

Machine Learning for the LHC and future machines: applications for simulations and operation

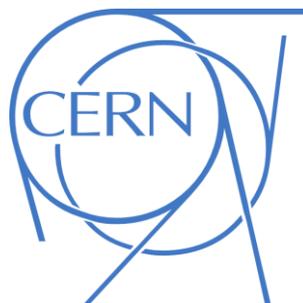
Dr Ing Gianluca Valentino

Department of Communications and Computer Engineering

University of Malta



**L-Università
ta' Malta**



*CERN Academic Training
Lecture Programme
6th May 2022*

Outline

- Why is ML useful for particle accelerators?
- Machine learning applications in the LHC
 - In simulations – anomaly detection in dynamic aperture estimates
 - In operation – improved tune measurements using a simulated dataset

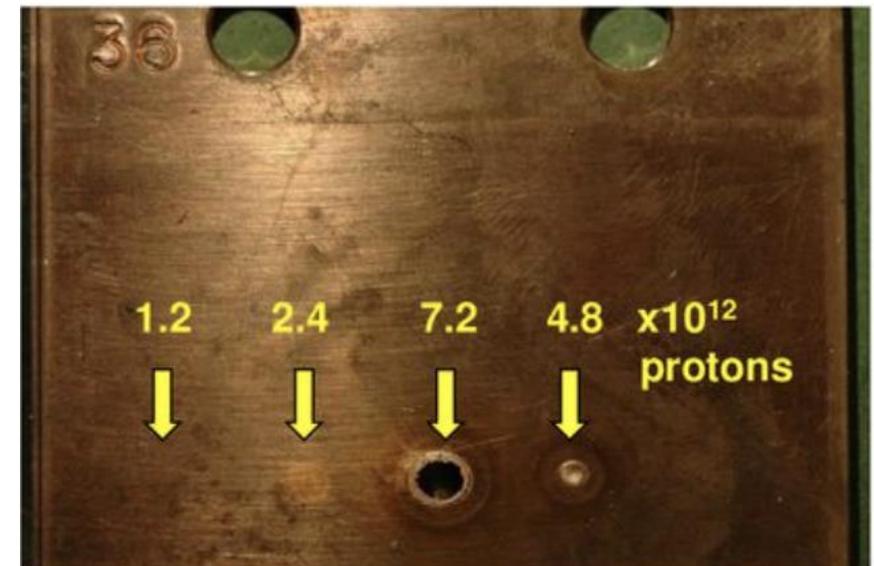
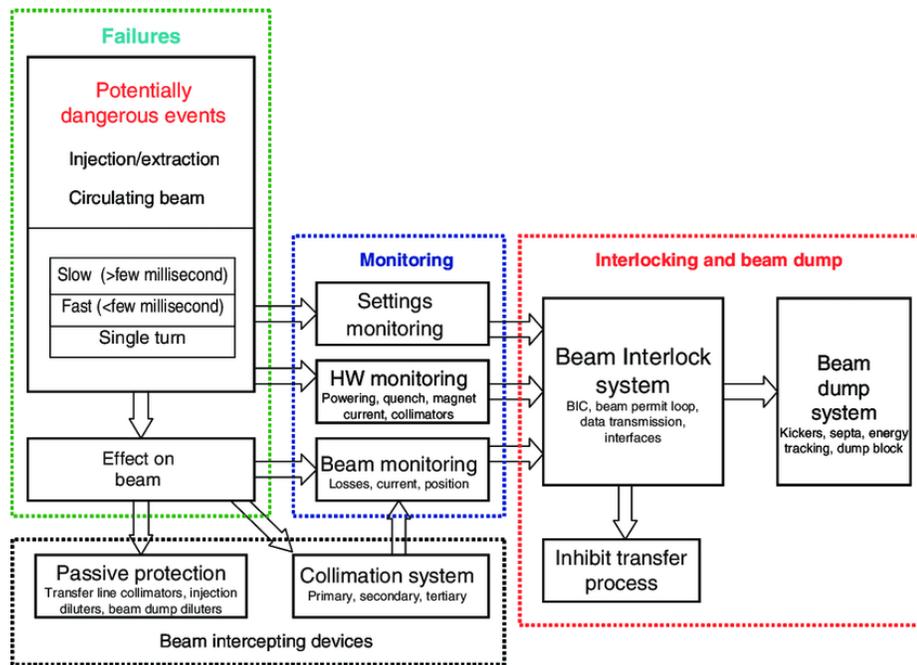
Outline

- **Why is ML useful for particle accelerators?**
- Machine learning applications in the LHC
 - In simulations – anomaly detection in dynamic aperture estimates
 - In operation – improved tune measurements using a simulated dataset

Why is ML useful for accelerators?

- **Anomaly detection and machine protection**

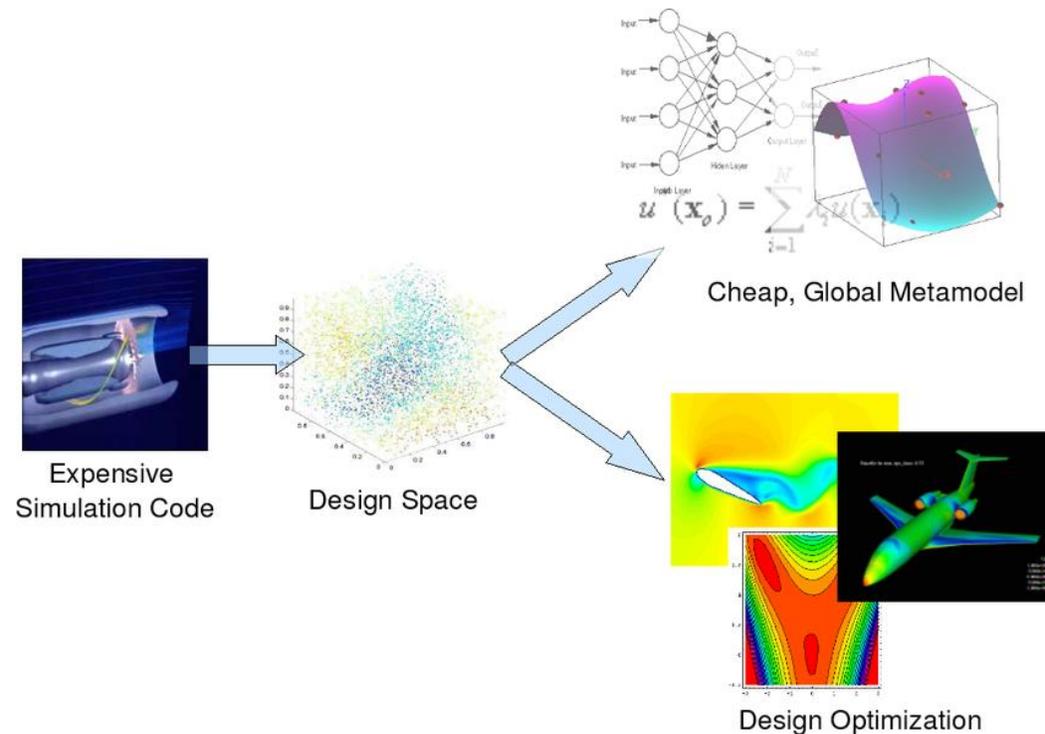
- High energy / high intensity machines are equipped with machine protection systems (MPS) which should extract the beam from the machine before catastrophic damage can occur.
- MPS are critical and therefore hardware-based.
- However, machine learning can be used to capture **operational issues** which impact beam quality, or **precursors of faults** which could lead to downtime (e.g. for machine refilling).



Why is ML useful for accelerators?

- **System modeling**

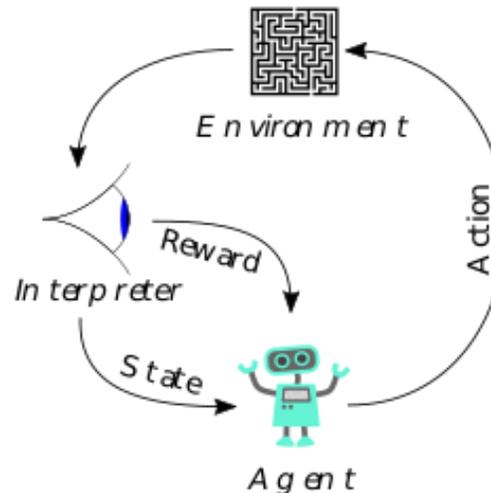
- A challenge in accelerators is to have machine models which are
 - Sufficiently accurate
 - Can execute quickly enough to be useful during operation, using real input from beam instrumentation (i.e. **online modeling**).
- Most simulation tools are too slow to be used in control systems or provide guidance to operators during machine operation.
- System modeling through ML allows to learn representations that combine information from physics-based simulations with measured data
 - aka **surrogate modeling**



Why is ML useful for accelerators?

- **Tuning and control**

- Operational settings tend to be established through theory and optimized during beam commissioning.
- These settings may need to be tuned over time due to effects of ground motion, changes in beam parameters, radiation to electronics etc.
- Accelerators may have dozens of operational modes and complex operational cycles.
- A lot of beam time is spent by operators 'reconfiguring' the machine to desired parameters.



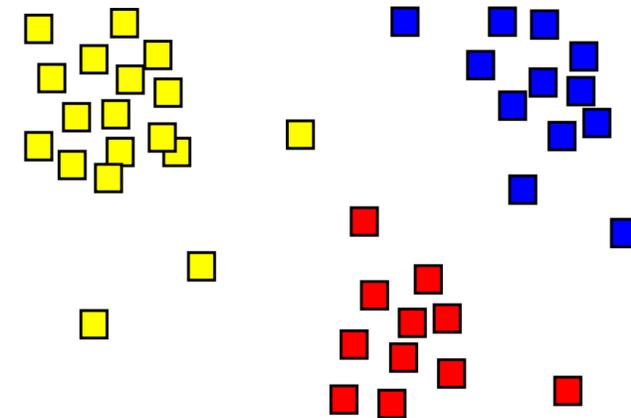
Reinforcement Learning and optimization techniques are well suited for these types of problems, and can be easily re-trained over time

Issues: sample efficiency and transfer learning

Why is ML useful for accelerators?

- **Advanced data analysis**

- Particle accelerators generate huge volumes of data
 - E.g. LHC: 1.5 million signals are logged from all the accelerator sub-systems = 2 TB/day
- Data analysis and visualization is a constant aspect of operation: to validate performance, try to diagnose and understand faults etc
- Unsupervised learning techniques could be very useful to understand hidden structures in data, e.g. understand which machine parameters contribute to beam losses
- If 'unknown' issues are uncovered, mitigation measures can be taken to improve machine performance and availability.



Outline

- Why is ML useful for particle accelerators?
- **Machine learning applications in the LHC**
 - **In simulations – anomaly detection in dynamic aperture estimates**
 - In operation – improved tune measurements using a simulated dataset

Anomaly detection

- The process of determining which points in a dataset are *different* than *most of* the others.
- We will first cover anomaly detection methods and algorithms, and then see how they were applied to dynamic aperture simulations.

