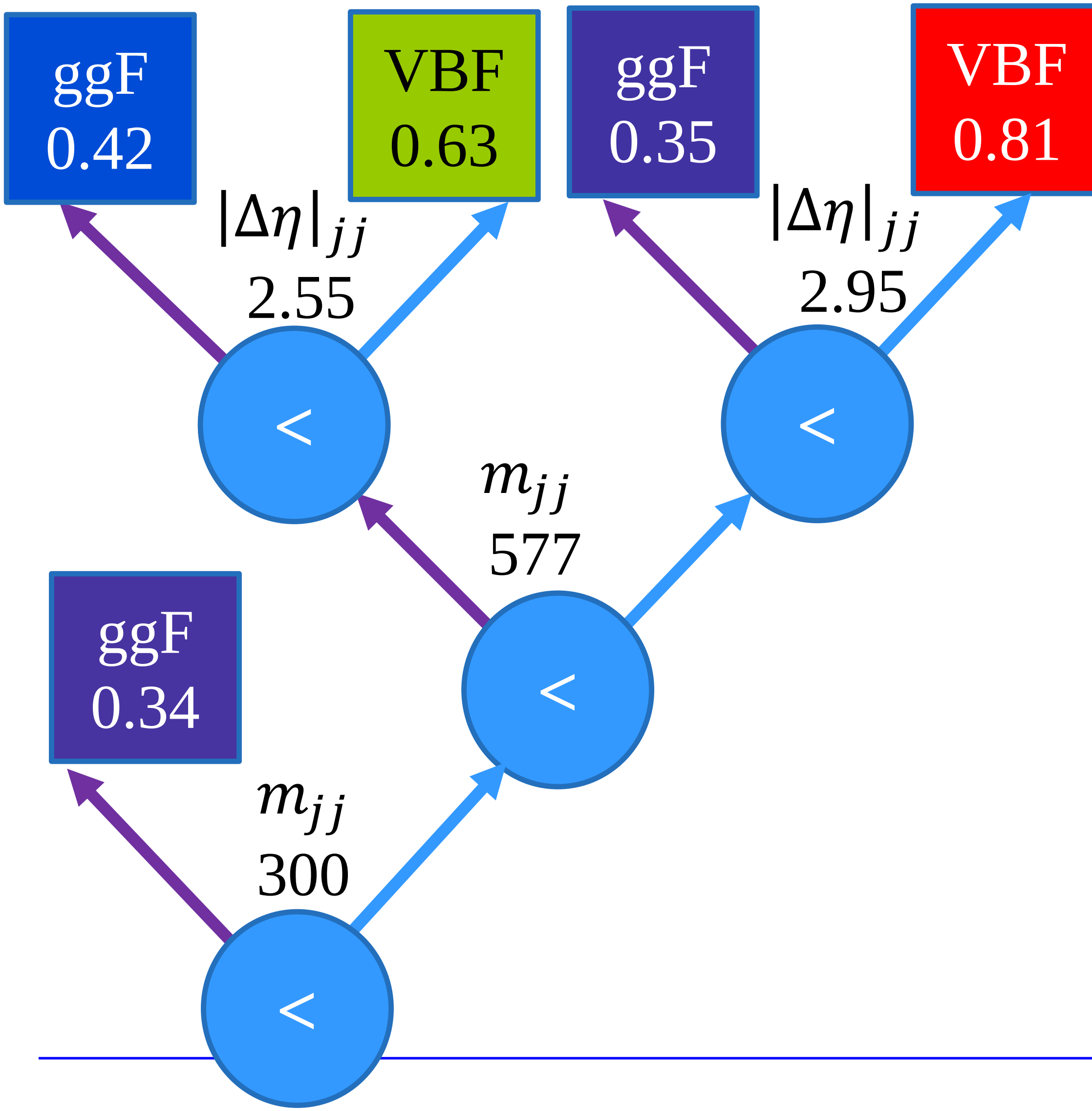
1. For each variable, find the partition ("cut") that gives the greatest *decrease* in impurity.
2. Choose the *best partition* among all partitions and split the data along that partition into *two* subsets.
3. Repeat 1. and 2. for each subset of data.



Decision Trees

Geometrically, a decision tree is just a *d-dimensional* histogram in which the bins are created *recursively*.





Decision Trees

Unfortunately, decision trees are *unstable*!

BOOSTED DECISION TREES

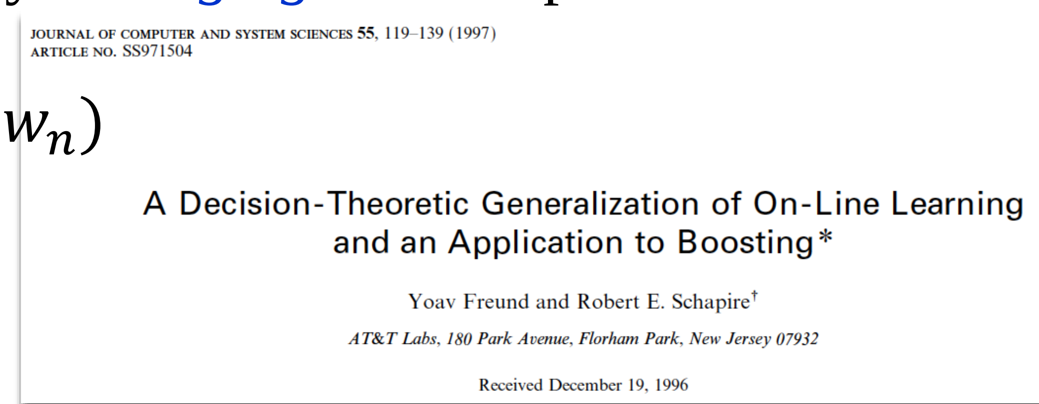
Boosted Decision Trees: AdaBoost

In 1997, AT&T researchers Freund and Schapire [Journal of Computer and Sys. Sci. **55** (1), 119 (1997)] published an algorithm that produced highly effective classifiers by combining many mediocre ones!

Their algorithm, called AdaBoost, was the first successful method to *boost* (i.e., enhance) the performance of poorly performing classifiers by *averaging* their outputs:

$$f(x, w) = \sum_{n=1}^{N} a_n \, T(x, w_n)$$

T = tree


JOURNAL OF COMPUTER AND SYSTEM SCIENCES **55**, 119–139 (1997)
ARTICLE NO. SS971504

A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting*

Yoav Freund and Robert E. Schapire†
AT&T Labs, 180 Park Avenue, Florham Park, New Jersey 07932

Received December 19, 1996

Averaging Methods

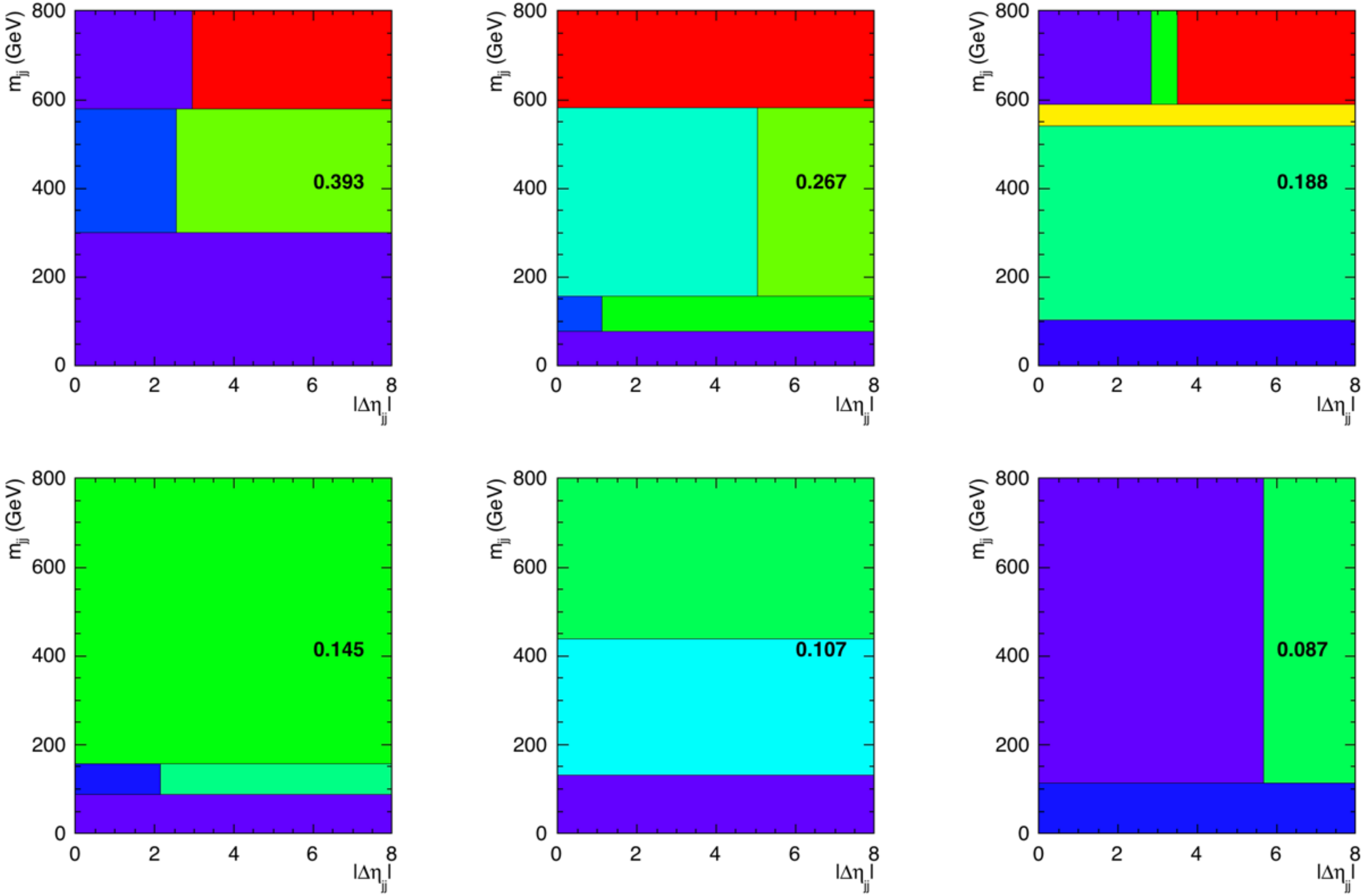
The most popular methods (used mostly with decision trees) are:

