CERN openlab

Deep Learning in HEP : an introduction

CERN openlab Summer Students Program

Sofia Vallecorsa

22/07/2019

Outline

Introduction

Basic Concepts

- Representational power and Depth
- Training DNN: Back-Propagation
- Evaluating the learning process

Convolutional Neural Networks

- CNN in HEP

Generative Models

- Examples from HEP

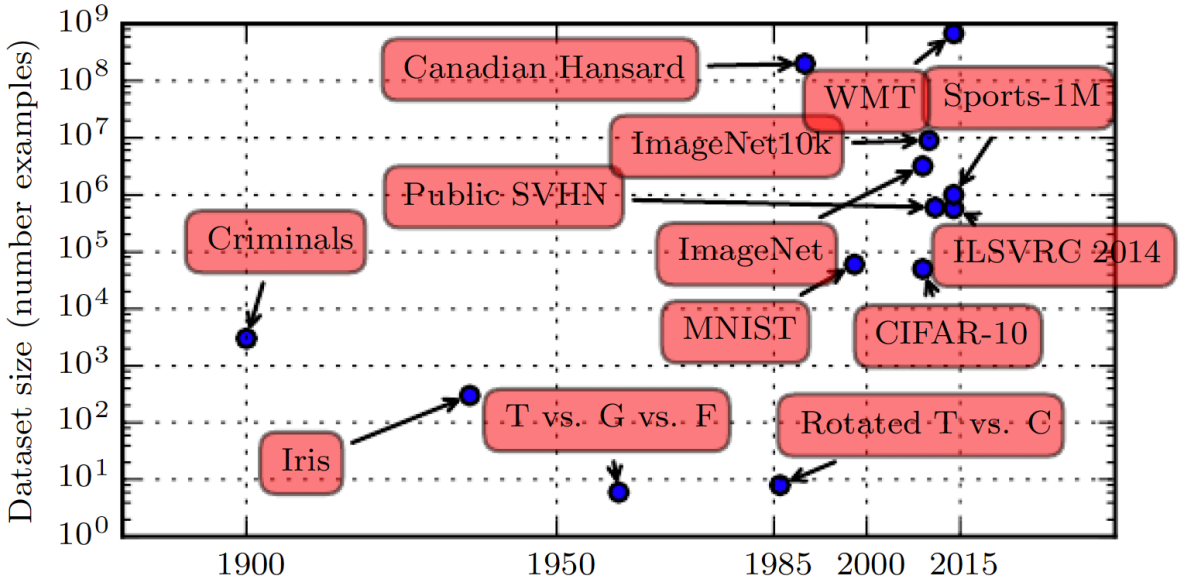
Hands-On:

- DNN for Higgs classification
- GANs for calorimeter simulation

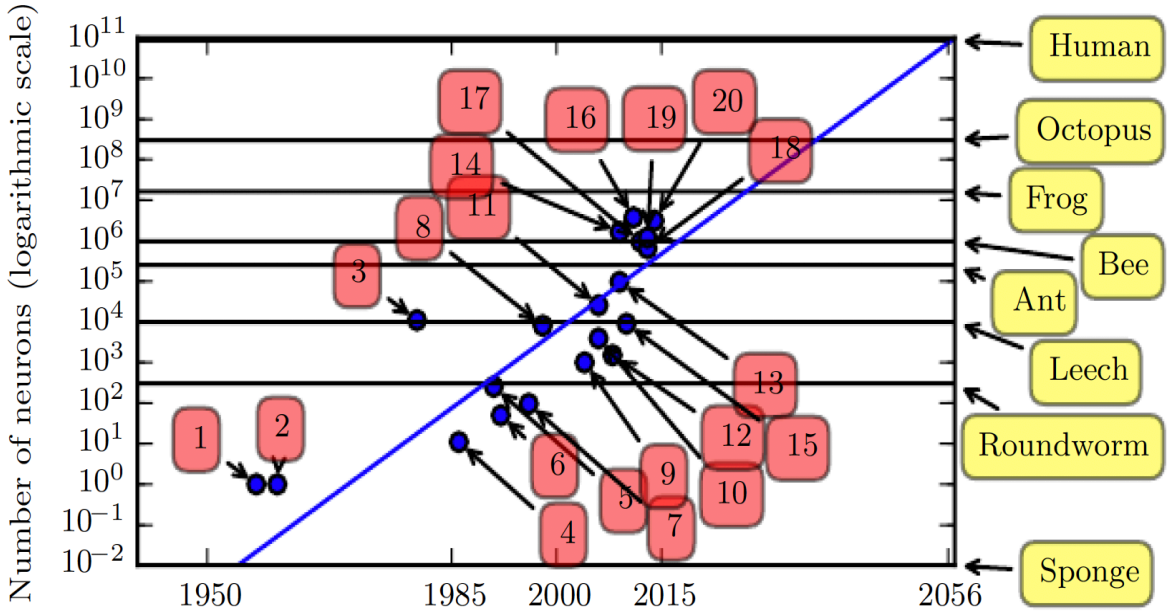
CERN openlab

Increasing sizes

Datasets:

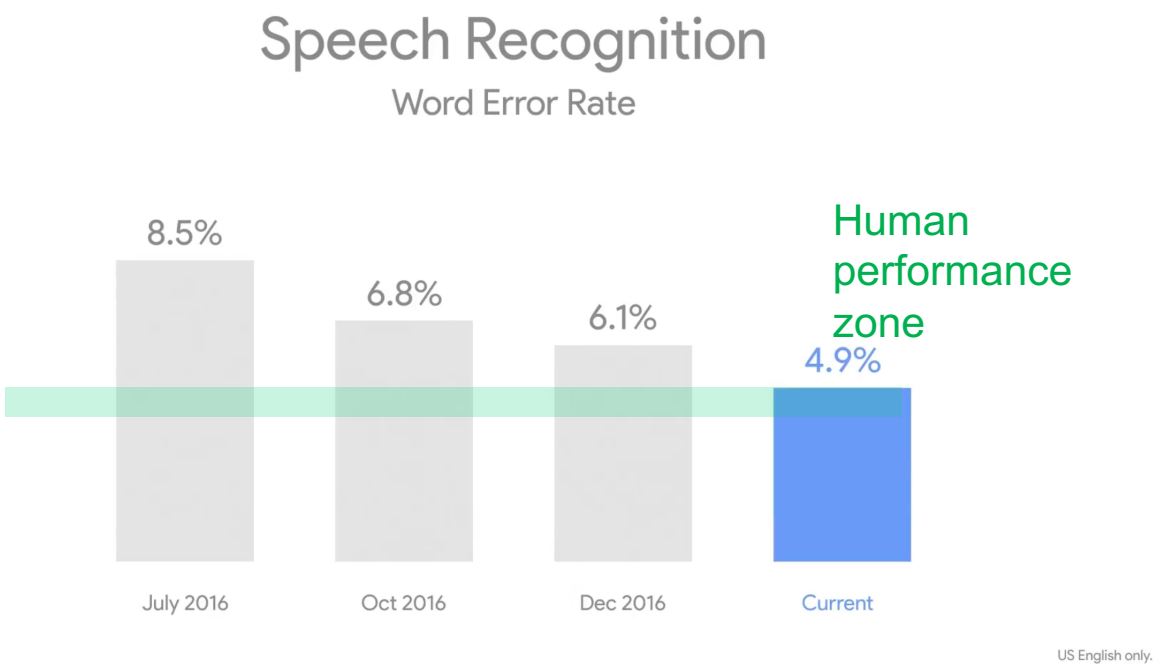


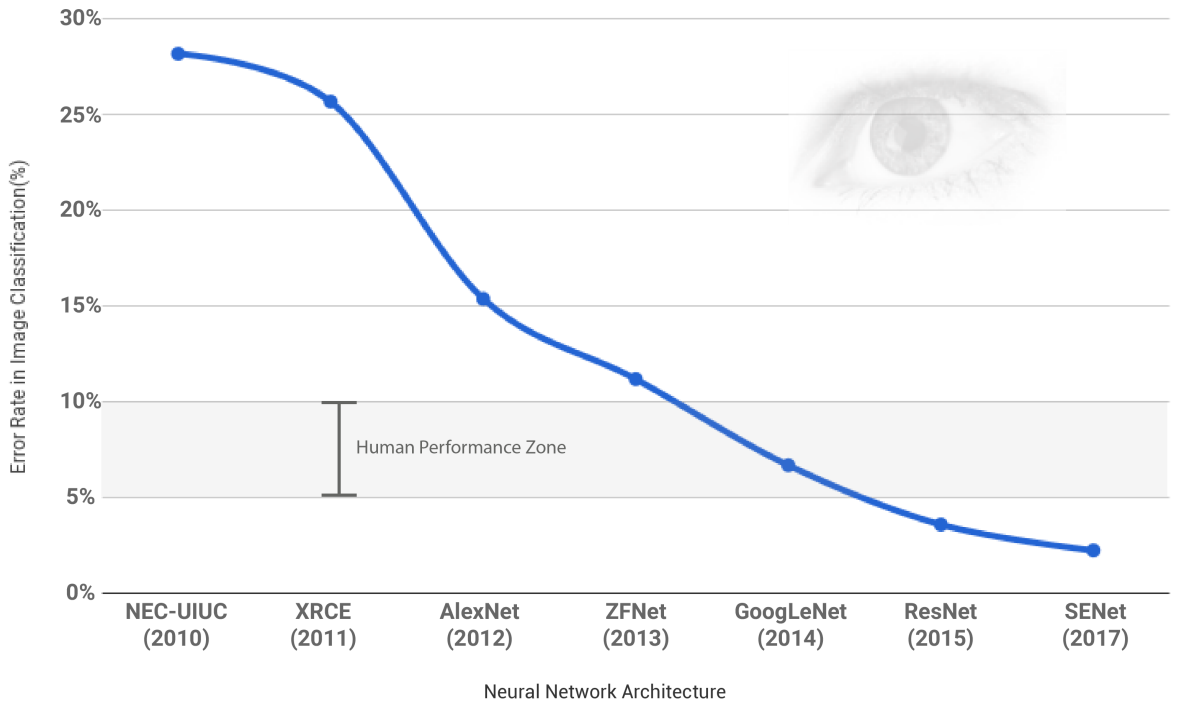
Models:



I. Goodfellow, Y. Bengio and A. Courville
MIT Press book
http://www.deeplearningbook.org/

Performance growth



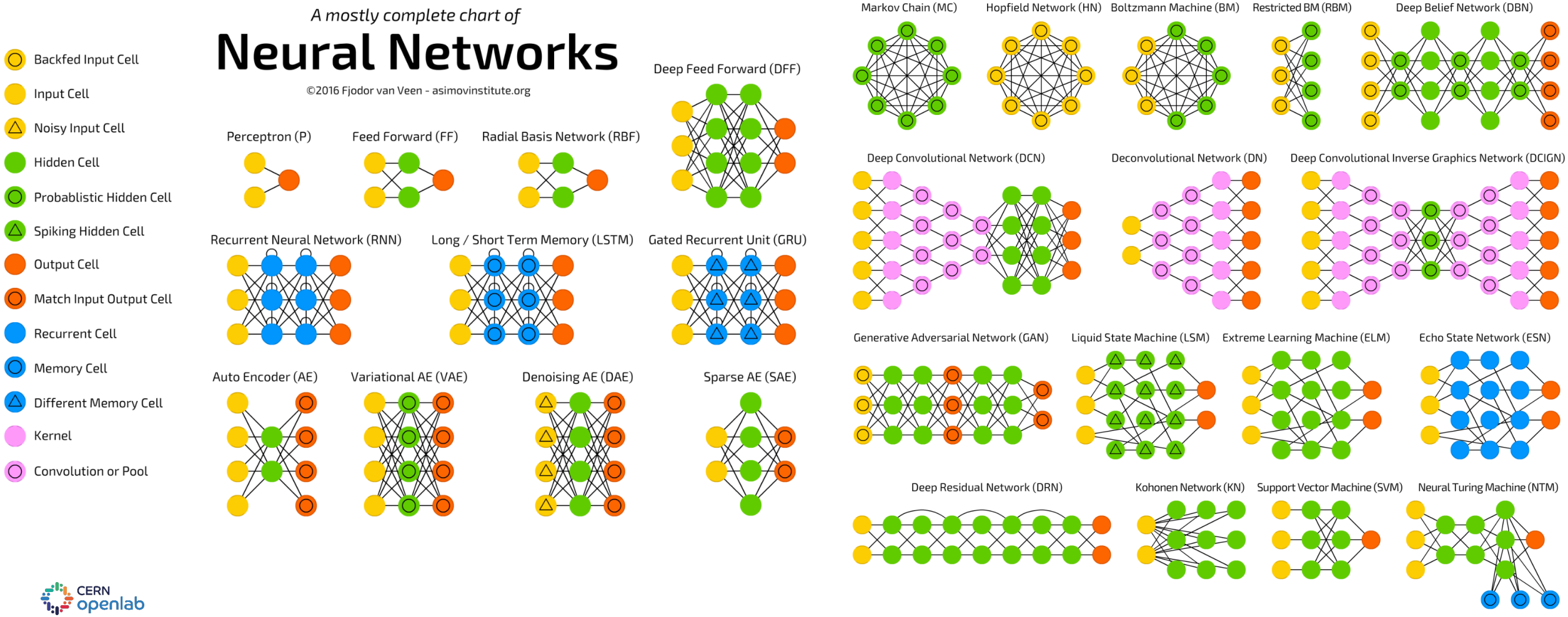


2017 Google results

https://arxiv.org/pdf/1409.0575.pdf

The model zoo

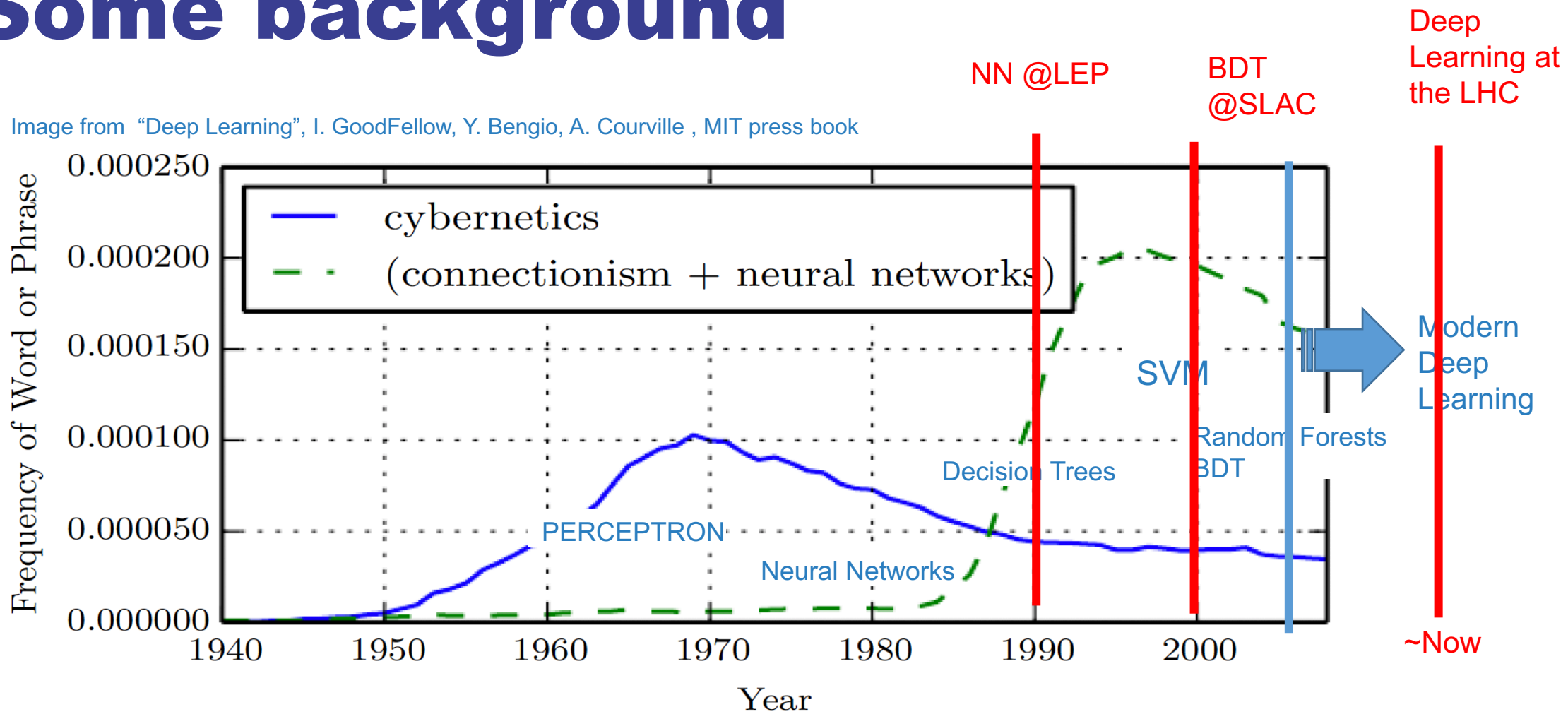
http://www.asimovinstitute.org/neural-network-zoo/



CERN openlab

Some background

Image from "Deep Learning", I. GoodFellow, Y. Bengio, A. Courville , MIT press book

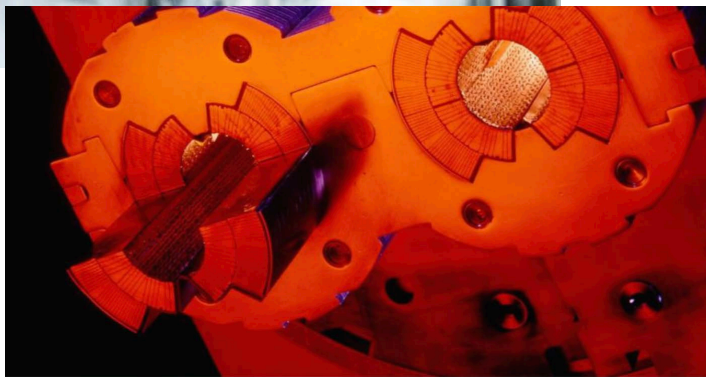


Theories on biological learning
First linear models
Layer-wise pre-training through greedy algorithms

Back-propagation

Why? ...Big Data

LHC is entering the Big Data era



Accelerators infrastructure (control systems, monitoring)

9600 magnets for Beam Control

1232 superconducting dipoles for bending

Experiments (detectors & physics data)

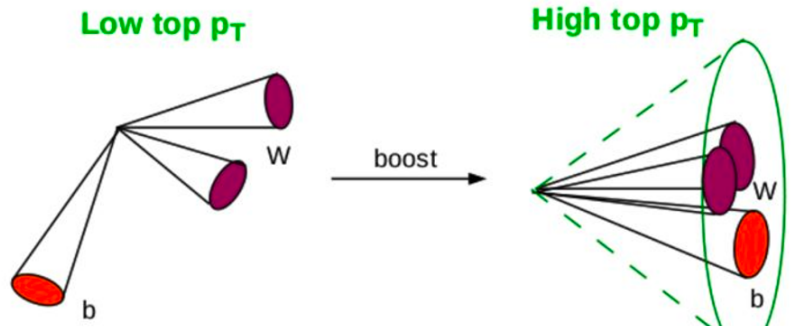
330 PB of collisions data stored by December 2018

The computing infrastructure:

Large sets of metrics collected from system components (CPU and batch, disk and archive storage, network topology and flows, and application throughput)

- LHC data is represent a challenge since it is **multi-structured, hybrid**
  - Metadata
  - Databases Aggregation
  - Evolving Data model

Why? ...New Challenges



Next generation colliders

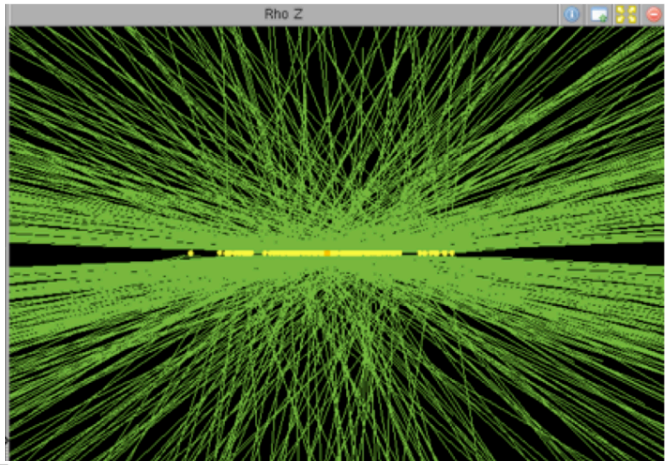
Will require **larger, highly granular detectors** (forward physics, high pT boosted objects)

- Larger, more complex datasets to analyse
- Challenging pattern recognition problems

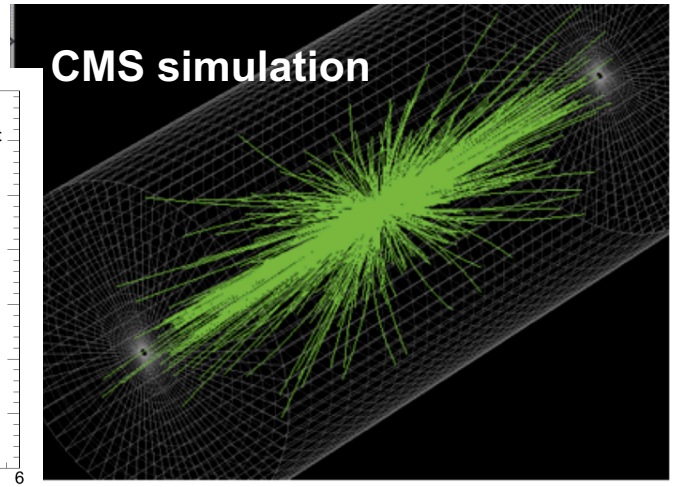
Will generate **huge particle data rates**

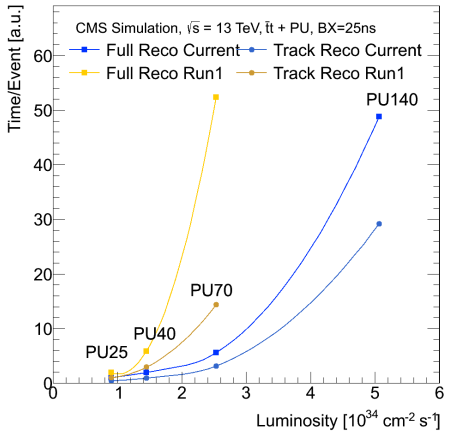
- Efficient, fast real-time selection will be essential
- HL-LHC bunch crossing frequency of 40 MHz and extreme data rates O(100 TB/s)
- L1 trigger latency ~1μs - 100 ms for HLT and offline processing

Related **computing challenges** will touch many aspects



$t\bar{t}$ event at <PU>=140 (94 vertices, 3494 tracks)





How? ... Deep Learning

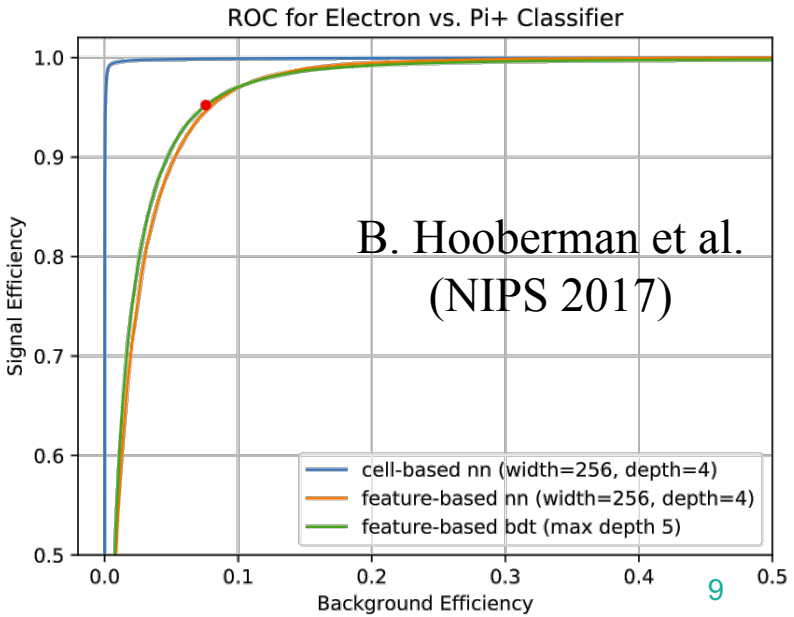
DL can **recognize patterns** in large complicated data sets
- DL algorithms can have better performances if applied directly to raw data

Re-cast physics problems as “DL problems”
- Interpret detector output as **images** and apply techniques borrowed from **computer vision** field
- Interpret physics events as **sentences** and apply NLP techniques

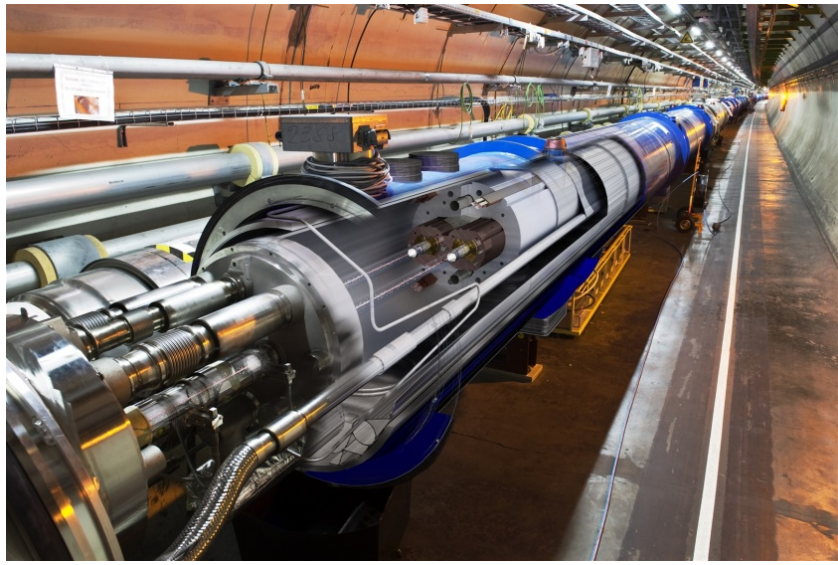
Intense R&D activity

Adapt DL to HEP requirements
- In terms of model **interpretability**
- Results **Validation** against classical methods
- Detailed **Systematics**

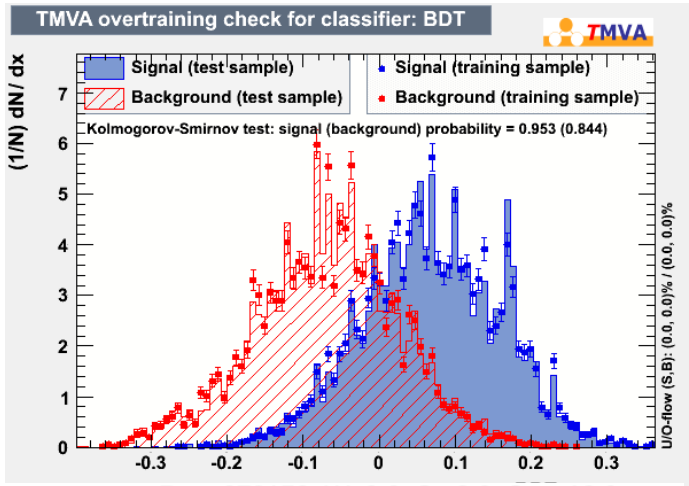
Adopting ”new” computing models
- **Accelerators** and dedicated hardware
- **HPC** integration
- **Cloud** environment
- **Big Data** platforms

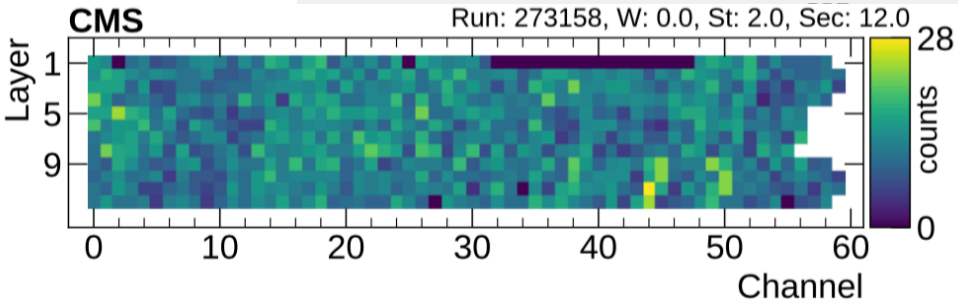


Deep Learning in HEP



- Analysis
- New physics searches as anomaly detection
- Reconstruction and particle identification
- Trigger and event filtering
- Data Quality Monitoring and Anomaly Detection in control systems
- Simulation
- Computing resources optimisation (dataset popularity, allocations, …)
- **“Theoretical” studies on model interpretability and systematics**