Types of anomalies

- **Point anomalies**

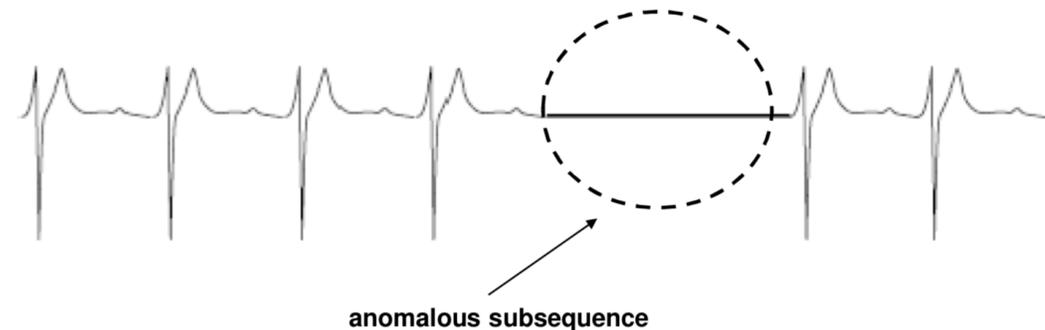
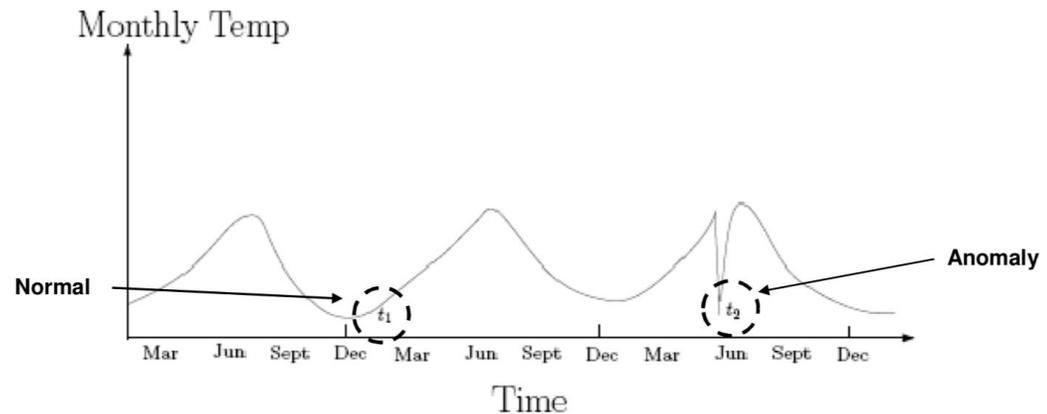
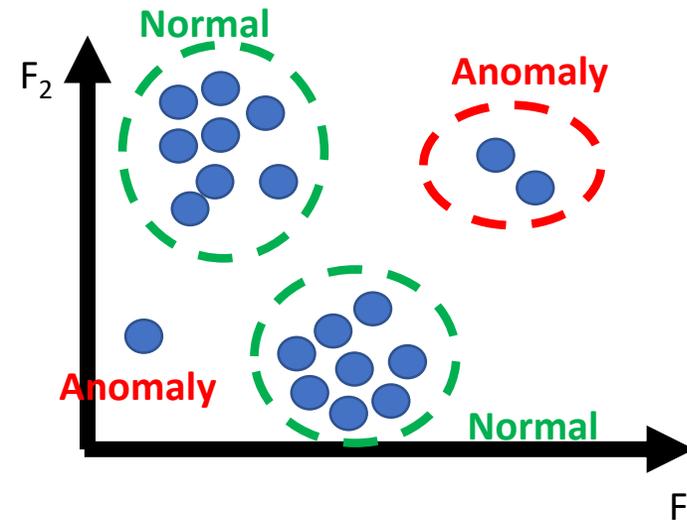
- An individual data point is anomalous with respect to the surrounding data
- The main focus for today's seminar

- **Contextual anomalies**

- An individual data instance is anomalous within a context
- Also referred to as a conditional anomaly

- **Collective anomalies**

- A collection of related data instances is anomalous
- Requires a relationship among data instances
- The individual instances within a collective anomaly are not anomalous by themselves



Proximity-based outlier detection

- Distance-based approach (kNN distance)
- Density-based approach (Local Outlier Factor)

kNN distance

- Outliers are objects **far away from other objects**
- kNN distance approach:
 - Compute an outlier score as distance to k^{th} nearest neighbor
 - Score is sensitive to choice of k

kNN distance

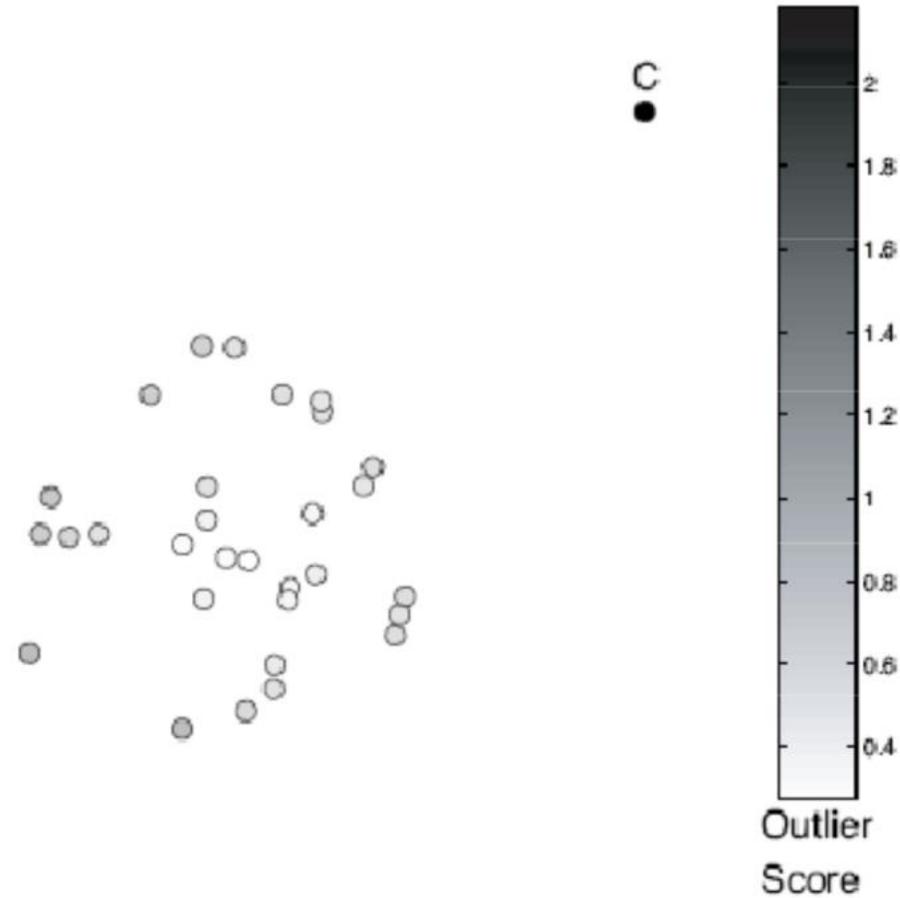


Figure 10.4. Outlier score based on the distance to fifth nearest neighbor.

kNN distance

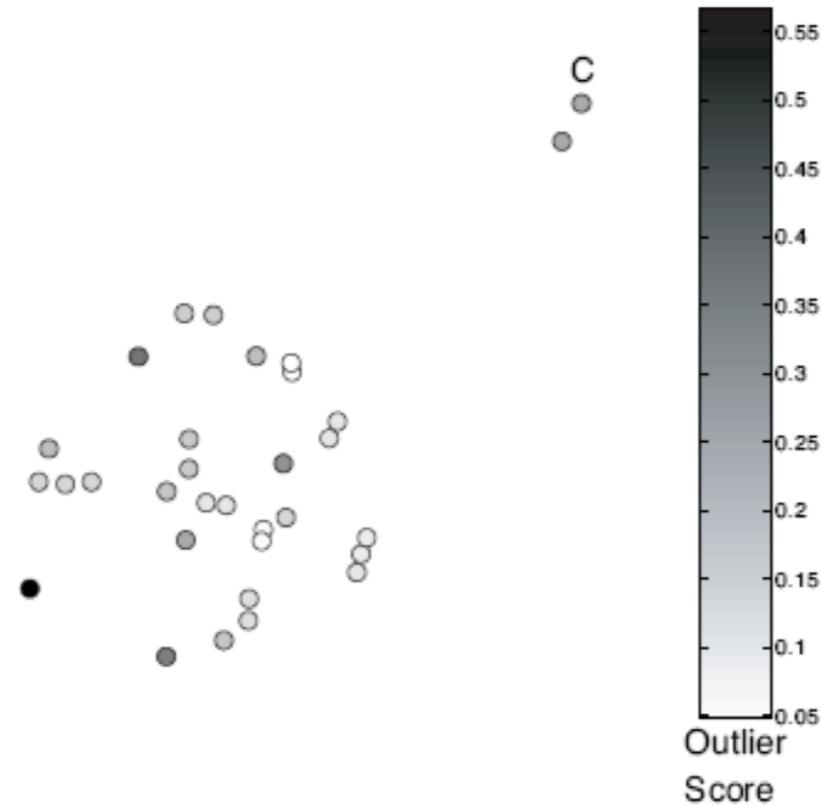


Figure 10.5. Outlier score based on the distance to the first nearest neighbor. Nearby outliers have low outlier scores.

kNN distance

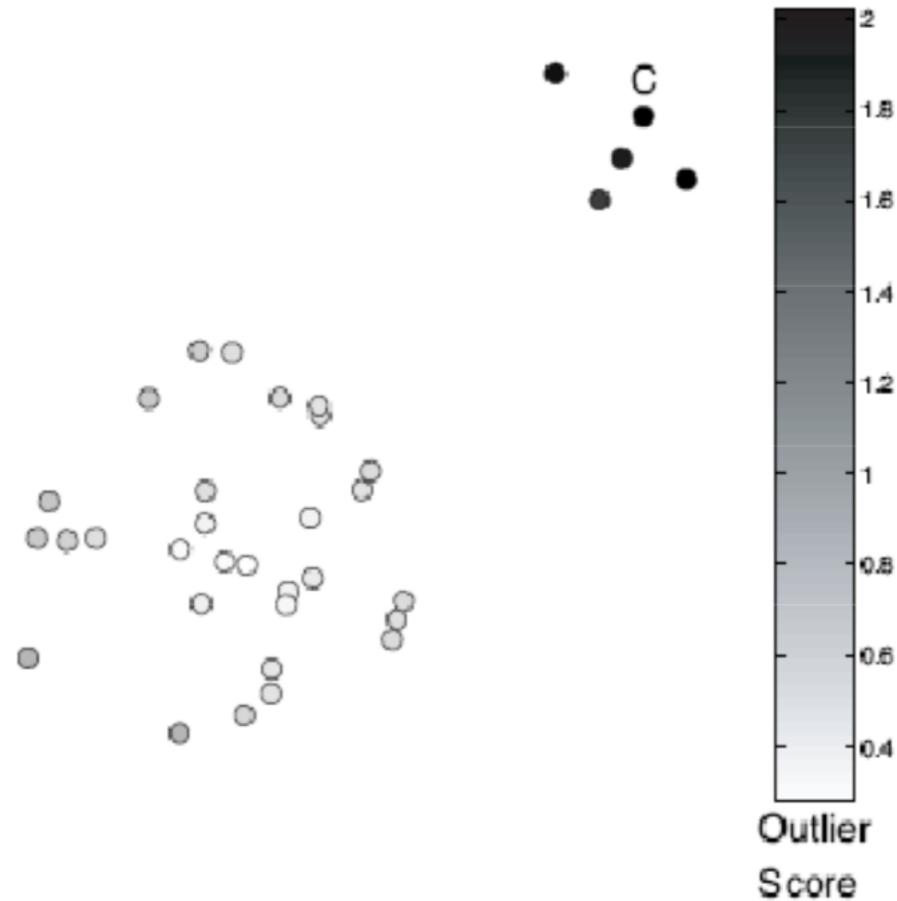


Figure 10.6. Outlier score based on distance to the fifth nearest neighbor. A small cluster becomes an outlier.