- Bagging: each tree is trained on a bootstrap* sample drawn from the training set
- Random Forest: bagging with randomized trees
- Boosting: each tree trained on a different reweighting of the training set

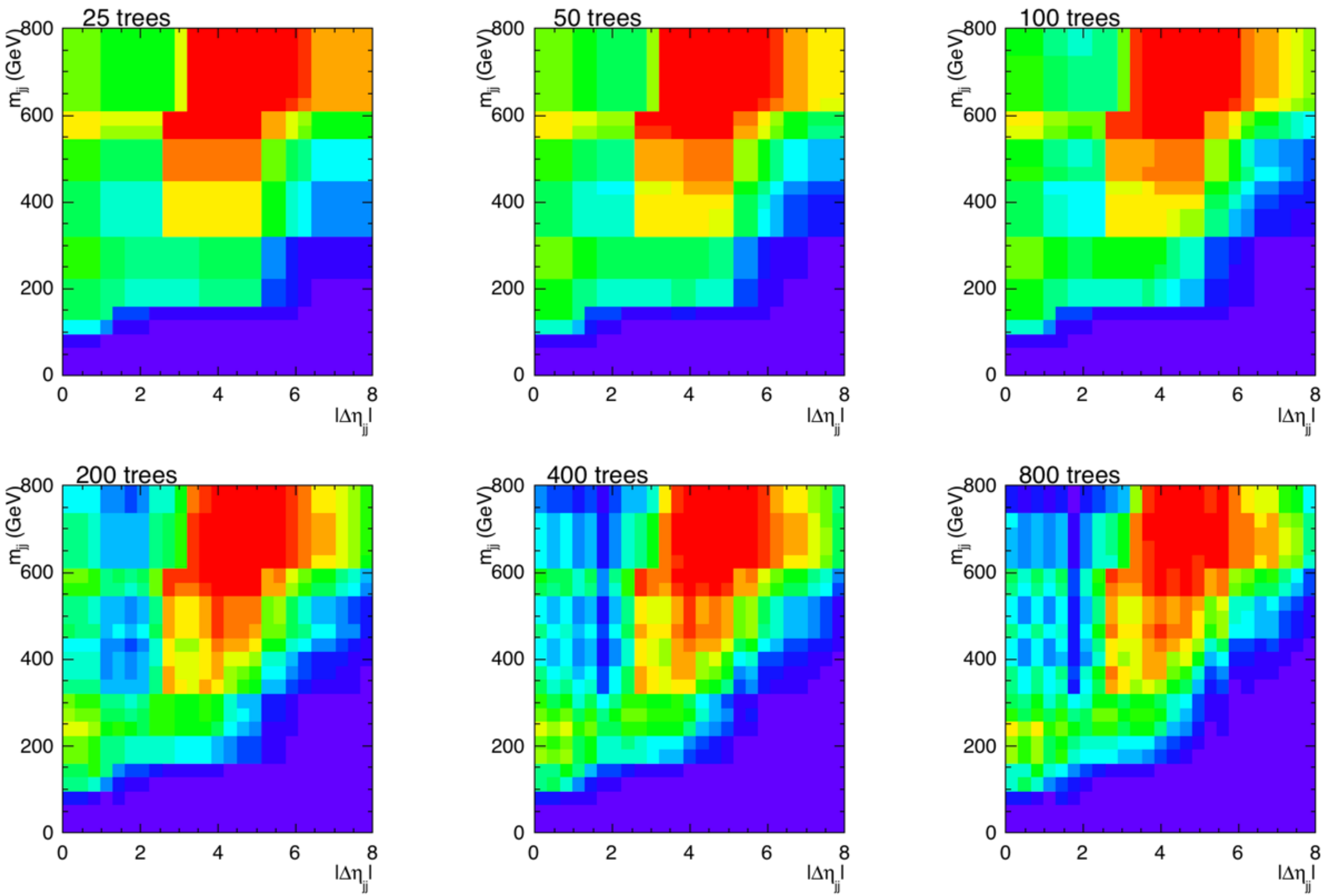
*A bootstrap sample is a sample of size N drawn, *with replacement*, from another of the same size. Duplicates can occur and are allowed.

EXAMPLE: $pp \rightarrow H \rightarrow ZZ \rightarrow 4l$

VBF vs. ggF: First 6 Decision Trees

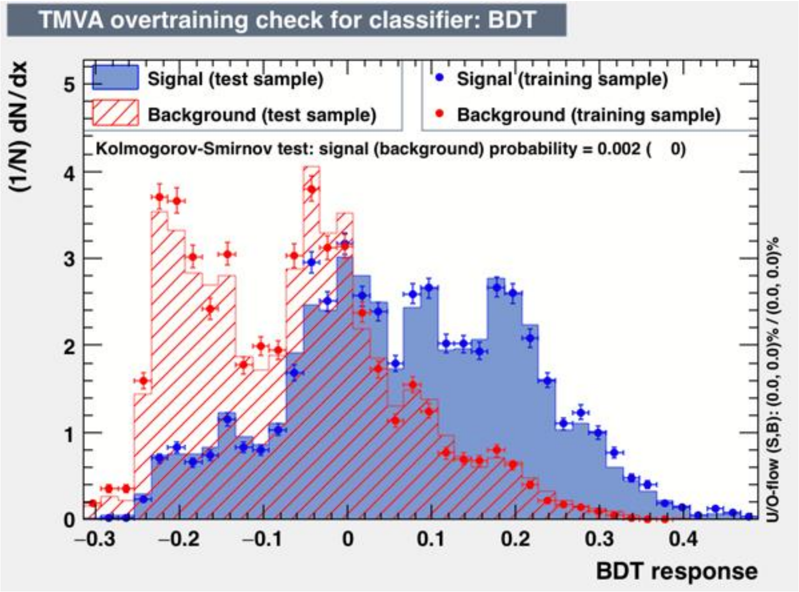


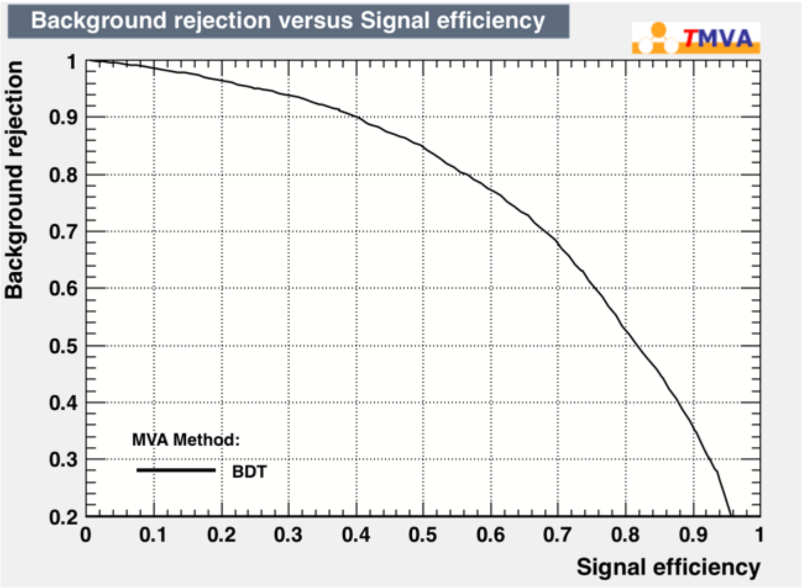
VBF vs. ggF: <Decision Trees>



VBF vs. ggF: Results

$$x = |\Delta\eta|_{jj}$$
$$y = m_{jj}$$



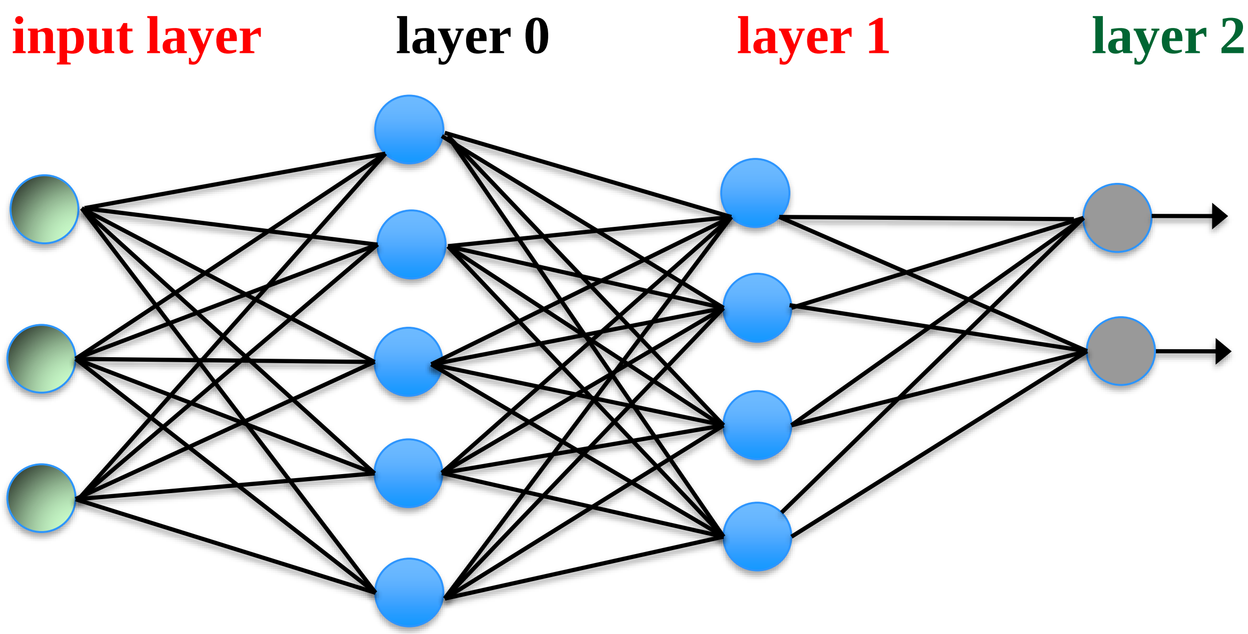


$$BDT(x,y) = \sum_{k=1}^{800} \alpha_k f(x,y,w_k)$$

Fraction of ggF events rejected for a given fraction of VBF events accepted.