CERN openlab

Part1: Basics

Representational Power and Network Depth

Training DNN: Back Propagation

Evaluating the learning process: Overfitting and regularisation

Representational power

NN with at least one hidden layer are *universal approximators*

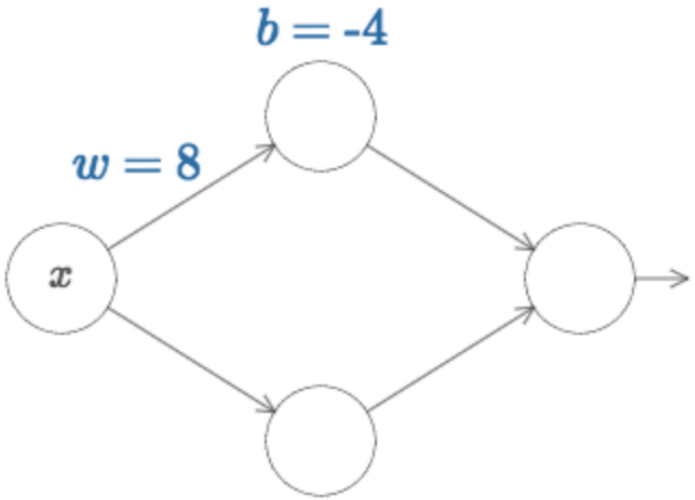
http://neuralnetworksanddeeplearning.com/chap4.html
Or Approximation by Superpositions of Sigmoidal Function

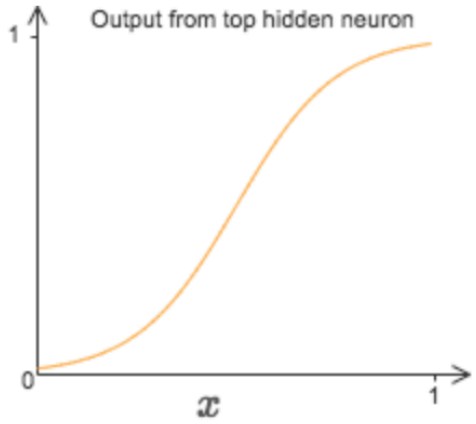
Representational power

NN with at least one hidden layer are *universal approximators*

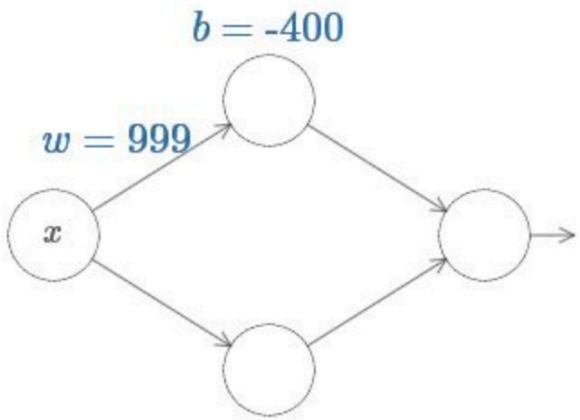
$\sigma(wx + b)$

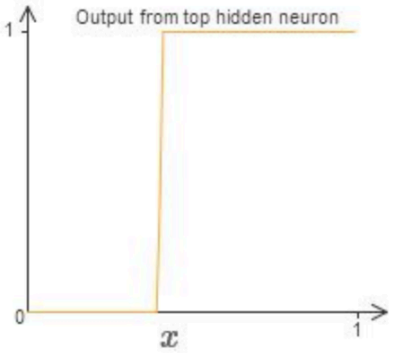
$\sigma(z) \equiv 1/(1 + e^{-z})$





Playing with the w, b parameters we can modify the shape of the sigmoid





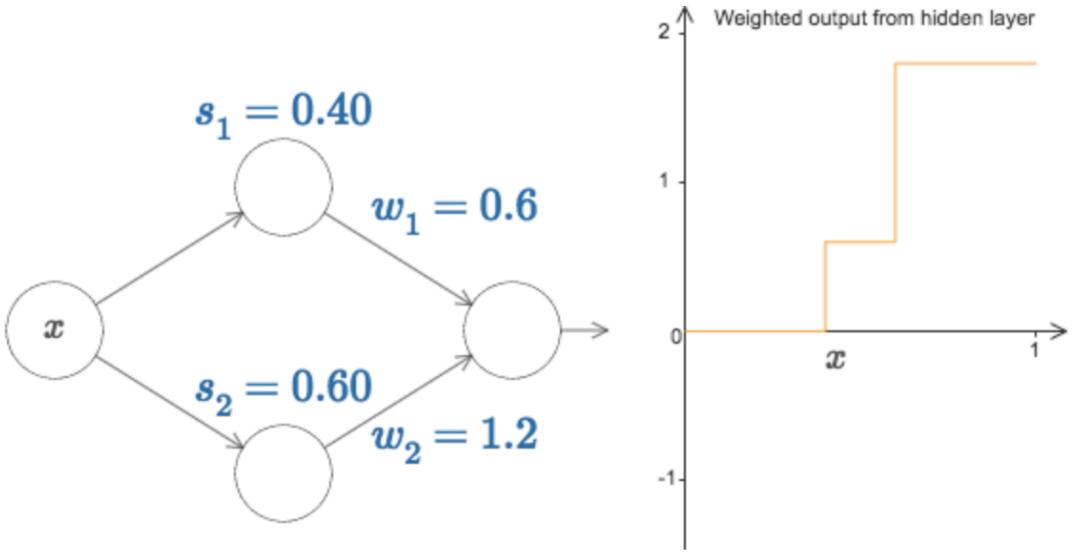
CERN openlab

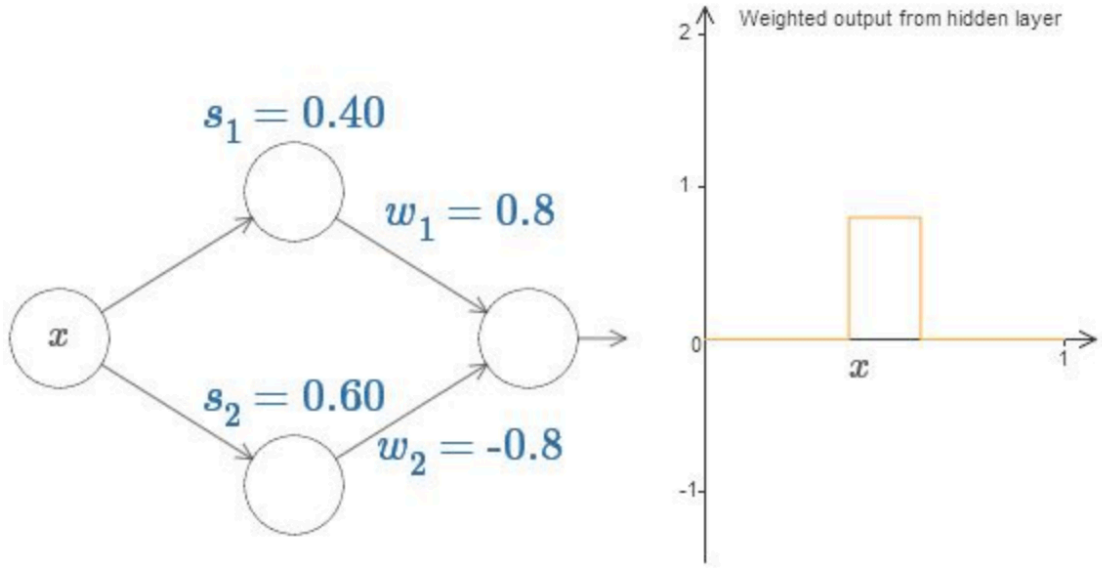
http://neuralnetworksanddeeplearning.com/chap4.html

Or Approximation by Superpositions of Sigmoidal Function

Representational power

We can add nodes and introduce “steps”

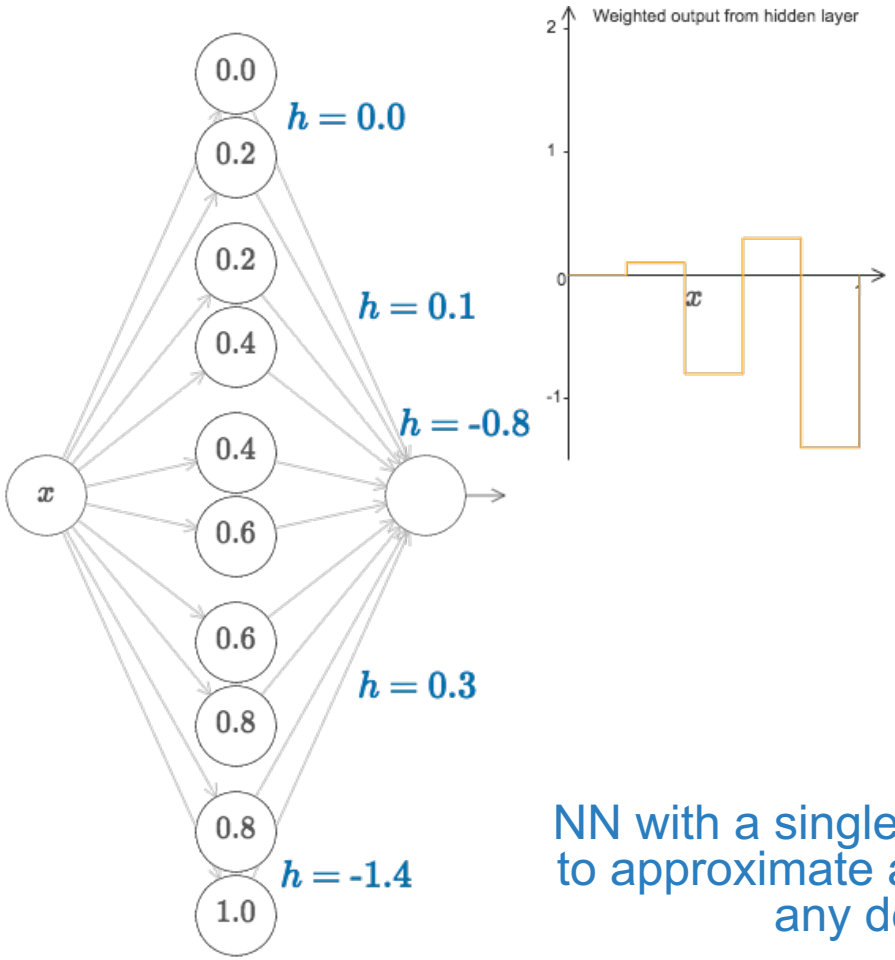


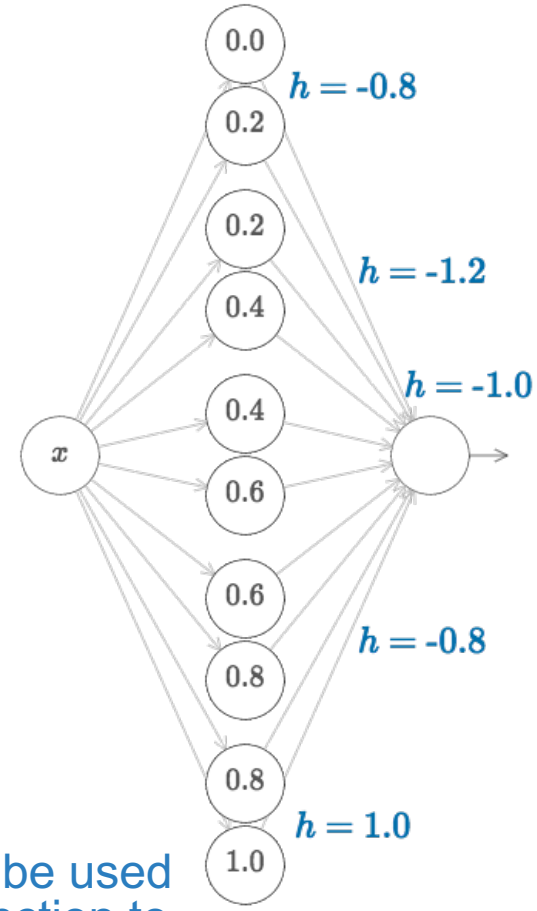


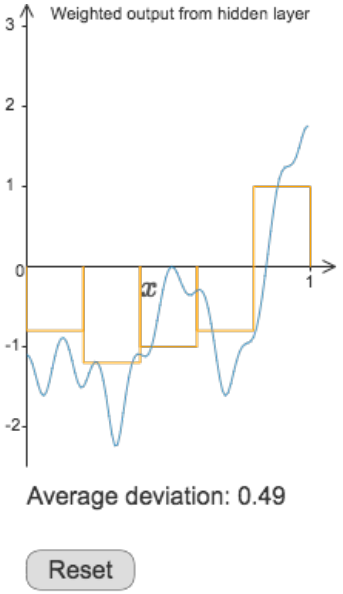
http://neuralnetworksanddeeplearning.com/chap4.html
Or Approximation by Superpositions of Sigmoidal Function

Representational power

Increasing complexity







NN with a single hidden layer can be used to approximate any continuous function to any desired precision

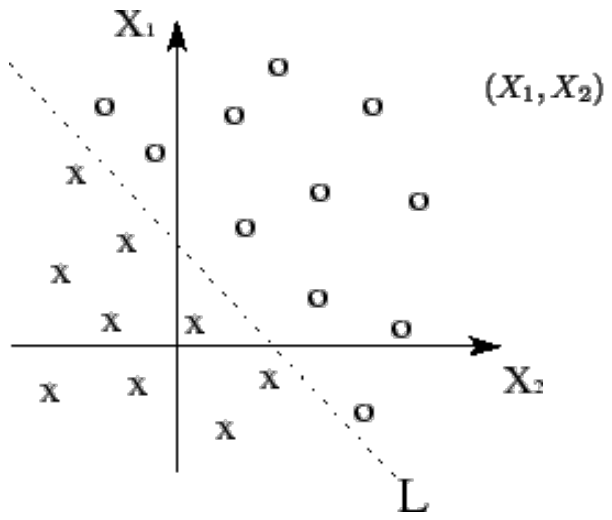
http://neuralnetworksanddeeplearning.com/chap4.html
Or Approximation by Superpositions of Sigmoidal Function

The need for depth

A single layer perceptron can categorize "linearly separable" patterns

Two classes classification:

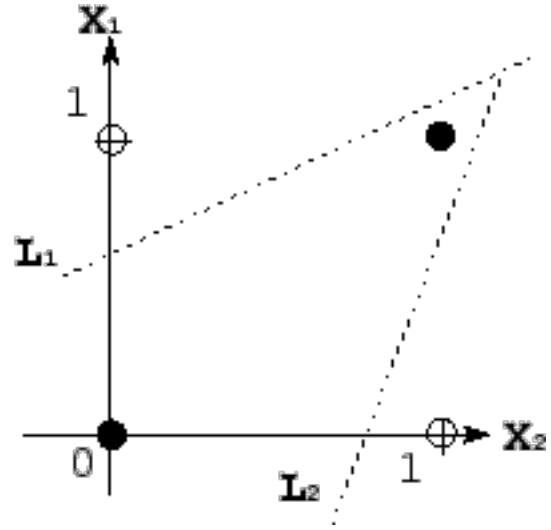
(OR function) (linearly separable)



Exclusive OR is an example of a non linearly separable pattern:

X_1	X_2	Y
0	0	0
0	1	1
1	0	1
1	1	0

$Y = X_1 \oplus X_2$

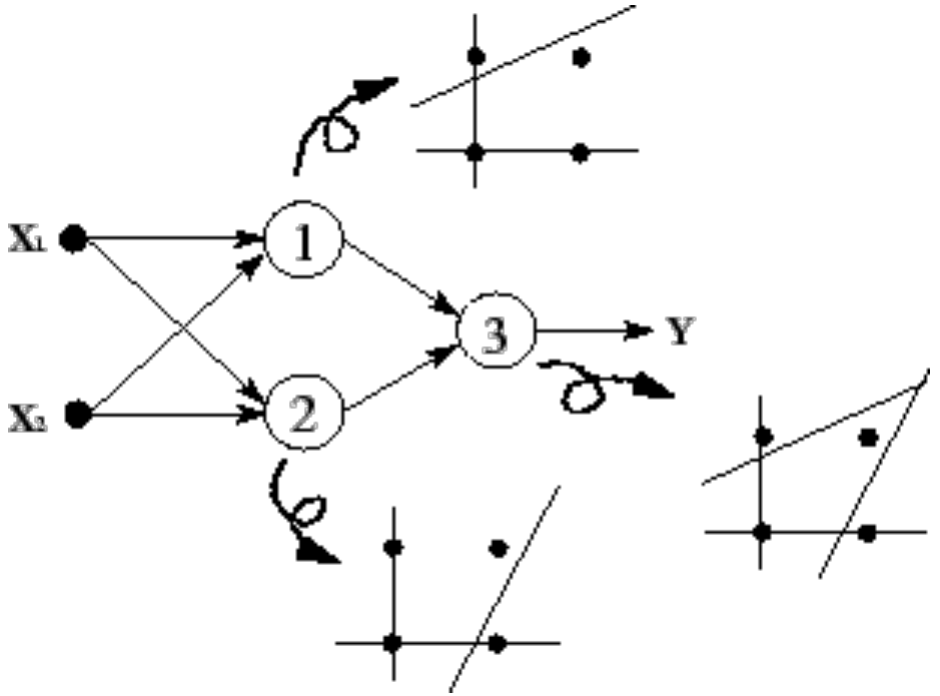


(tutorial) http://www.theprojectspot.com/tutorial-post/introduction-to-artificial-neural-networks-part-1/7
(images)http://www.ece.utep.edu/research/webfuzzy/docs/kk-thesis/kk-thesis-html/node19.html

The need for depth

Need a Multi-Layer architecture to solve the exclusive OR problem:

Two-stages approach



(tutorial) http://www.theprojectspot.com/tutorial-post/introduction-to-artificial-neural-networks-part-1/7
(images)http://www.ece.utep.edu/research/webfuzzy/docs/kk-thesis/kk-thesis-html/node19.html

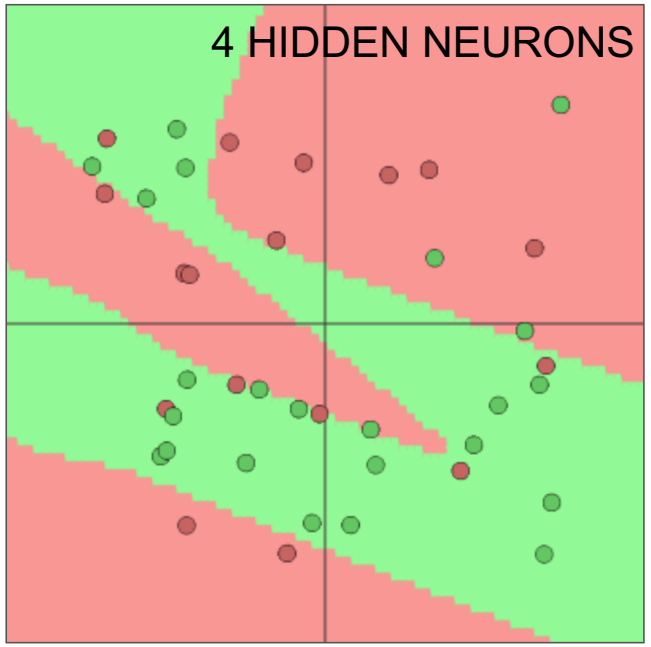
“Shallow” MLP

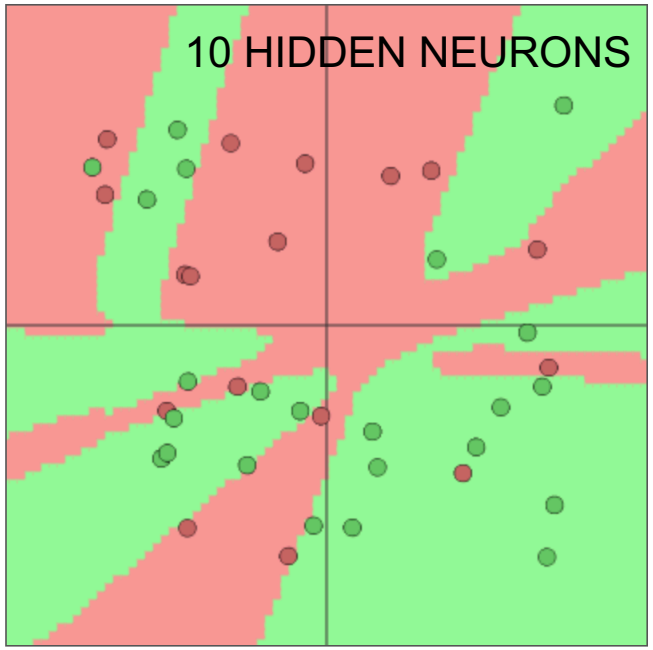
https://cs.stanford.edu/people/karpathy/convnetjs/demo/classify2d.html

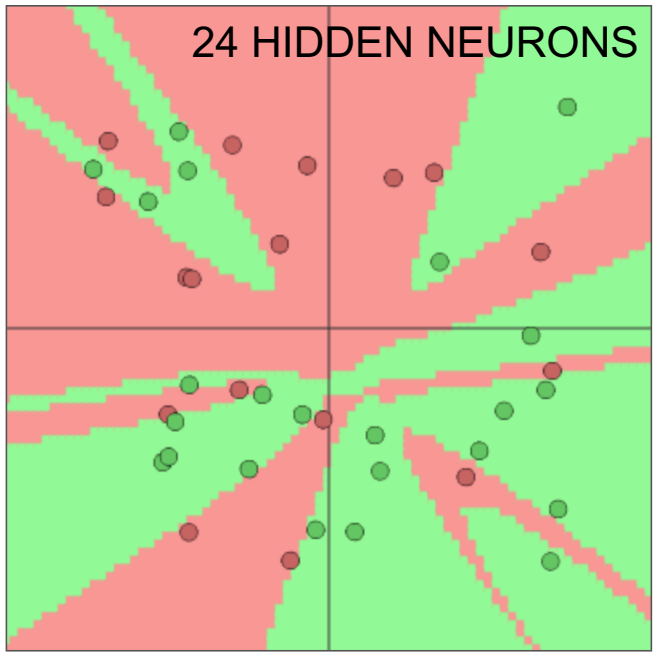
Empirical observation:

- 3-layers often outperform 2-layer nets
- going deeper rarely helps.

Overfitting?





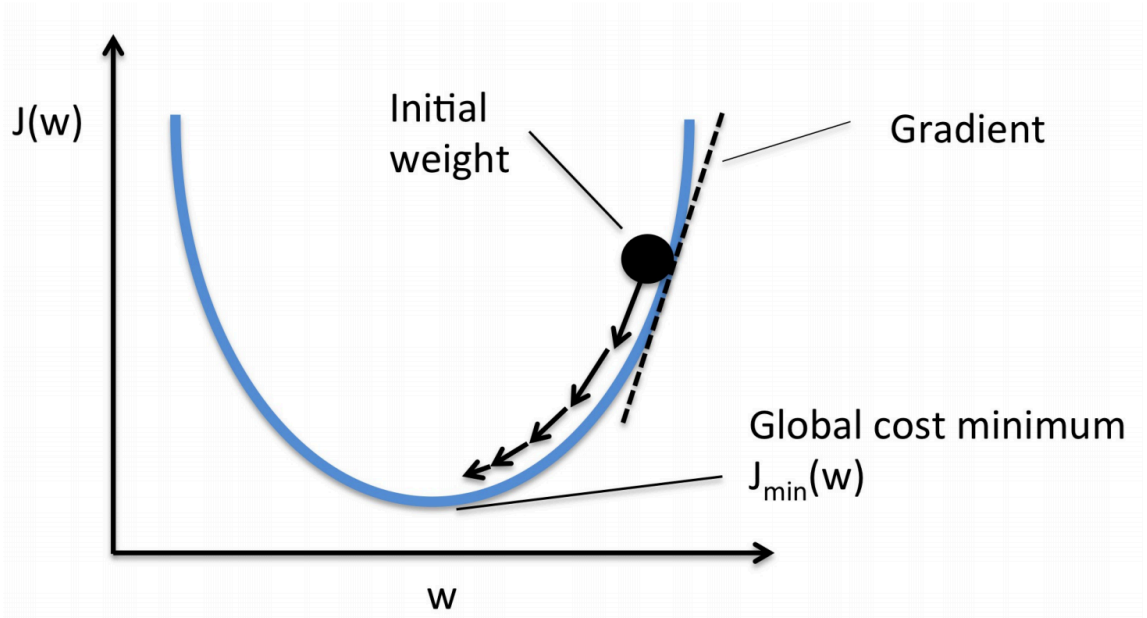


NB: CNN behave differently (depth helps discovering more features)

Training DNNs

Training process consists in:

1. Calculating the network output Y for some input X
2. Calculating the corresponding value of the loss function
3. Calculating the loss gradient wrt to the weights
4. Finding the value of the weights which minimizes the loss gradient.



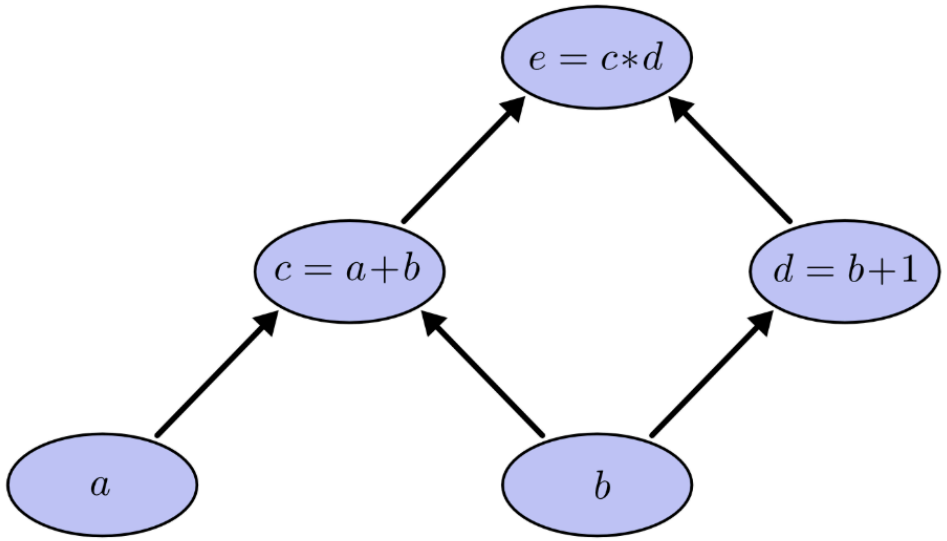
CERN openlab

Back-propagation: the problem

"the difference between a model taking a week to train and taking 200,000 years"

Simple classification examples involve hundreds of weights, complicated activations and losses

How do we actually implement step 4 in a computationally efficient way ?



$$c = a + b$$

$$d = b + 1$$

$$e = c * d$$

Ex: a simple computational graph
How does a change in **a** affect **e**?

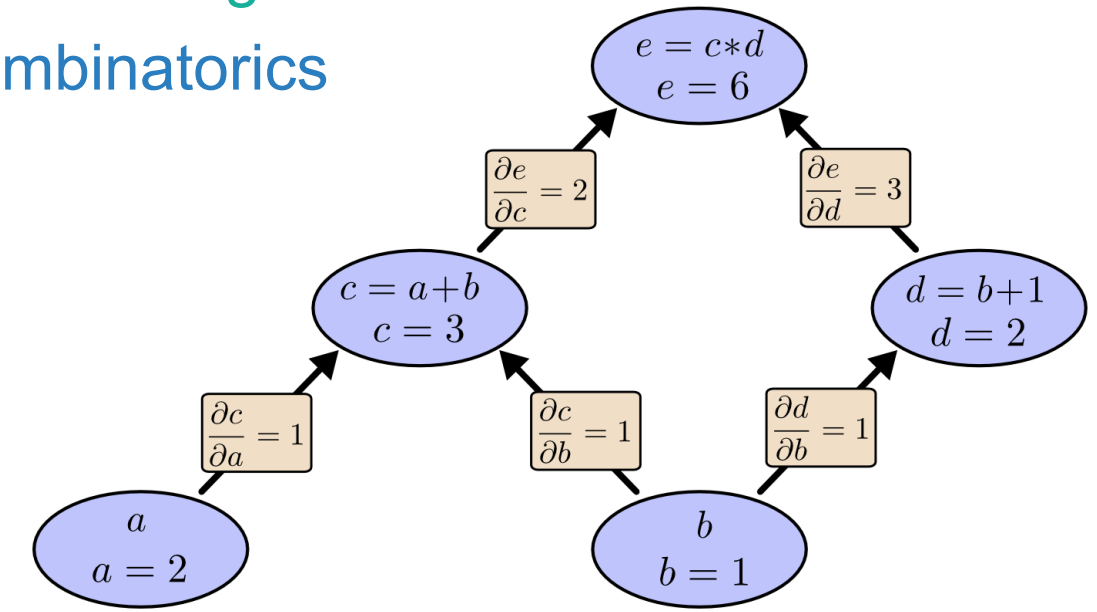
Back-propagation: the problem

Given a function **e** and a set of variables (**a** & **b**) :

How does a change in **a** affect **e**?

