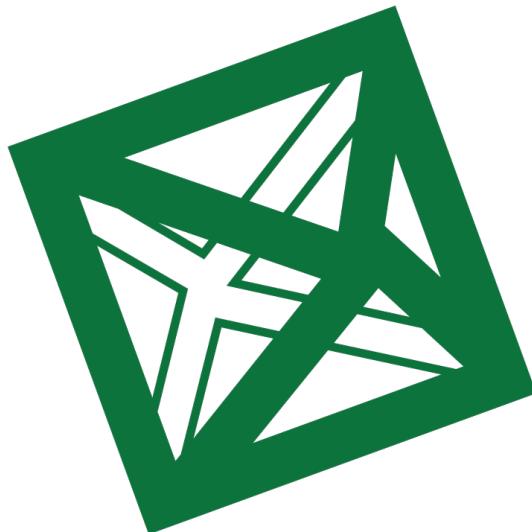


UNIVERSITÀ DEGLI STUDI DI MILANO-BICOCCA

---

SCHOOL OF SCIENCE  
DEPARTMENT OF PHYSICS



Master Degree in Particle Physics

---

# Overcoming medical data scarcity: transfer learning from synthetic particle jets images to lung CT using deep neural networks

---

**Supervisor:**

Prof. Pietro Govoni

**Co-Supervisor:**

Dott. Simone Gennai

**Candidate:**

Beatrice Scotti  
Student ID: 858535

17 september 2025

---

Academic Year 2024-2025

# Contents

<b>Abstract</b>	<b>3</b>
<b>1 Computed Tomography</b>	<b>4</b>
1.1 Tomography setup . . . . .	5
1.2 Image Reconstruction . . . . .	5
1.3 Windowing . . . . .	8
1.4 CT Planes . . . . .	9
<b>2 Machine Learning in a nutshell</b>	<b>11</b>
2.1 Neural networks and deep learning . . . . .	12
2.2 YOLO: You Only Look Once . . . . .	12
2.2.1 Architecture of YOLOv8 . . . . .	12
2.3 Object Detection Metrics . . . . .	13
2.4 Transfer learning . . . . .	17
2.5 Curriculum Learning . . . . .	18
<b>3 Hadronic Collisions and Jet Reconstruction</b>	<b>20</b>
3.1 High energy collisions in hadronic accelerators . . . . .	20
3.2 Useful variables to describe an hadronic process . . . . .	21
3.3 Algorithms for jet reconstruction . . . . .	22
3.3.1 Cone Algorithms . . . . .	22
3.3.2 Sequential Clustering Algorithms . . . . .	23
<b>4 Synthetic Particle Dataset</b>	<b>26</b>
4.1 Blob Generation . . . . .	26
<b>5 Medical Dataset</b>	<b>28</b>
5.1 NLST - National Lung Screening Trial Dataset . . . . .	28
5.1.1 Dataset Pruning . . . . .	28
5.1.2 Extraction of the pixel array . . . . .	29
5.1.3 Image Metadata . . . . .	29
5.2 DLCS - Duke Lung Cancer Screening Dataset . . . . .	29
5.2.1 Extraction of the 2D slices . . . . .	29
5.2.2 Dataset Pruning . . . . .	29
5.3 Pre processing the images . . . . .	29
5.3.1 Masking . . . . .	32

<b>6</b>	<b>Dataset creation and Training Strategies</b>	<b>34</b>
6.1	YOLO dataset . . . . .	34
6.2	Curriculum Learning . . . . .	34
<b>7</b>	<b>Training Results</b>	<b>35</b>
7.1	Medical Images . . . . .	35
7.2	Jet Images - Curriculum Learning . . . . .	35
7.3	Transfer Learning in a data scarcity regime . . . . .	35
7.4	Transfer learning in a normale data regime . . . . .	35
	<b>Conclusion</b>	<b>36</b>

# Abstract

Early diagnosis is the most effective weapon in the fight against cancer, with computed tomography and medical imaging playing a crucial role in this process. Recently, artificial intelligence has shown great potential to identify and classify tumor lesions in their early stages. However, the application of these techniques faces a fundamental limitation: the scarcity of annotated data available for training.