DEEP NEURAL NETWORKS

Deep Neural Networks



A (3, 5, 4, 2)-DNN

$$o = g(\boldsymbol{b_2} + \boldsymbol{w_2} h(\boldsymbol{b_1} + \boldsymbol{w_1} h(\boldsymbol{b}_0 + \boldsymbol{w_0} x)))$$

$h(z)$ = ReLU(z) [= max(0, z)], tanh(z)

$g(z)$ = Identity(z), logistic(z) = 1/[1 + exp(-z)]

Deep Neural Networks

- In 2006, Hinton, Osindero, and Teh[1] (U. of Toronto) succeeded in training a deep neural network for the first time using a very clever training algorithm.

- But, in 2010, Cireşan *et al.*[2] showed that cleverness was not needed! Just a lot of computing power!
- The authors succeeded in training a DNN with architecture (**784**, 2500, 2000, 1500, 1000, 500, **10**) to classify the hand-written digits in the MNIST database.

1 Hinton, G. E., Osindero, S. and Teh, Y. (HOT), A fast learning algorithm for deep belief nets, Neural Computation 18, 1527-1554.
2 Cireşan DC, Meier U, Gambardella LM, Schmidhuber J. , Deep, big, simple neural nets for handwritten digit recognition. Neural Comput. 2010 Dec; 22 (12): 3207-20. http://yann.lecun.com/exdb/mnist/

Deep Neural Networks

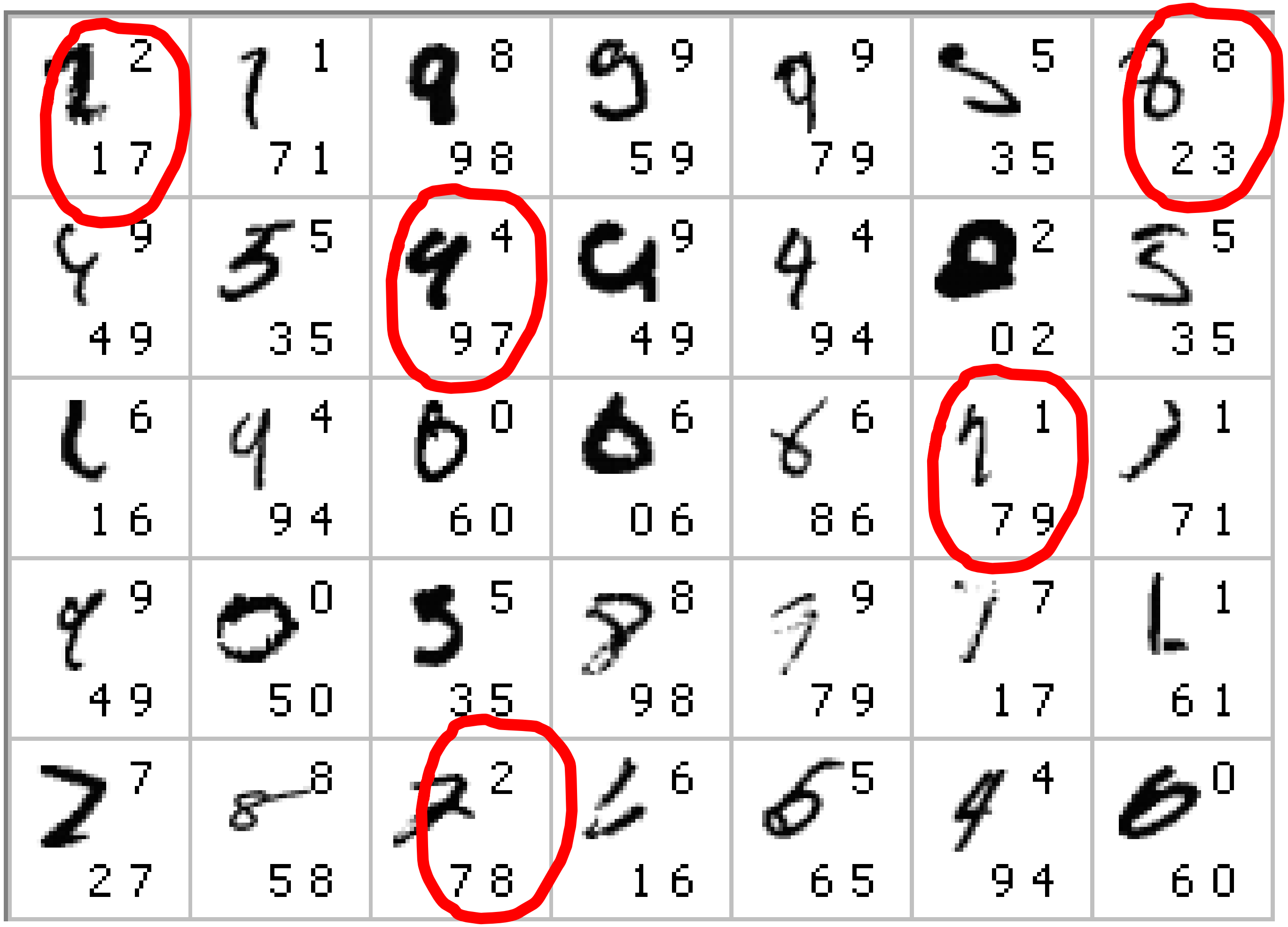
- The database comprises 60,000 $28 \times 28 = 784$ pixel images for training and validation, and 10,000 for testing.

- The error rate of their (**784**, 2500, 2000, 1500, 1000, 500, **10**) DNN was 35 images out of 10,000.

- The misclassified images are shown on the next slide.

2 Cireşan DC, Meier U, Gambardella LM, Schmidhuber J. , Deep, big, simple neural nets for handwritten digit recognition. Neural Comput. 2010 Dec; 22 (12): 3207-20. http://yann.lecun.com/exdb/mnist/

(784, 2500, 2000, 1500, 1000, 500, 10)

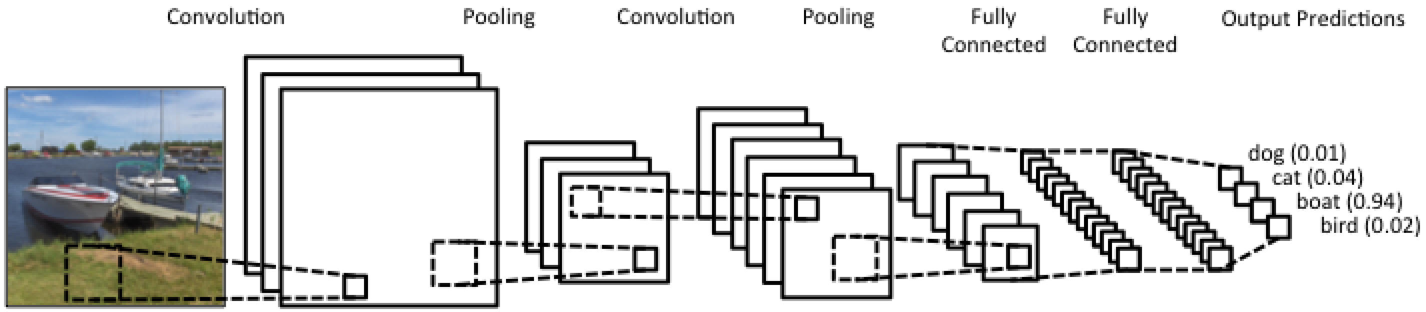


Upper right: correct answer; lower left answer of highest DNN output; lower right answer of next highest DNN output.

Convolutional Neural Networks

Many of the remarkable breakthroughs in tasks such as face recognition use a type of DNN called a convolutional neural network (CNN).

CNNs are *functions* that compress data and classify objects using their compressed representations using a fully connected NN. The compression dramatically reduces the dimensionality of the space to be searched.