- Represent each branch with a derivative
- Follow the path through each of the connecting branches

Increasing number of layers increases combinatorics



http://colah.github.io/posts/2015-08-Backprop/

Back-propagation

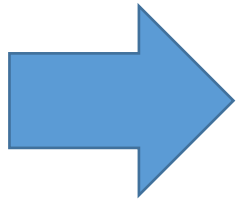
the key algorithm that makes training deep models computationally tractable

- A technique for calculating derivatives quickly
- Based on the derivatives chain rules

$$u_1 = f(v)$$

$$u_2 = g(v)$$

$$t = h(u_1, u_2)$$



$$\frac{dt}{dv} = \sum_i \frac{dt}{du_i} \frac{du_i}{dv}$$

Back-propagation

A simple visual example

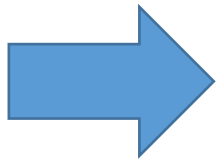
$$f(x, y, z) = (x + y)z.$$

$$q = x + y$$

$$f = qz$$

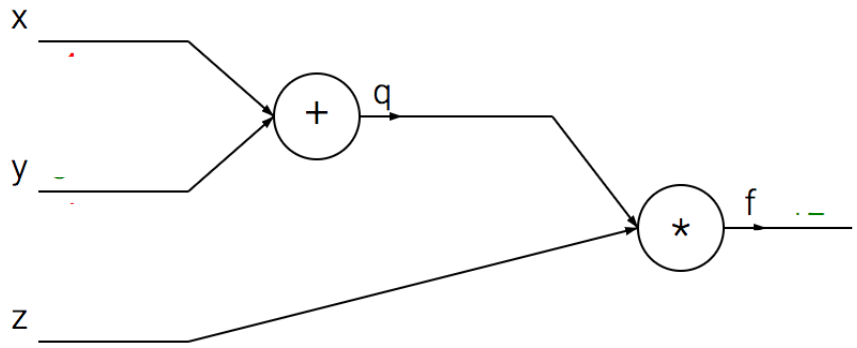
$$\frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

$$\frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$



$$\frac{\partial f}{\partial x} = \frac{\partial f}{\partial q}\frac{\partial q}{\partial x}$$

Forward pass computes values from inputs to output
X = -2, Y = 5, Z = -4



How does a change in Y affect f?
Calculate (forward) derivatives

OR
use **backward derivatives**: starts at the end and recursively applies the chain rule

Ex: Back-propagation example

Example circuit for a 2D neuron with a sigmoid activation function

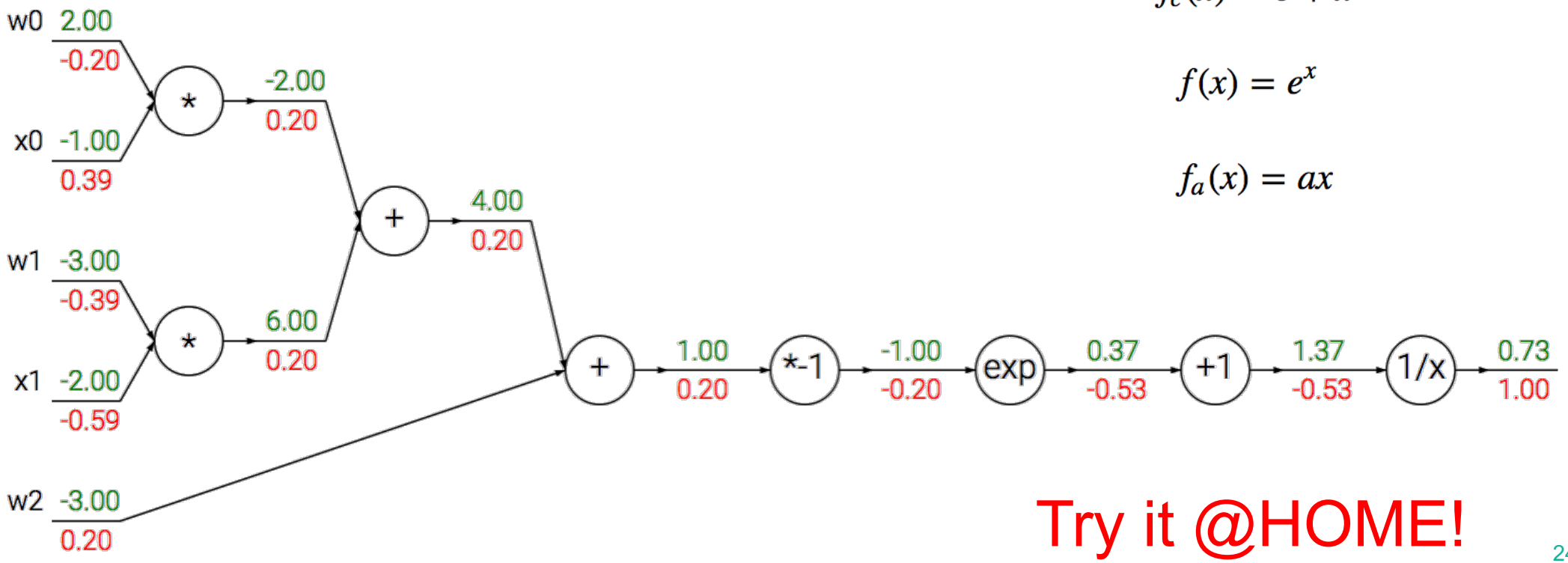
$$f(w,x) = \frac{1}{1+e^{-(w_0x_0+w_1x_1+w_2)}}$$

Decompose using : $f(x) = \frac{1}{x}$

$$f_c(x) = c + x$$

$$f(x) = e^x$$

$$f_a(x) = ax$$



Try it @HOME!

Back-propagation

A visual example

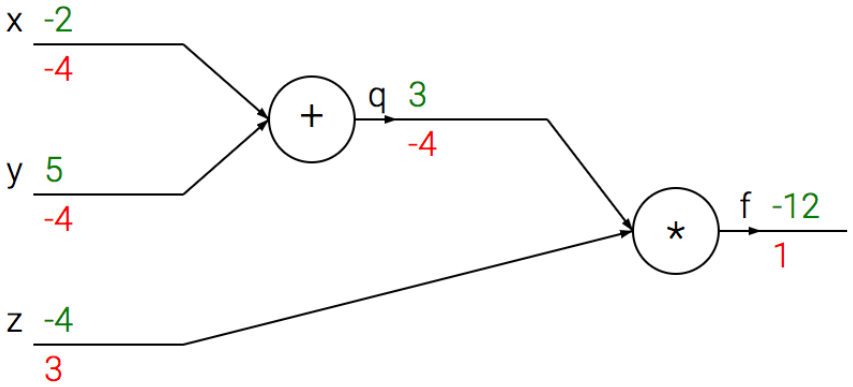
Back-propagation approach is much more **efficient** in the case of large graphs

Locality: Because of the chain rules, at each step the derivative depends only

- On the derivatives already calculated for the parent nodes
- On the node values calculated during the forward pass

Gradients flow "**backward**" through the graph

- Generalised through **Auto-Differentiation** techniques



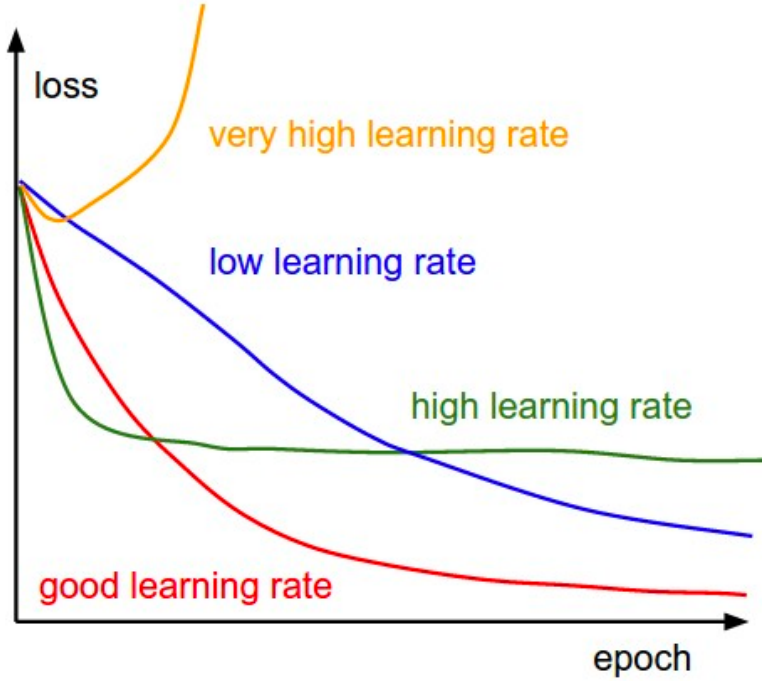
https://arxiv.org/pdf/1502.05767.pdf

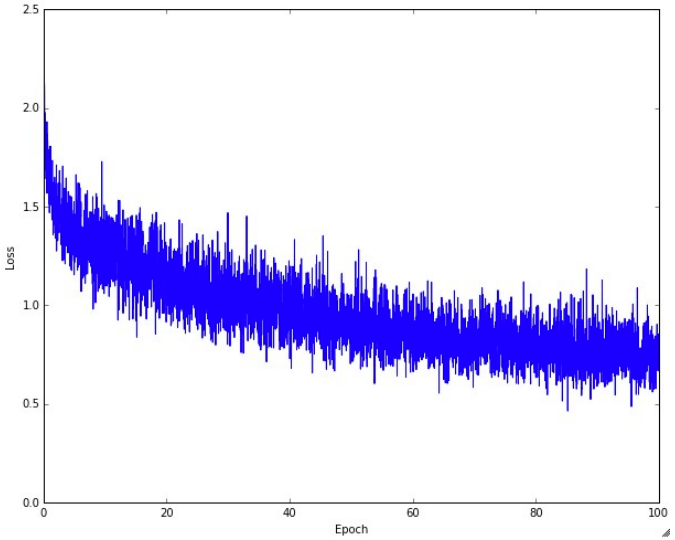
Evaluating the learning process

High learning rate will quickly decay the loss faster

- Can get stuck at worse values because the parameters bounce around chaotically

Low learning rate induces very low convergence speed



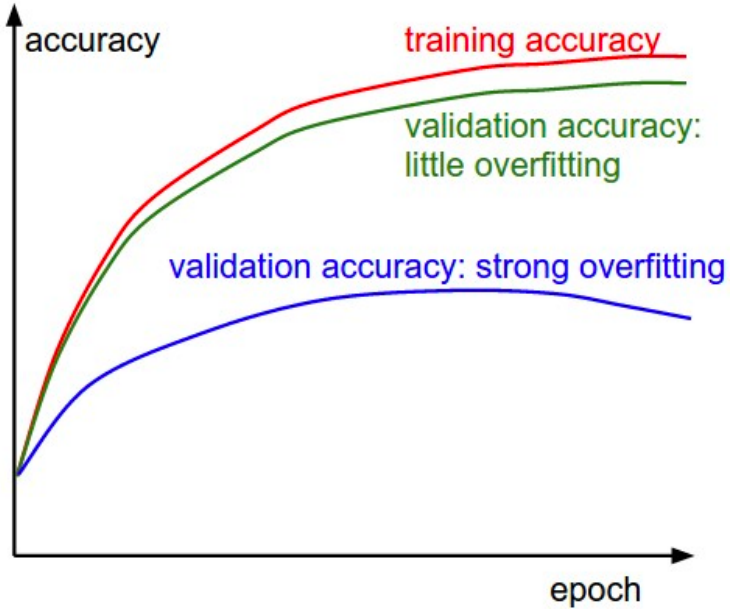


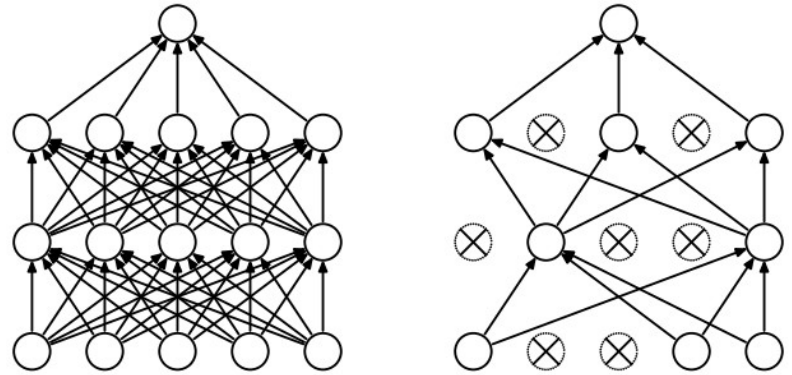
Typical loss function over time

- A slightly too small learning rate ?
- Too low batch size ?

Overfitting & Regularisation

- Validation accuracy slightly below the training accuracy
  - Model capacity might not be large enough
    - increase the number of parameters
- Validation error curve shows small accuracy compared to training
  - Strong overfitting → increase regularization or training data
- L2 or L1 regularisation
  - Introduce terms penalising large weights
- Dropout
- …

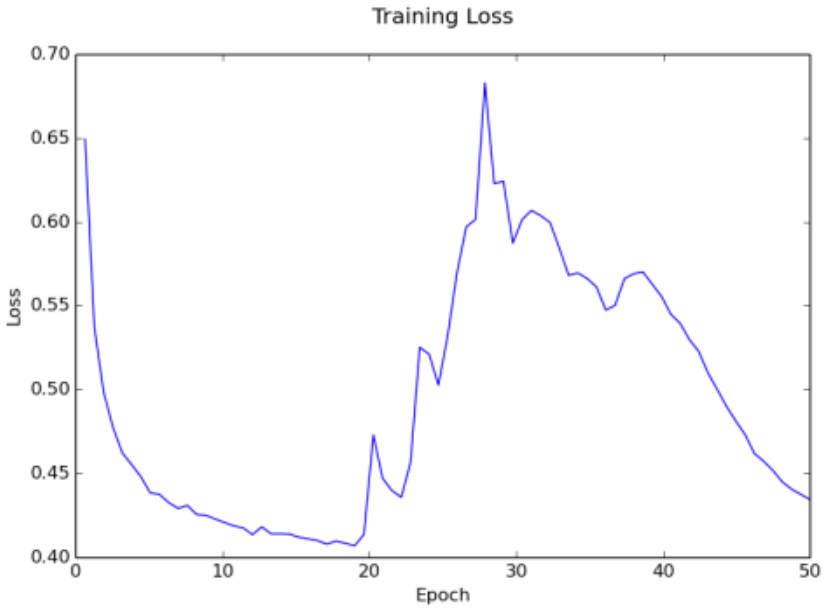




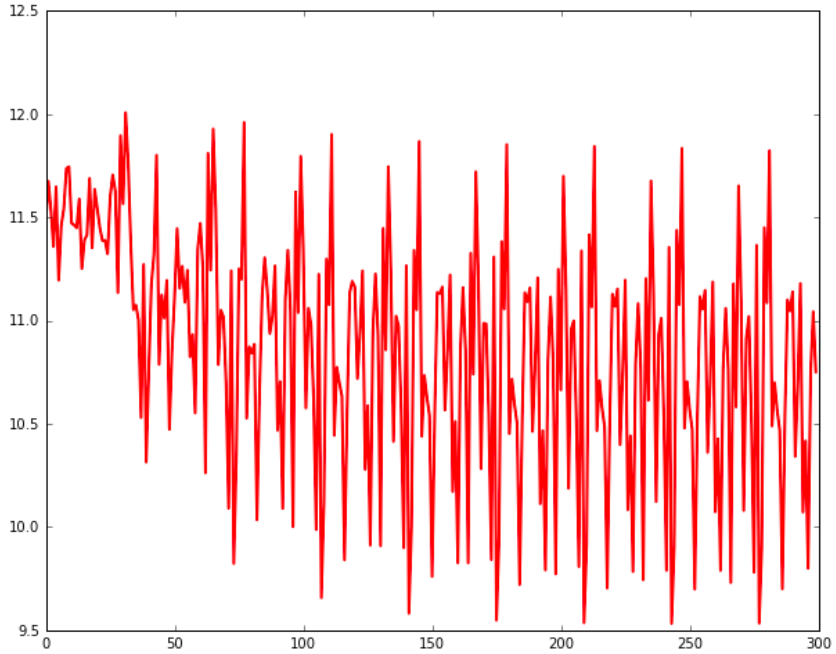
(a) Standard Neural Net (b) After applying dropout.

“Art” pieces

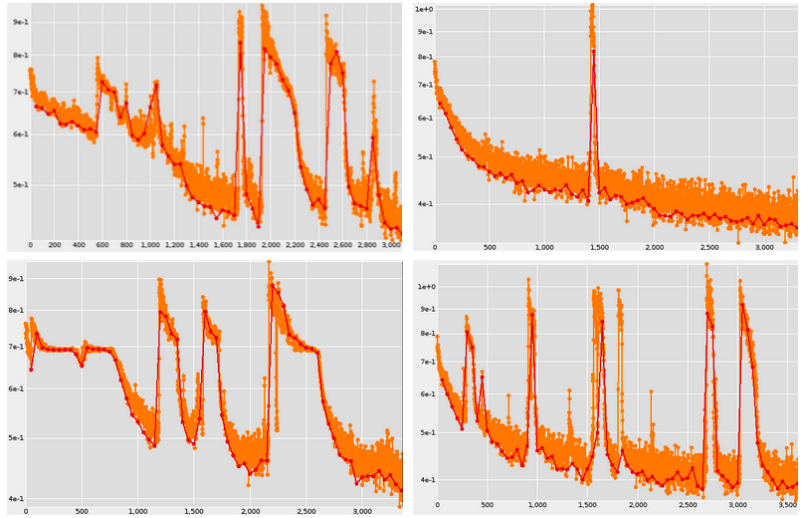
From https://lossfunctions.tumblr.com



"This RNN smoothly forgets everything it has learned. God knows what happened… »

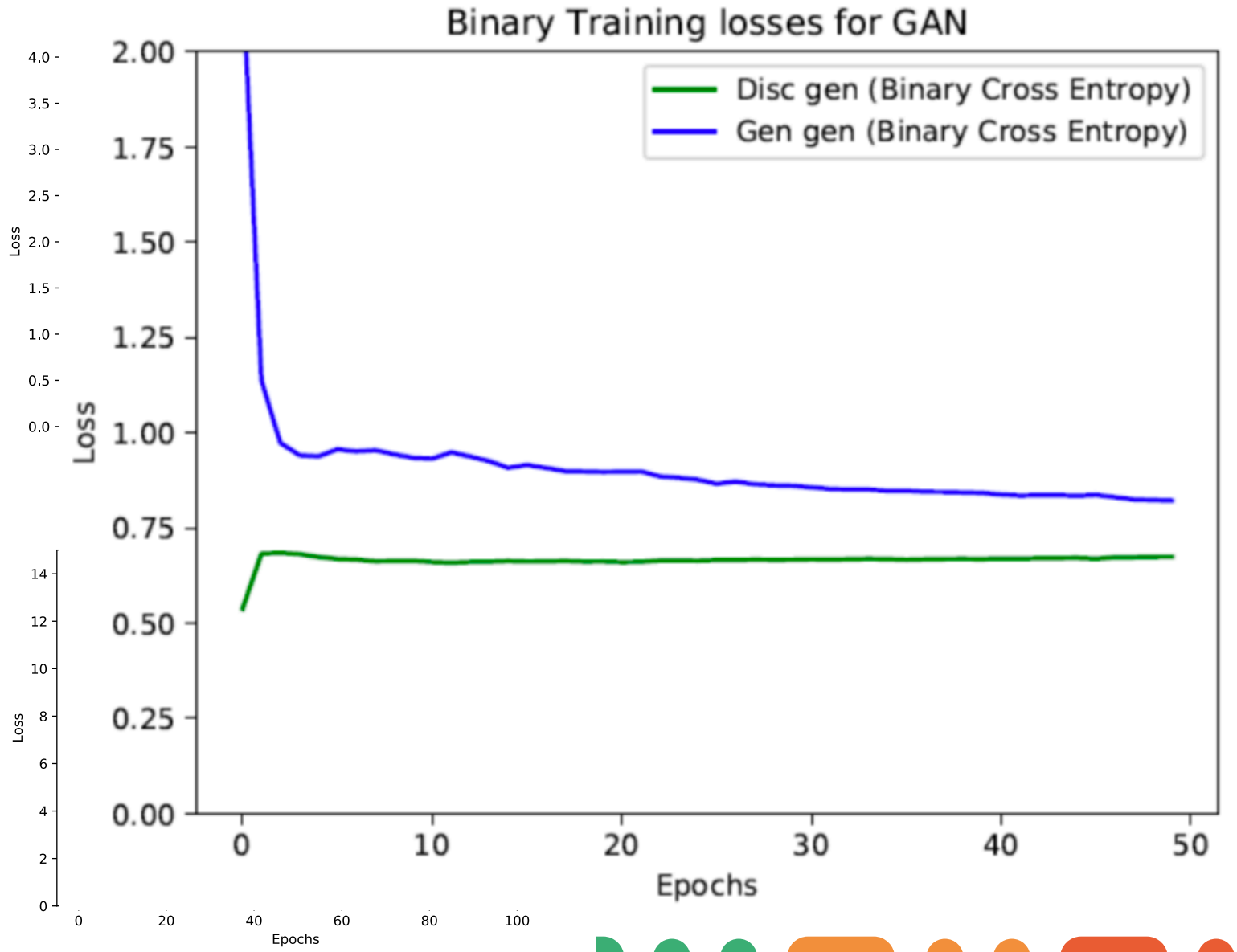


“A heart rate or a loss function? :)”



“Taming Spatial Transformer Networks, contributed by Diogo. For the record, it’s not supposed to look like that”

“Art” in HEP


Binary Training losses for GAN
Disc gen (Binary Cross Entropy)
Gen gen (Binary Cross Entropy)
Loss
Epochs
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
0
10
20
30
40
50
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
14
12
10
8
6
4
2
0
0
20
40
60
80
100

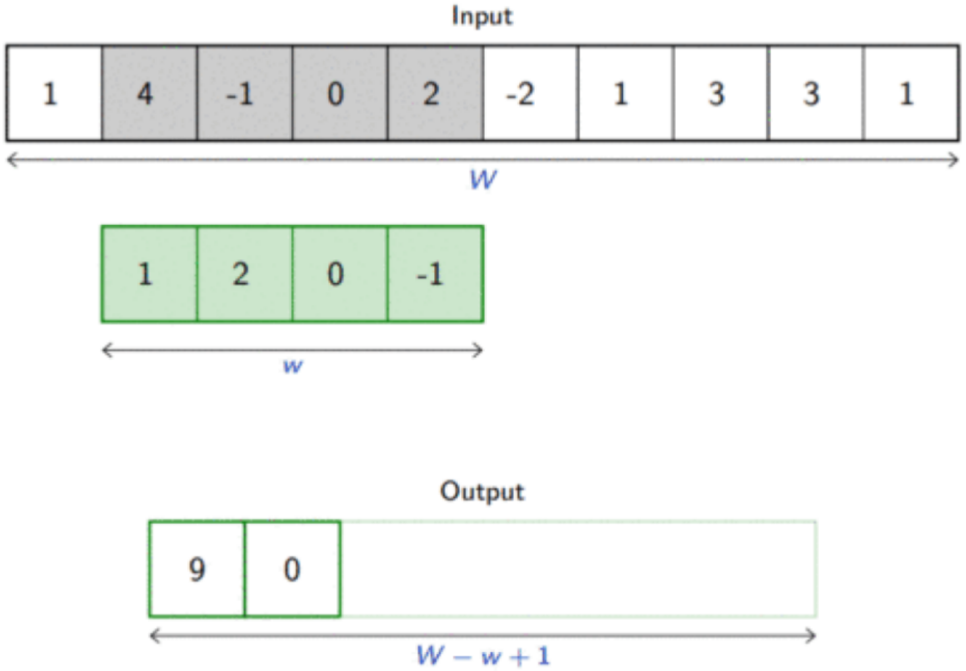
Part 2: Convolutional Neural Network and Generative models

Convolutions

$\mathbf{x} \in \mathbb{R}^M$ and kernel $\mathbf{u} \in \mathbb{R}^k$

discrete convolution $\mathbf{x} * \mathbf{u}$ is vector of size M-k+1

$$(x * \mathrm{u})_i = \sum_{b=0}^{k-1} x_{i+b} u_b$$



2D convolutions extract features from input image using “small squares of input data”

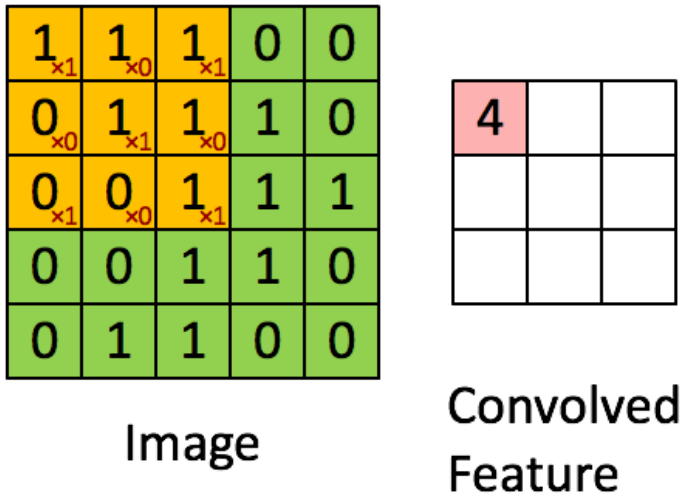
preserve spatial relationship between pixels.

Ex: 5 x 5 input image, 3 x 3 filter

Slide the filter matrix

Element wise multiplication

Sum of the multiplication outputs

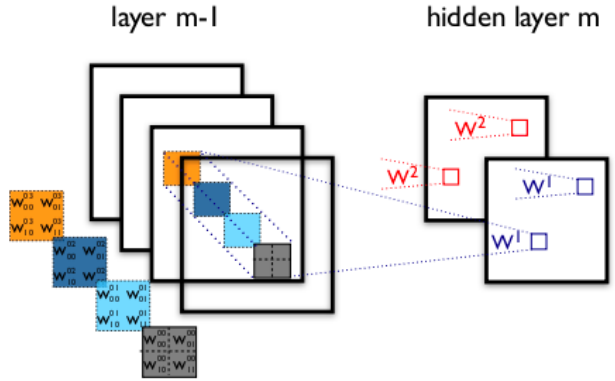


CERN openlab

Example

Image from CVPR 2012 tutorial

Convolutional Neural Networks

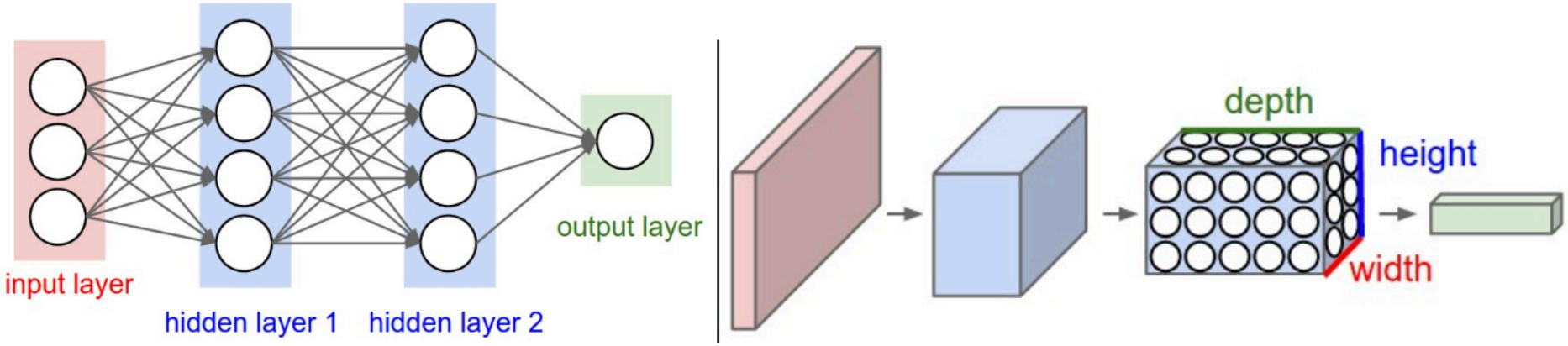


CNNs exploit spatially-local correlation

Enforce local connectivity pattern between neurons of adjacent layers

Neurons are arranged in 3 dimensions (width, height, depth)

Every layer transforms the 3D input volume to a 3D output volume of neuron activations.