As is well known in machine learning, the quality of results depends strictly on the quality and completeness of the training data ("garbage in, garbage out"). To overcome this limitation, this thesis explores the use of transfer learning, a technique that allows the transfer of knowledge from a data-rich source domain to a data-limited target domain.

Specifically, the research focuses on the use of simulated images of high-energy particle jets, constructed specifically for this study, characterized by two distinct classes of physical phenomena: a noisy background with physical characteristics similar to the main jets and morphological properties similar to tumor structures.

The proposed approach involves a pretraining phase on a large high-energy physics dataset, transferring the learned weights to the medical domain, and then fine-tuning them on limited clinical datasets. This method aims to leverage the feature extraction capabilities developed in the particle physics domain, adapting them to the medical context where annotated data are scarce, but diagnostic accuracy is crucial.

# Chapter 1

## Computed Tomography

Computed Tomography (CT) is a medical imaging technique that uses X-rays to generate detailed cross-sectional images of the human body. Since its introduction in the 1970s, CT has revolutionized diagnostic radiology by enabling the visualization of internal structures with high spatial resolution and three-dimensional reconstruction capabilities. Unlike conventional radiography, which produces a single projection image, CT acquires multiple X-ray measurements from different angles around the patient and uses computer algorithms to reconstruct the corresponding anatomical slices.

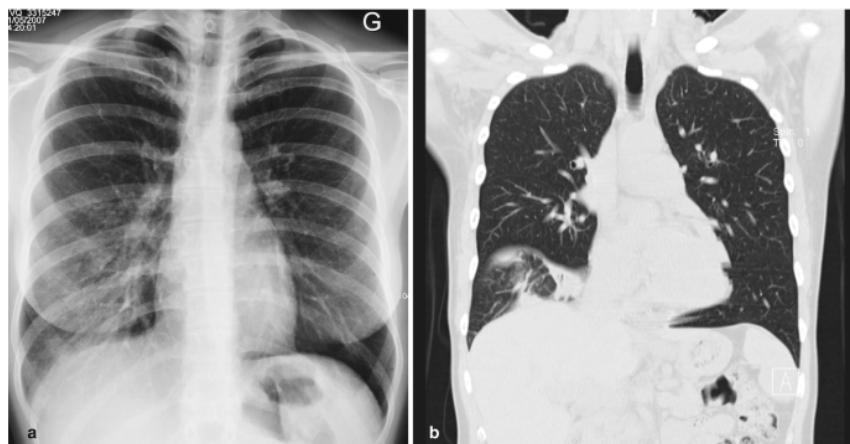


Figure 1.1: On the left (figure a.) we can see a chest radiography, on the right (figure b.) we can see a CT scan of the same area. The CT scan provides a much more detailed view of the internal structures, allowing for better diagnosis and treatment planning.

CT plays a crucial role in clinical practice due to its ability to provide fast, accurate, and non-invasive diagnostic information for a wide range of applications. Modern CT scanners are equipped with advanced technologies that significantly improve image quality, reduce scan time, and minimize radiation dose.

This chapter provides an overview of the main aspects of CT imaging that are relevant for medical physics applications. The first part briefly describes the experimental setup of a CT scanner, the second part focuses on the mathematical principles behind image reconstruction, highlighting the algorithms used to convert projection data into tomographic slices. Finally, the chapter introduces the concept of image post-processing, with particular attention to windowing and its impact on image interpretation.

## 1.1 Tomography setup

The basic idea behind tomography is to obtain a three-dimensional image of an object from a series of two-dimensional images acquired from different angles. For this purpose, a CT scanner consists, like any radiation measurement system, of a radiation source (an X-ray tube) and an array of particle detectors. To acquire projections from multiple angles, typically ranging from 0 to 180 degrees, the X-ray tube and the detectors are mounted on a rotating gantry, which allows the system to rotate around the patient or, more generally, around the object being examined.