Source: https://www.clarifai.com/technology

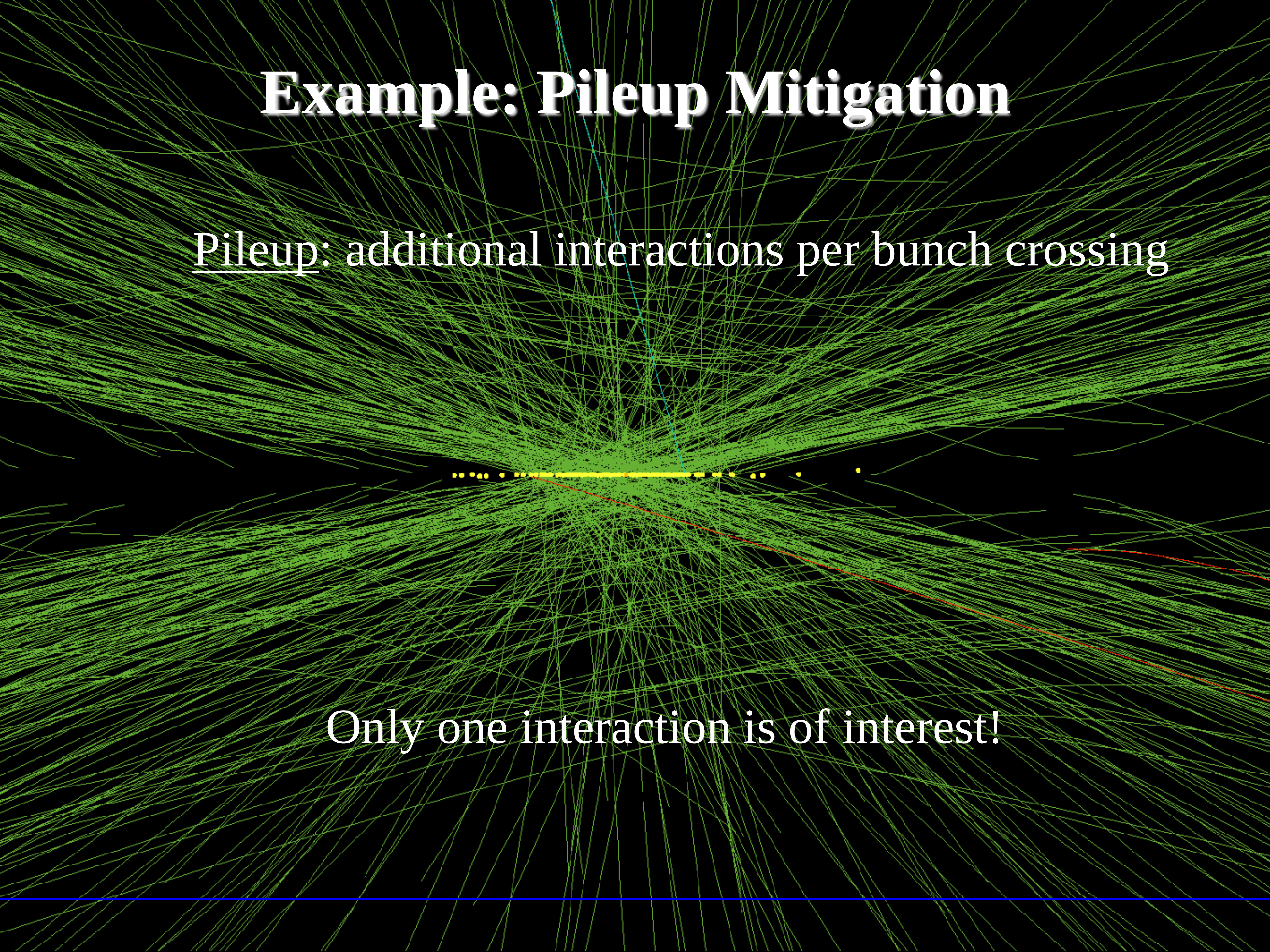
THE FUTURE OF MACHINE LEARNING

The Future of Machine Learning

- In Particle Physics
  - The standard approach to classification and regression problems is to use physical insight to arrive at suitable variables and functions.
  - However, in recent work, machine learning has matched or *outperformed* the work of expert physicists.

- In Society
  - The most far-reaching application of machine learning is *artificial intelligence* (AI), a technological development that could transform our societies more profoundly than did the Industrial Revolution.

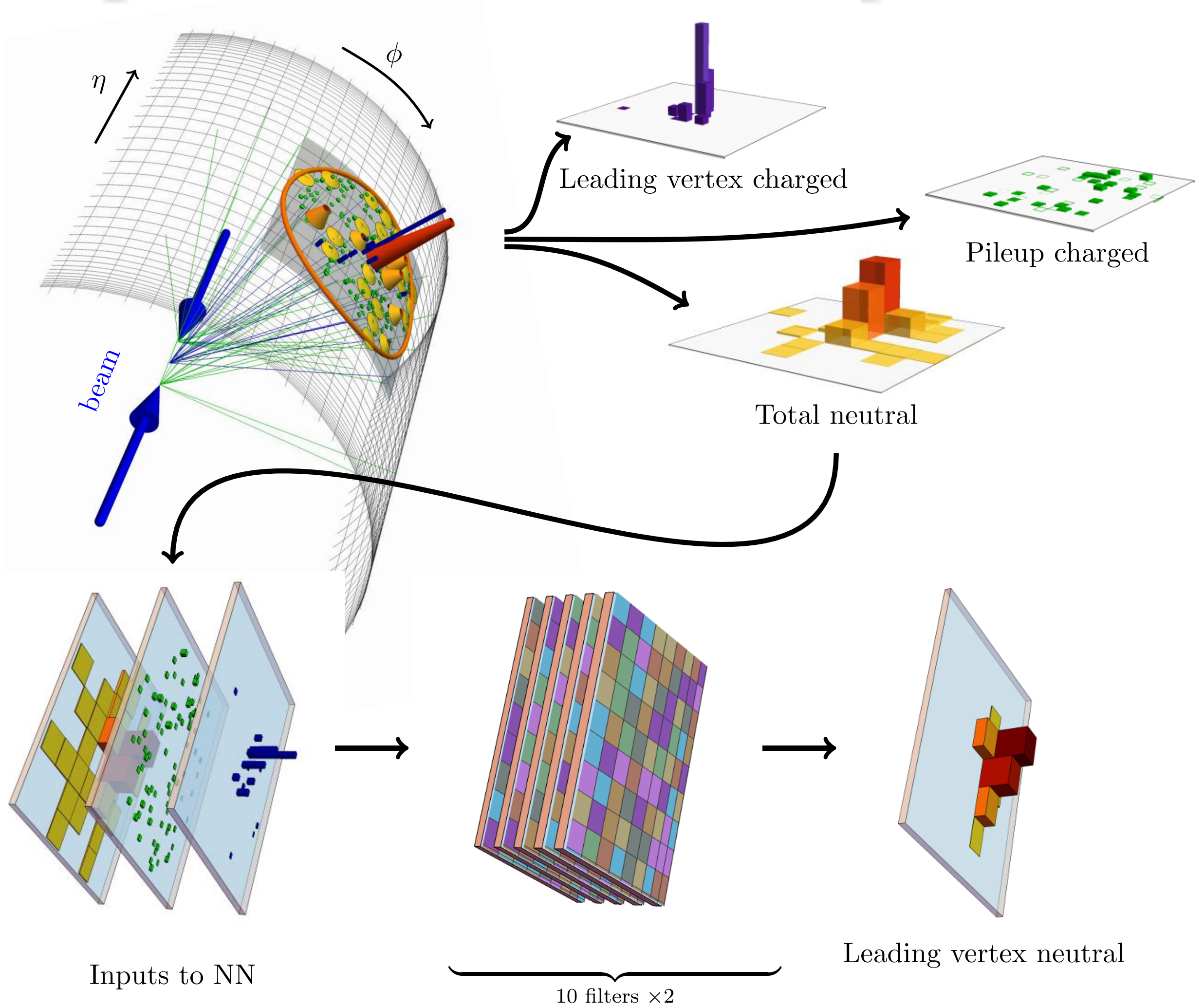
Example: Pileup Mitigation



Pileup: additional interactions per bunch crossing

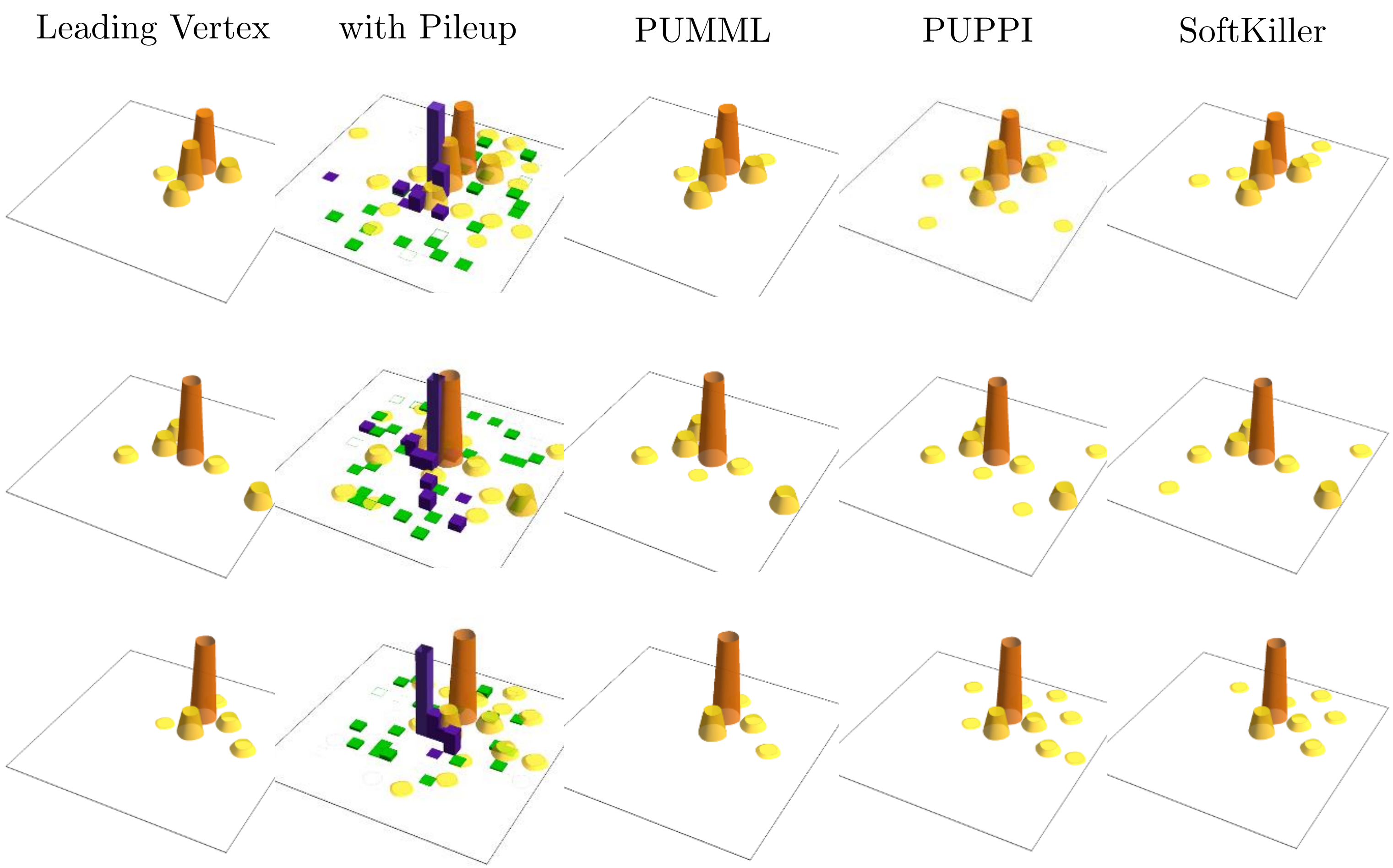
Only one interaction is of interest!