Input: image width x image height x RGB channel

https://github.com/rasbt/python-machine-learning-book/tree/master/code

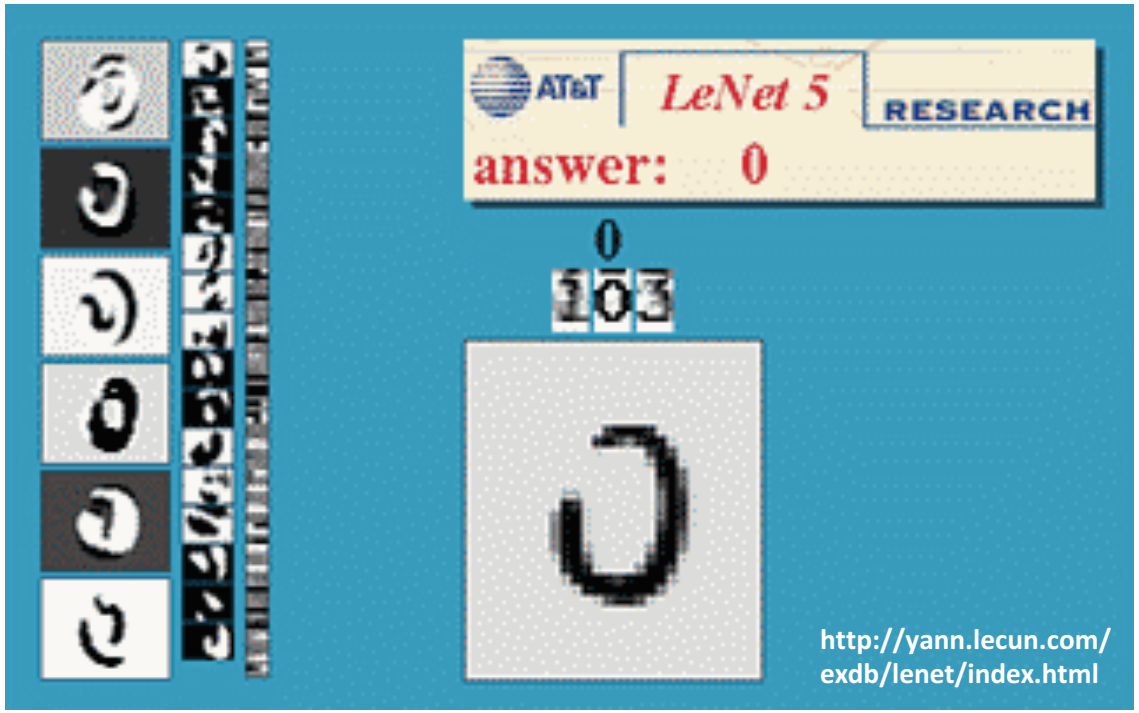
LeNet

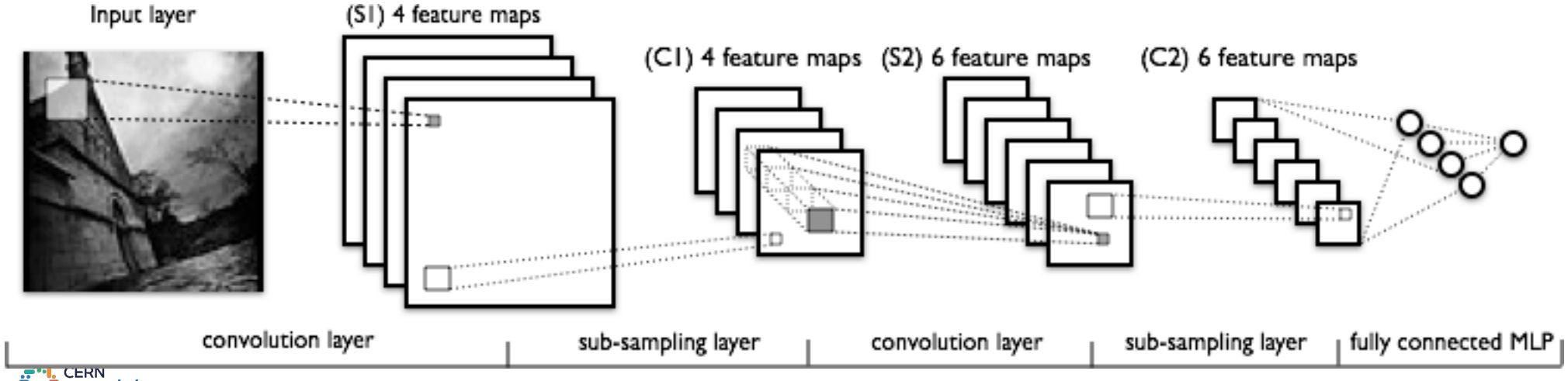
7-layers CNN to recognise hand-written numbers on checks

- digitized in 32x32 pixel greyscale input images.

to process higher resolution images need larger and more convolutional layers

- availability of computing resources!





LeCun et al.[34] 1998

ImageNet dataset

Imagenet : dataset of labelled images across hierarchical categories

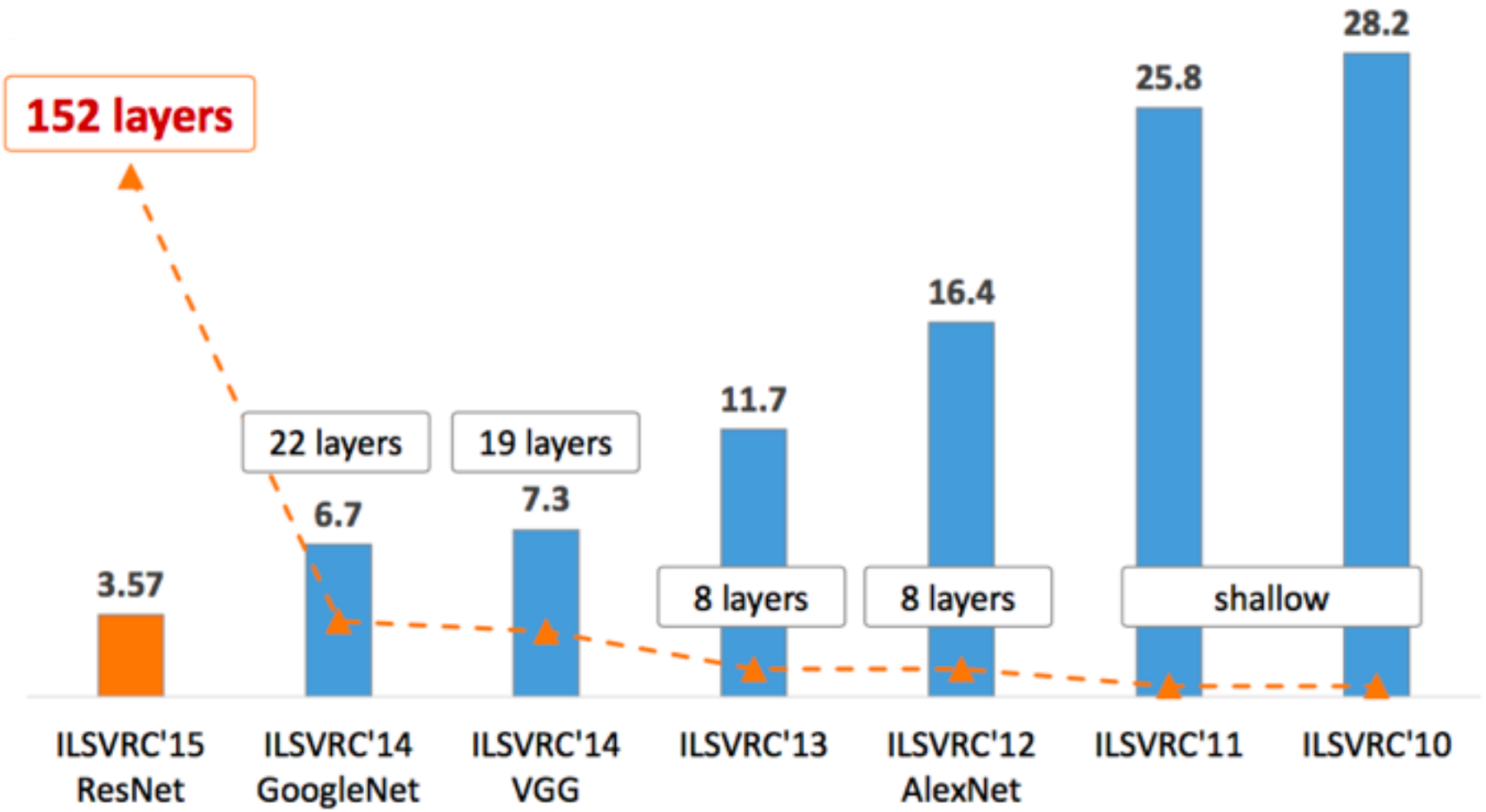
- Now: >14 M images with 20k classes

ImageNet Large Scale Visual Recognition Challenge (ILVRC) started in 2010 with 100 classes.

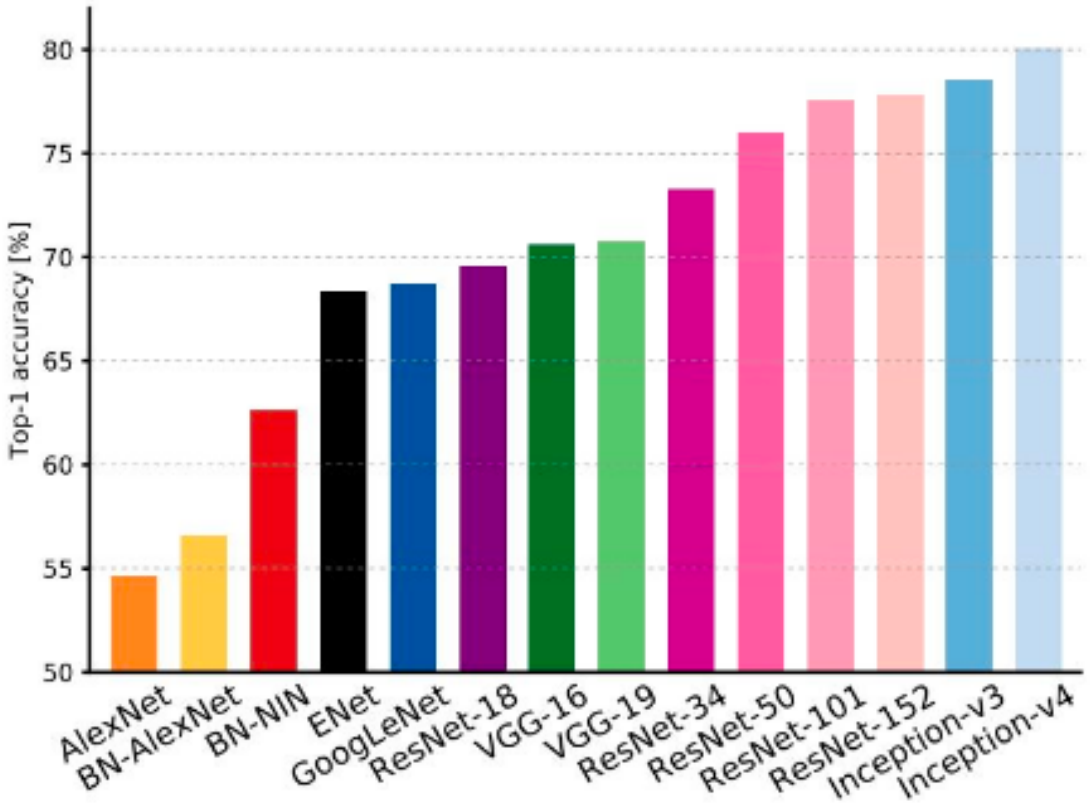
- 1000 classes now

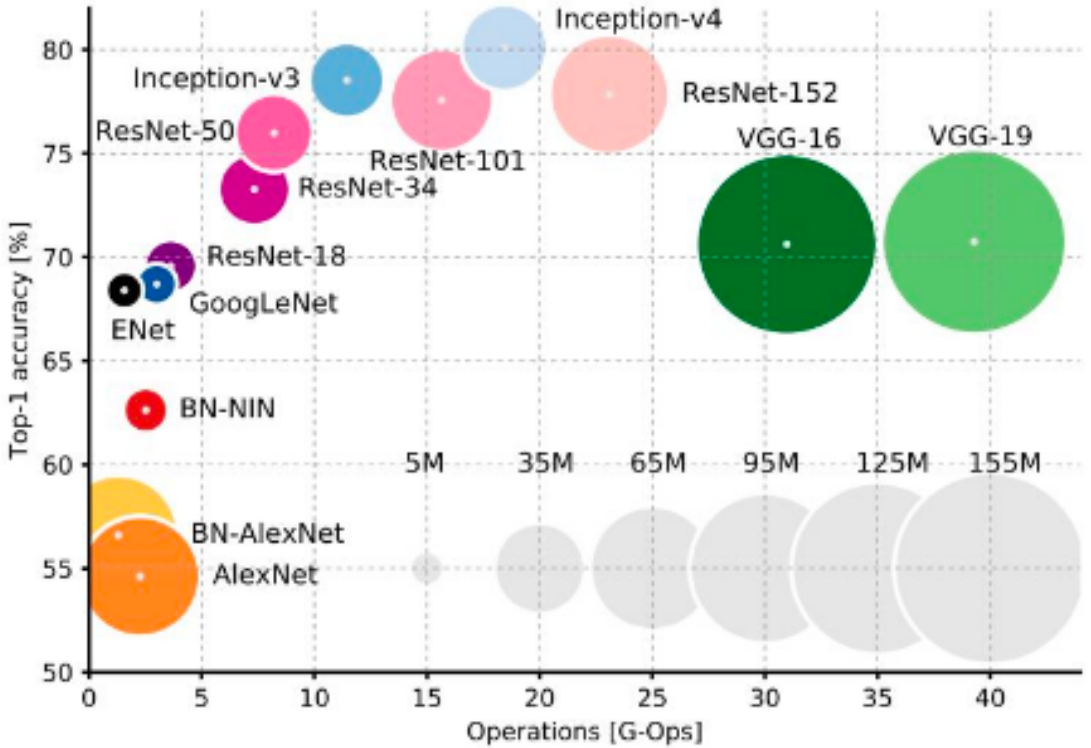
Original Paper: http://www.image-net.org/papers/imagenet_cvpr09.pdf

ILSVRC winners



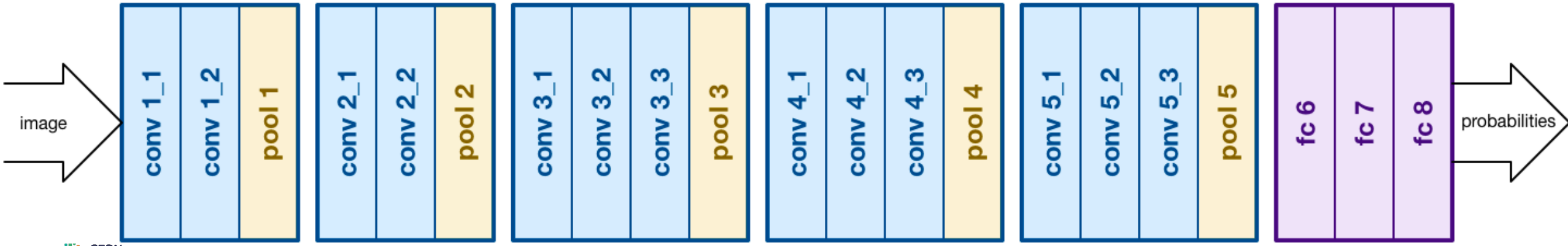
Some statistics





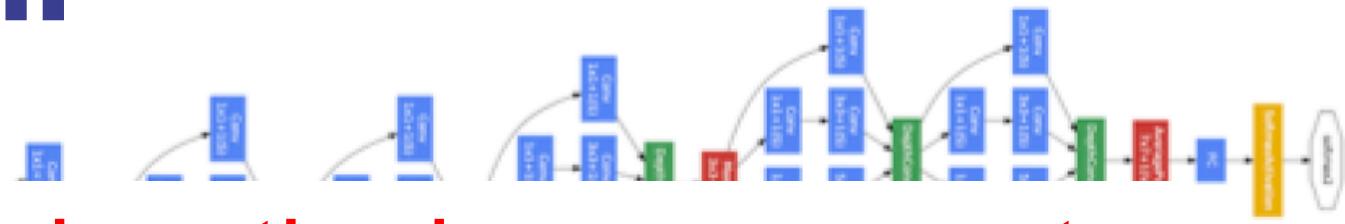
ILSVRC 2014 winner: VGGNet

- very uniform architecture
  - 16 convolutional layers
  - 3x3 convolutions (lots of filters) –> Total 138 million parameters
- Trained on 4 GPUs for 2–3 weeks.
- Currently the most preferred choice for image feature extraction
  - Weight configuration is publicly available (an example in our tutorial)



Simonyan and Zisserman, 2014

GoogleNet/Inception



After a few days of training, the human expert (A.Karpathy) was able to achieve a top-5 error rate of 5.1%

- batch normalization, image distortions and RMSprop
- And the “inception” model
  - several very small convolutions to drastically reduce number of parameters

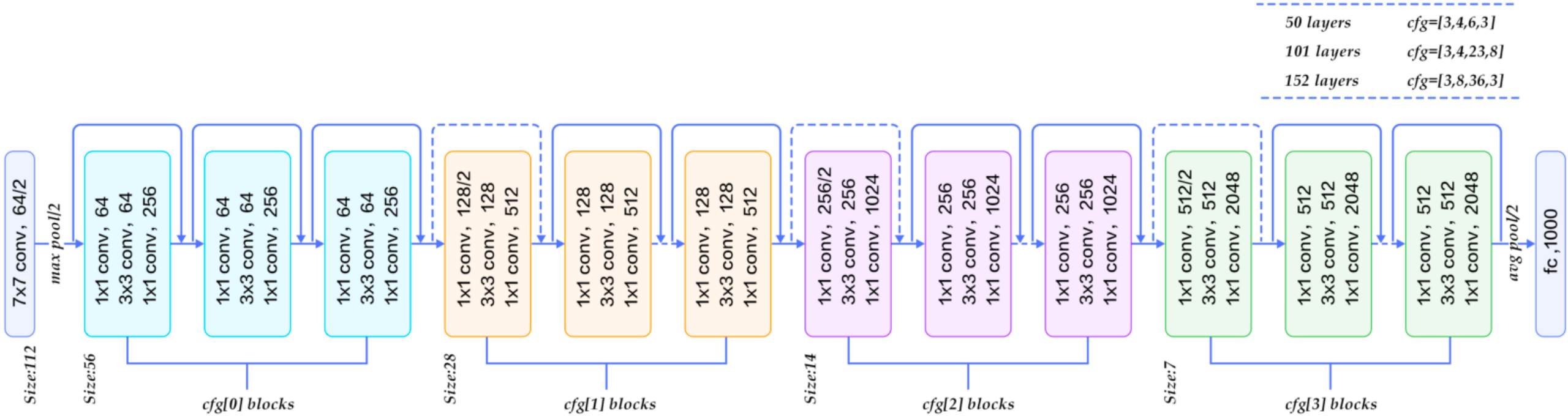
22 layer deep ~4M parameters (AlexNet ~60M)

6.6 % error rate

Convolution
Pooling
Softmax
Other

ILSVRC 2015: ResNet

Residual Neural Network (ResNet) by introduced a novel architecture with “skip connections” (gated recurrent units) and heavy batch normalization.

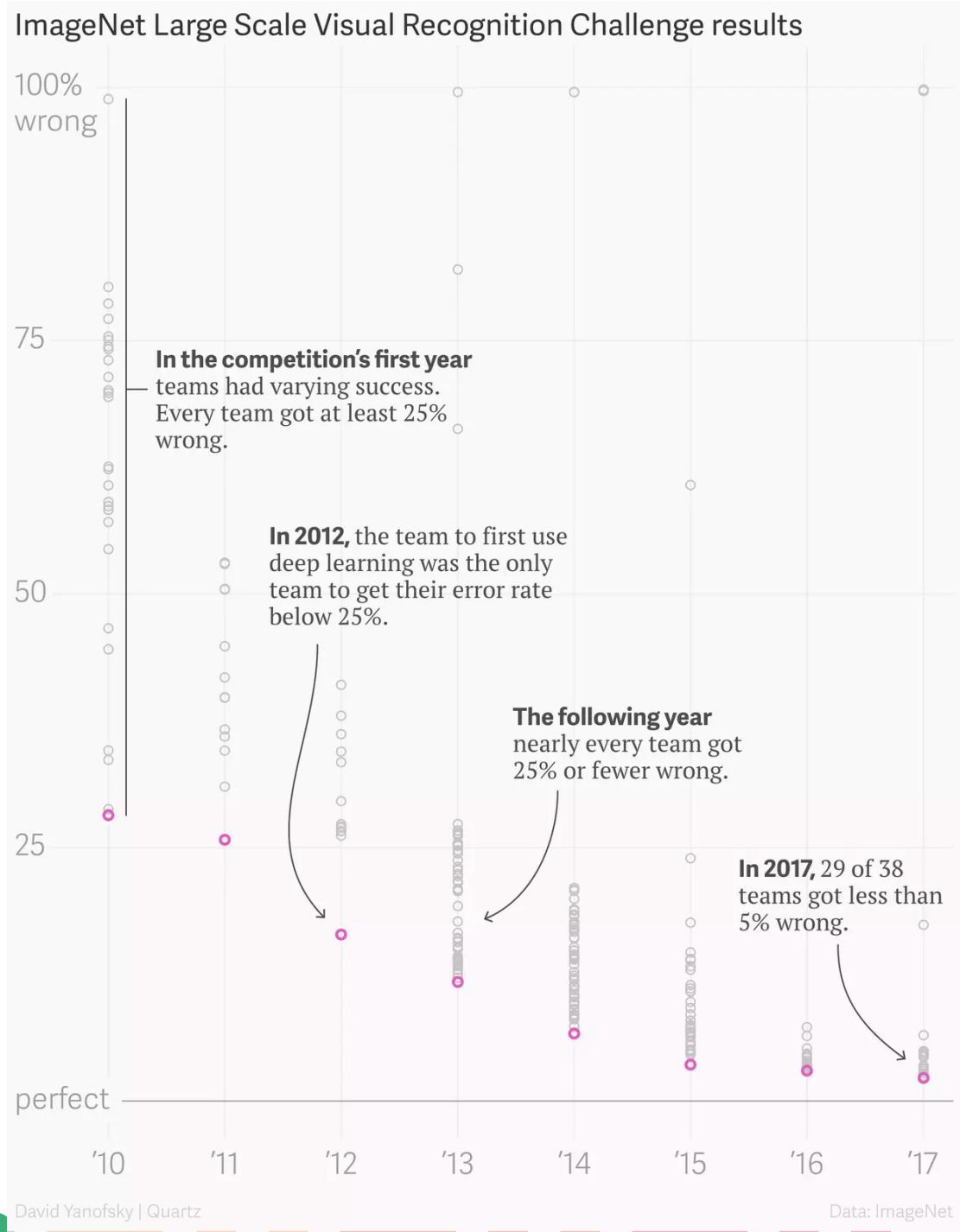


Kaiming He et al, 2015

What now?

In 2017 29/38 competitors got better than human score.

Since 2018 community is discussing increasing difficulty to 3D images

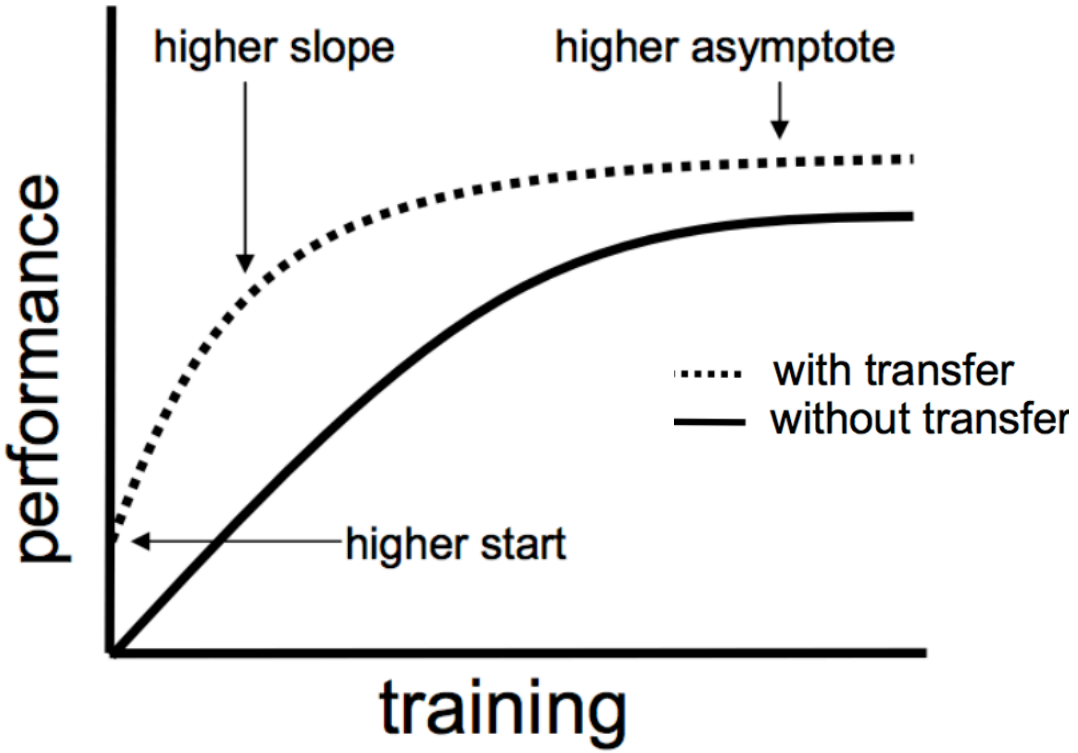


Transfer learning

"Transfer learning and domain adaptation refer to the situation where what has been learned in one setting … is exploited to improve generalization in another setting"

Page 526, Deep Learning, 2016.

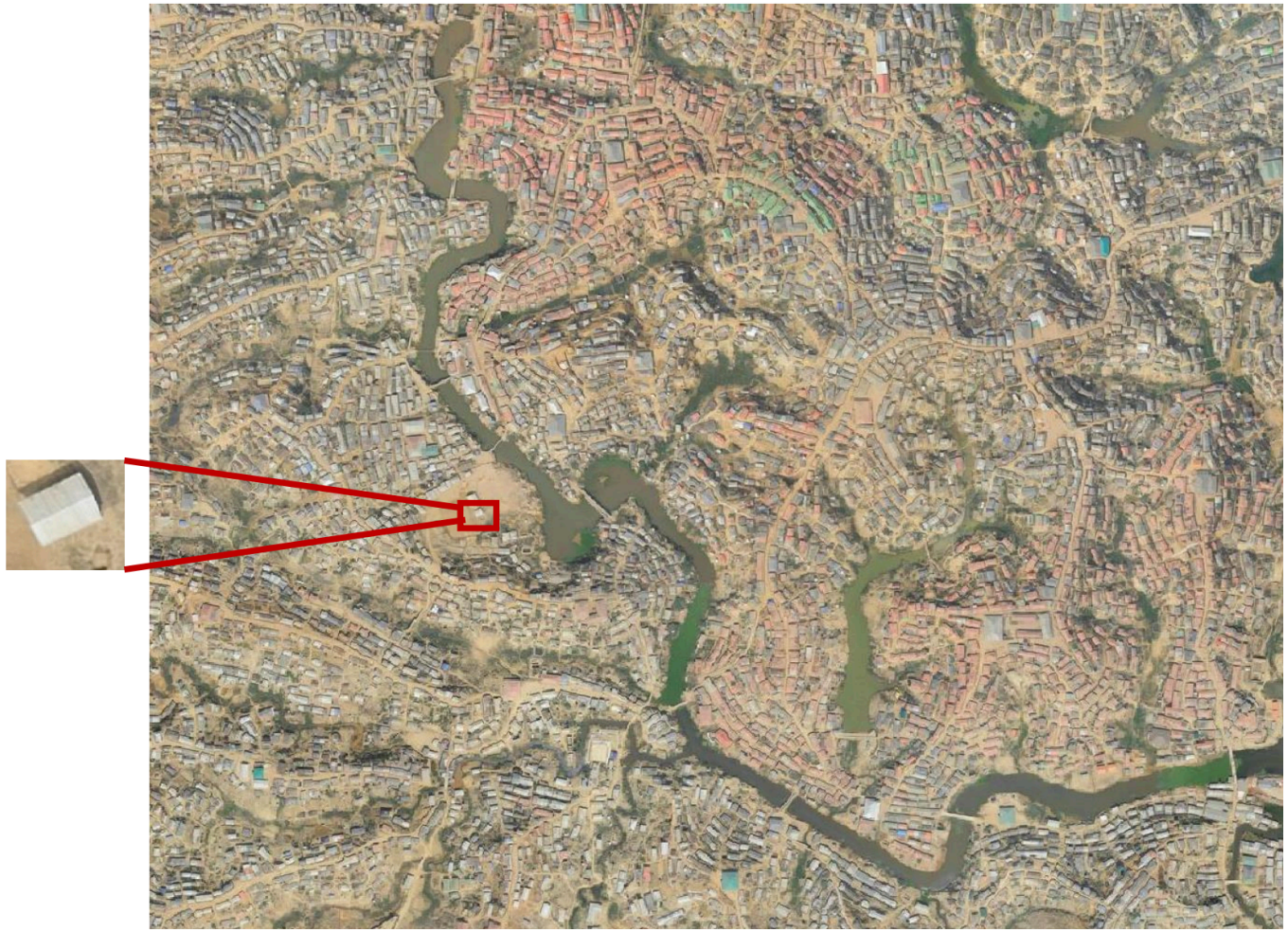
- Improvement of learning on a new task, transferring the knowledge already learned on a similar task
  - Might be the only option given the amount of resources needed from training
- Carefully choose how much of the pre-trained model to use in new one
  - CNN features are more generic in early layers and more dataset-specific in later layers



Counting shelters in refugee camps

CERN openlab and UNOSAT collaboration

(UN Operational Satellite Applications Centre)



Scan million pixels satellite photos for disaster relief:

- Evolution of refugee camps
- Natural disasters
- Building damage

Because of the high level of precision required

- it’s done MANUALLY!!!!