As shown in Figure 1.2, the detectors are not arranged on a plane perpendicular to the X-ray source, but rather along a circular arc. This configuration ensures that all pixels in the projection correspond to the same physical size, thus avoiding geometric distortions in the reconstructed image. Depending on the scanner design, the acquisition can be performed in a step-and-shoot mode or using continuous rotation, as in helical (spiral) CT. The X-ray beam is collimated into a fan or cone shape to cover the entire detector array. Each detector element typically consists of a scintillator coupled to a photodiode, which converts the X-ray photons into an electrical signal proportional to the received dose. Synchronizing the gantry rotation with data acquisition allows the collection of a sufficient number of projections to ensure accurate image reconstruction.

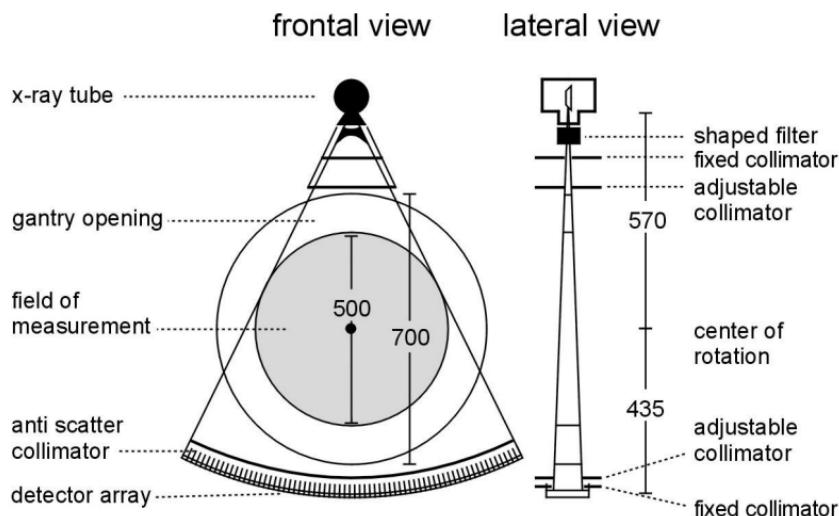


Figure 1.2: In the frontal view we can see, in order, the X-ray tube (radiation source), the rotating gantry (support for the X-ray tube and detectors), the field of measurement (where the patient or the object is allocated), the anti scatter collimators (to reduce the amount of scattered radiation reaching the detectors), and the detectors (which measure the intensity of the X-rays after they pass through the object). In the lateral view we can see the presence of collimators, used to adjust the narrowness of the beam depending on the type of scan and the area of interest.

## 1.2 Image Reconstruction

In computed tomography (CT), image reconstruction refers to the process of transforming raw projection data, acquired from different angles, into a cross-sectional image of the scanned object. The core idea is to recover a 2D function  $f(x, y)$  — representing the

internal structure — from its projections. Each projection contains information about how much an X-ray beam is attenuated as it travels through the object along a certain direction.

## 1. The Radon Transform

Mathematically, each X-ray beam travels along a straight line. For a beam oriented at angle  $\theta$ , the line is defined by:

$$x \cos \theta + y \sin \theta = s$$

where  $s$  is the perpendicular distance from the origin to the line.

The projection  $p(s, \theta)$  is obtained by integrating the object's density  $f(x, y)$  along that line:

$$p(s, \theta) = \int_{-\infty}^{+\infty} f(x, y) \delta(x \cos \theta + y \sin \theta - s) dx dy$$

This is called the **Radon transform** of the function  $f(x, y)$ .

## 2. The Fourier Slice Theorem

To reconstruct the image, a key result is the **Fourier Slice Theorem**. It says that the 1D Fourier transform of a projection at angle  $\theta$  gives a slice of the 2D Fourier transform of  $f(x, y)$  taken in the same direction.

We compute the 1D Fourier transform of the projection:

$$P(\theta, u) = \int_{-\infty}^{+\infty} p(s, \theta) e^{-2\pi i us} ds$$

Plugging in the expression for  $p(s, \theta)$ , we get:

$$P(\theta, u) = \iint f(x, y) e^{-2\pi i u(x \cos \theta + y \sin \theta)} dx dy$$

This expression matches the 2D Fourier transform of  $f(x, y)$ , evaluated at:

$$u_x = u \cos \theta, \quad u_y = u \sin \theta$$

so we can write:

$$P(\theta, u) = F(u \cos \theta, u \sin \theta)$$

In other words, each projection gives us a radial slice of the 2D Fourier transform of the image.