Pileup Mitigation Example: PUMML



* Pileup Mitigation with Machine Learning (PUMML)
Metodiev, Komiske, Nachman, Schwarz, JHEP 12 (2017) 051, arXiv:1707.08600

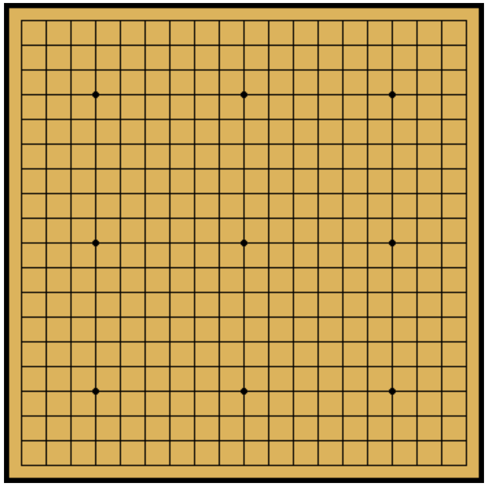
Pileup Mitigation Example: PUMML



* Pileup Mitigation with Machine Learning (PUMML)
Metodiev, Komiske, Nachman, Schwarz, JHEP 12 (2017) 051, arXiv:1707.08600

AlphaGo 4, Homo sapiens 1

2016 – Google's AlphaGo program beats Go champion Lee Sodol.



Photograph: Yonhap/Reuters

ARTICLE

doi:10.1038/nature24270

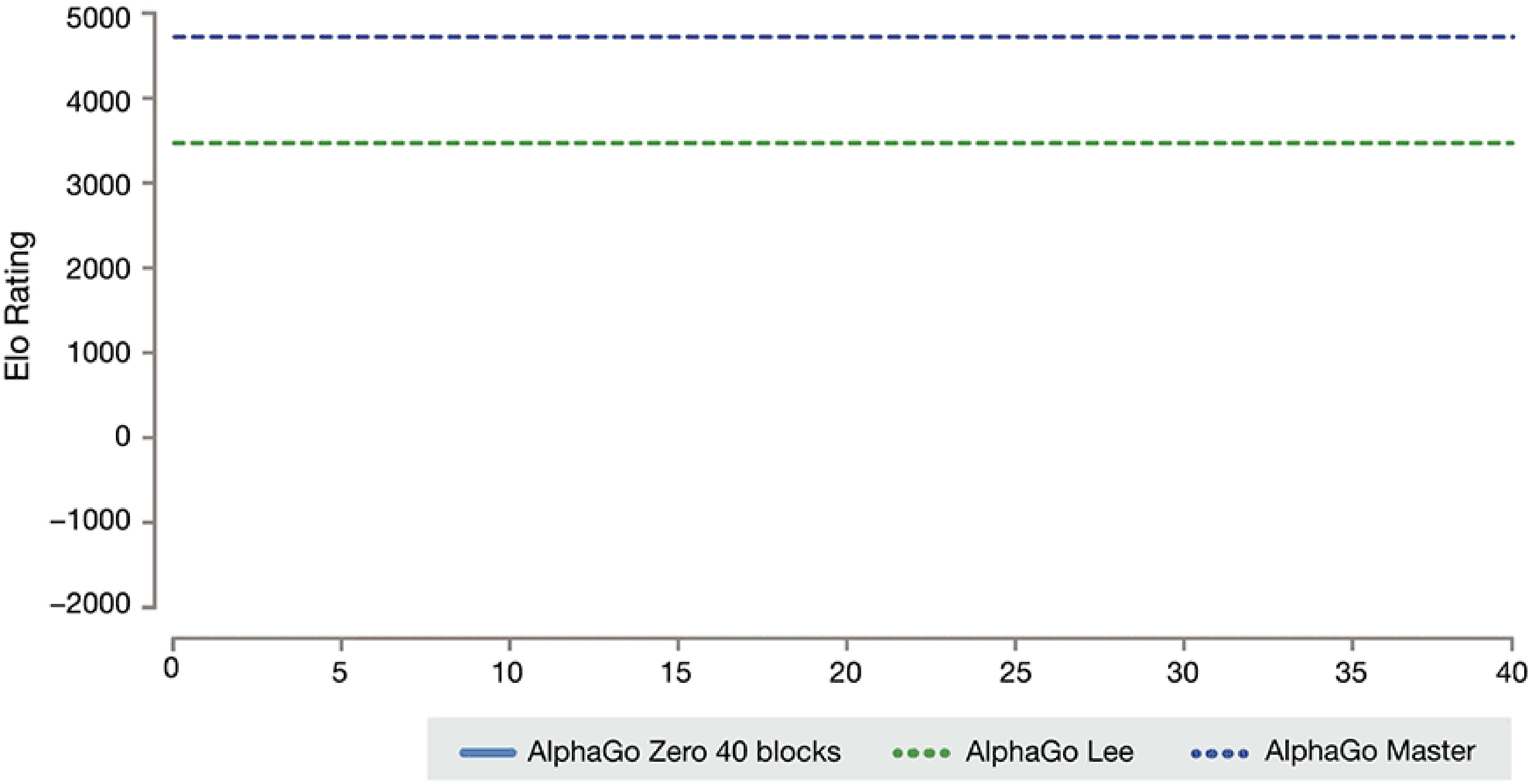
Mastering the game of Go without human knowledge

David Silver[1]*, Julian Schrittwieser[1]*, Karen Simonyan[1]*, Ioannis Antonoglou[1], Aja Huang[1], Arthur Guez[1], Thomas Hubert[1], Lucas Baker[1], Matthew Lai[1], Adrian Bolton[1], Yutian Chen[1], Timothy Lillicrap[1], Fan Hui[1], Laurent Sifre[1], George van den Driessche[1], Thore Graepel[1] & Demis Hassabis[1]

A long-standing goal of artificial intelligence is an algorithm that learns, *tabula rasa*, superhuman proficiency in challenging domains. Recently, AlphaGo became the first program to defeat a world champion in the game of Go. The tree search in AlphaGo evaluated positions and selected moves using deep neural networks. These neural networks were trained by supervised learning from human expert moves, and by reinforcement learning from self-play. Here we introduce an algorithm based solely on reinforcement learning, without human data, guidance or domain knowledge beyond game rules. AlphaGo becomes its own teacher: a neural network is trained to predict AlphaGo's own move selections and also the winner of AlphaGo's games. This neural network improves the strength of the tree search, resulting in higher quality move selection and stronger self-play in the next iteration. Starting *tabula rasa*, our new program AlphaGo Zero achieved superhuman performance, winning 100–0 against the previously published, champion-defeating AlphaGo.

356 | NATURE | VOL 550 | 19 OCTOBER 2017

https://deepmind.com/blog/alphago-zero-learning-scratch/



Contents

07 DECEMBER 2018
VOL 362, ISSUE 6419

AlphaZero

Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm

David Silver,[1*] Thomas Hubert,[1*] Julian Schrittwieser,[1*]
Ioannis Antonoglou,[1] Matthew Lai,[1] Arthur Guez,[1] Marc Lanctot,[1]
Laurent Sifre,[1] Dharshan Kumaran,[1] Thore Graepel,[1]
Timothy Lillicrap,[1] Karen Simonyan,[1] Demis Hassabis[1]

[1]DeepMind, 6 Pancras Square, London N1C 4AG.
*These authors contributed equally to this work.

.AI] 5 Dec 2017

arXiv:1712.01815v1

"Starting from random play, and given no domain knowledge except the game rules, *AlphaZero* achieved within 24 hours a superhuman level of play in the games of chess and shogi (Japanese chess) as well as Go, and convincingly defeated a world-champion program in each case."

McKinsey&Company

MCKINSEY GLOBAL INSTITUTE

A FUTURE THAT WORKS: AUTOMATION, EMPLOYMENT, AND PRODUCTIVITY

JANUARY 2017

EXECUTIVE SUMMARY

"Almost half the activities people are paid almost $16 trillion in wages to do in the global economy have the potential to be automated by adapting currently demonstrated technology, according to our analysis of more than 2,000 work activities across 800 occupations."

McKinsey & Company,

A FUTURE THAT WORKS: AUTOMATION, EMPLOYMENT, AND PRODUCTIVITY

Executive Summary January 2017

The Future of Machine Learning

By 2050, the following AI systems might be in routine use:

1. personal predictive medical systems
2. personal tutors
3. autonomous house servants
4. autonomous vehicles that can drive safely in Cairo!