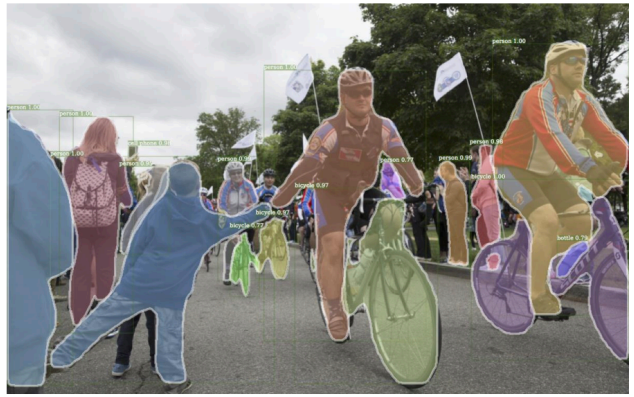
Counting shelters in refugee camps

CERN openlab and UNOSAT collaboration

(UN Operational Satellite Applications Centre)

CERN openlab Summer Students Program

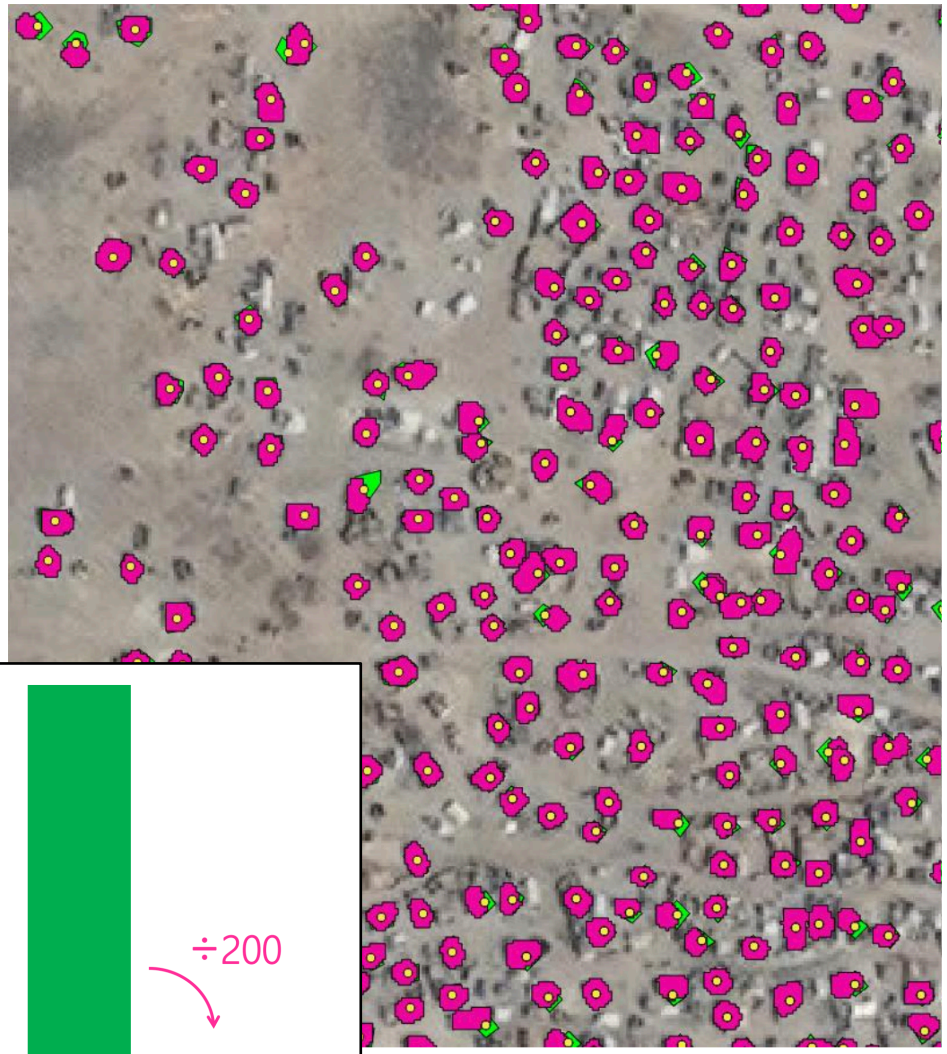
Why not use CNN instead??



Detectron Framework (FacebookAI)

Retrain & encode point data cleverly

Unosat Adapted model



Transfer learning from RCNN model
Average precision is 82%
Speedup is x200

CERN openlab

https://indico.cern.ch/event/727274/contributions/3100369/

CNN in HEP

Energy regression

CERN openlab Summer Students Program

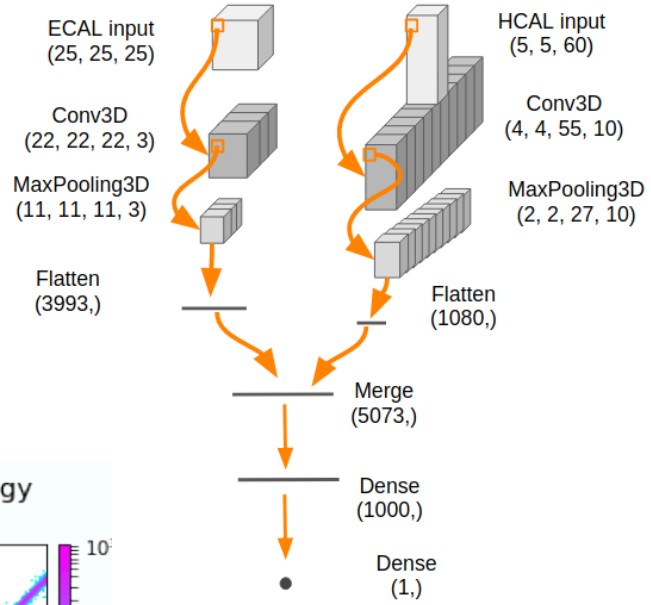
3D CNN can learn true energy of an incoming particle from the recorded hit pattern

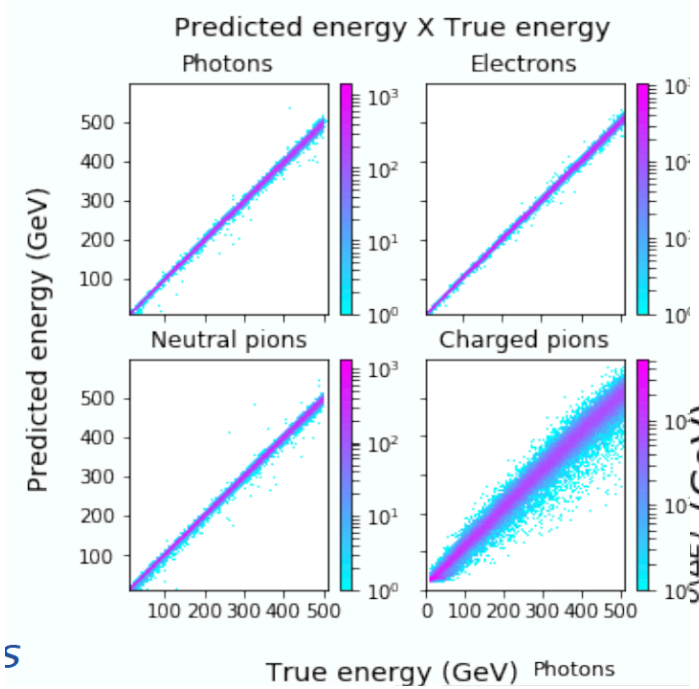
No high-level knowledge of physics

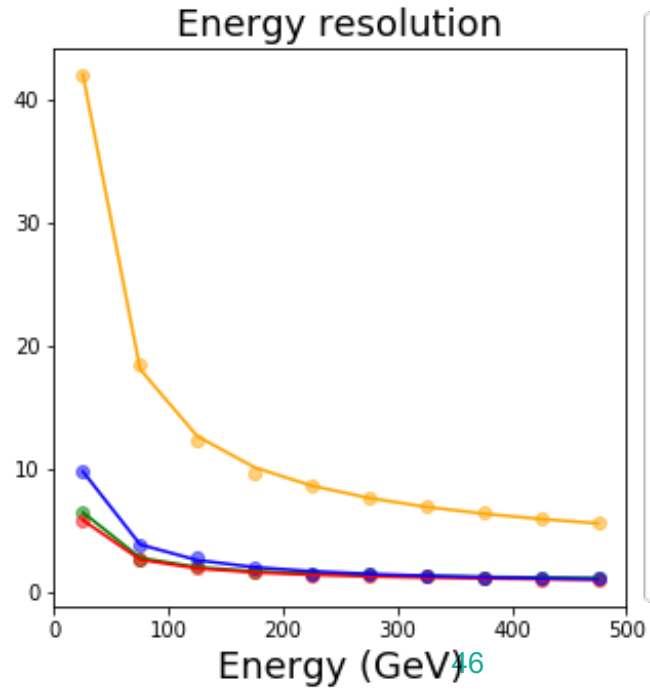
and/or detector features

used only RAW data as inputs

could be used offline and trigger



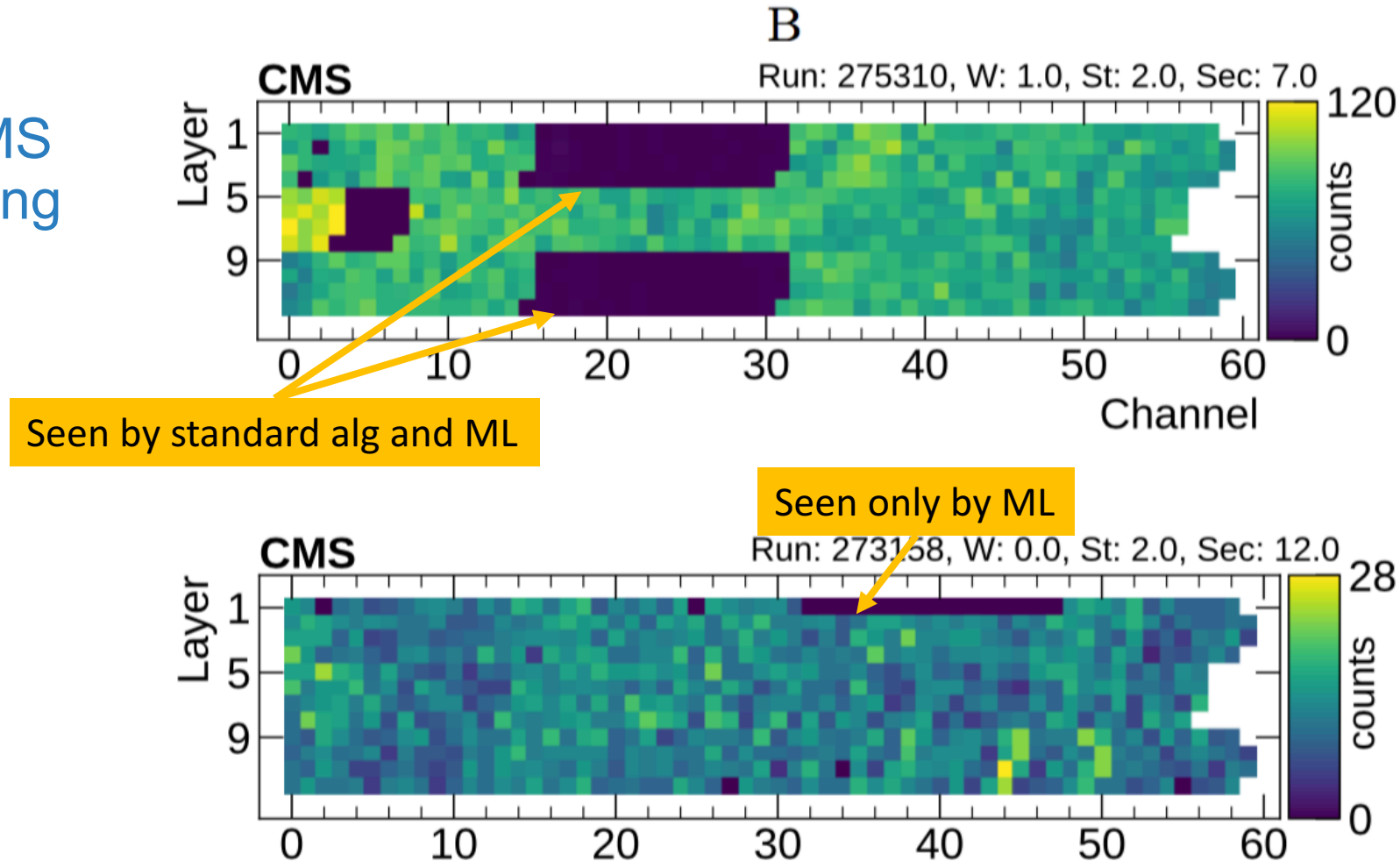




Work by K. Datta & V. Pacela, (2016/2017 Caltech Summer Students) and J.Mohapatra (openLab Summer student 2016)

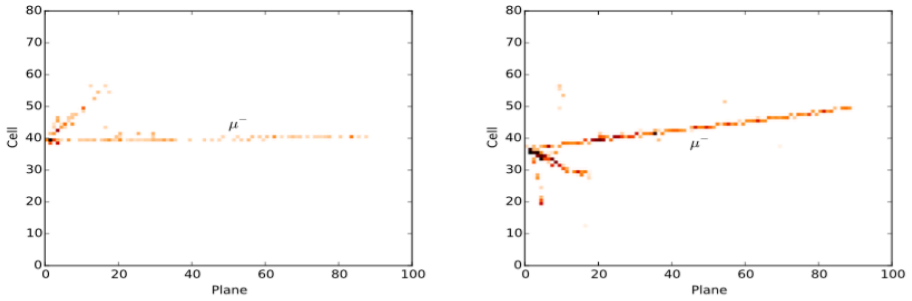
Anomaly : DQM application

Example application CMS muon chamber monitoring

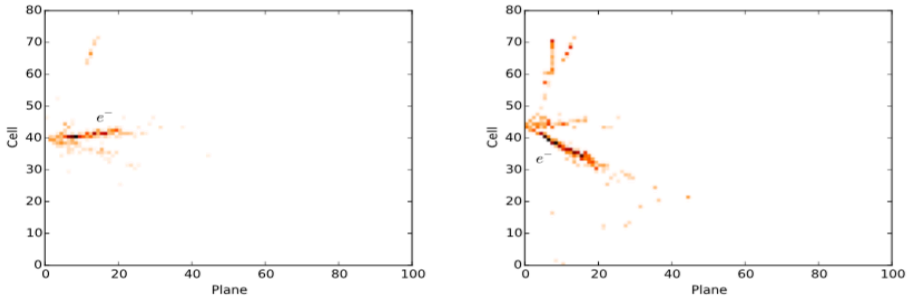


A. A. Pol, CHEP2018

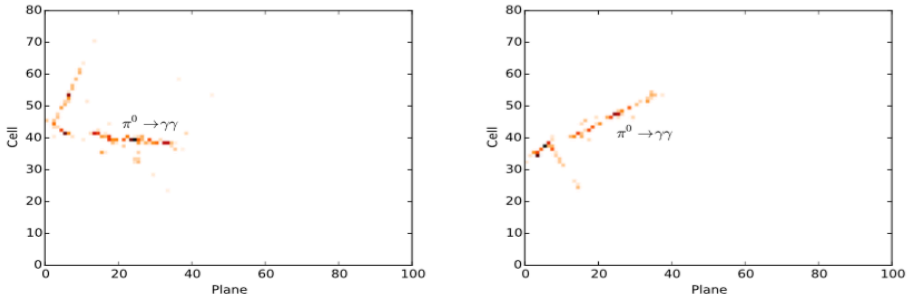
Neutrinos in NOVA



(a) ν_μ CC interaction.

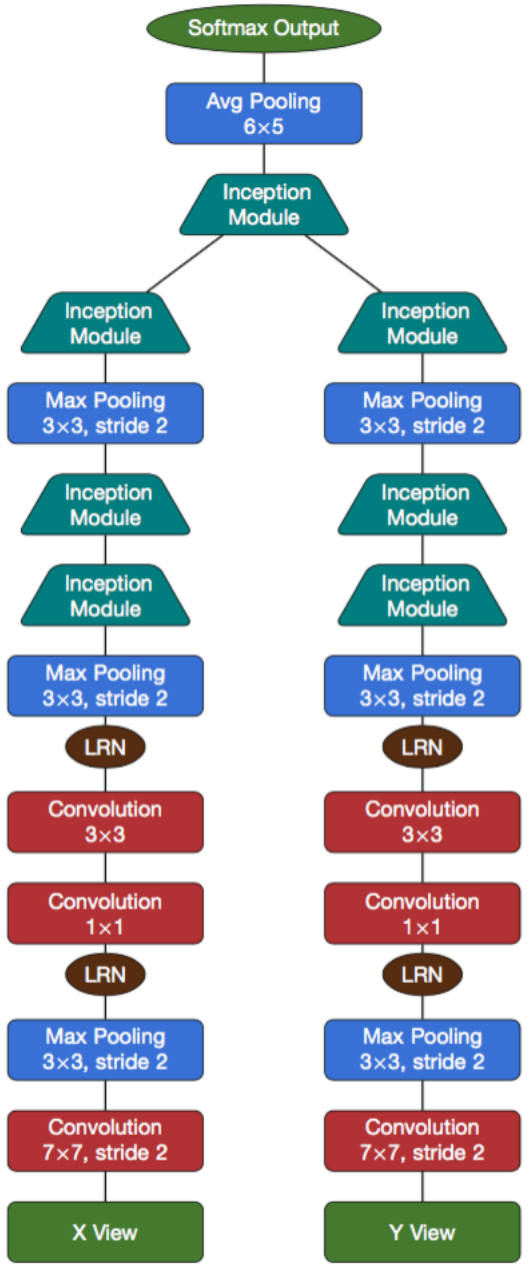


(b) ν_e CC interaction.



(c) NC interaction.

Neutrino interaction classification (GoogleNet)
Used in 1703.03328 1706.04592



arXiv 1604.01444 Aurisano et al

Generative models

The problem:

Assume data sample follows p_{data} distribution

Can we draw samples from distribution p_{model} such that $p_{model} \approx p_{data}$?

A well known solution:

Assume some form for p_{model} (using prior knowledge, parameterized by θ)

Find the maximum likelihood estimator

$$\theta^* = \arg\max_{\theta} \sum_{\mathbf{x} \in \mathcal{D}} \log(p_{\text{model}}(\mathbf{x}; \theta))$$

draw samples from $p_{\theta*}$

Generative models don't assume any prior form for p_{models}

Extract meaningful representation from training data

Deep Generative Models

Internal representations learned by shallow systems are simple (Bengio & LeCun 2007, Bengio 2009)

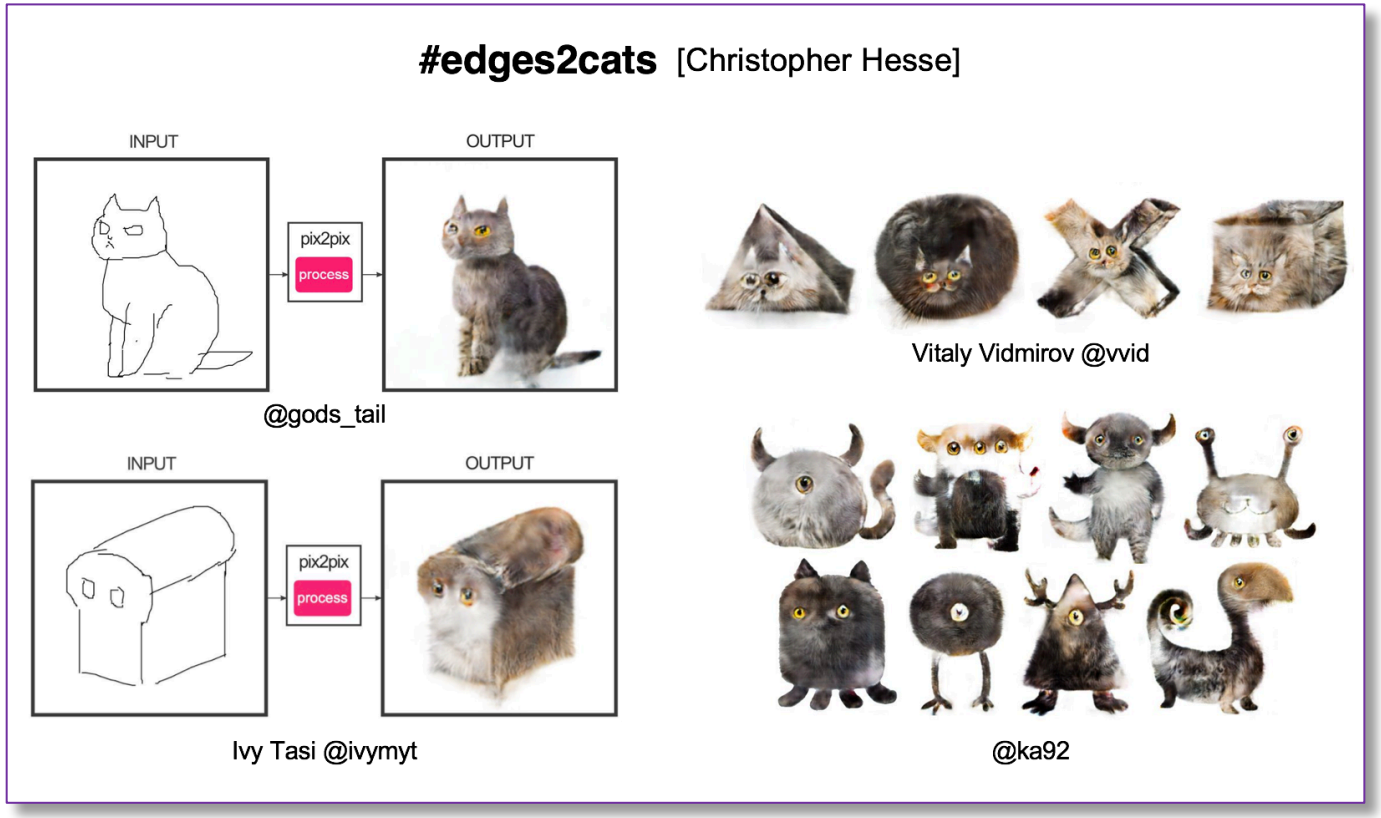
- Incapable of learning complex hidden structures
- Require large amounts of labeled data

→ Deep Generative Models

- → Allow higher levels of abstractions
- → Improve generalisation and transfer

→ Multiple applications

- Find Underlying Factors (**Discovery**)
- Detect Rare events (**Anomaly Detection**)
- Predict future events (**Planning**)
- Find Analogies (**Transfer Learning**)



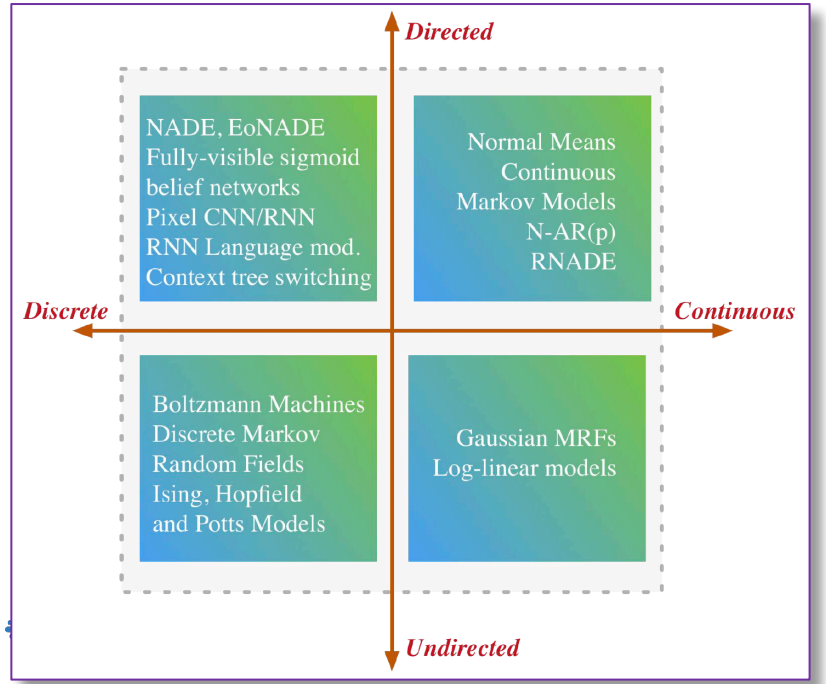
CERN openlab

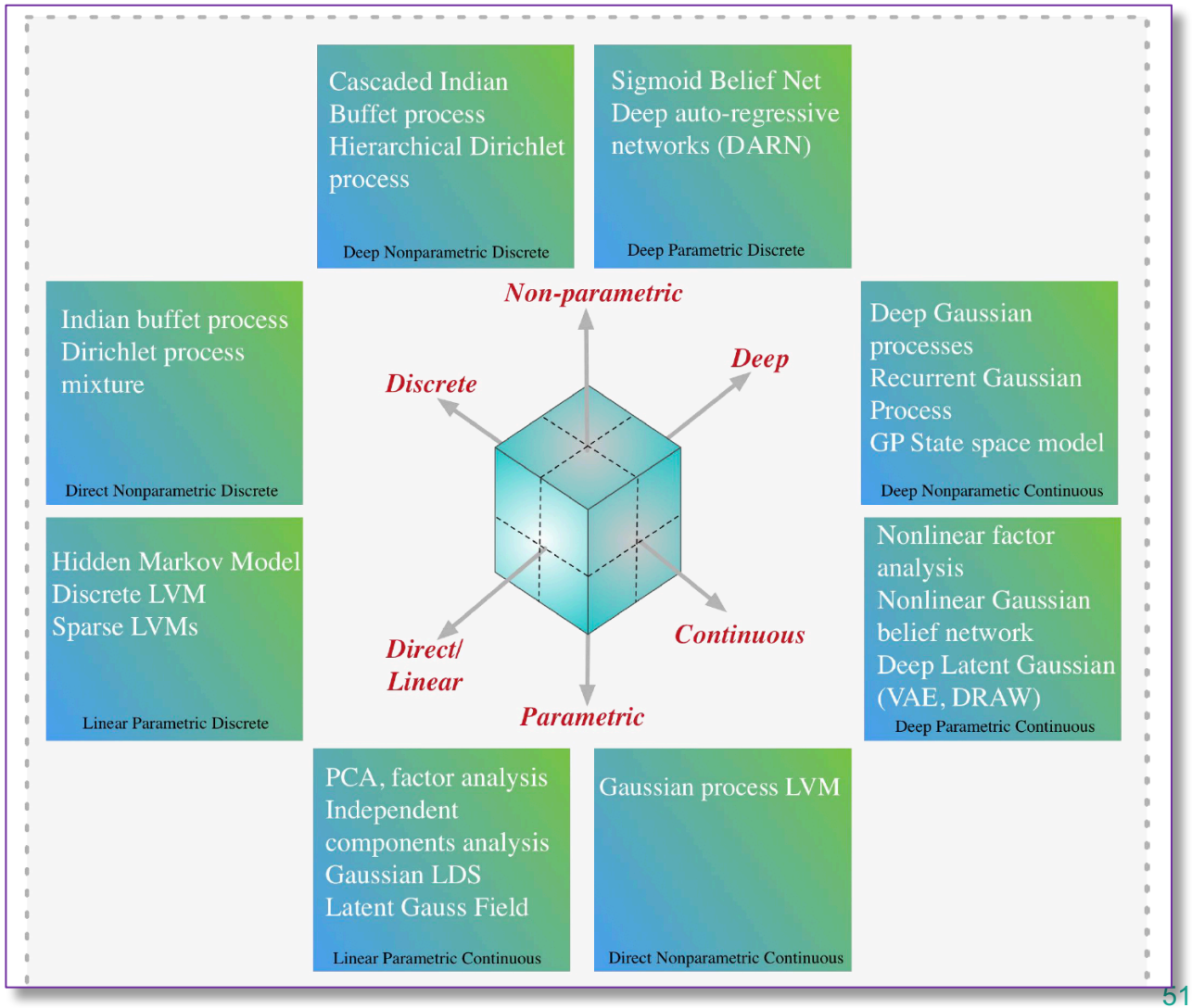
Pictures from Danilo Rezende's Tutorial on Deep Generative Models