## 3. Inverse Transform and Filtering

If we collect projections over many angles, we can build the 2D Fourier space of the image. Then, to reconstruct  $f(x, y)$ , we apply the inverse 2D Fourier transform:

$$f(x, y) = \iint F(u_x, u_y) e^{2\pi i (u_x x + u_y y)} du_x du_y$$

Switching to polar coordinates:

$$u_x = u \cos \theta, \quad u_y = u \sin \theta$$

the Jacobian of the transformation gives:

$$du_x du_y = |u| du d\theta$$

Substituting into the inverse Fourier formula:

$$f(x, y) = \int_0^\pi \int_{-\infty}^{+\infty} |u| P(\theta, u) e^{2\pi i u(x \cos \theta + y \sin \theta)} du d\theta$$

Here, the factor  $|u|$  acts as a high-pass filter, enhancing high-frequency components and correcting for the blurring introduced during projection. The equation above gives the foundation for the **Filtered Back Projection** algorithm, one of the most common reconstruction methods in CT. It works as follows:

1. Acquire projections  $p(s, \theta)$  at many angles  $\theta \in [0, \pi]$ .
2. Compute their 1D Fourier transforms  $P(\theta, u)$ .
3. Multiply by  $|u|$  (or an equivalent ramp-like filter) to enhance sharpness.
4. Invert the Fourier transform to get filtered projections.
5. Back-project the filtered projections over the image domain and sum them.

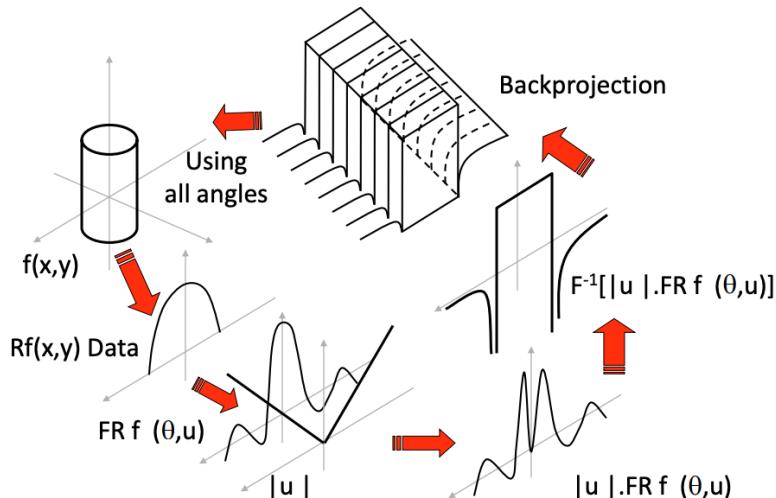


Figure 1.3: Radon transform and reconstruction via filtered back projection. Each projection contributes a slice in frequency space; the image is recovered by inverse Fourier transform.

The filtered back projection is fast and effective, which is why it is still widely used in clinical CT. However, modern scanners often integrate more advanced techniques such as iterative reconstruction or machine learning, which can offer better noise suppression and lower radiation dose. Nevertheless, understanding the Radon transform and the Fourier Slice Theorem provides deep insight into how tomographic images are formed.

## 1.3 Windowing

The pixel values are quantified in **Hounsfield units (HU)**, a standardized scale that reflects the radiodensity of the tissue. This scale is defined relative to the attenuation coefficients of water and air:

$$HU_{tissue} = 1000 \times \frac{\mu_{tissue} - \mu_{water}}{\mu_{water} - \mu_{air}} \quad (1.1)$$

where  $\mu$  represents the linear attenuation coefficient. Key reference points: water = 0 HU for definition, air = -1000 HU.