kNN distance

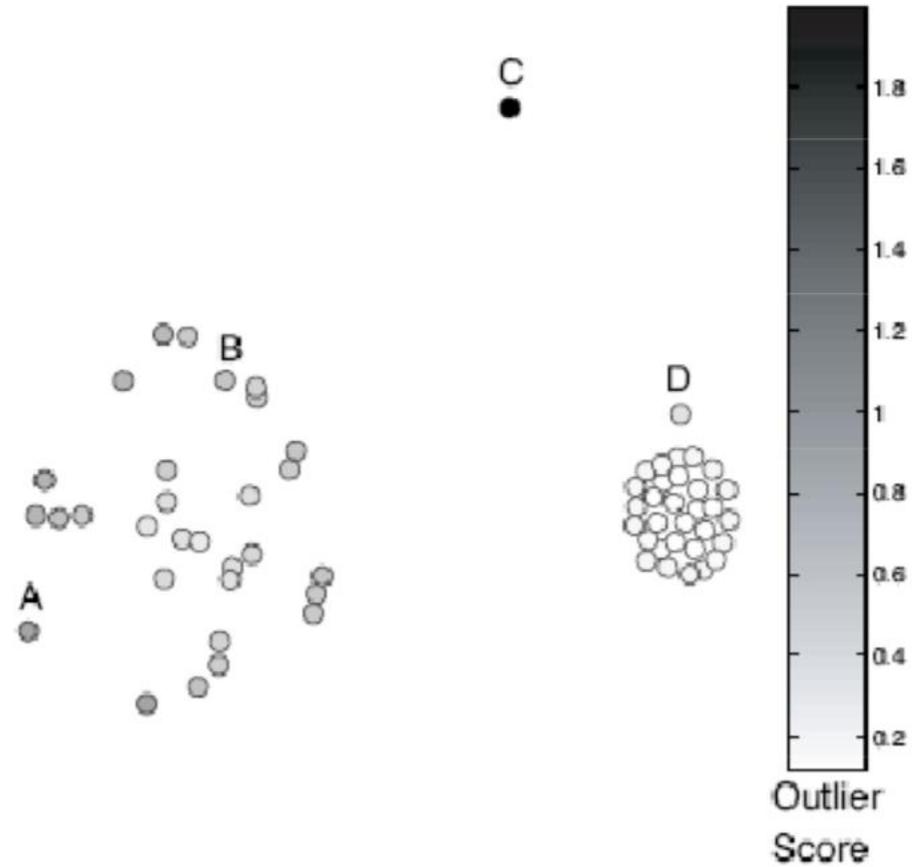


Figure 10.7. Outlier score based on the distance to the fifth nearest neighbor. Clusters of differing density.

kNN distance

- Advantages:
 - Easier to define a proximity measure for a dataset than determine its statistical distribution
 - Quantitative measure of degree to which object is an outlier (i.e. the score)
 - Deals naturally with multiple “structures” in the data (e.g. clusters)
- Disadvantages:
 - $O(n^2)$ complexity
 - Score sensitive to choice of k
 - Does not work well if data has widely variable density

Local Outlier Factor

- Outliers are objects in regions of **low density**
- Relative density outlier score
 - Reciprocal of average distance to k nearest neighbours, relative to that of those k neighbours

Local Outlier Factor

Definitions: $N_k(A)$ = set of k nearest neighbours of object A

k-distance(A) = distance of object A to kth nearest neighbour

reachability-distance_k(A, B) = max{k-distance(B), d(A, B)}

Local Reachability Distance $\text{lrd}_k(A) := 1 / \left(\frac{\sum_{B \in N_k(A)} \text{reachability-distance}_k(A, B)}{|N_k(A)|} \right)$

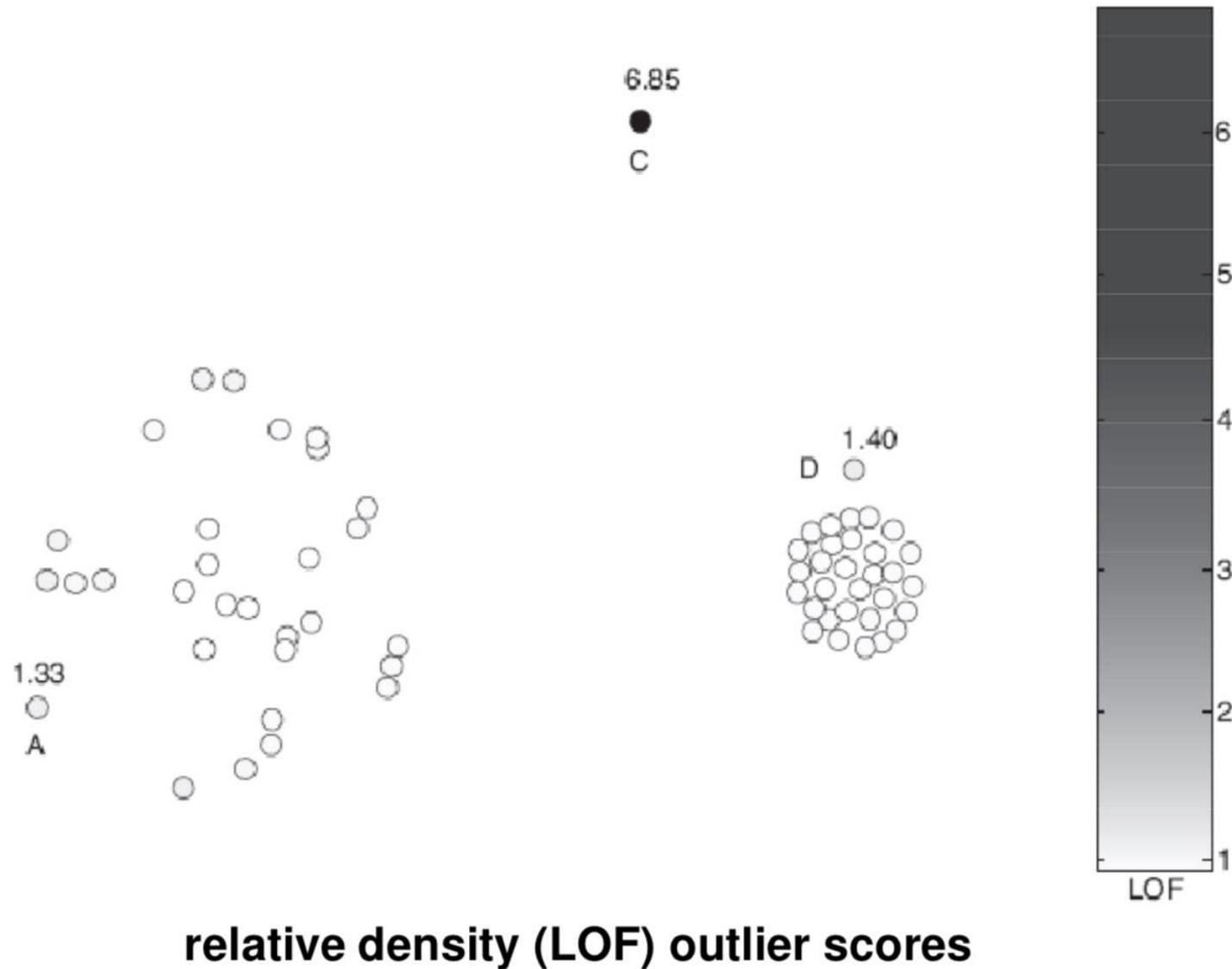
$$\text{LOF}_k(A) := \frac{\sum_{B \in N_k(A)} \frac{\text{lrd}(B)}{\text{lrd}(A)}}{|N_k(A)|} = \frac{\sum_{B \in N_k(A)} \text{lrd}(B)}{|N_k(A)|} / \text{lrd}(A)$$

LOF(A) ~ 1 => **similar density as neighbours**

LOF(A) < 1 => **higher density than neighbours (inlier)**

LOF(A) > 1 => **lower density than neighbours (outlier)**

Local Outlier Factor



Local Outlier Factor

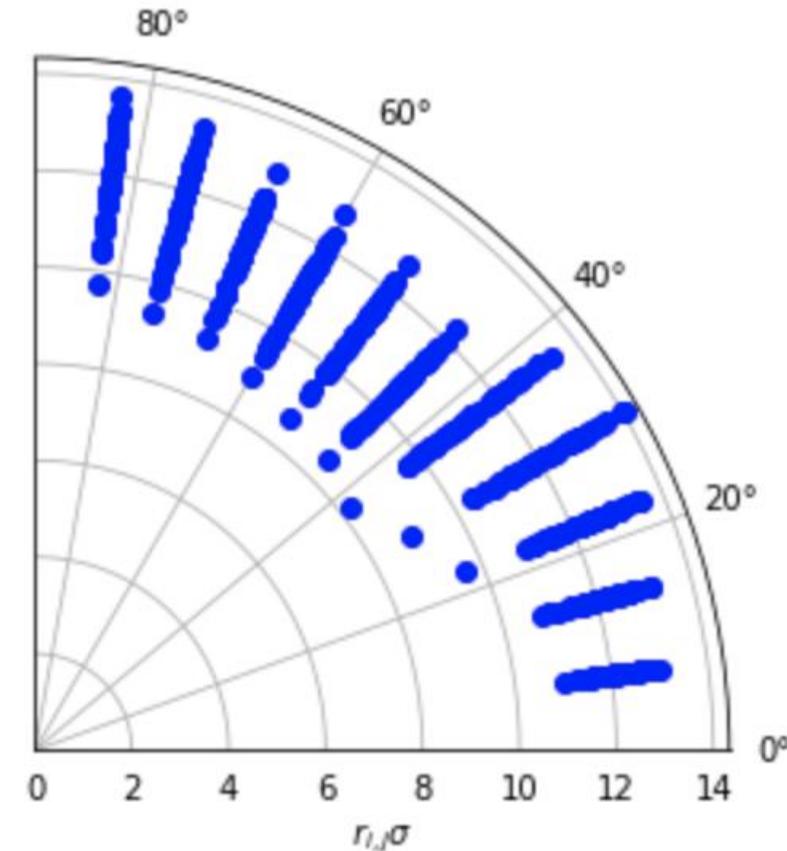
- Advantages
 - Quantitative measure of degree to which object is an outlier (score)
 - Can work well even if data has variable density
- Disadvantages
 - $O(n^2)$ complexity
 - Need to choose k

Dynamic aperture simulations

- DA: the extent of the simply-connected region of phase space in which the particle's motion remains bounded over a finite number of turns.
- Such a volume is shaped by e.g. the non-linear magnetic errors in the LHC superconducting magnets.
 - Detailed knowledge of the magnetic field errors might be difficult to obtain, e.g. due to practical difficulties in measuring the whole ensemble of superconducting magnets, or the limited precision of the magnetic measurements.
- Therefore, DA evaluation entails a Monte Carlo approach
 - Tracking simulations are used to calculate the DA for different random realisations of the machine (seeds) and over a given set of initial conditions uniformly distributed in polar co-ordinates (angles) in physical space.

DA and anomalies

- For a given angle, the stable amplitude may differ considerably from seed to seed, producing a distribution of stable amplitudes over seeds.
- Such a distribution might feature outliers, which can strongly affect the minimum (and, to a lesser extent, the mean) DA for a given machine.
- The physical reason for these outliers may be linked with the fact that the distribution of non-linear magnetic errors excites particular resonances, which is highly seed-dependent.
- It is also clear that outliers possibly represent unlikely configurations, which could be removed from the analysis of the numerical data in view of the computation of the minimum DA.