Generative Models Zoo

Deep Generative Models → **Latent variables**

↓ **Fully-observed**





arXiv:1406.2661

Generative adversarial networks

Simultaneously train two networks that compete and cooperate with each other

Generator G generates data from random noise

Generator learning is supervised by a second network, **discriminator D**

Arxiv:1701.00160

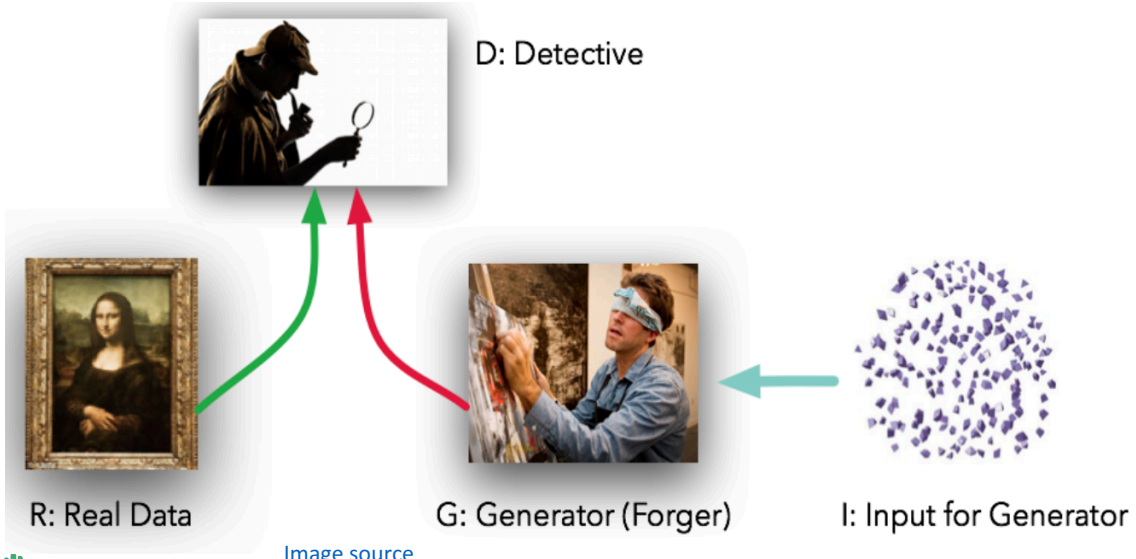


Image source

The forger/detective case

Forger shows its Monalisa to the detective

Detective says it is fake

Forger makes new Monalisa based on feedback

Iterate until detective is fooled

Generative adversarial training

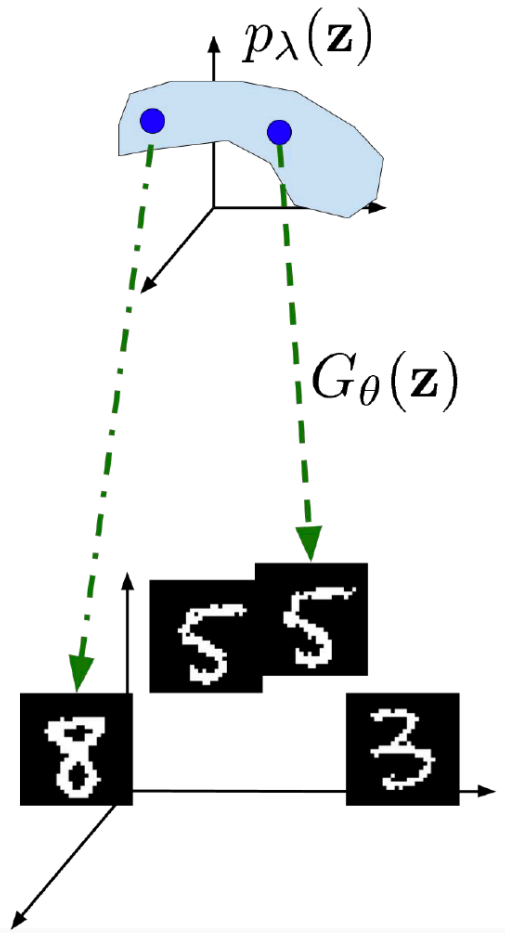
Assume a deterministic generator: $\mathbf{x} = G_\theta(\mathbf{z})$

A prior over latent space: $p_\lambda(\mathbf{z})$

Define a discriminator: $D_\psi(\mathbf{x}) \in [0, 1]$

A learnable loss function from the min-max game

$$\min_\theta \max_\psi \mathbb{E}_{\mathbf{x}\sim p_{data}}\Big[\ln D_\psi(\mathbf{x})\Big] - \mathbb{E}_{\mathbf{z}\sim p_\lambda(\mathbf{z})}\Big[\ln\big(1 - D_\psi(G(\mathbf{z}))\big)\Big]$$



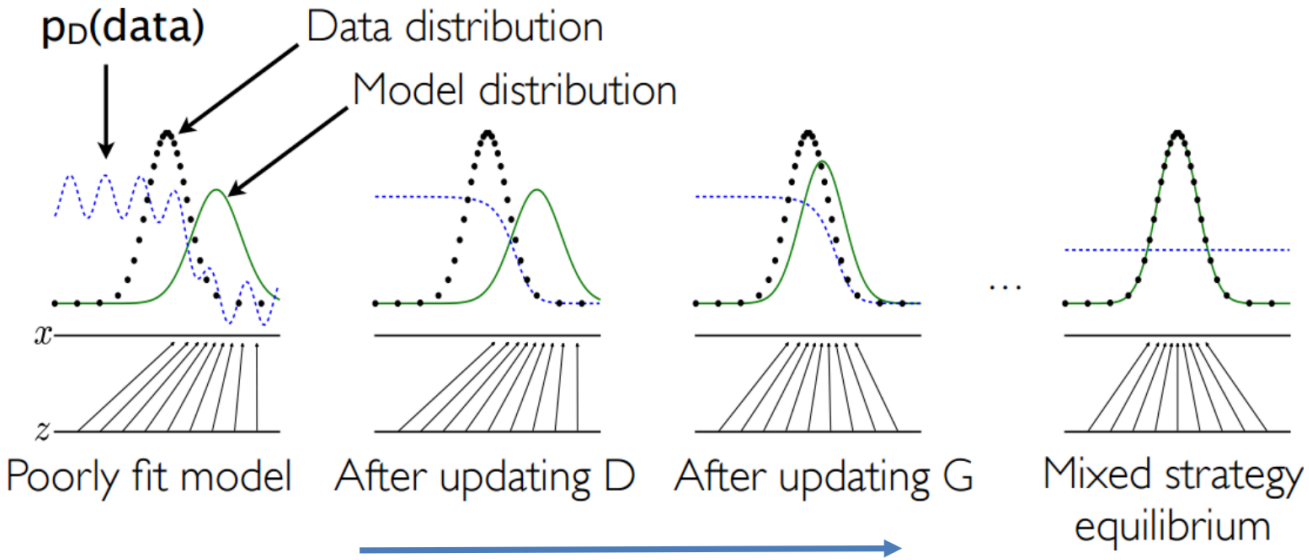
Equilibrium when **Jensen-Shannon divergence** between real and generated samples is minimized

arXiv:1406.2661v1

Generative adversarial training

Generator is trained to maximize the probability of Discriminator making a mistake

D gradient guides G to regions more likely to be classified as data



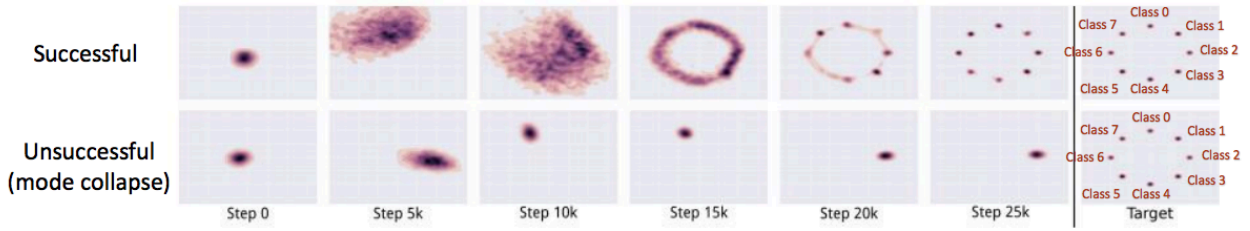
D is not an accurate classifier

G and D don't improve anymore.
D is unable to differentiate

D is trained to discriminate samples from data

Arjowski et al ’17, arxiv:1701.07875

Wasserstein GAN



Luke Metz, Ben Poole, David Pfau, and Jascha Sohl-Dickstein. 2016

Standard GAN loss formulation can lead to

- **vanishing gradients** when discriminator too powerful
- **mode collapse** (generating only a subset of the target distribution

Alternative **Wasserstein metric** (from Earth Mover’s distance)

- solution may not be optimal due to **biased** gradients

$$W(\mathbb{P}_r, \mathbb{P}_g) = \inf_{\gamma \in \Pi(\mathbb{P}_r, \mathbb{P}_g)} \mathbb{E}_{(x,y)\sim\gamma} \left[\|x - y\| \right]$$

γ «optimal transport plan»

Reformulate the loss using Cramer distance → remove bias on gradients → **Cramer GAN**

Other GAN flavors

Original GAN was based on MLP in 2014

Deep Convolutional GAN in 2015

Conditional GAN

- Extended to learn a parameterized generator $p_{model}(x|\theta)$;
- Useful to obtain a single generator object for all θ configurations
- Interpolate between distribution

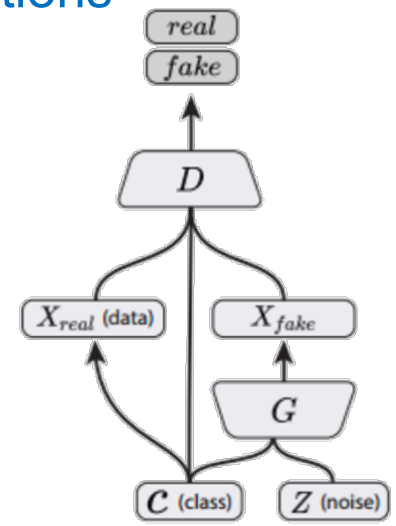
Auxiliary Classifier GAN

- D can assign a class to the image

Progressive growing GAN

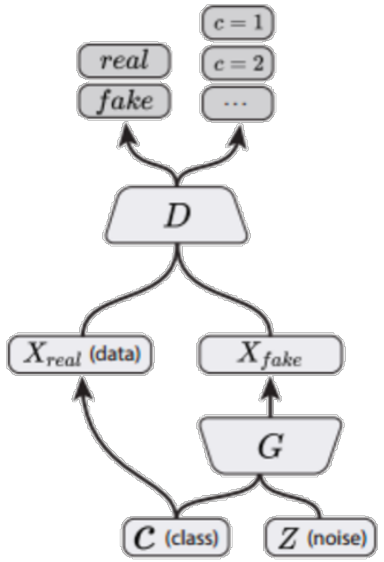
Stack GAN

BiGAN ..



Conditional GAN
(Mirza & Osindero, 2014)

arXiv: 1411.1784



AC-GAN
(Present Work)

arXiv:1610.0858

How well does it work?

Applications

GANs for ATLAS LAr calorimeter (I)

arxiv:1705.02355
arxiv:1701.05927

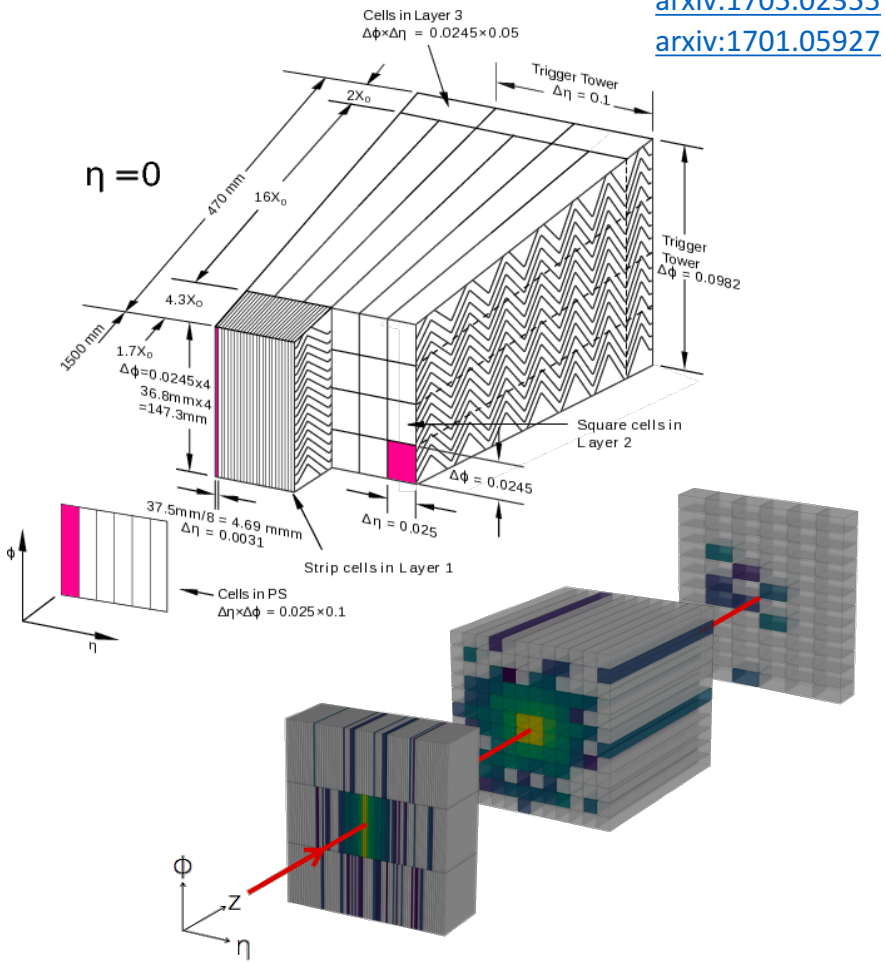
Complex geometry

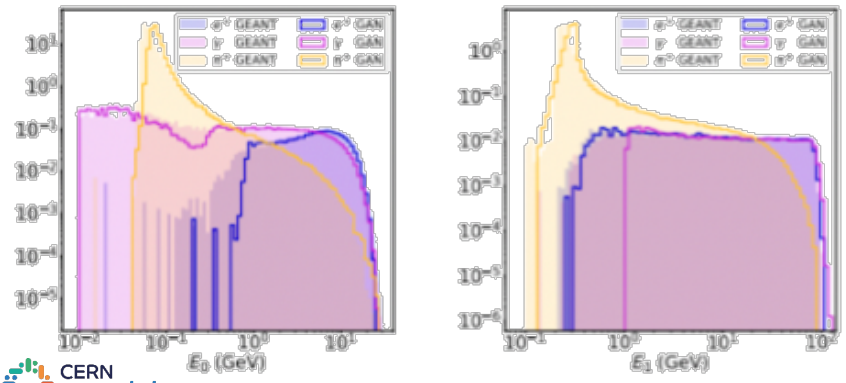
- Heterogeneous longitudinal segmentation into 3 layers
- Irregular granularity in eta and phi

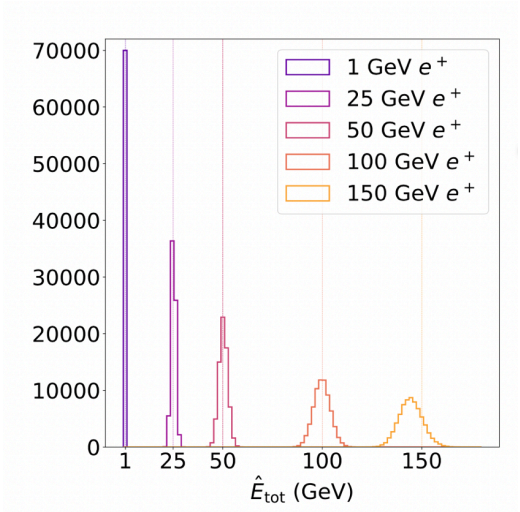
Energy deposition in each layer as a 2D image

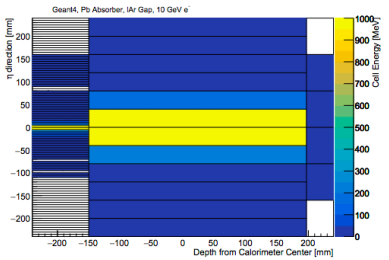
CaloGAN Build one LAGAN per layer

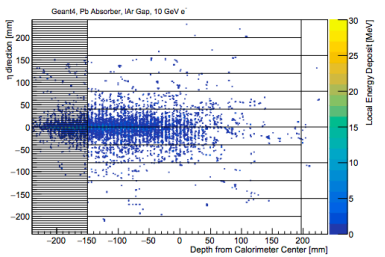
Result is a concatenation of 2D images that reproduce full 3D picture











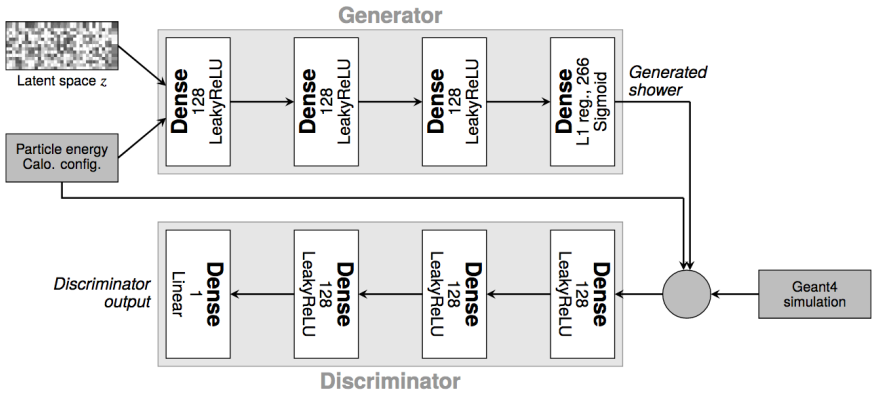
CERN openlab

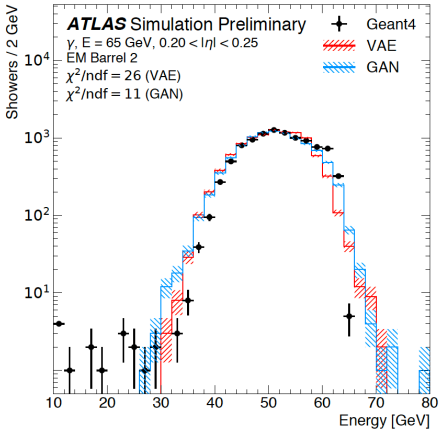
GANs for ATLAS LAr calorimeter (II)

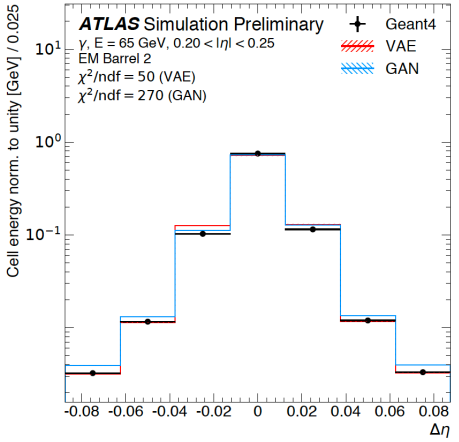
ATL-SOFT-PUB-2018-001

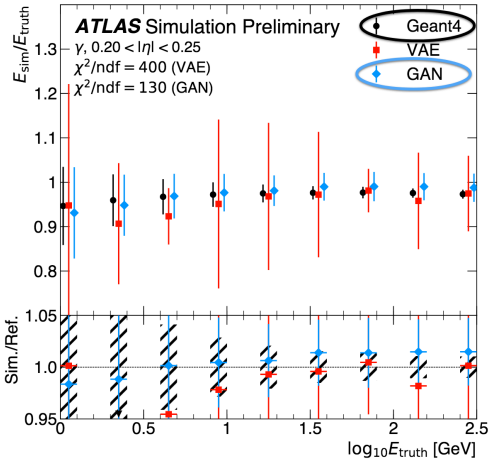
Wasserstein GAN model reproduces the mean energy distribution but not its width

At convergence critic can't see the difference in real and fake images anymore.