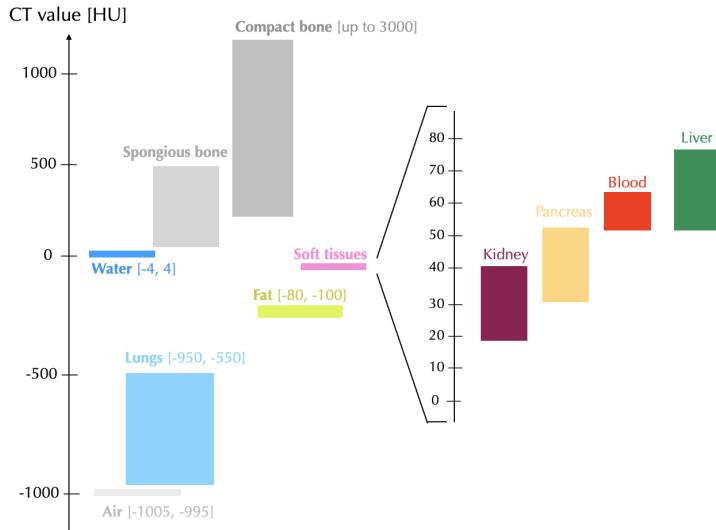


Figure 1.4: CT Hounsfield Unit (HU) ranges for key tissues. Highlighted regions show typical HU values for diagnostic reference, from dense compact bone ( $\sim 3000$  HU) to air (1000 HU), with soft tissues (20 – 80 HU) and fluids (blood  $\sim 45$  HU, water 0 HU) in intermediate ranges.

**Windowing** is a technique used in computed tomography (CT) to **enhance the visibility of specific tissues or structures within the body**. It involves adjusting the range of Hounsfield units (HU) displayed in the CT image, allowing radiologists to focus on particular types of tissues, such as bones, soft tissues, or air-filled spaces. The window width and center determine the contrast and brightness of the image, respectively.

- **Window width (WW):** This parameter controls the range of Hounsfield units displayed in the image. A narrow window width enhances the contrast between tissues with similar densities, while a wider window width allows a broader range of densities to be visualized.
- **Window center (WC):** This parameter sets the midpoint of the Hounsfield unit range displayed in the image. Adjusting the window level changes the brightness of the image, allowing radiologists to focus on specific tissue types.

The choice of window width and level depends on the specific clinical question and the type of tissue being examined.

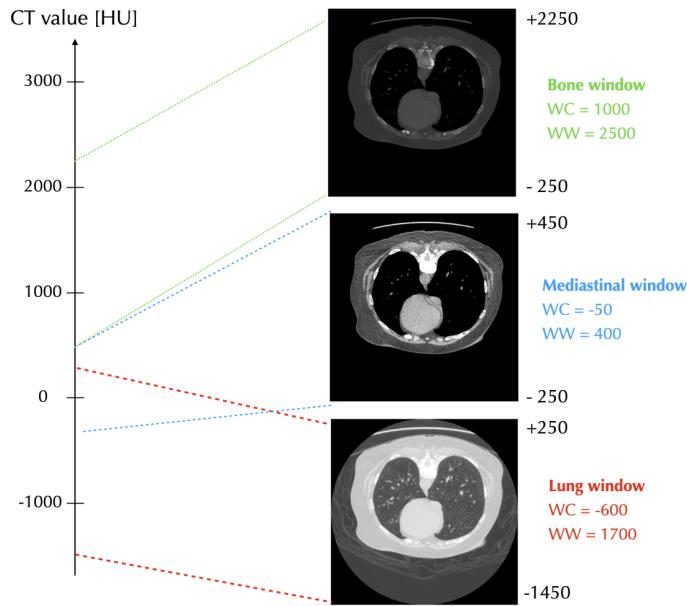


Figure 1.5: Standard CT windowing settings for different anatomical structures. Full CT value range ( $[-1000, 3000]$  HU). Clinical presets with window center (WC) and width (WW) values for bone (WC=1000, WW=2500), mediastinum (WC=-50, WW=400), and lung (WC=-600, WW=1700) visualization.

## 1.4 CT Planes

Images can be visualized in three standard orthogonal planes: axial, coronal, and sagittal. Each plane offers a unique perspective on anatomical structures, aiding in comprehensive clinical assessment.