The potential of machine learning and AI is vast and exciting.

But, some have argued (e.g, Henry Kissinger, Bill Gates, Elon Musk, the late Stephen Hawking) that the *dangers* are also vast: autonomous drone soldiers, AI computer viruses…

Your lives may well come to depend on AI systems…

"Doubt is not a pleasant condition, but certainty is an absurd one"

Voltaire

THANK YOU!

Tutorials

Dependencies

python 2.7.x, x > 9

numpy	array manipulation
pandas	DataFrame manipulation
matplotlib	plotting
scikit-learn	simple machine learning toolkit

Also useful:

scipy	mathematical stuff for scientists
sympy	amazing symbolic algebra package

Installation

git clone https://github.com/hbprosper/ENHEP

Tutorials

1. Use a BDT to separate VBF produced Higgs boson events from events produced via ggF. If $F(x)$ is the output of a BDT trained using the AdaBoost algorithm then

$$D(x) = \frac{1}{1 + \exp\left[-2F(x)\right]}$$

   where

$$D(x) = \frac{\text{vbf}(x)}{\text{vbf}(x) + \text{ggf}(x)}$$

2. Repeat, but using a DNN. Note: a DNN approximates $D(x)$ directly.

BACKUP

Ensemble Methods

Suppose you have an ensemble of classifiers $f(x, w_k)$, which, individually, perform only marginally better than random guessing. Such classifiers are called weak learners.

It is possible to build highly effective classifiers by *averaging* their outputs:

$$f(x) = a_0 + \sum_{n=1}^{N} a_n f(x_n, w_n)$$

Jerome Friedman & Bogdan Popescu (2008)

Adaptive Boosting

The **AdaBoost** algorithm of Freund and Schapire uses decision trees $f(x, \mathbf{w})$ with weights $\mathbf{w}$ assigned to each object to be classified, and each assigned a target value of either $\boldsymbol{y} = \mathbf{+1}$, or $\mathbf{-1}$, e.g., +1 for signal, –1 for background.

The value assigned to each leaf of $f(x, \mathbf{w})$ is also ± 1.

Consequently, for object $\boldsymbol{n}$, associated with values (y_n, x_n)

$f(x_n, \mathbf{w})\, y_n > 0$ for a correct classification

$f(x_n, \mathbf{w})\, y_n < 0$ for an incorrect classification

Y. Freund and R.E. Schapire. Journal of Computer and Sys. Sci. **55** (1), 119 (1997)

Adaptive Boosting

Initialize weights $\boldsymbol{w}$ in training set (e.g., setting each to $\mathbf{1}/\boldsymbol{N}$)

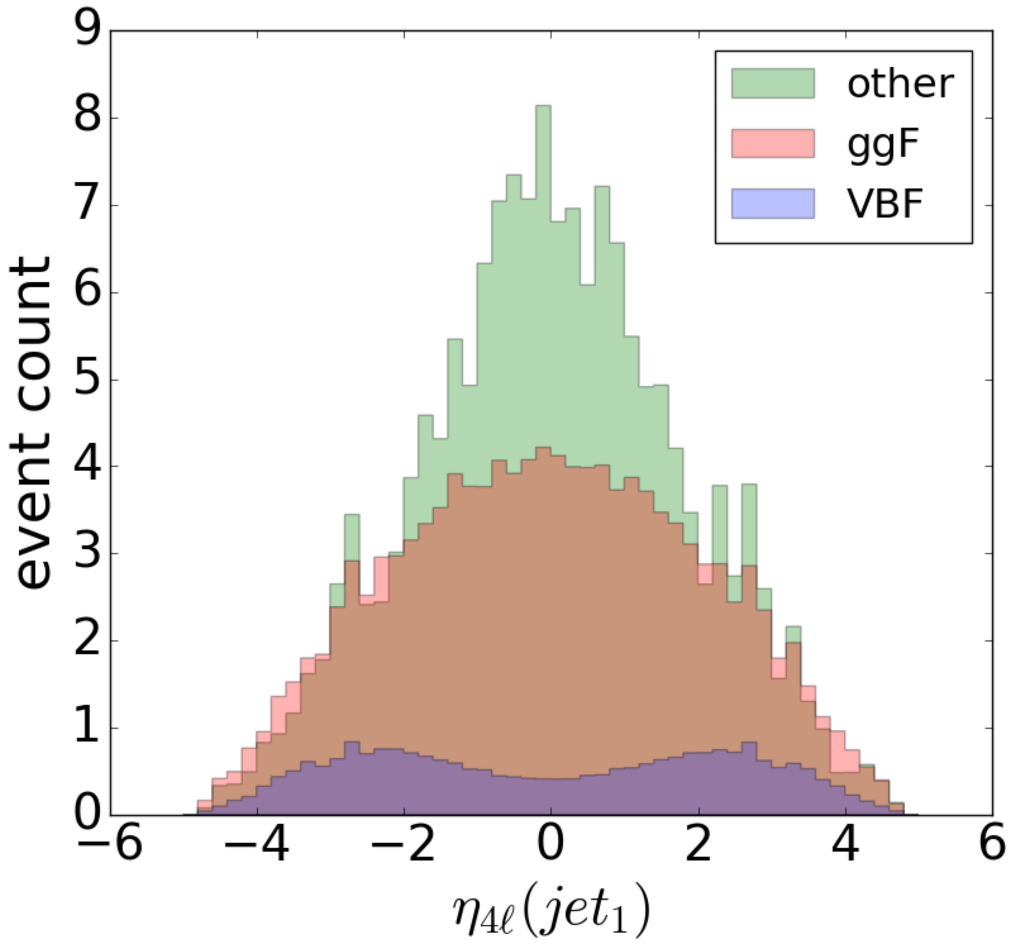
for k = 1 to K:

1. Create a decision tree $f\,(x, \mathbf{w})$ using the current weights.
2. Compute its error rate Σ on the ***weighted*** training set.
3. Compute $\langle\ = \ln(1 - \Sigma) / \Sigma$ and store as $\langle_k = \langle$
4. Update each weight $\boldsymbol{w_n}$ in the training set as follows:
   $\boldsymbol{w_n} = \boldsymbol{w_n} \exp[-\langle_k\ f\,(x_n, \mathbf{w})\, y_n] / A$, where A is a normalization constant such that $\sum \boldsymbol{w_n} = 1$. Since $f\,(x_n, \mathbf{w})\, y_n < 0$ for an incorrect classification, the weight of misclassified objects is ***increased***.

At the end, compute the average $f\,(x) = \sum \langle_k\ f\,(x, \mathbf{w_k})$

Y. Freund and R.E. Schapire. Journal of Computer and Sys. Sci. **55** (1), 119 (1997)

CMS Run 2 Simulated Dataset



$L_{int} = 150\ \mathrm{fb}^{-1}$
Jet_1
pseudo-rapidity
distributions

Adaptive Boosting

AdaBoost is a highly non-intuitive algorithm. However, soon after its invention, Friedman, Hastie and Tibshirani showed that the algorithm is mathematically equivalent to minimizing the following average loss function

$$R(F) = \int \exp(-y\, F(x))\, \boldsymbol{p}(\boldsymbol{x}, \boldsymbol{y})\, dx\, dy$$

where $F(x) = \sum_{n=1}^{N} a_n\, f(x_n, w_n)$,

Minimizing this loss function yields

$$D(x) = \text{logistic}(2F) = 1/(1 + \exp(-2\, F(x)))$$

which can be interpreted as a probability, even though F cannot!

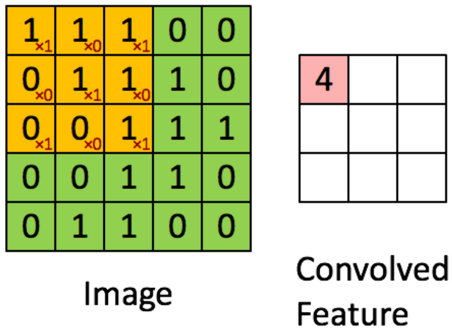
J. Friedman, T. Hastie and R. Tibshirani, "Additive logistic regression: a statistical view of boosting," The Annals of Statistics, 28(2), 377-386, (2000)

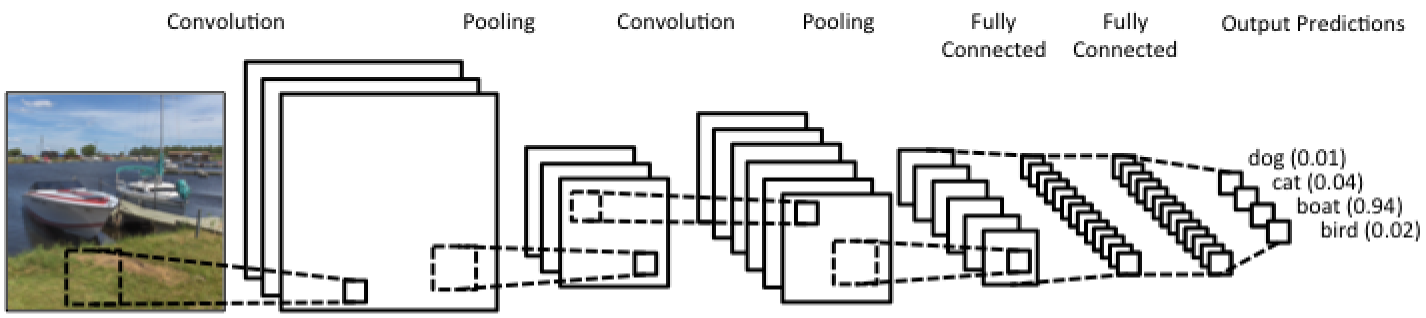
Convolutional Neural Networks

A CNN comprises three types of processing layers: 1. convolution, 2. pooling, and 3. classification.