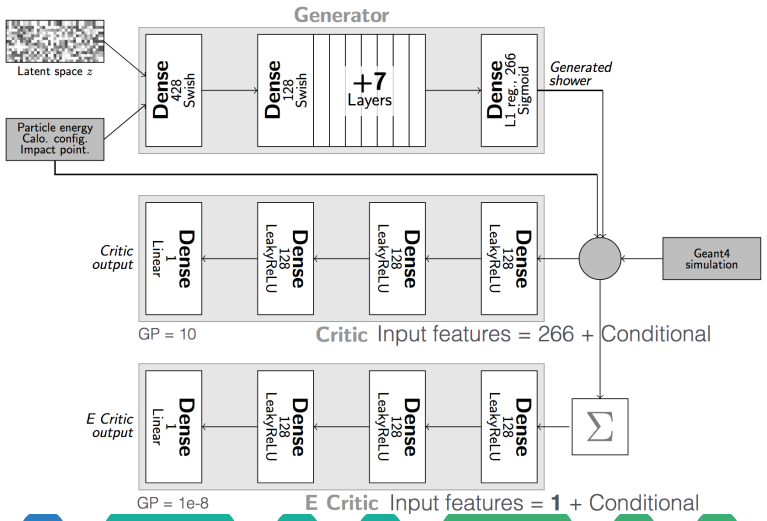


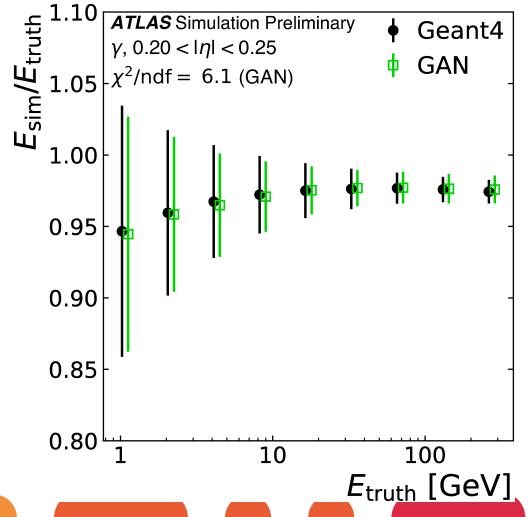






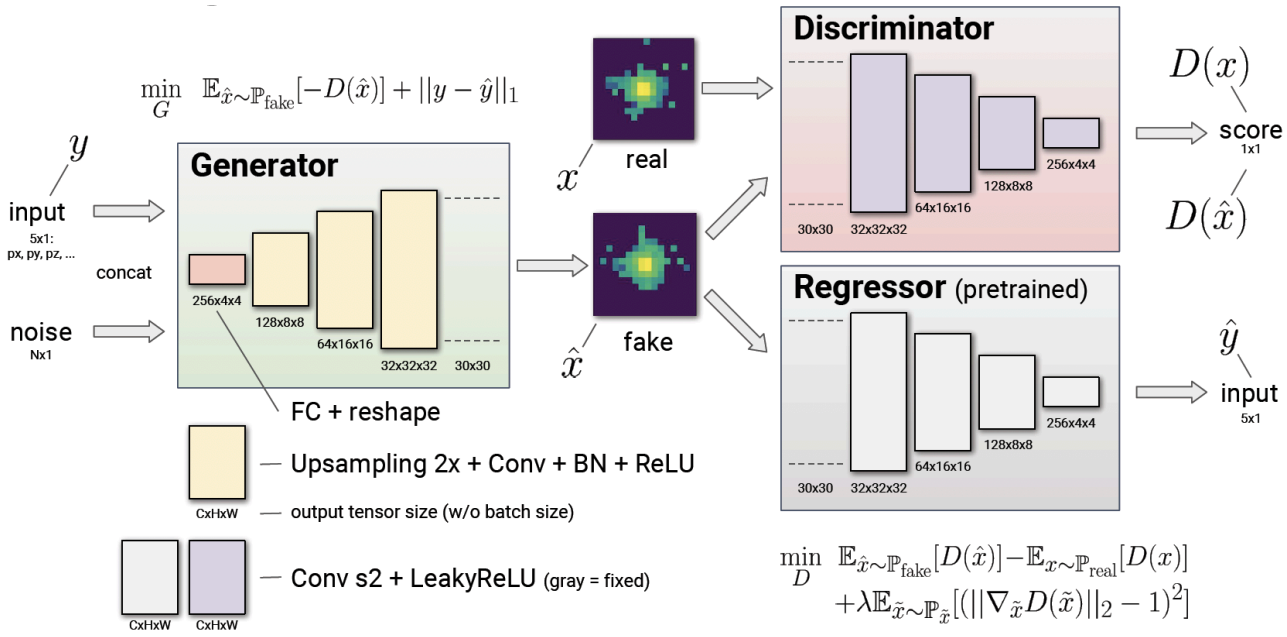
Train the generator against two critics

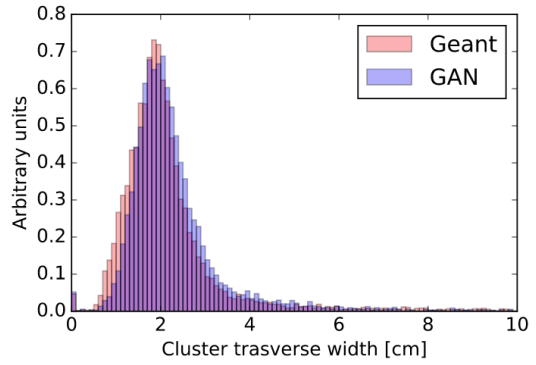


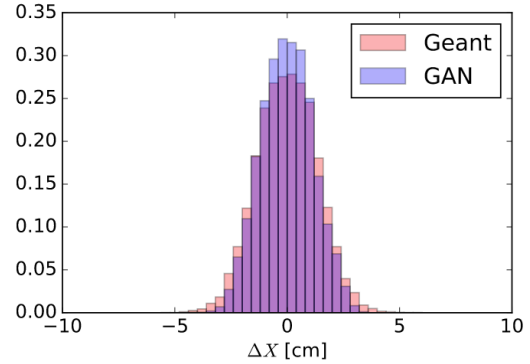


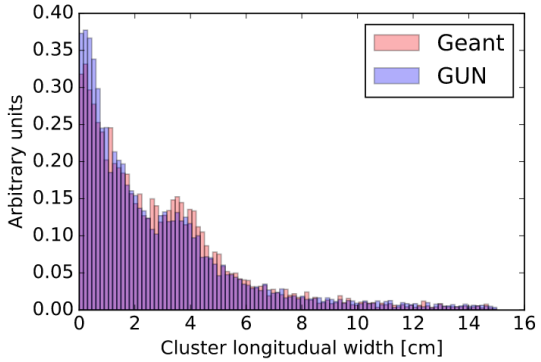
LHCb Calorimeter fast simulation

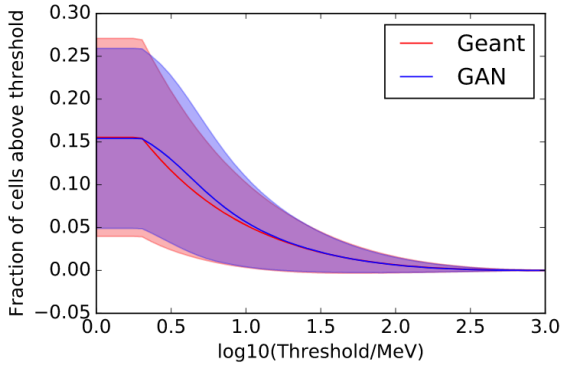
Wasserstein Convolutional GAN

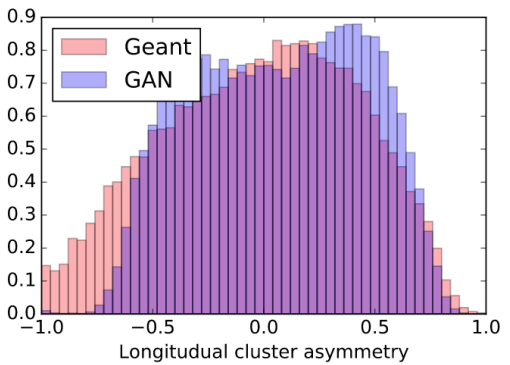


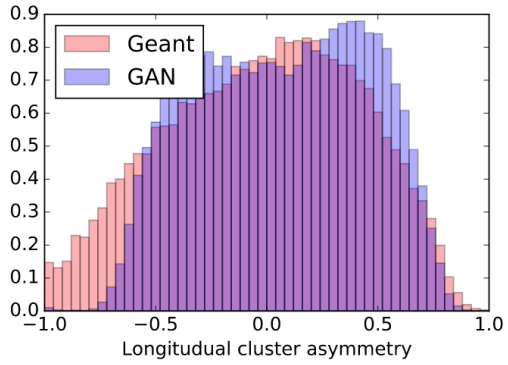






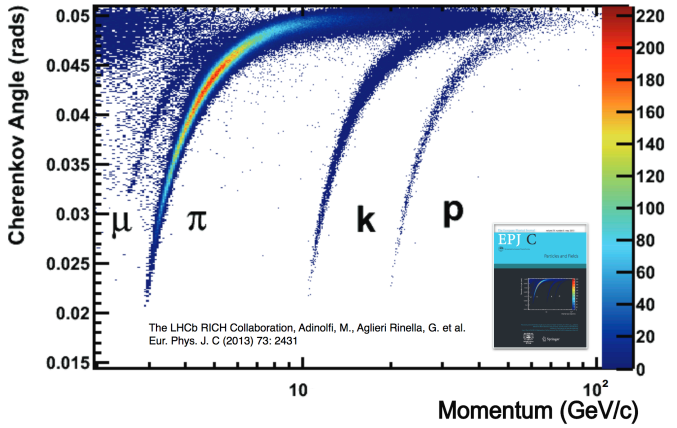






CERN openlab

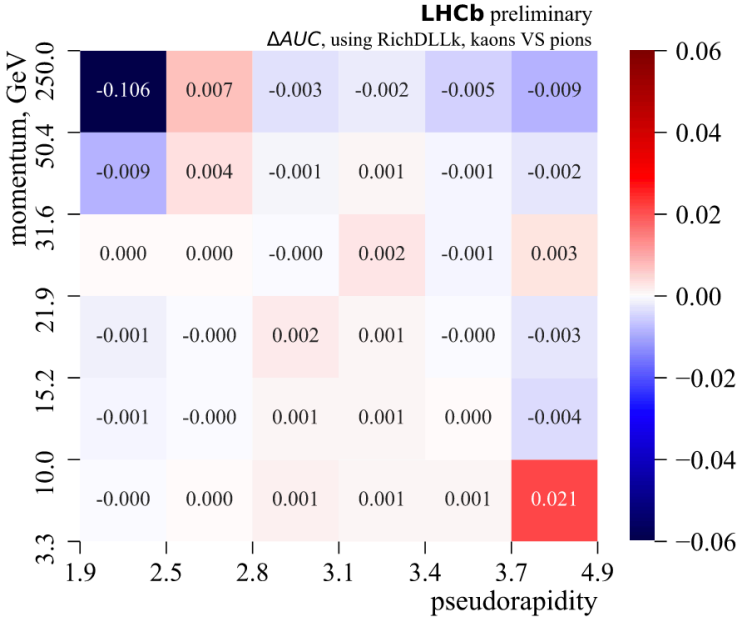
LHCb RICH fast simulation



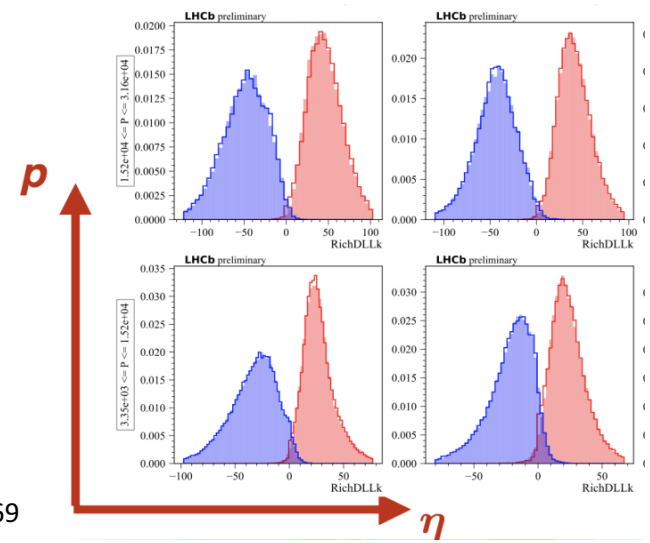
PID information encoded in log-likelihood differences (DLL) between particle type hypotheses

A fully connected layers GAN to simulate DLLs

- Input track parameters and total number of tracks
- Cramer distance to avoid gradients bias
- Train on real data using sPlot* technique to extract signal distributions



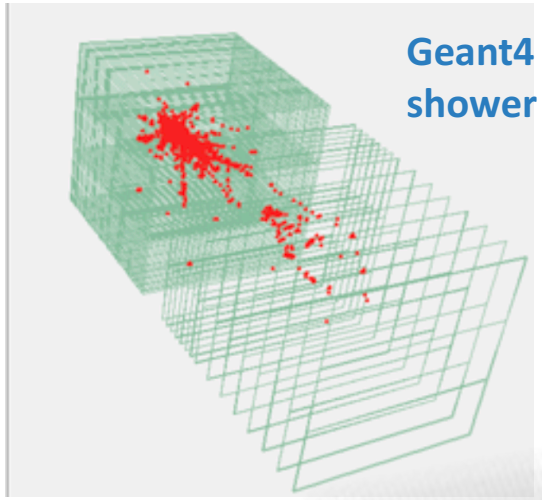
AUCs differences between few sigmas



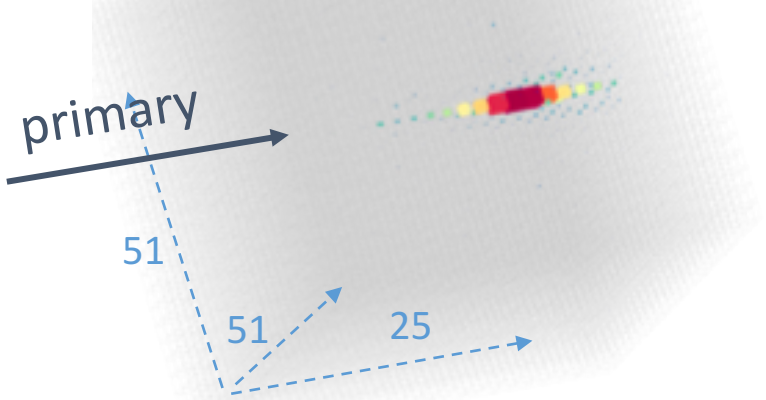
*Pivk, Muriel et al. Nucl.Instrum.Meth.A 555 (2005) 356-369

Increasing detector granularity

Open data set developed for ML applications [1]



Pixelized
3D image



CLIC[2] Electromagnetic calorimeter detector design

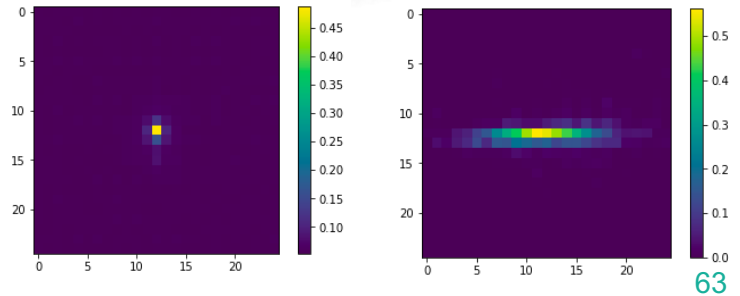
Example **next generation detector**: 5 mm×5 mm segmentation

Single particle samples (e,γ,π) with flat energy spectrum (10-500) GeV

α=+/- 30° random incident angle

Detector response as **3D images** (51x51x25 pixels)

Large dynamic range



CERN openlab

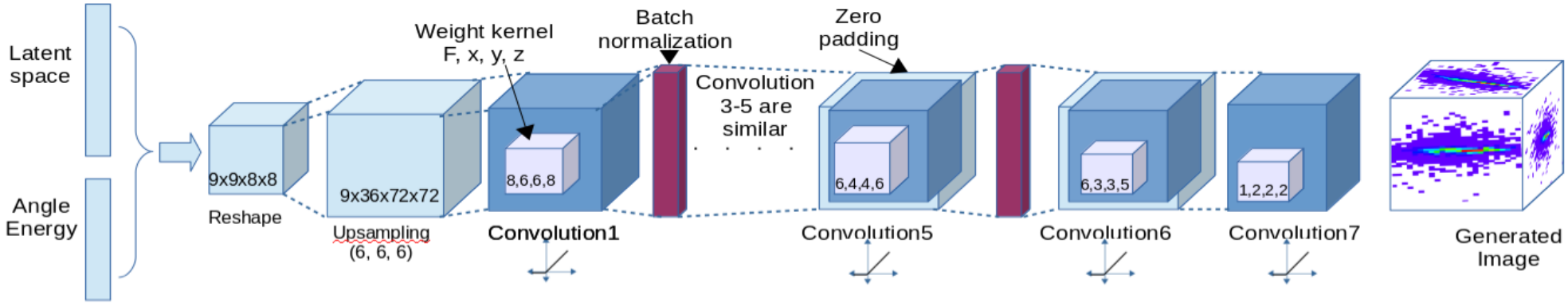
[1] M. Pierini et al. NIPS 2017 (2) http://cds.cern.ch/record/2254048# (updated April 2019)

3D convolutional GAN

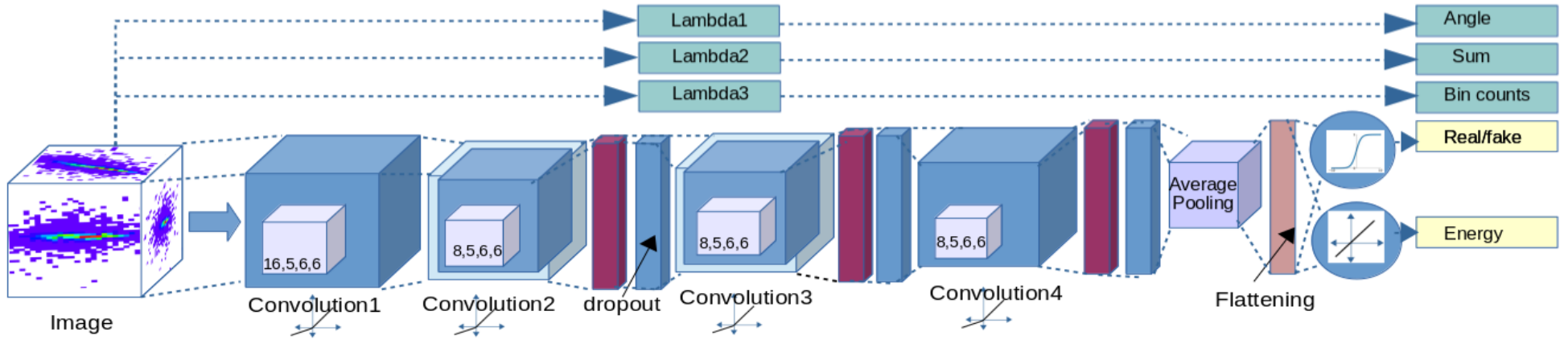
Condition training on input **particle energy** and **incident angle**, **Custom losses**

Auxiliary regression tasks assigned to the discriminator

Generator:

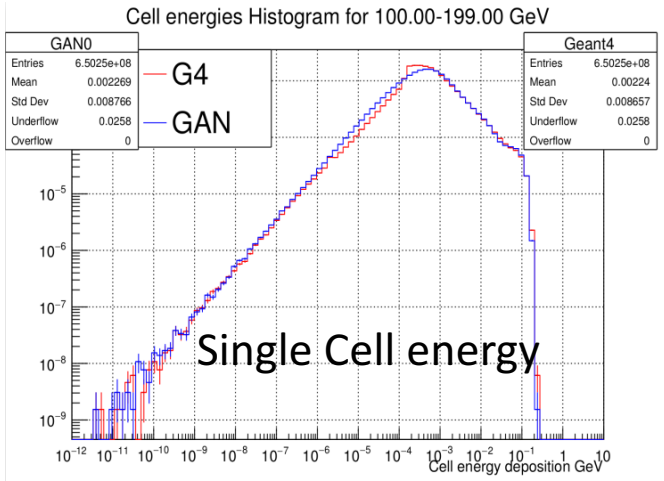


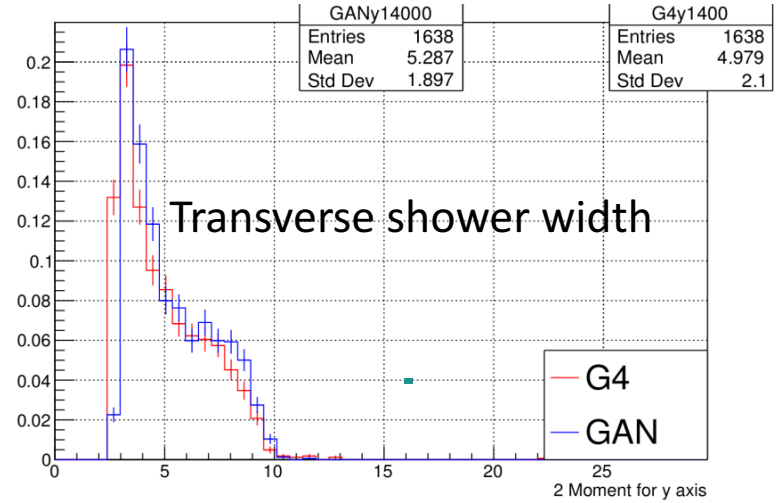
Discriminator:

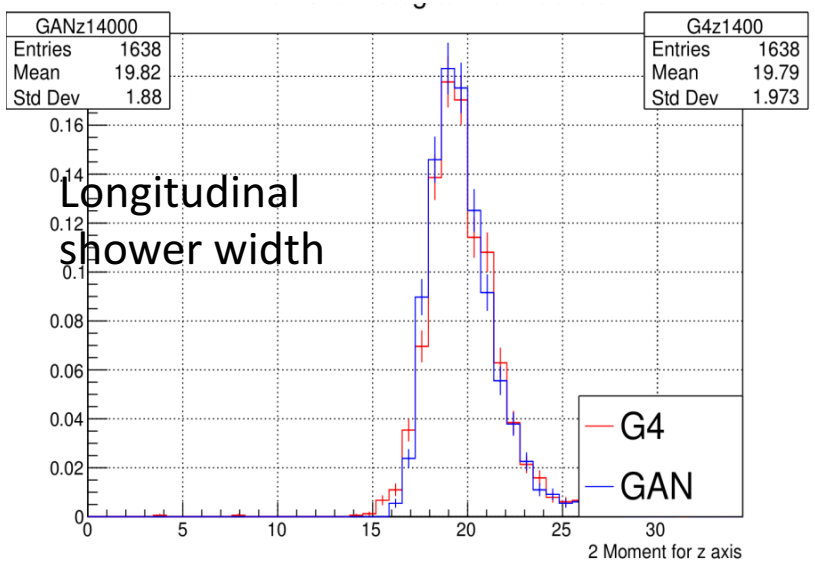


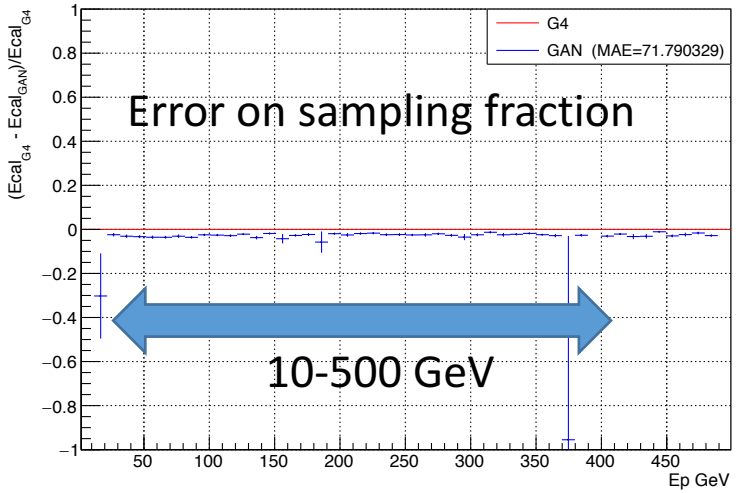
Two steps training

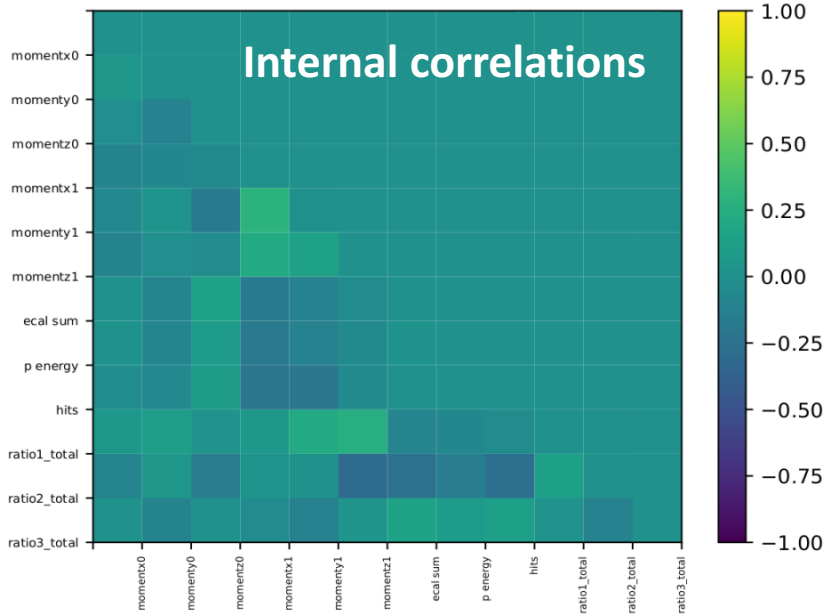
- Train on 100-200 GeV energy range
- Transfer learning to full spectrum







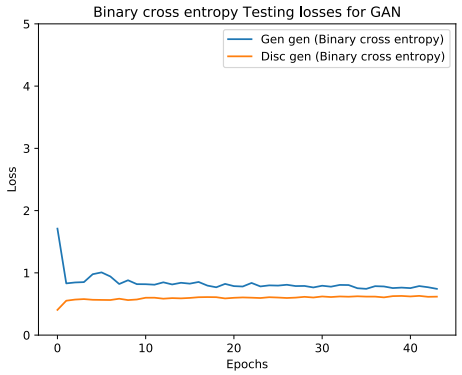


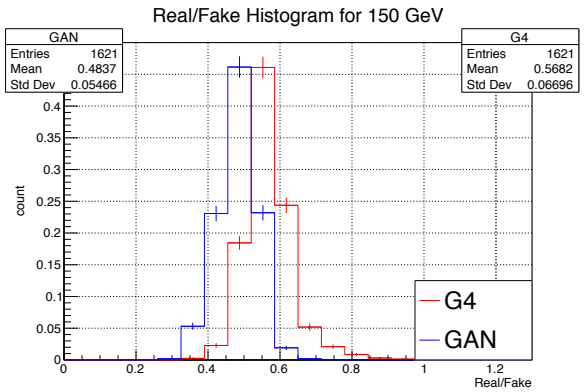


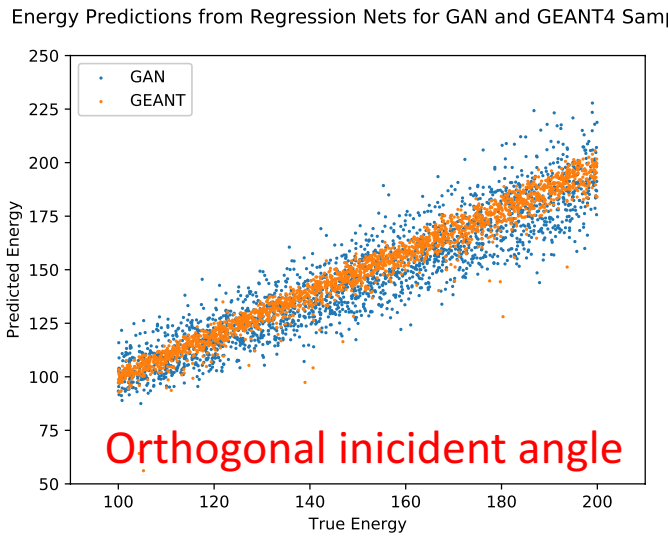
CERN openlab

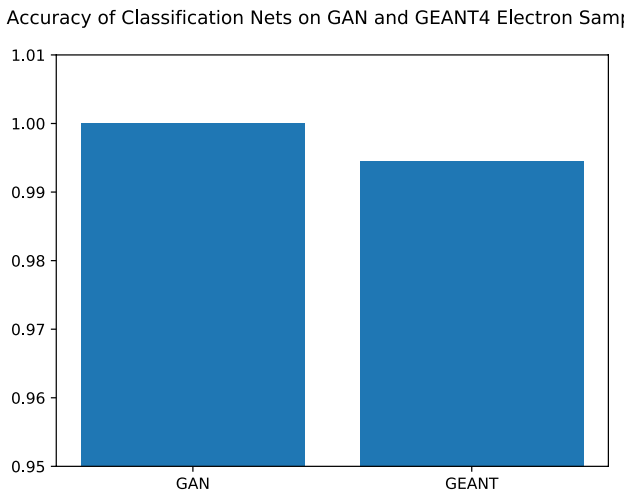
3DGAN Performance

- Convergence and discriminator performance
  - Stable test loss
  - Discriminator Real/Fake probability peaks at ~50%
  - Correct incident angle
- Comparison to Monte Carlo
  - Shower Shapes, Sampling Fractio
  - Correlations
  - Sparsity, etc..
- “In-house inception score”
  - Compare TriForce[(1)] classification and r on GAN/GEANT4
- Image Quality Analysis





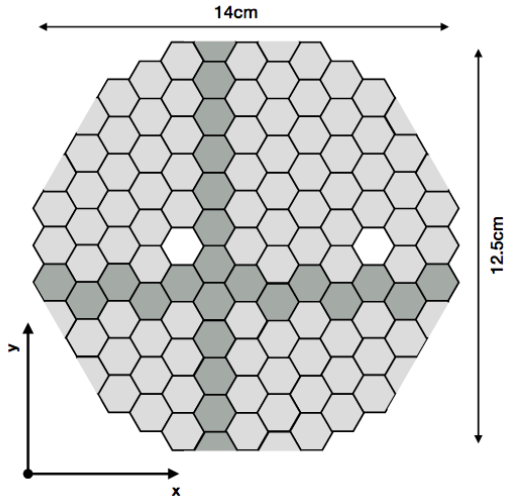




[(1)]Matt Zhang, https://github.com/BucketOfFish/Triforce_CaloML

CMS HGCAL prototype

High Granularity and Hexagonal cells prototype for CMS upgrade



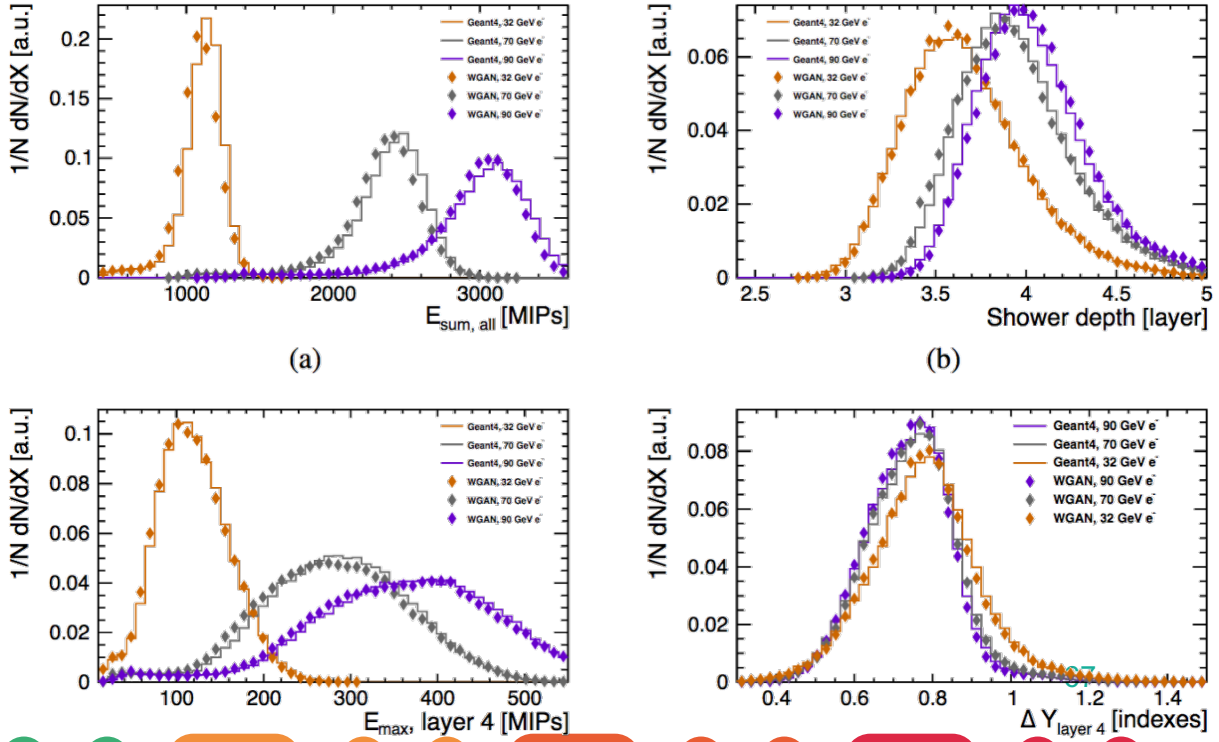
Wasserstein conditional GAN (convolutions)

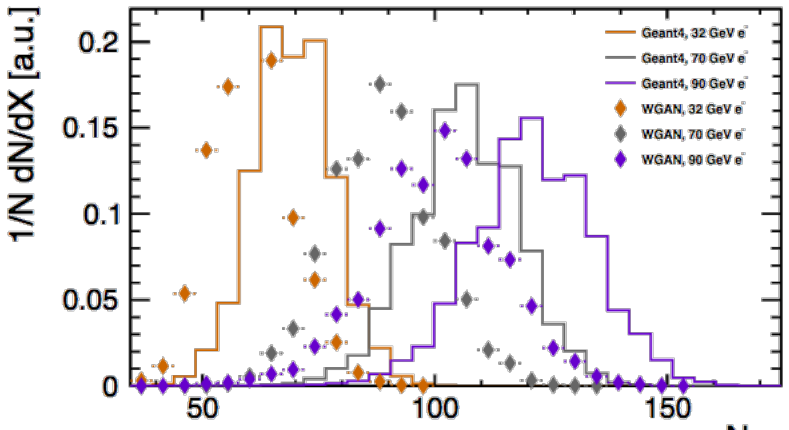
Train a generator against a critics and 2 constrainer networks reconstructing energy and impact point coordinates

Good agreement to Geant4

Some problems at low energy

arxiv.org: 1807.01954





CERN openlab

Other applications in HEP

Generative models for ALICE TPC simulation (ACAT2019)