- **Axial plane (transverse):** A horizontal plane that divides the body into superior (upper) and inferior (lower) parts. It is the primary acquisition plane in most CT scans.
- **Coronal plane:** A vertical plane that divides the body into anterior (front) and posterior (back) parts. Often used for viewing the frontal anatomy.
- **Sagittal plane:** A vertical plane that divides the body into left and right portions. It is particularly useful for evaluating asymmetries between the two sides.

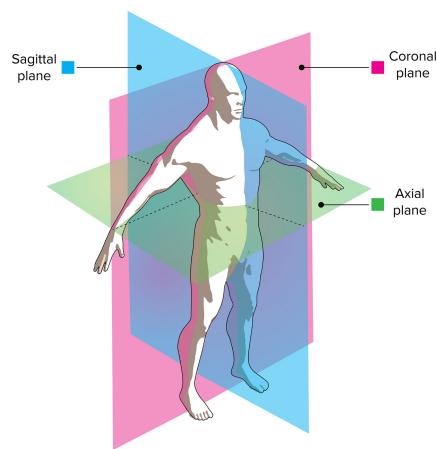


Figure 1.6: CT planes: axial, coronal, and sagittal. The axial plane is horizontal, the coronal plane is vertical and divides the body into front and back, and the sagittal plane is vertical and divides the body into left and right.

# Chapter 2

## Machine Learning in a nutshell

Machine learning is a subfield of artificial intelligence that focuses on the development of algorithms and statistical models that enable computers to perform tasks without explicit instructions. Instead, these systems learn from data, identifying patterns and making decisions based on the information they have been trained on. Machine learning consists of designing efficient and accurate prediction algorithms. More generally, learning techniques are data-driven methods combining fundamental concepts in computer science with ideas from statistics, probability and optimization.

### Types of tasks

The following are some standard types of tasks in machine learning:

- **Classification:** Assigning labels to data points based on learned patterns (e.g., e-mail spam detection).
- **Regression:** Predicting continuous values based on input features (e.g., predicting house prices). In regression, the penalty for an incorrect prediction depends on the magnitude of the difference between the true and predicted values, in contrast with classification problem, where there is typically no notion of closeness between various categories.
- **Object detection:** Identifying and localizing objects within images or videos (e.g., detecting tumours in lung CT scans).
- **Clustering:** Grouping similar data points together without predefined labels.
- **Anomaly detection:** Identifying unusual patterns that do not conform to expected behavior.
- **Ranking:** Ordering items based on their relevance or importance.

Algorithms that solve a learning task based on semantically annotated historical data are said to operate in a **supervised learning** mode. In contrast, algorithms that use data without any semantic annotation are said to operate in an **unsupervised learning** mode. In the latter case, the algorithm is expected to discover patterns in the data without any prior knowledge of the labels or categories. In this thesis I'll mainly focus on supervised learning.

**Label set:** We use  $Y$  to denote the set of all possible labels for a data point of a given learning problem. Note that the labels can be of two different types: categorical

labels, which are discrete and finite and define classification problems, and continuous labels, which can take any value in a continuous range and define regression problems.

## 2.1 Neural networks and deep learning

Neural networks are a class of machine learning algorithms inspired by the structure and function of the human brain. They consist of interconnected nodes (neurons) that process information in layers. Deep learning refers to the use of neural networks with many layers (deep neural networks) to model complex patterns in large datasets. This approach has led to significant advancements in areas such as image recognition, natural language processing, and game playing.

## 2.2 YOLO: You Only Look Once

Object detection is a task that involves identifying and classifying objects present in images or videos. Initially, object detection was approached as a pipeline consisting of three main steps: proposal generation, feature extraction and region classification. However, this approach was computationally expensive and often led to suboptimal results. The emergence of deep learning brought a significant change in object detection, with deep convolutional neural networks (CNNs) playing a crucial role in this transformation. CNNs are designed to automatically learn hierarchical features from raw pixel data, eliminating the need for manual feature engineering. This shift allowed for more efficient and accurate object detection systems.