1. **Convolution layers**

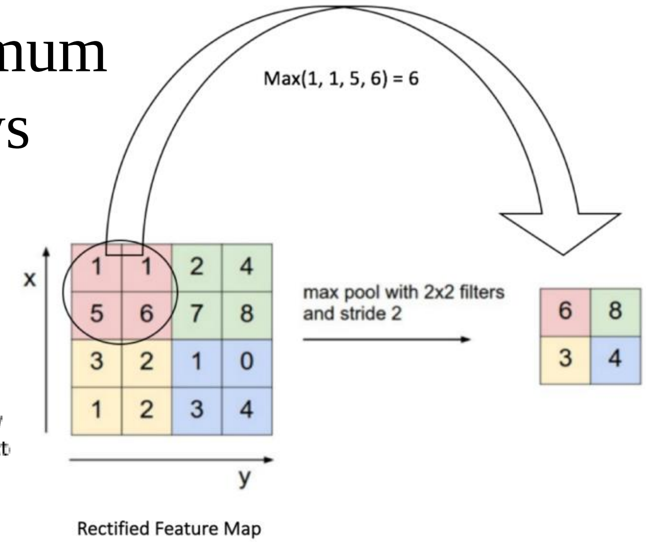
   The input layer is "convolved" with one or more matrices using element-wise products that are then summed. In this example, since the sliding matrix fits 9 times, we compress the input from a 5 x 5 to a to a 3 x 3 matrix.

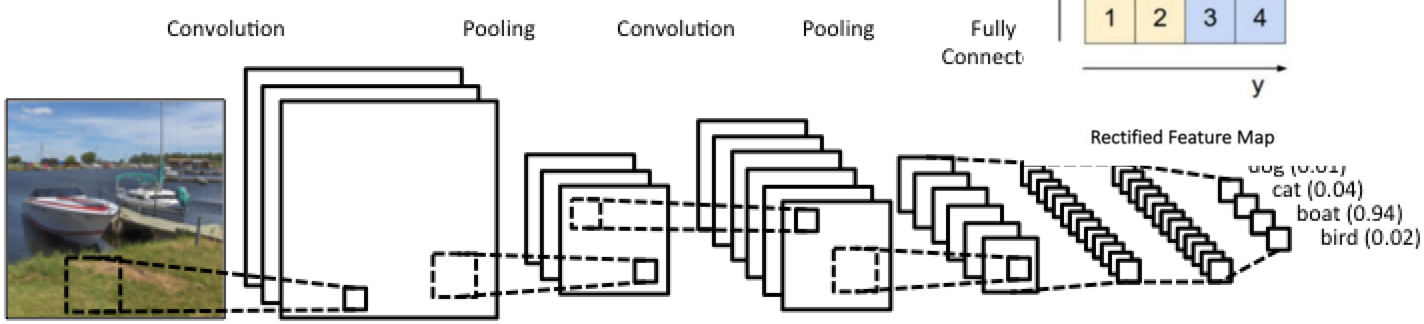




Convolutional Neural Networks

2. **Pooling Layers**
   After convolution, and a pixel by pixel non-linear map (using, e.g., the function $y = \max(0, x) = \mathrm{ReLU}(x)$), a coarse-graining of the layer is performed called max pooling in which the maximum values within a series of small windows are selected and become the output of a pooling layer.



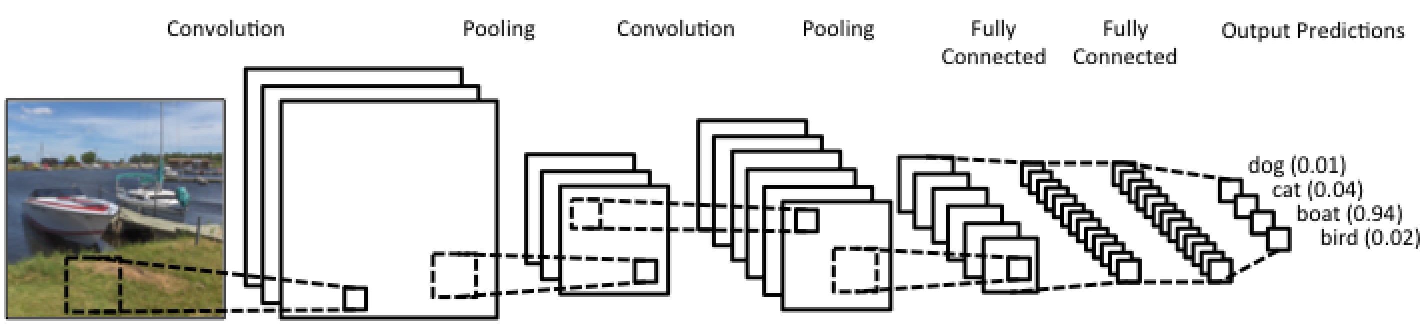


Convolutional Neural Networks

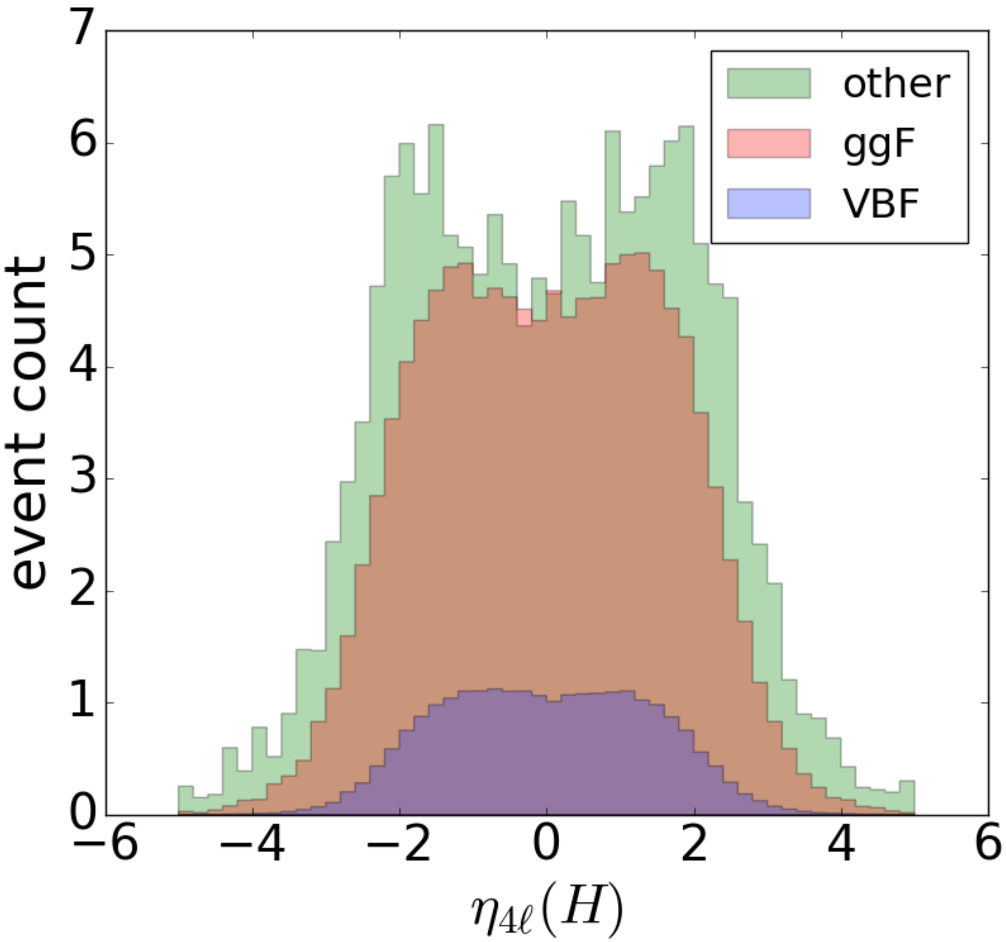
3. Classification Layers

After an alternating sequence of convolution and pooling layers, the outputs go to a standard neural network, either shallow or deep. The final outputs correspond to the different classes and like all flexible classifiers, a CNN approximates,