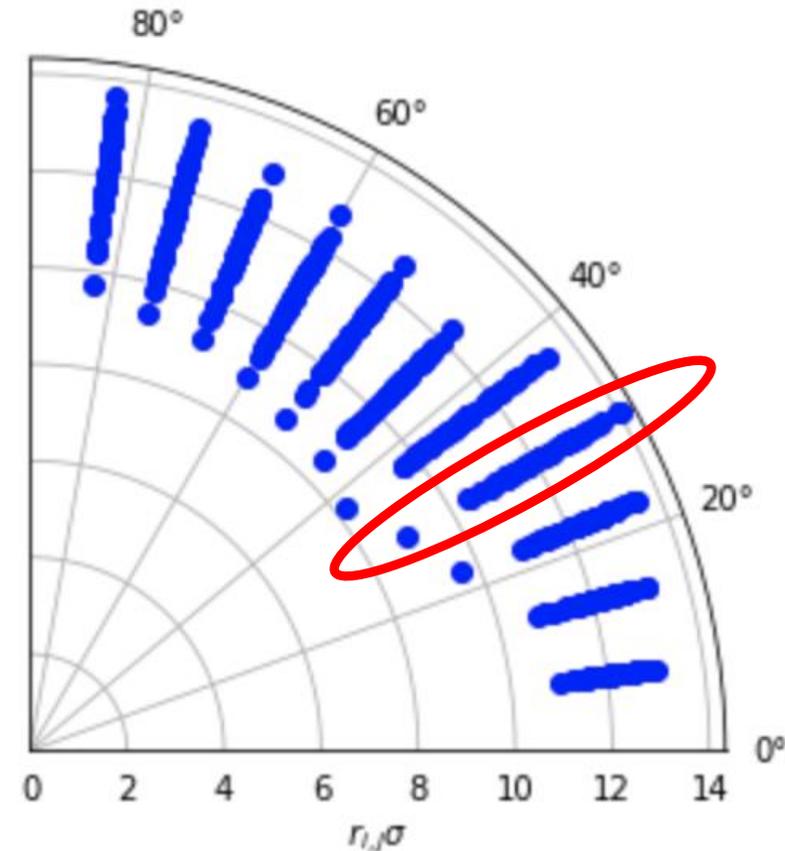
Result of a DA simulation with 60 seeds per angle

Anomaly detection for DA outliers

- Dataset of DA simulations with the following categories:
 - Accelerator (LHC or HL-LHC)
 - Beam energy (450 GeV or 7 TeV)
 - Circulating beam (B1 rotating clockwise or B2 rotating counterclockwise)
 - Optical configuration of the accelerator:
 - LHC: nominal or ATS
 - HL-LHC: V1.0, V1.3, V1.4
 - Strength of octupole magnets used to stabilise the beams against collective effects.
- Two approaches were attempted:
 - **Supervised learning** using Support Vector Machines trained on a (subset) labelled dataset (~300 studies = DA points for ~140,000 seeds and angles)
 - **Unsupervised learning** using DBSCAN & LOF on the full dataset (~5100 studies = DA points for ~3.4E6 seeds and angles)

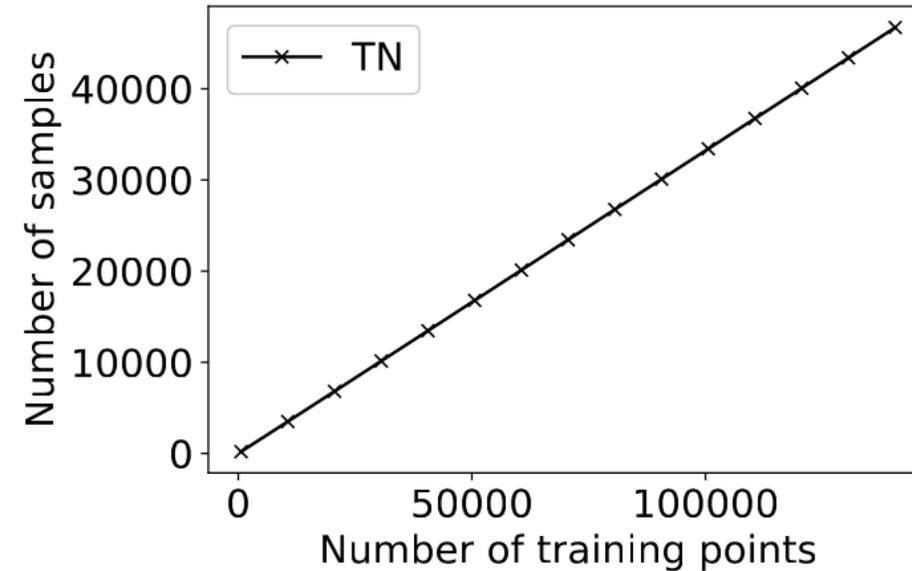
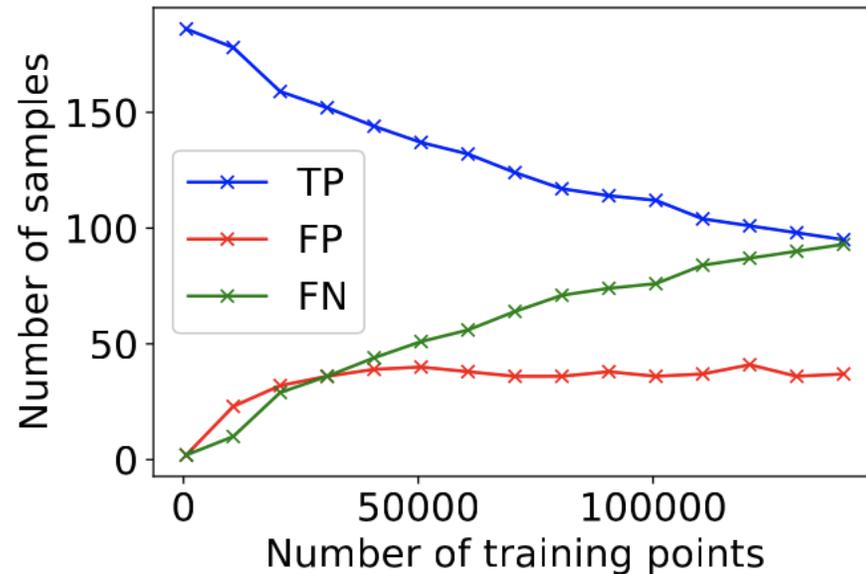
Feature extraction and scaling

- Constraint: outliers can exist only as minima or maxima (not in between).
- The 60 DA points (corresponding to the 60 seeds) for each angle are scaled separately to have zero mean and unit variance.
- The vector of DA points with the corresponding binary labels (0 = normal, 1 = anomaly) are then used to train the SVM.



Results with supervised learning

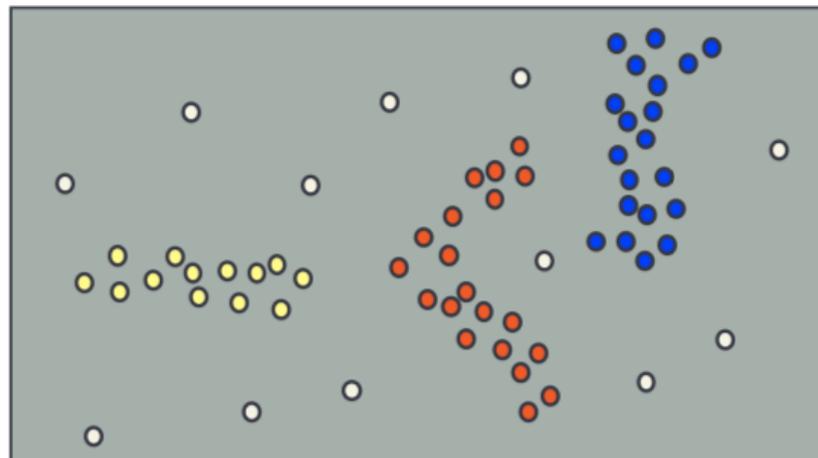
Learning curves for the trained SVM model:



- TP = True Positives (ground-truth = anomaly, prediction = anomaly), FP = False Positives, TN = True Negatives (ground-truth = normal, prediction = normal), FN = False Negatives
- The results show that when the training data set is close to being balanced between anomalous and normal points, the number of TP is quite high, while the FP and FN are low.
- However, as the data set becomes skewed towards normal points, the model achieves a lower performance.
 - This is understandable given the assumption of balance in the SVM algorithm.

DBSCAN

- DBSCAN = **D**ensity-**B**ased **S**patial **C**lustering of **A**pplications with **N**oise
 - M. Ester, H.-P. Kriegel, J. Sander, X. Xu, Proc. KDD, 1996.
- Relies on a density-based notion of cluster; #clusters not specified in advance like k-means.
- Basic idea:
 - Group together points in high-density
 - Mark as outliers the points which lie alone in low-density regions.



DBSCAN

- Local point density at a point p defined by two parameters:

1. ϵ -> radius for the neighbourhood of point p

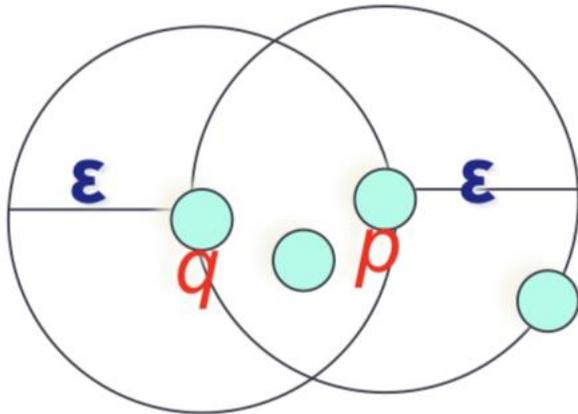
ϵ -Neighbourhood: all points within a radius of ϵ from the point p such that

$$N_{\epsilon}(p) = \{q \text{ in dataset } D \mid \text{dist}(p, q) \leq \epsilon\}$$

2. MinPts -> minimum number of points in the given neighbourhood $N(p)$

DBSCAN

- How do we define high-density?
 - ϵ -Neighbourhood of a point contains at least MinPts



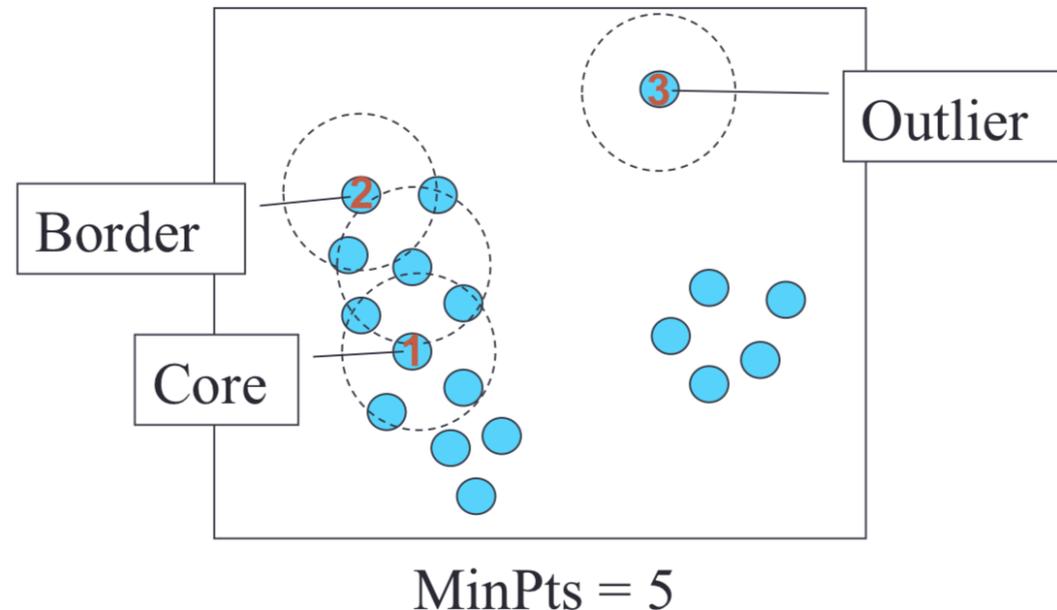
ϵ -Neighbourhood of $p = 4$ points
 ϵ -Neighbourhood of $q = 3$ points

Q. When MinPts = 4?

Density of p is “high”
Density of q is “low”

DBSCAN

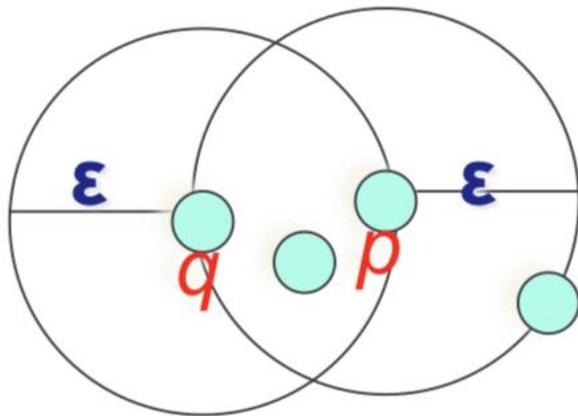
- A point can fall into one of three categories: core, border and outlier
 - Core point: if its density is **high**
 - Border point: density is low (but in the **neighbourhood of a core point**)
 - Noise point: any point that is not a core point nor a border point



DBSCAN

- Directly Density-reachable

- A point q is **directly density-reachable** from a point p if p is a **core point** and q is in p 's ϵ -Neighbourhood.