Conditional Wasserstein GANs for fast simulation of electromagnetic showers in a CMS HGCAL prototype (IML WG 04/18)

Variational AutoEncoders to simulate ATLAS LAr calorimeter (PASC18)

Wasserstein GANs to generate high-level physics variables based on Monte Carlo ttH (superfast-simulation) (IML WG 04/18)

Particle-GAN for Full Event Simulation at the LHC (ACAT2019)

Refining Detector Simulation using Adversarial Networks (IML WG 04/18)

Model-Assisted GANs for the optimisation of simulation parameters (IML WG 04/19)

CosmoGAN

Deep networks for generating cosmology mass maps

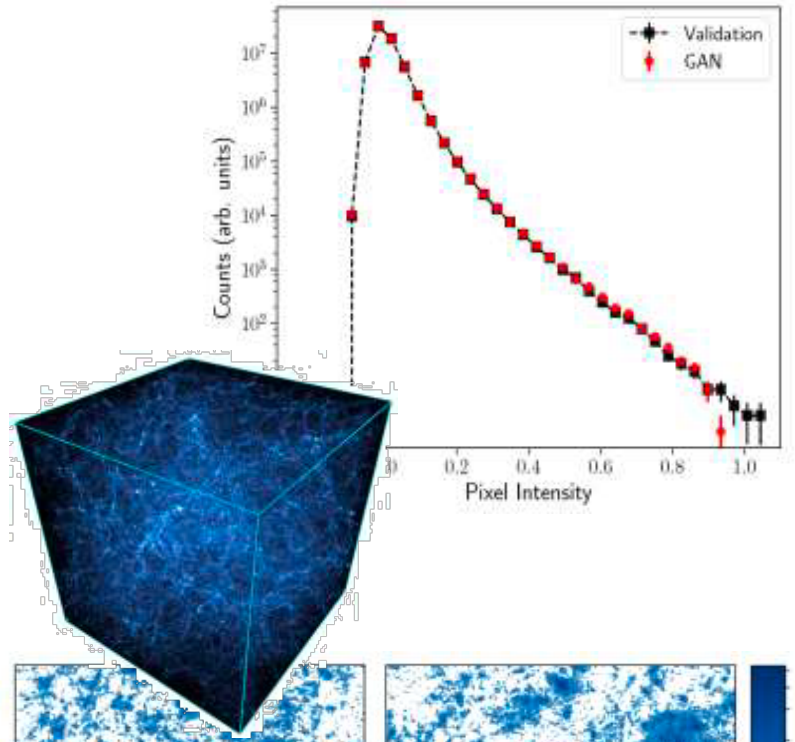
Cosmology simulations create ‘virtual universes’ (weak lensing convergence maps)

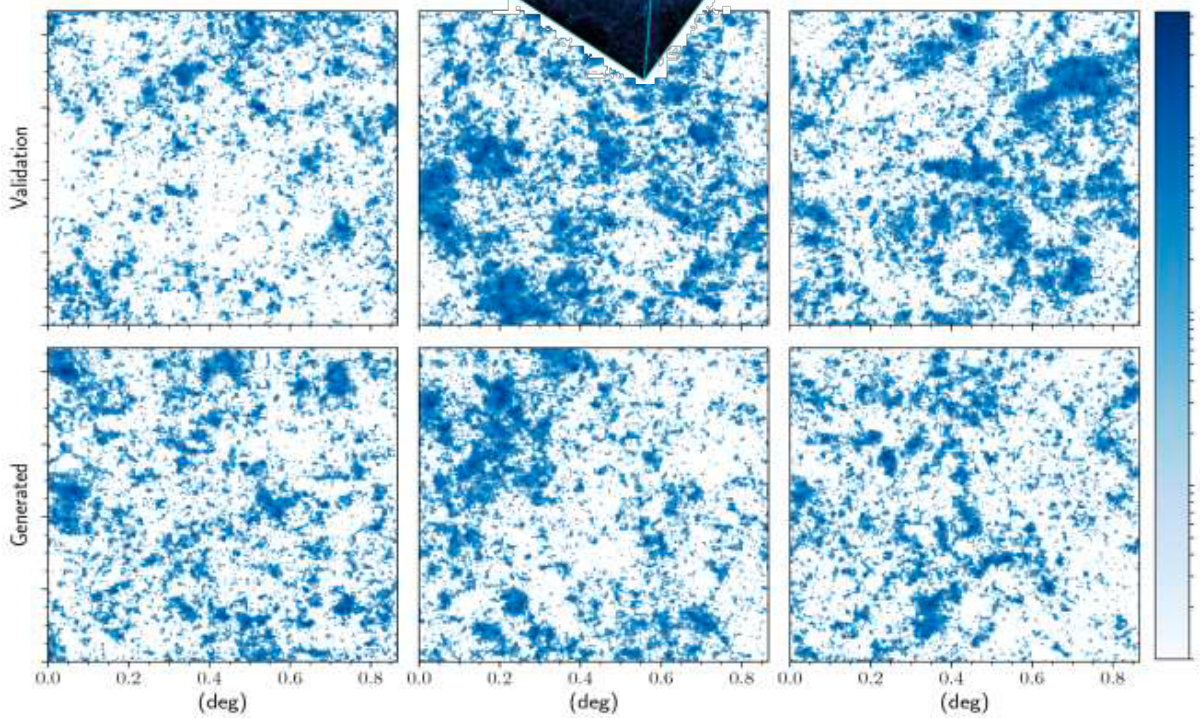
Train DCGAN on N-body simulation 3D “images”

Validate power spectrum for generated images (256x256 pixels)

K-S test p_{value}>0.995 for 246/248 moments

Also higher-order quantities (Minkowski functionals) are correctly reproduced





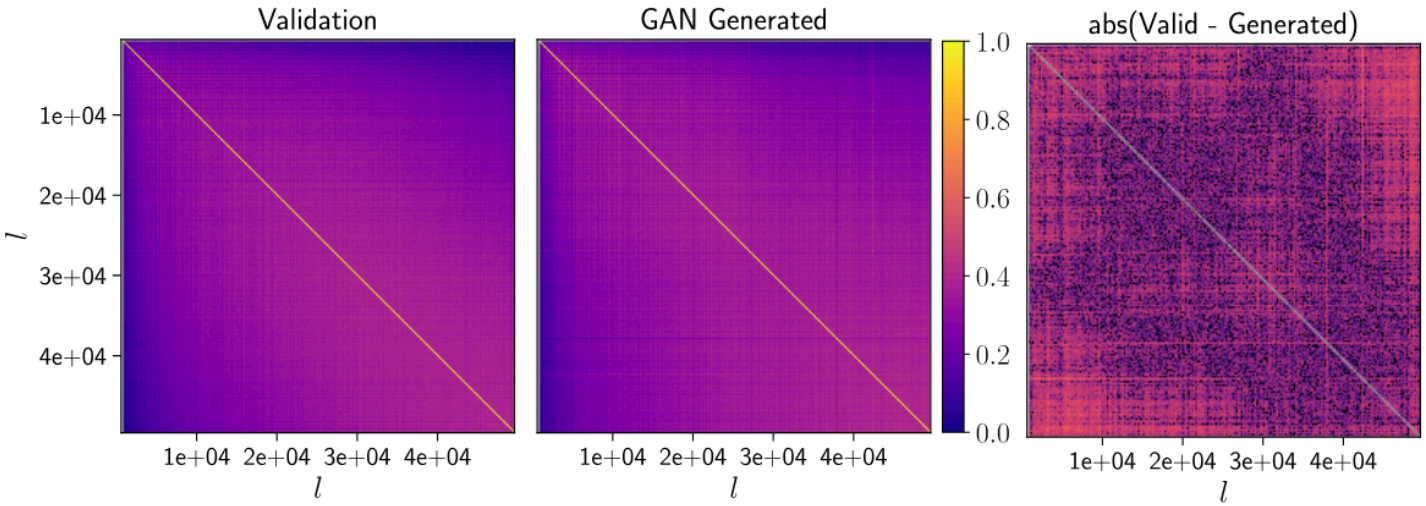
https://github.com/MustafaMustafa/cosmoGAN

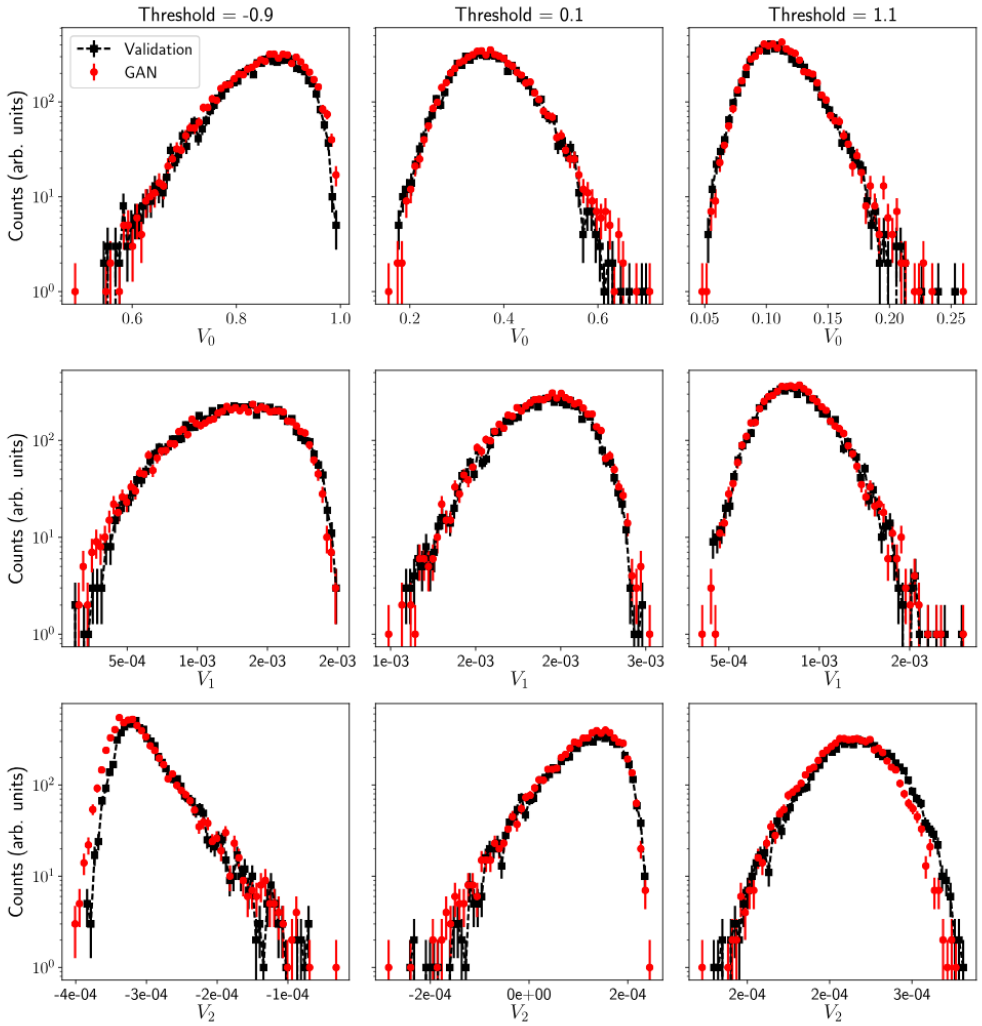
CERN openlab

CosmoGAN (II)

Training sample: 45 simulations using using the Gadget2 + Inspector Gadget (for Weak Lensing)

- 512^3 particles in a box of size of $240h^{-1}$ Mpc
- each map covers 12 square degrees, (2048×2048pixels), downsampled
- to 1024×1024 pixels.
- 200 randomly cropped maps (256×256)



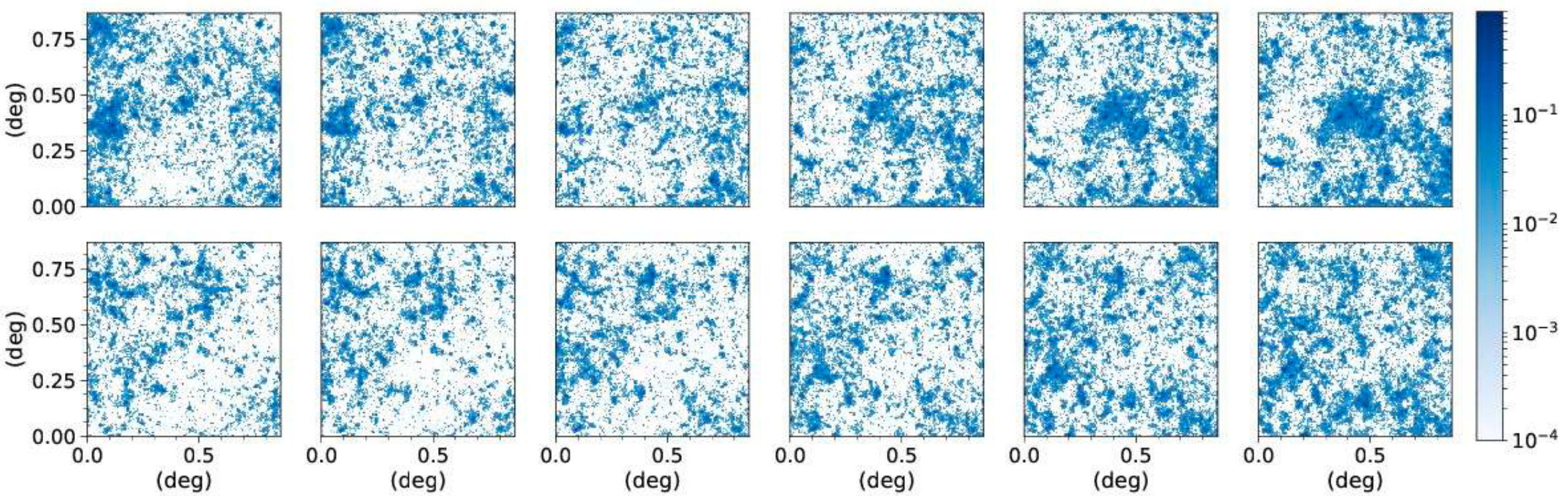


Further validation tests

Network does not copy existing training samples

Performs smooth interpolation

“Birthday Paradox” test indicates that coverage is >> 1B maps



Keras hands-on

High-level neural networks API library written in Python

Runs on top of TensorFlow, CNTK, Theano, …

User friendliness. "… an API designed for human beings, not machines…. »

Modularity. A model is a sequence (graph) of standalone, fully-configurable modules.

Easy extensibility. Add new modules as new classes and functions

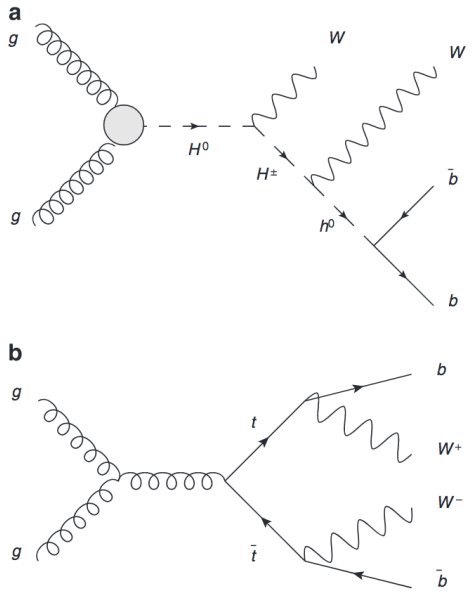
Selected a couple of exercises from a tutorial we prepared for the 6th EIROforum school last May:
https://github.com/svalleco/ml-tutorial-EIROforum

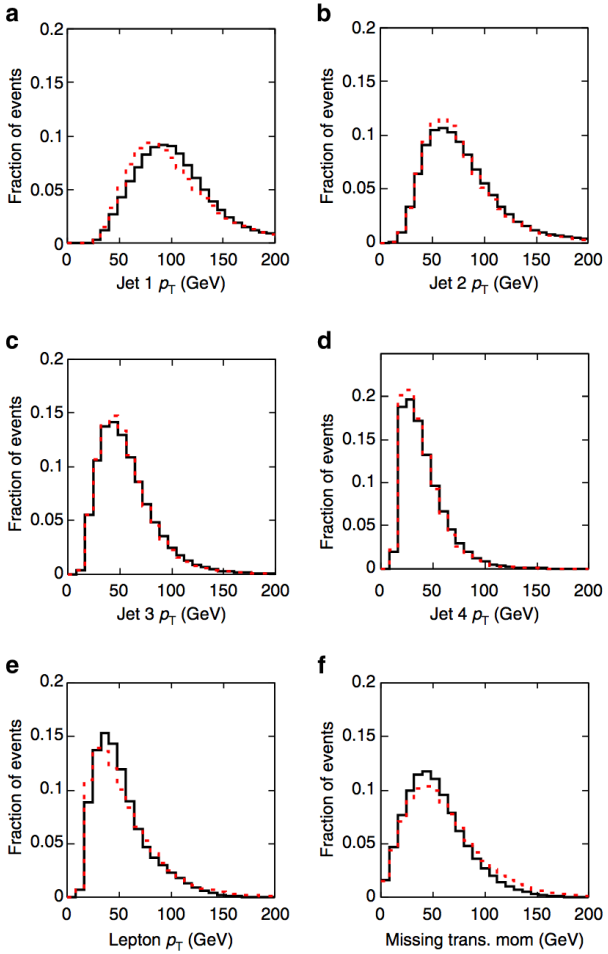
J. Martinez-Heras, L. Moneta, S.V.

Higgs Classification

Signal/Background classification task on Higgs benchmark selection:

- Compare performance of a shallow and a deep classifier
- Compare performance using low level and high level features





https://github.com/svalleco/ml-tutorial-EIROforum/blob/master/Higgs_classification_example.ipynb

Searching for Exotic Particles in High-Energy Physics with Deep Learning by Baldi et al., https://www.nature.com/articles/ncomms5308

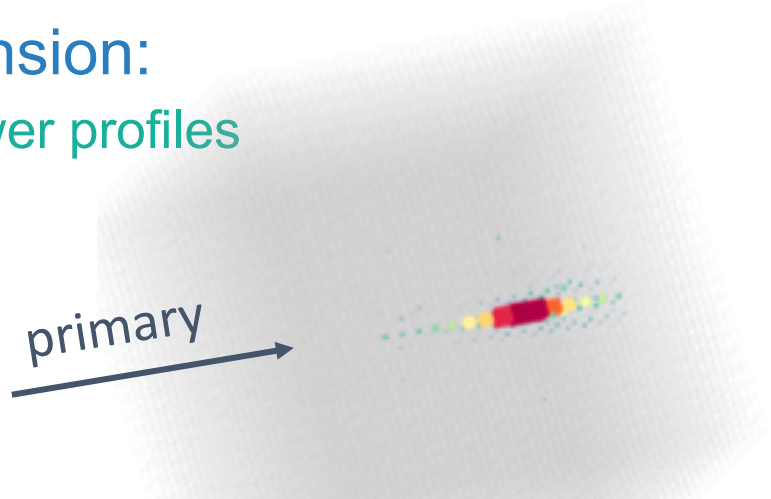
Keras.io

Keras

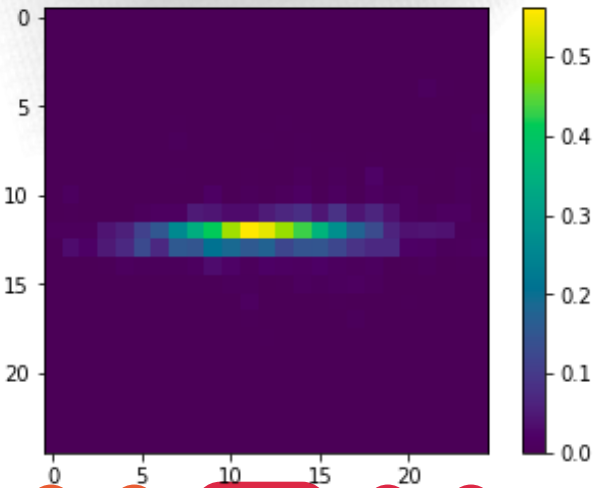
2D GAN

We simplified the 3DGAN model by eliminating one dimension:

- 2D projection in the transverse and longitudinal plane of 3D shower profiles
- Train the 2D GAN
- Generate shower profile images
- Compare to Monte Carlo shower profiles



https://github.com/svalleco/ml-tutorial-EIROforum/blob/master/Ecal2DGan.ipynb



CERN openlab

Thanks!

Questions?

References

http://cs231n.github.io/

- Pattern Recognition and Machine Learning, Bishop (2006)
- Elements of Statistical Learning (2nd Ed.) Hastie, Tibshirani & Friedman 2009
- Introduction to machine learning, Murray: http://videolectures.net/bootcamp2010_murray_iml/
- What is Machine Learning, Ravikumar and Stone: http://www.cs.utexas.edu/sites/default/files/legacy_files/research/documents/MLSSIntro.pdf
- CS181, Parkes and Rush, Harvard University: http://cs181.fas.harvard.edu
- CS229, Ng, Stanford University: http://cs229.stanford.edu/
- Machine learning in high energy physics, Alex Rogozhnikov: https://indico.cern.ch/event/497368/

http://scs.ryerson.ca/~aharley/vis

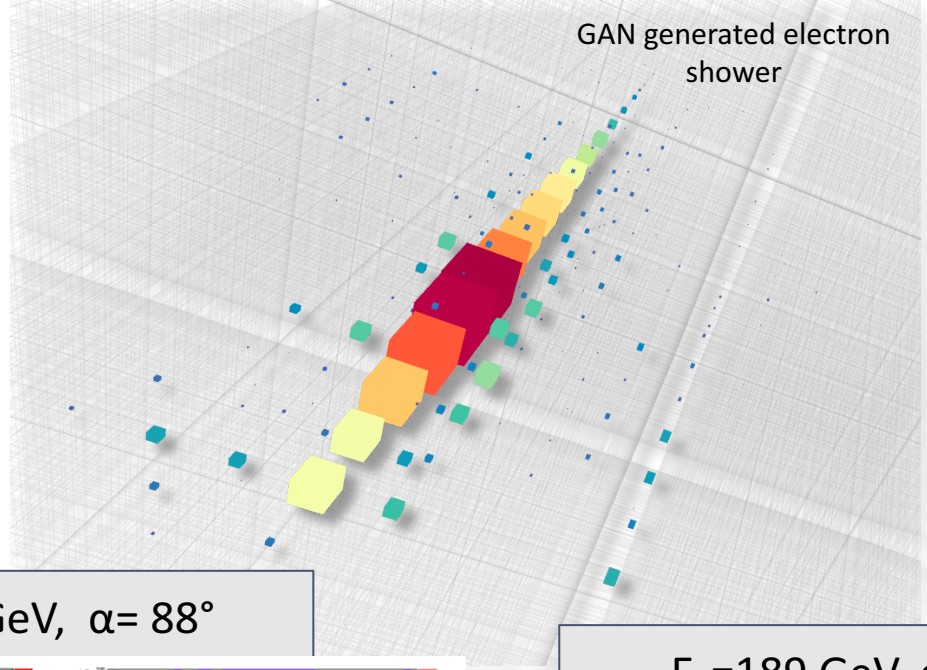
http://cs. stanford.edu/people/karpathy/convnetjs/

Keras.io

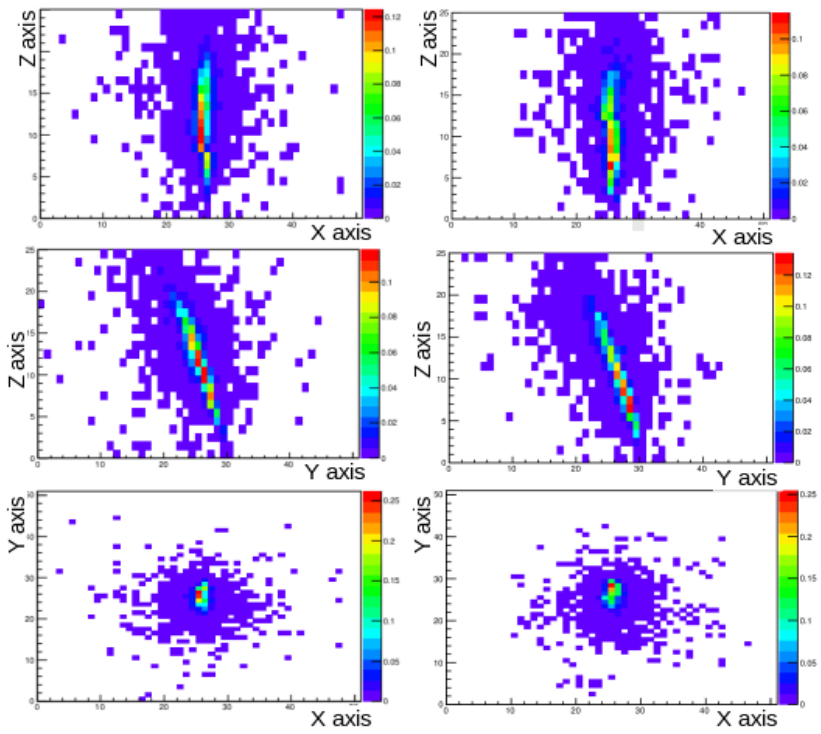
http://www.asimovinstitute.org/neural-network-zoo/

http://scikit-learn.org/

3DGAN Generated events

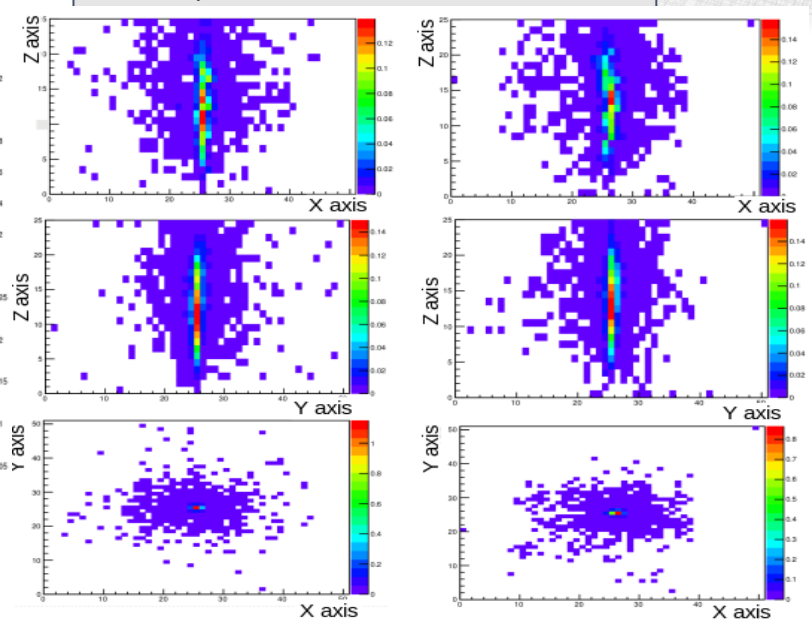


E_p=111 GeV, α=115°



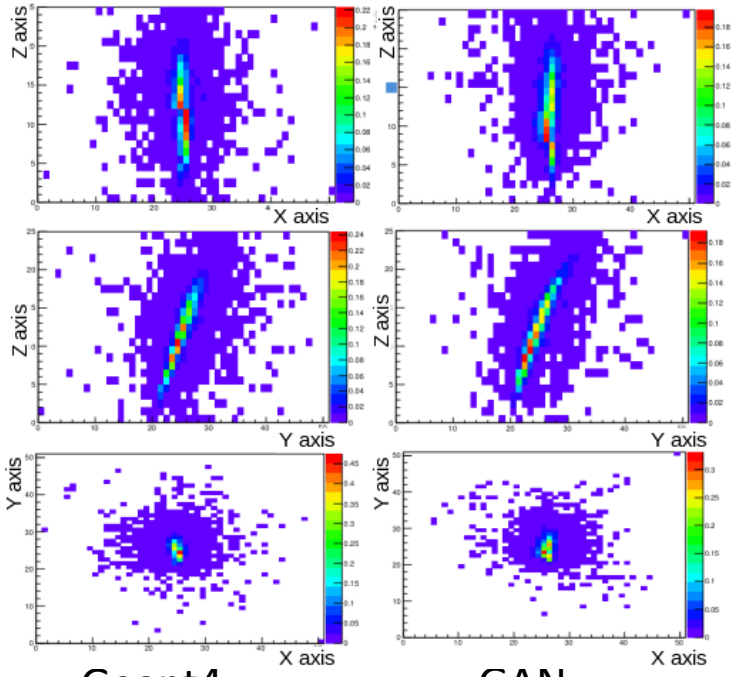
Geant4 GAN

E_p=147 GeV, α= 88°



Geant4 GAN

E_p=189 GeV, α= 63°



Geant4 GAN

CERN openlab