$$p(C_k|x) = p(x|C_k)p(C_k)/\sum_{m=1}^{M} p(x|C_m)p(C_m)$$



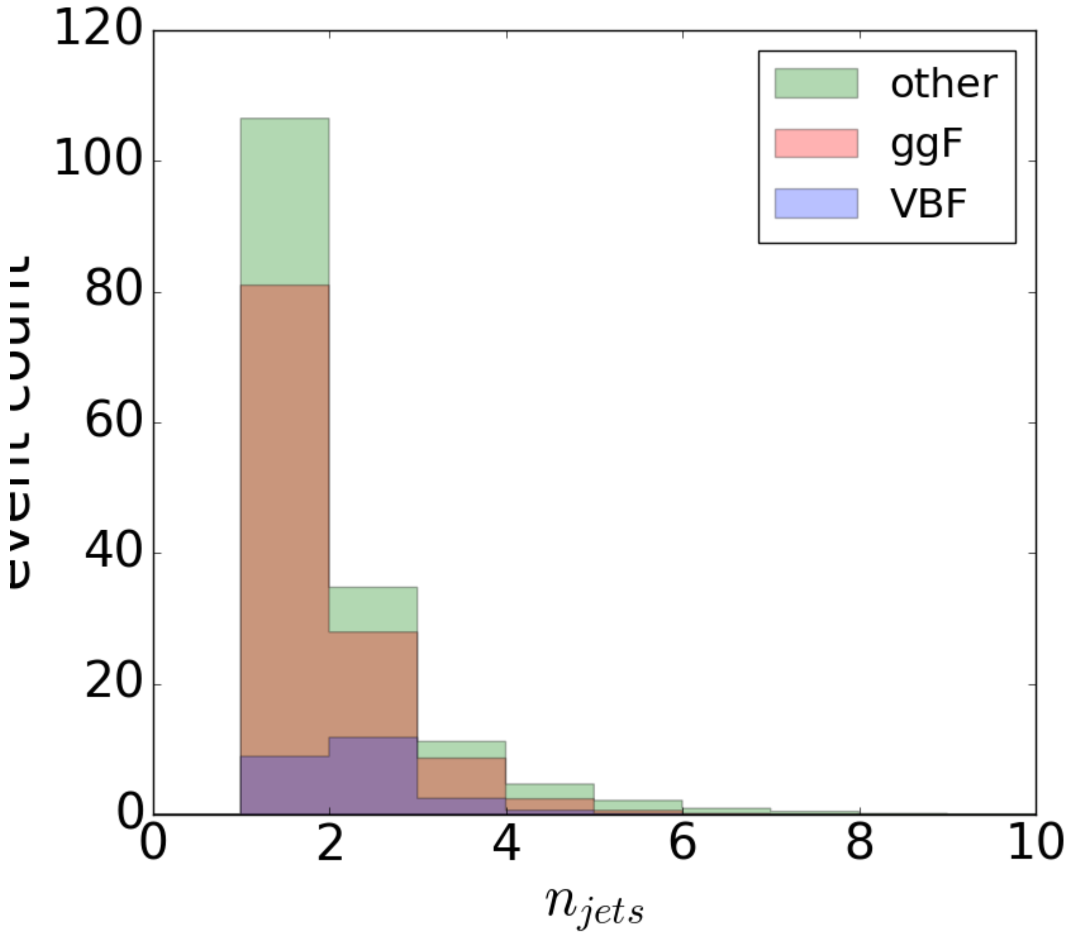
CMS Run 2 Simulated Dataset



$L_{int} = 150\ \mathrm{fb}^{-1}$

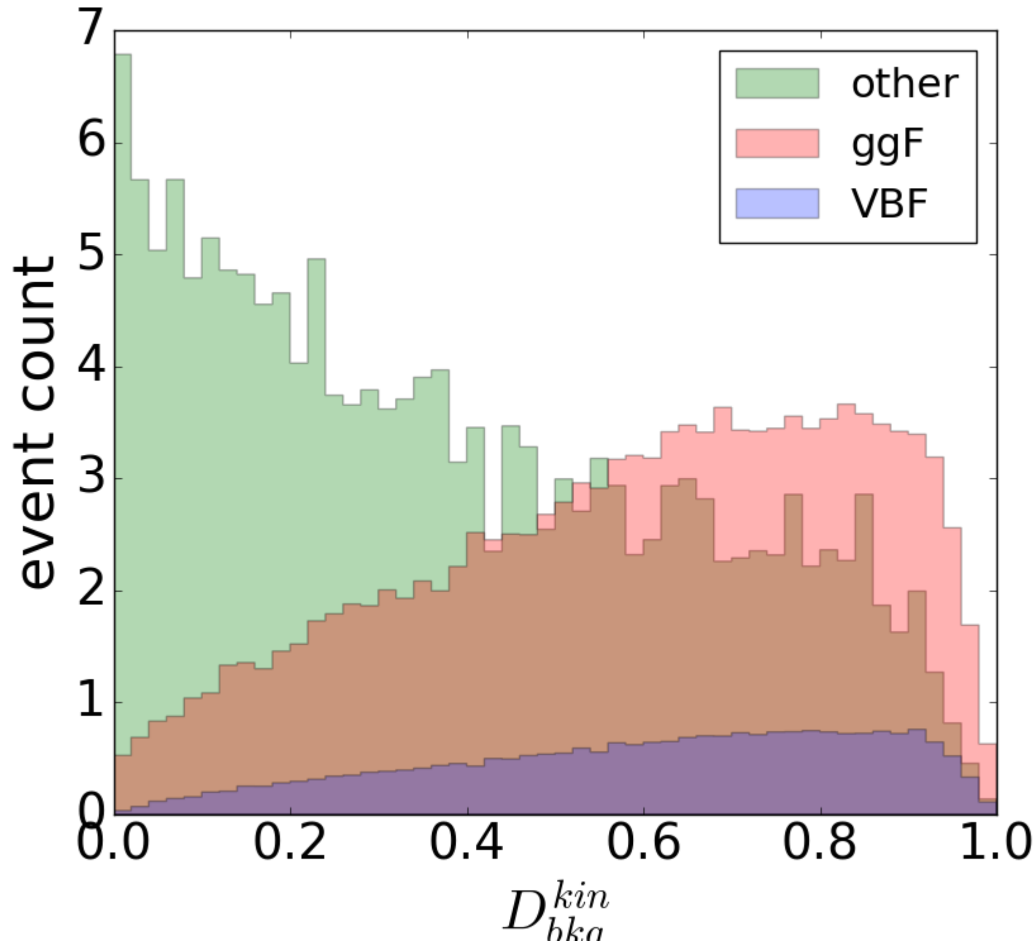
Higgs boson pseudo-rapidity distributions.

CMS Run 2 Simulated Dataset



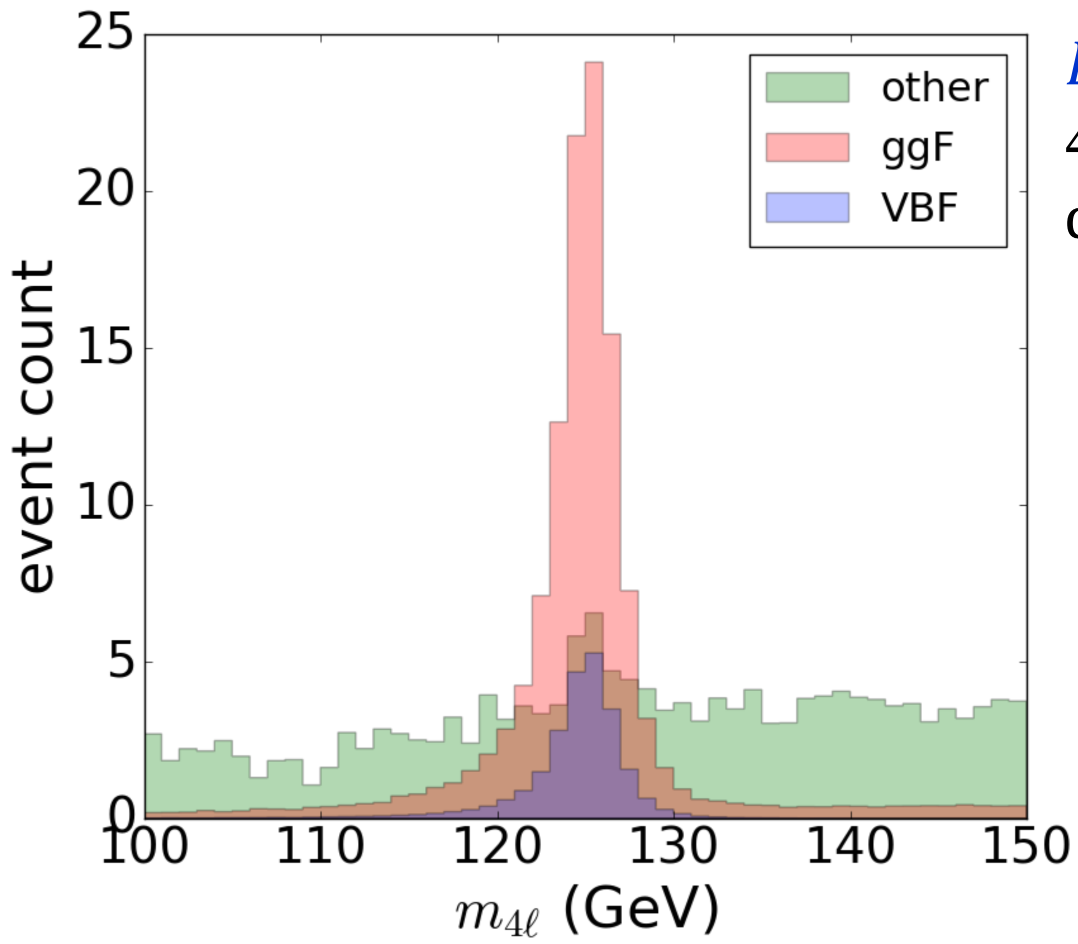
$L_{int} = 150\ \text{fb}^{-1}$
jet multiplicity distributions

CMS Run 2 Simulated Dataset



$L_{int} = 150\ \text{fb}^{-1}$
signal/bkg.
discriminant
distributions

CMS Run 2 Simulated Dataset



$L_{int} = 150\ \mathrm{fb}^{-1}$
4-lepton mass distributions

Pileup Mitigation Example: PUMML

Basic idea*

Treat a jet as a 3-color image in the (η, φ)-plane, where each color corresponds to be different category of particle.

1. Red p_T of *all* neutral particles
2. Green p_T of charged particles from pileup (PU)
3. Blue p_T of charged particles from the primary interaction, i.e., leading vertex (LV)

Use machine learning to map the 3-color image to an image of the p_T of neutral particles from the leading vertex. The jet is then formed from the charged and neutral particles from the leading vertex.

* Pileup Mitigation with Machine Learning (PUMML)
Metodiev, Komiske, Nachman, Schwarz, JHEP 12 (2017) 051, arXiv:1707.08600

Classification

The result

$$f(x) = p(S|x) = \frac{p(x|S)p(S)}{p(x|S)p(S) + p(x|B)p(B)}$$

was derived in 1990* in the context of *neural networks*.

But notice, our discussion so far made *no* mention of neural networks!

* Ruck et al., *IEEE Trans. Neural Networks* 4, 296-298 (1990); Wan, *IEEE Trans. Neural Networks* 4, 303-305 (1990);

Richard and Lippmann, *Neural Computation.* 3, 461-483 (1991)