MinPts = 4

Q. Is p directly density-reachable from q ?

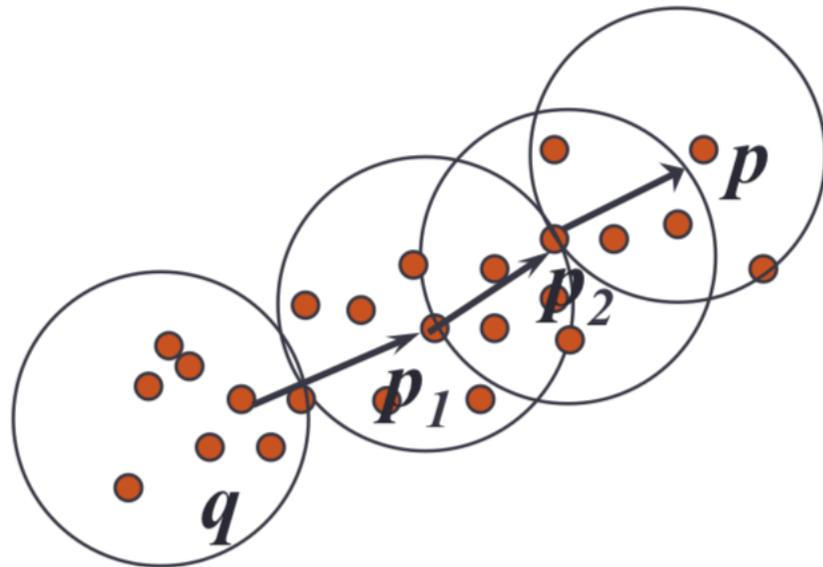
A. No: as q is not a core point.

Density-reachability is **asymmetric**.

DBSCAN

- Density-reachable

- A point p is **density-reachable** from a point q if there is a chain of points p_1, p_2, \dots, p_n with $p_1 = q, p_n = p$ such that p_{i+1} is directly density-reachable from p_i



MinPts = 7

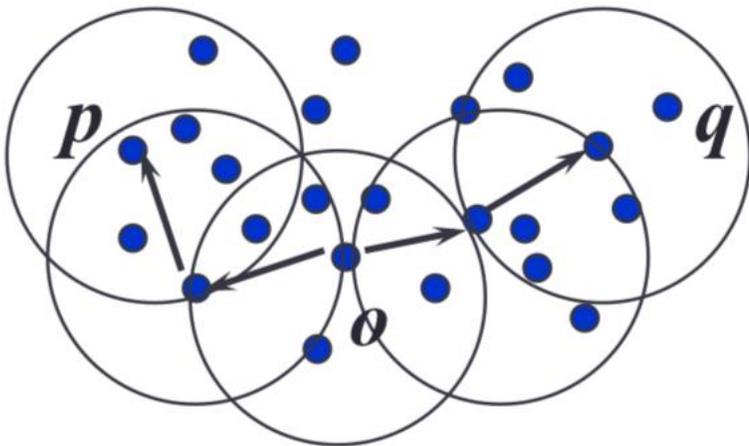
- p_1 is directly density-reachable from q
- p_2 is directly density-reachable from p_1
- p is directly density-reachable from p_2
- There is a chain from q to p ($q \rightarrow p_1 \rightarrow p_2 \rightarrow p$)

Q. q is density-reachable from p ?

A. No: as p is not a core point itself.

DBSCAN

- Density-connectivity:
 - A pair of points p and q are density-connected if they are commonly density-reachable from a point o



MinPts = 7

- Density-connectivity is **symmetric**

DBSCAN

- Formal description of cluster:
 - Given a dataset D , and parameters ϵ and MinPts , a cluster C is a subset of D if the these two criteria are satisfied:
 - **Maximality:** $\forall p, q$ if $p \in C$ and if q is **density-reachable** from p , then also $q \in C$
 - **Connectivity:** $\forall p, q \in C$, p and q are **density-connected**
 - Note: cluster contains core points as well as border points

DBSCAN

- Clustering algorithm:

$\forall p \in D$ **do**

if p is not yet classified **then**

if p is a core-point **then**

collect all points density-reachable from p and assign them to a new cluster

else

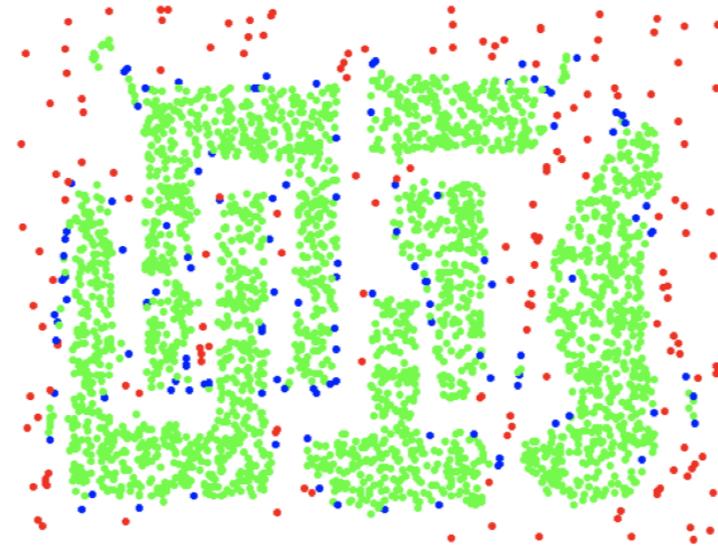
assign p to Noise (i.e. outlier)

DBSCAN

- Example:



Original Points

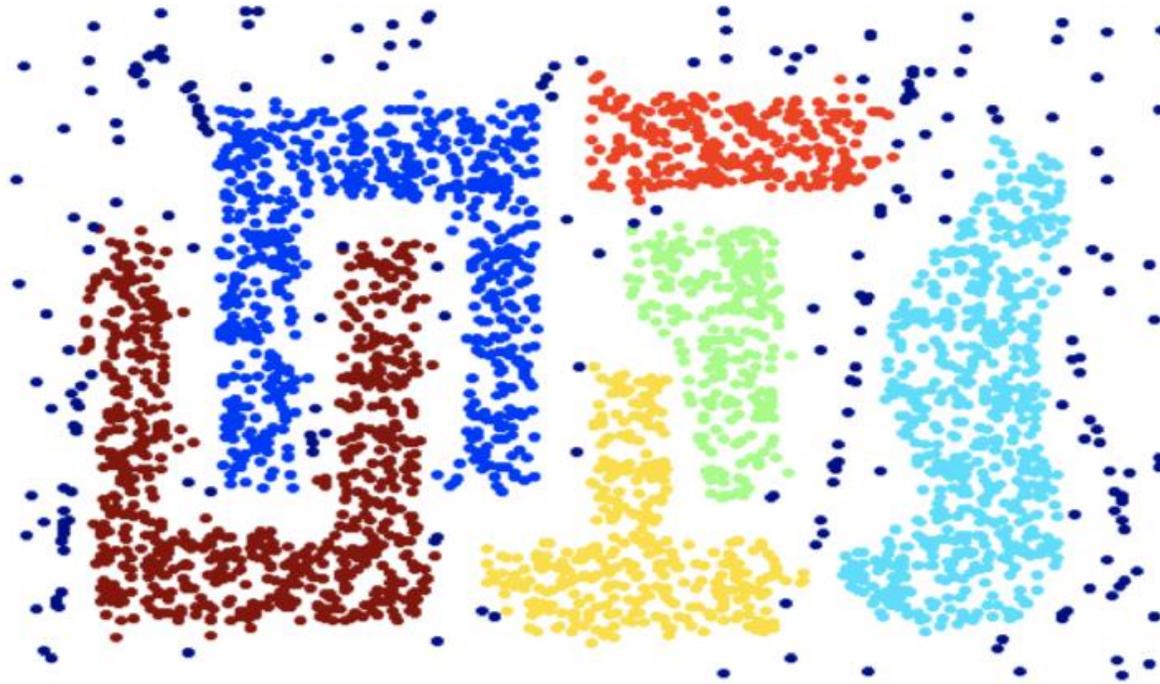


Point types: **core**,
border and **outliers**

$\epsilon = 10$, MinPts = 4

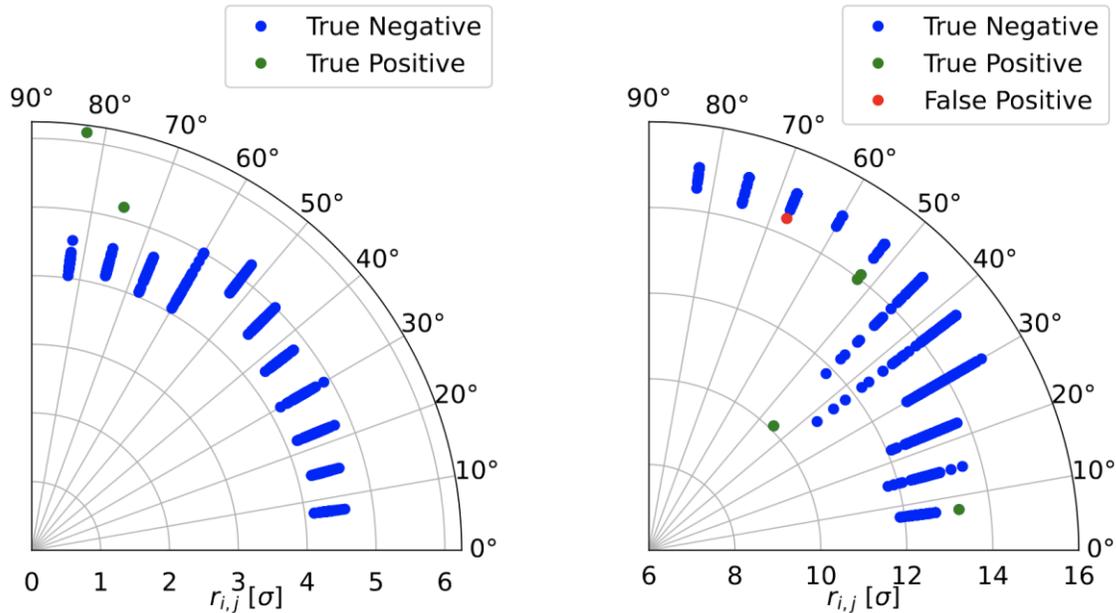
DBSCAN

- Example:

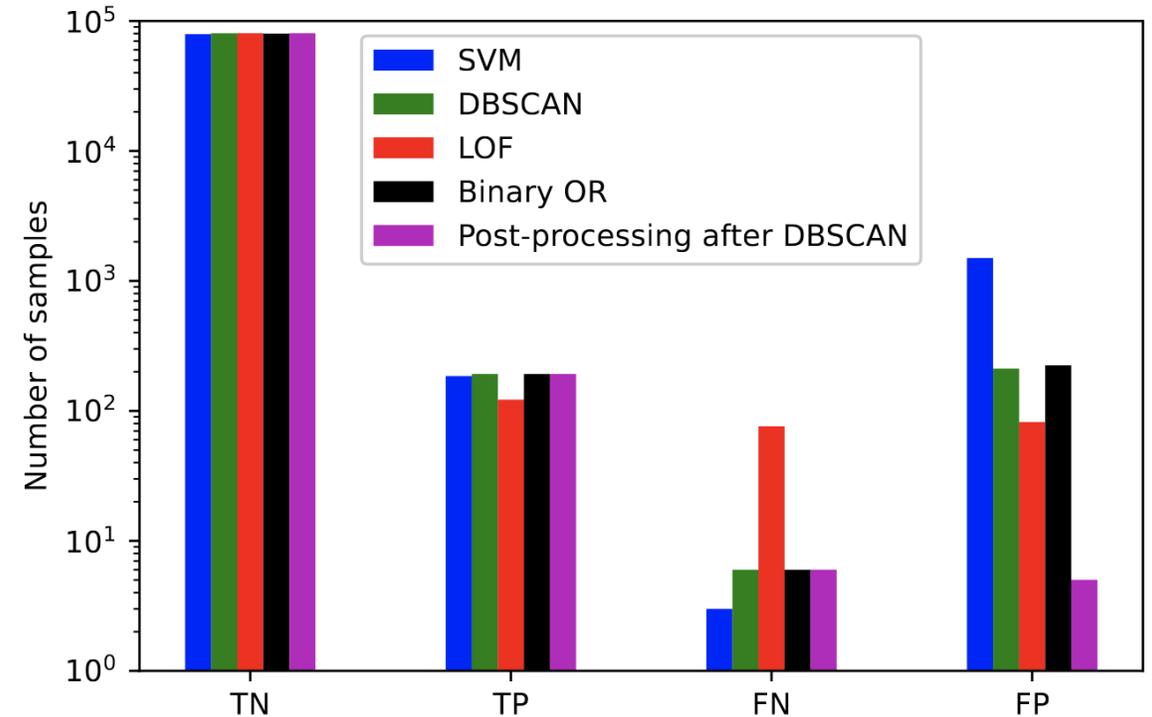


$\epsilon = 5$, MinPts = 4

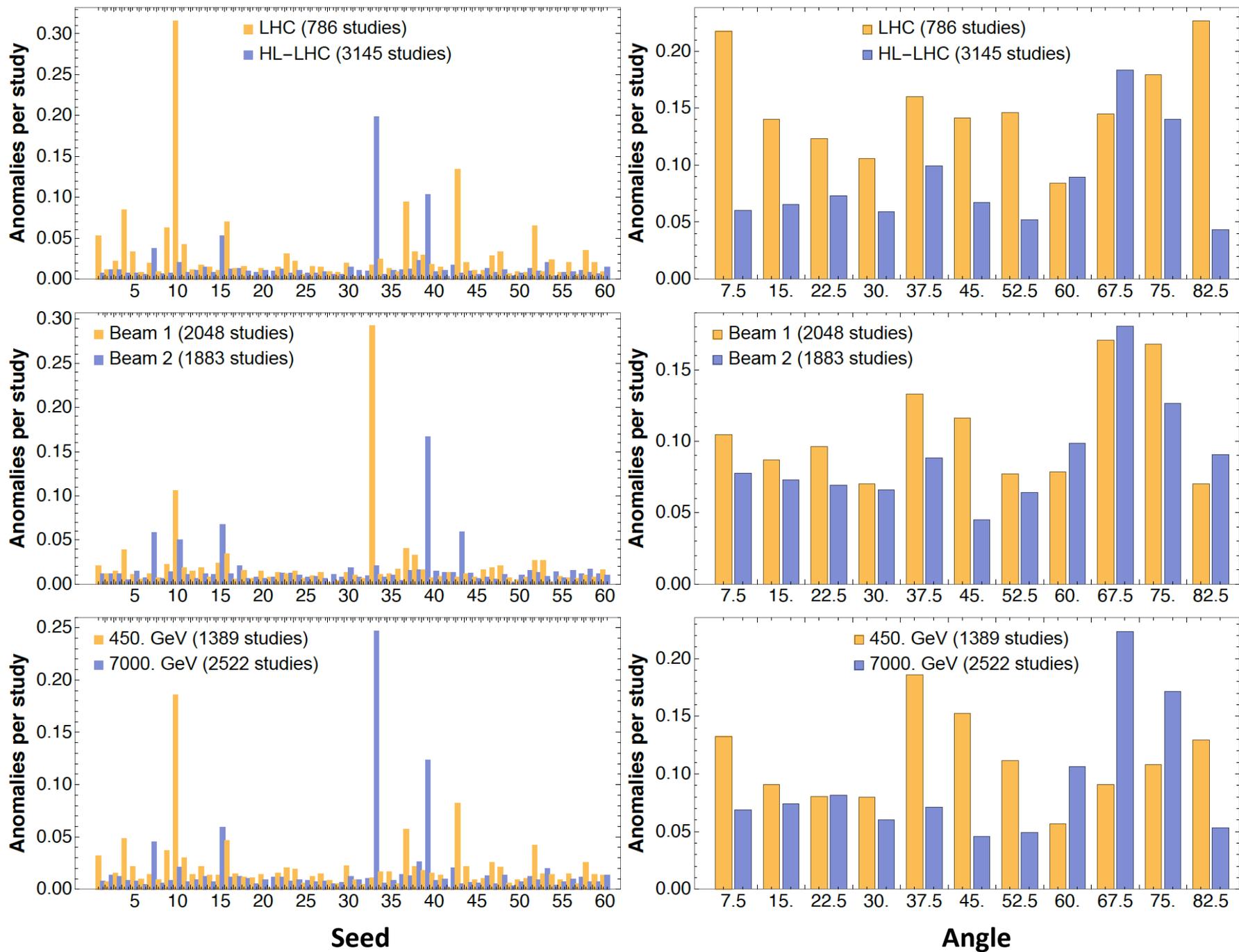
DA and anomalies



Two outliers correctly flagged



- The labels predicted by the DBSCAN and LOF algorithms were also combined through a binary OR operation to produce a further set of labels.
- The results show that the unsupervised methods perform an order of magnitude better than SVM in terms of false positives, however they are worse in terms of false negatives, especially when using LOF.
- The method of post-processing following DBSCAN clearly contributes to reducing the number of false positives, while maintaining the TP and FN rates.



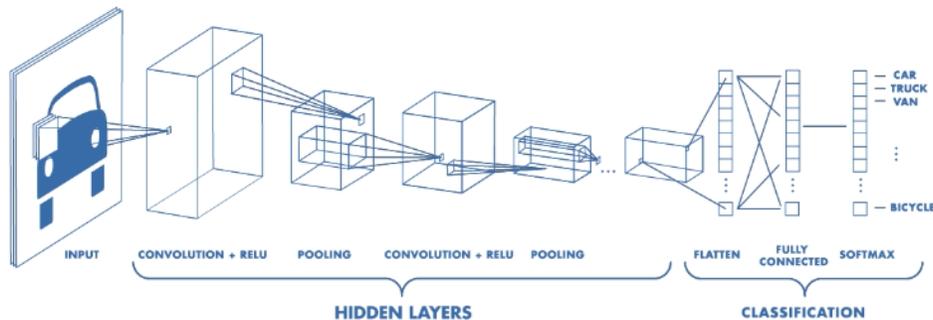
Outline

- Why is ML useful for particle accelerators?
- **Machine learning applications in the LHC**
 - In simulations – anomaly detection in dynamic aperture estimates
 - **In operation – improved tune measurements using a simulated dataset**

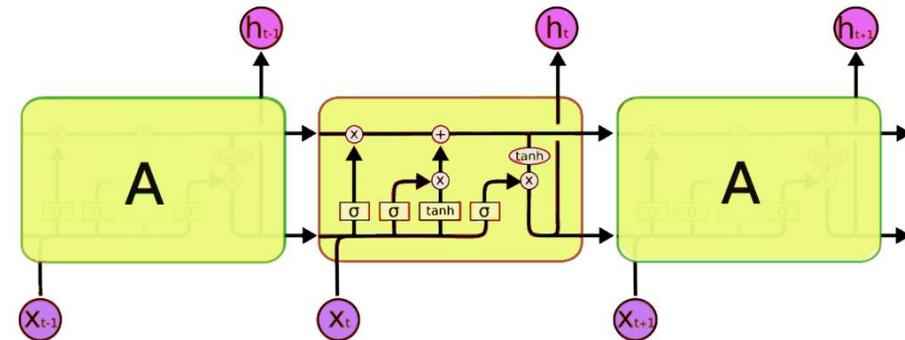
Neural networks

- Neural networks are by far the most versatile and pervasive ML model.
- Neural network models are used in supervised learning problems (e.g. classification, regression), unsupervised learning problems (e.g. dimensionality reduction) and reinforcement learning problems (e.g. to model the value/policy function).
- They can also be used to extract features from complex types of data such as images and sequences.

CNN:

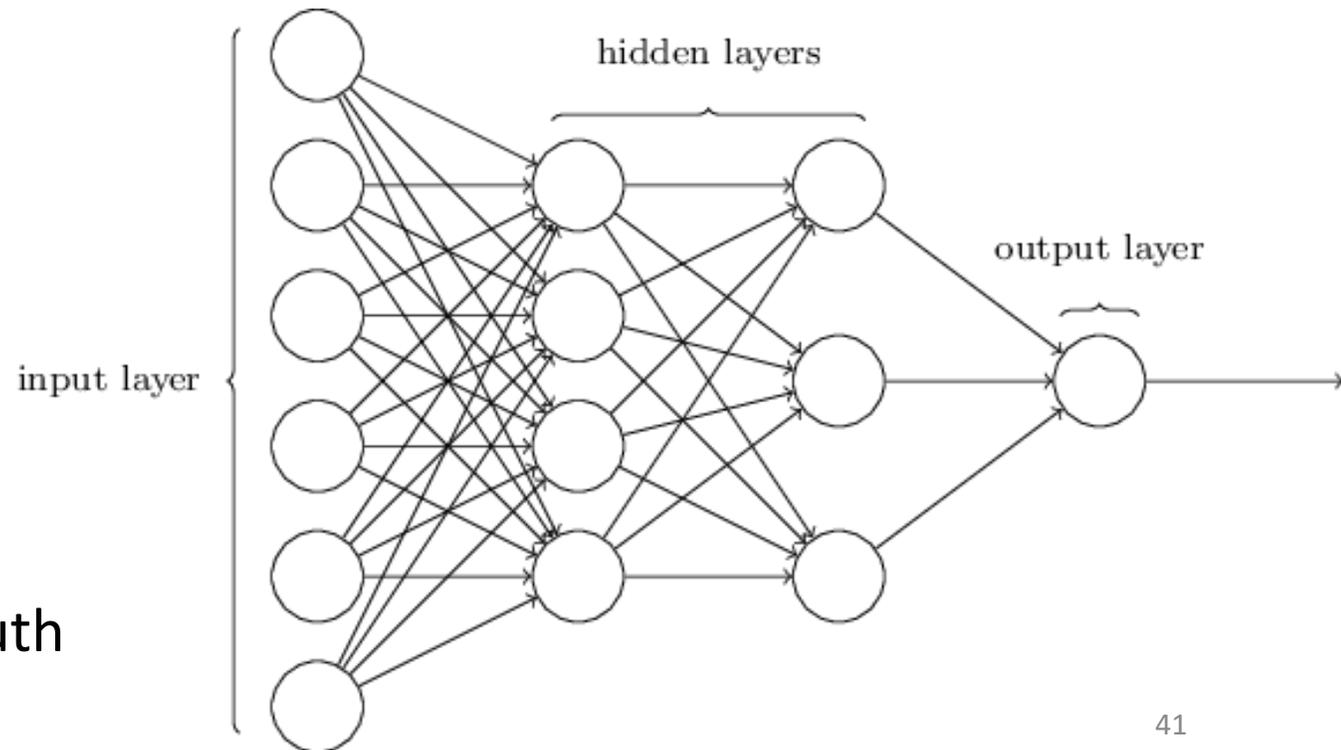


LSTM:



Basic network architecture

- Number of neurons in input and output layer: constrained by the problem dataset
- **Hyperparameters:**
 - Number of hidden layers, neurons in each layer
 - Activation function (e.g. tanh, sigmoid, ReLU)
 - Learning rate
 - Batch size & # epochs
- Training procedure: iterative
backpropagation of errors wrt ground truth



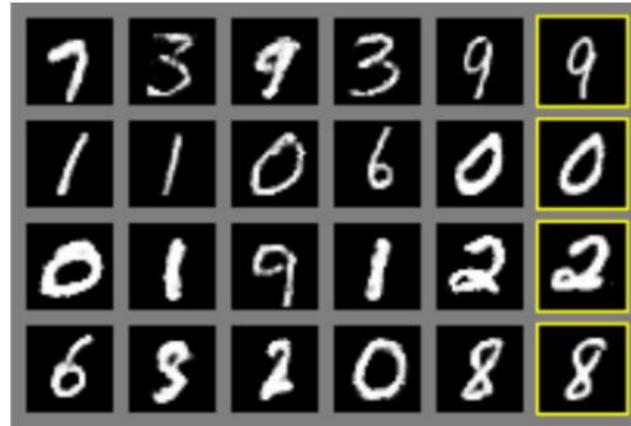
Generative Adversarial Networks (GANs)

- Generative networks are used to generate samples from an unlabeled distribution $P(X)$ given samples X_1, \dots, X_n .
- For example:
 - Learn to generate new, realistic images given exemplary images
 - Learn to generate new, realistic music given exemplary recordings
 - Learn to generate new, realistic text given exemplary corpus

Original GAN paper (Goodfellow et al, 2014)

3 different sets of images:
MNIST, TFD, CIFAR

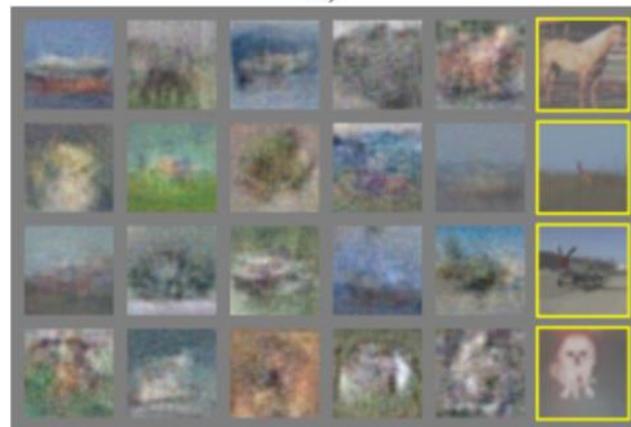
The first 20 images are all
generated, the
highlighted ones are the
nearest training example



a)



b)



c)



d)

Evolution in GAN output quality over 5 years



Unsupervised Image-to-Image Translation (Liu et al, 2018)



Generative vs Discriminative Networks

- Given a distribution of inputs X and labels Y
 - **Discriminative networks** model the conditional distribution $P(Y | X)$
 - **Generative networks** model the joint distribution $P(X, Y)$

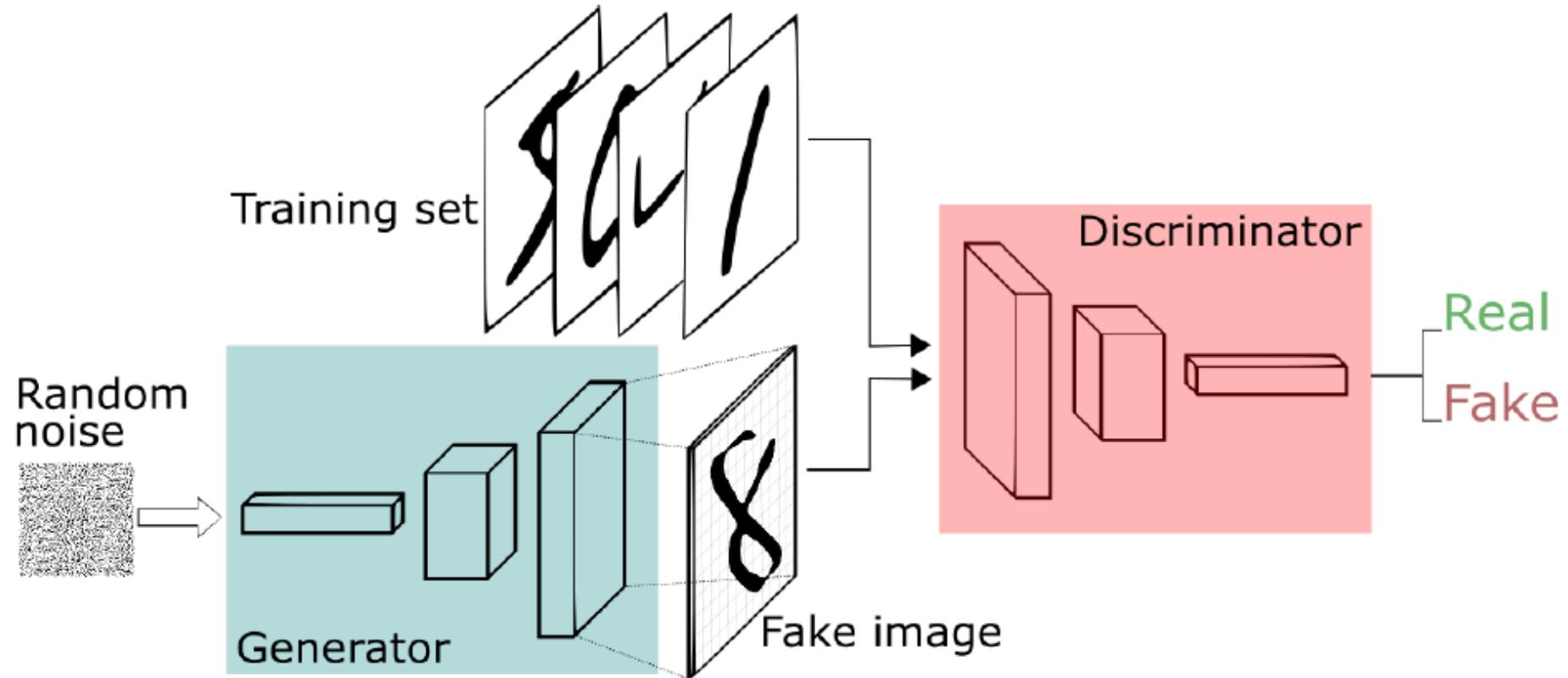
Why Generative Networks?

- Model understands the joint distribution $P(X, Y)$
 - Can calculate $P(X | Y)$ using Bayes rule.
 - Can perform other tasks like $P(X | Y)$, generating data from the label.
 - “Deeper” understanding of the distribution than a discriminative model.
- If you only have X , you can still build a model. Many ways to leverage unlabeled data. Not every problem is discriminative.
- However, model for $P(X, Y)$ is harder to learn than $P(Y | X)$
 - Map from X to Y is typically many to one
 - Map from Y to X is typically one to many
 - Dimensionality of X typically \gg dimensionality of Y

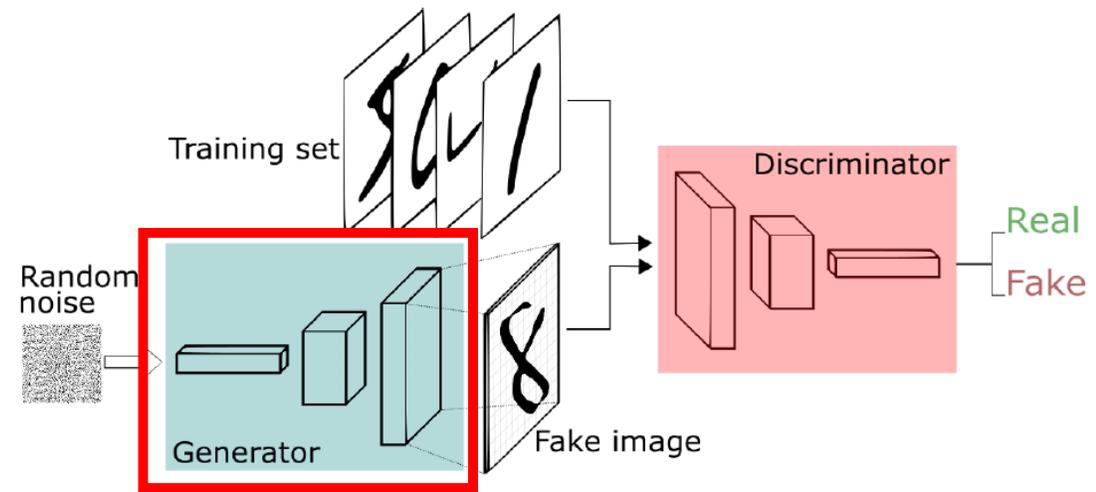
Generative Adversarial Networks (GANs)

- GANs were introduced in 2014 by Ian Goodfellow et al.
 - Goal is to model $P(X)$, the distribution of the training data
 - Model can generate samples from $P(X)$
 - Trained using a pair of “adversaries” (two players with conflicting loss functions)
 - A generator
 - A discriminator

GAN Architecture Diagram

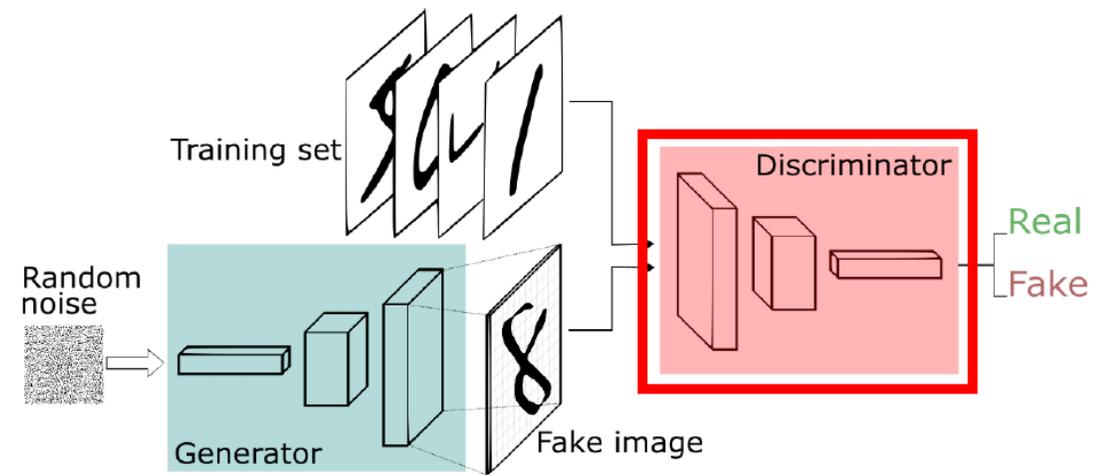


Generator



- The generator learns $P(X | Z)$ to produce realistic looking output samples X given samples from a hidden space Z
 - Hidden representation Z is sampled from a known prior, such as a Gaussian or uniform
 - Generator learns to map between a simple known distribution and a complicated output distribution
 - However, no simple loss function is available to measure the divergence between the **generated distribution** and the **real distribution**
 - Easy to measure distance between individual samples, harder to measure distance between complicated distributions
 - Instead of a traditional loss function, loss is calculated by a discriminator (another network)

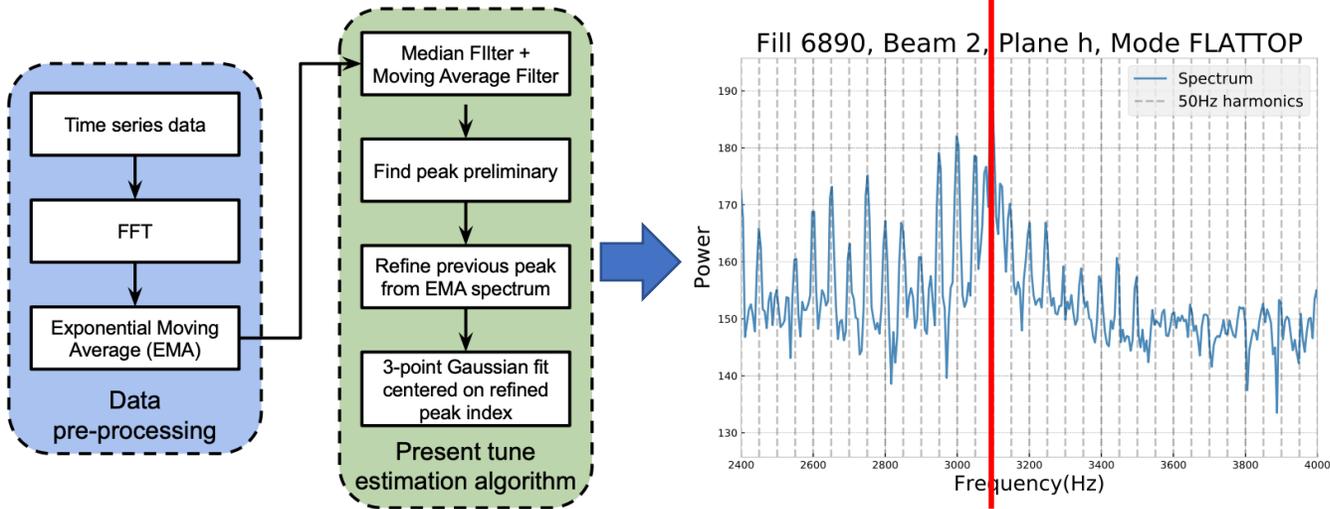
Discriminator



- The discriminator is a secondary neural network that guides the generator
 - Trained to tell the difference between real and generated data
 - Generator tries to “confuse” the discriminator, so it can’t tell the difference between real and generated data
 - From the perspective of the generator, the discriminator is like an adaptive loss function
- It is a “throwaway” network only really useful to train the generator.

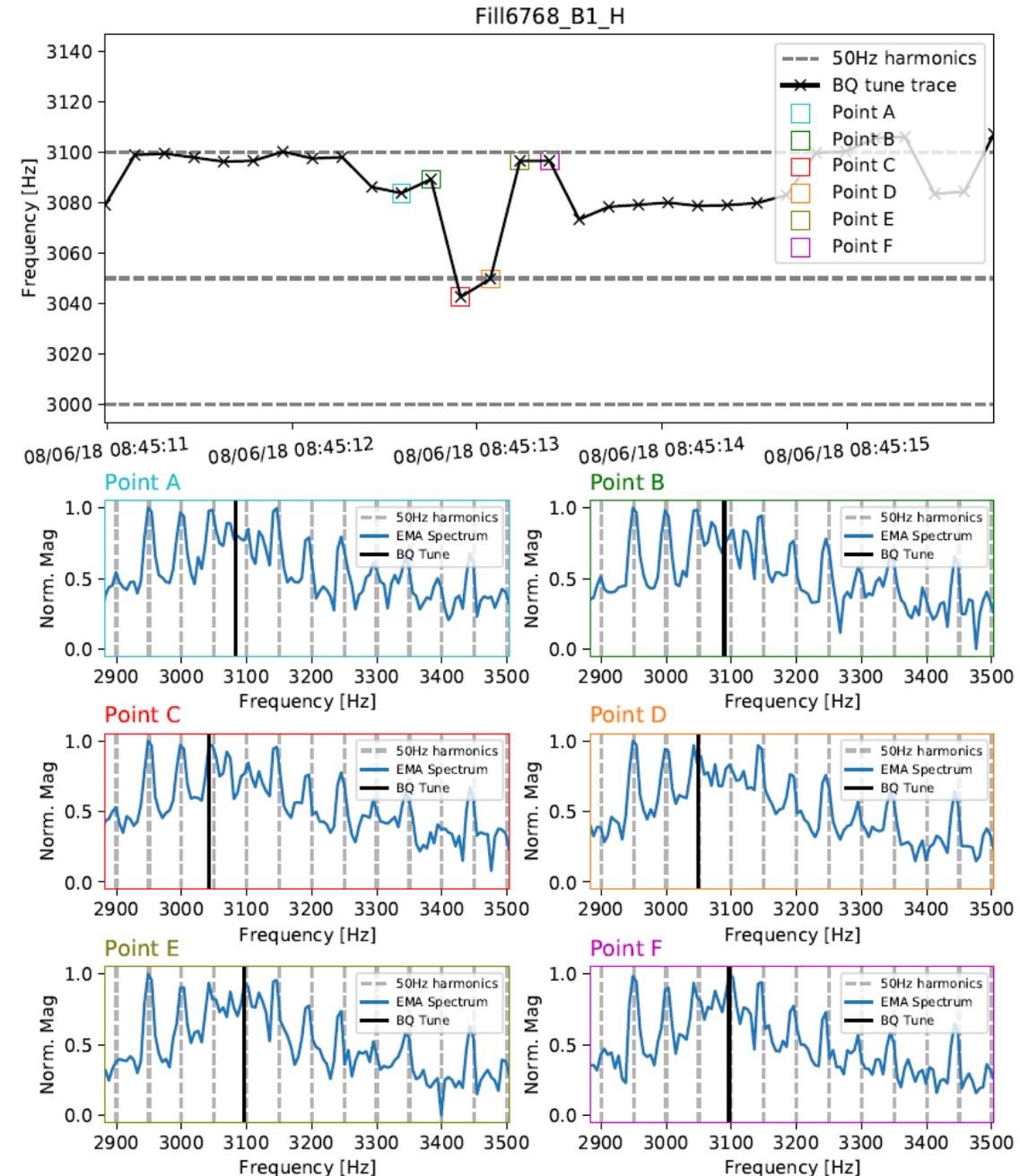
Tune estimation in the LHC

- The LHC tunes in H and V, B1 and B2 are measured by observing turn-by-turn betatron oscillations using a beam position monitor.



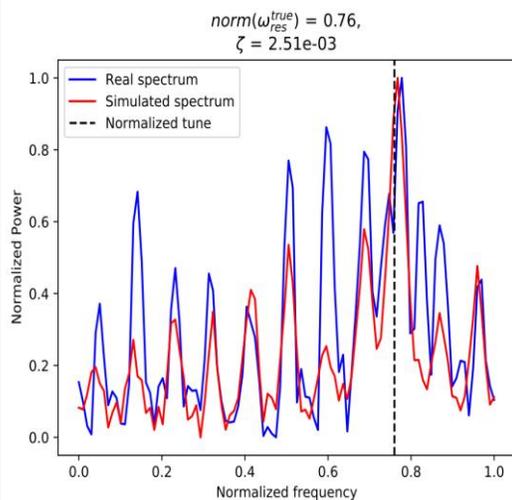
- 50 Hz harmonics
 - Present since start of LHC – due to main dipole magnets.
 - Harmonics perturb the spectrum, which affects reliability of the tune estimates.
 - Unstable tunes cause the Tune Feedback (QFB) system to switch itself off as a preventive measure.

L. Grech et al., MDPI Information, 2021.

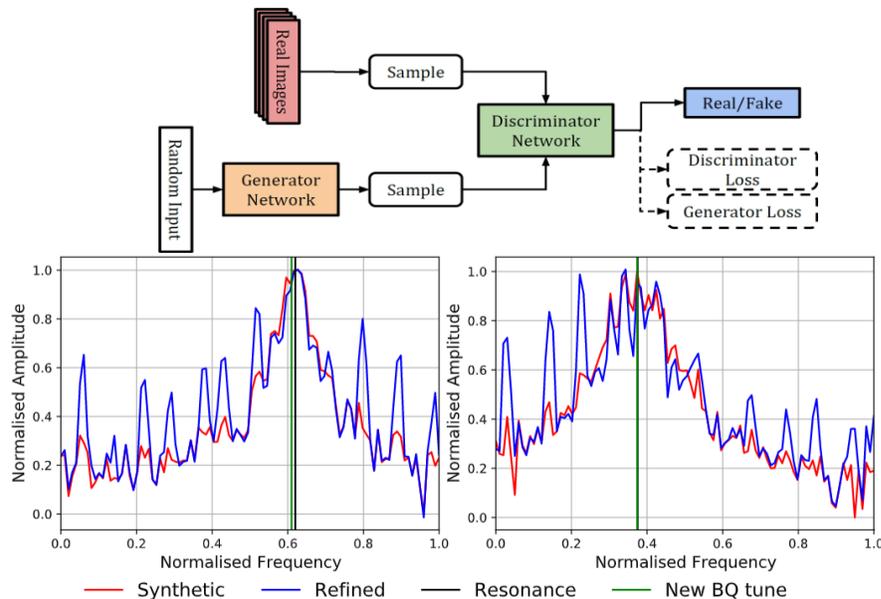


Tune estimation in the LHC

Second order system simulation of real BBQ spectra:



Variant of GAN called SimGAN used to improve simulated spectra:

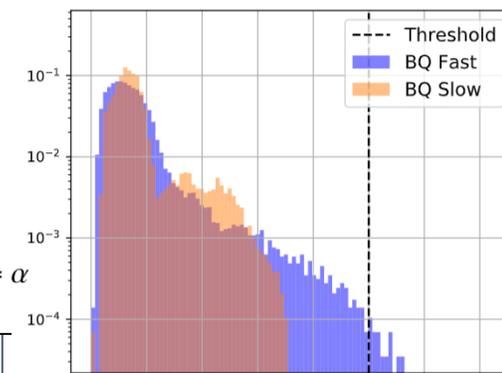


First spectra were simulated, passed through trained SimGAN, and a dataset was created to train ML models.

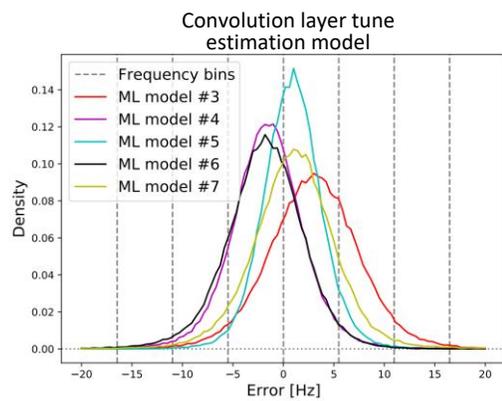
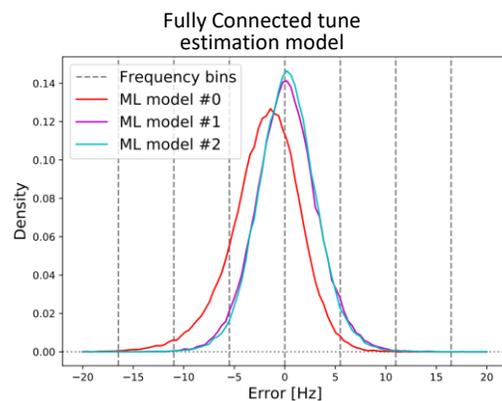
ML-Refined has same architecture as best simple model (Model #1) but trained over improved SimGAN dataset.

ML-Refined shows better stability than the current tune estimation algorithm, meaning better tune control in LHC.

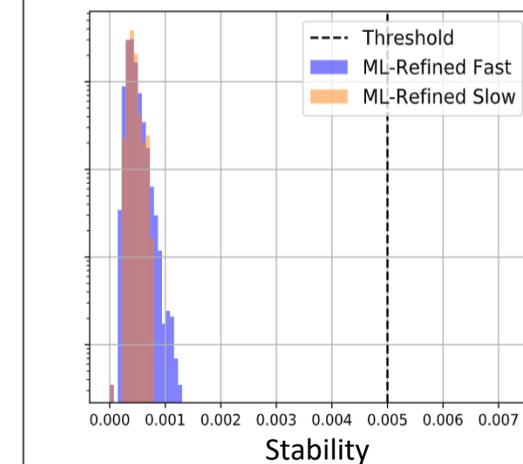
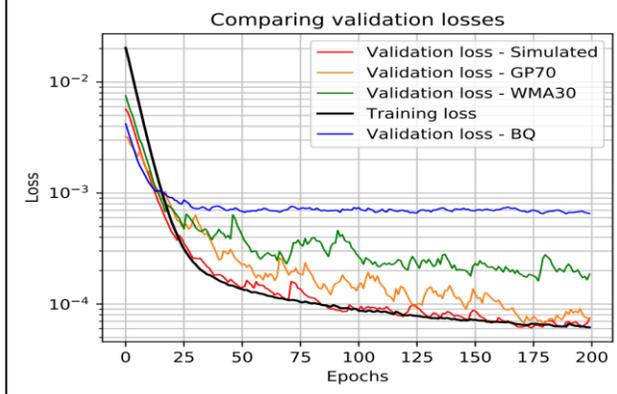
$$Stability = q_{t+1} * (1 - \alpha) + q_t * \alpha$$



Simple approach
Simulated data trains DNN and CNN



Some limitations, over-fitting to simulated training data:



Conclusions

- The few years have seen an high growth in the take-up of ML by the accelerator community, driven by:
 - Deep learning developments
 - Increase in scale and complexity of machines
 - Availability of data
- Some of the latest activities involving DA anomaly detection and LHC tune estimation were reviewed.
- ML will be a key tool to help meet demands for higher beam brightness, energy and intensity.