Currently, deep learning-based object detection frameworks can be classified in two families:

- **Two-stage detectors:** These methods first generate region proposals and then classify them. Examples include R-CNNs (Region-based Convolutional Neural Networks), that first generate region proposals using a selective search algorithm and then extracts features from these regions using a CNN; the extracted features are then fed into an SVM for object classification.
- **One-stage detectors:** These methods perform detection in a single pass, directly predicting bounding boxes and class probabilities. Examples include YOLO (You Only Look Once), which exists in eleven versions. The YOLO models are popular for their accuracy and compact size. It is a state-of-the-art model that could be trained on any hardware. YOLOv8, in particular, was developed by Ultralytics and introduced on January 2023. It is used to detect objects in images, classify images and distinguish objects from each other.

### 2.2.1 Architecture of YOLOv8

The YOLOv8 architecture is composed of two major parts, namely the **backbone** and the **head**, both of which use a fully convolutional neural network.

- The **backbone** is responsible for extracting features from the input image. It consists of a modified version of the CSPDarknet53 architecture, which has 53 convolutional layers and employs a technique called cross-stage partial connections to

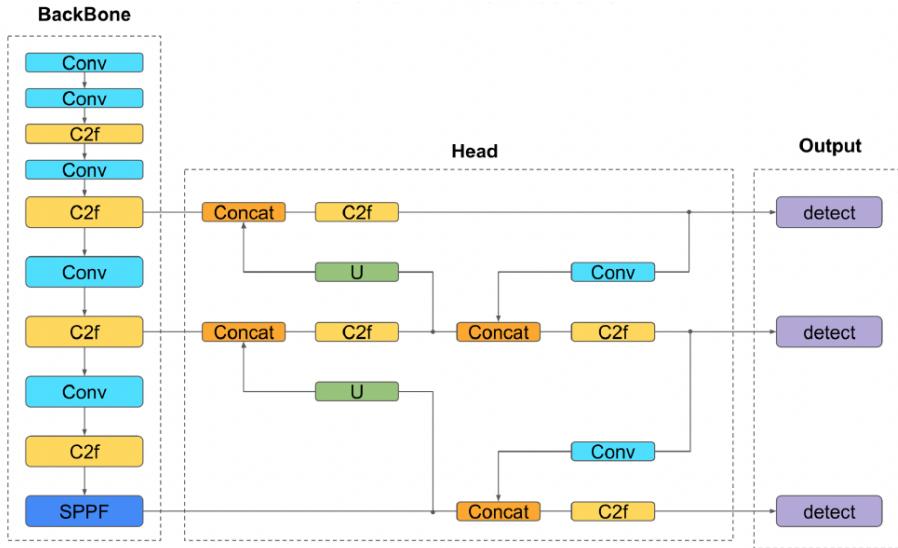


Figure 2.1: Architecture of YOLOv8. The backbone extracts features from the input image, while the head predicts bounding boxes and class probabilities.

enhance the transmission of information across the various levels of the network. The convolutional layers are organized in a sequential manner to extract relevant features from the input image.

- The **head** is responsible for predicting bounding boxes and class probabilities. It consists of a series of convolutional layers that take the features extracted by the backbone and apply additional operations to predict the bounding boxes and class probabilities for each object in the image. The head uses a technique called anchor boxes to handle objects of different sizes and aspect ratios.

The YOLOv8 framework can be used to perform computer vision tasks such as detection, segmentation, classification and pose estimation and comes with pre-trained models for each task. For detection, the models are pre trained on the COCO dataset, while for classification on ImageNet dataset. There are different versions of YOLOv8, each designed for different tasks and with different architectures. The most common versions are YOLOv8n, YOLOv8s, YOLOv8m, YOLOv8l and YOLOv8x, where the letter indicates the size of the model (n for nano, s for small, m for medium, l for large and x for extra large). The larger the model, the more parameters it has and the more computational resources it requires to train and run.

## 2.3 Object Detection Metrics

A metric is a function that quantifies and evaluates the performance of a model. In the context of object detection, metrics are used to assess how well a model can detect and classify objects in images.

### Intersection over Union (IoU)

When we talk about object detection, we often refer to the concept of **bounding boxes**. A bounding box is a rectangular box that is used to define the location of an object in an

image, typically represented by its top-left corner coordinates ( $x, y$ ), width, and height, or top-left and bottom-right coordinates. The bounding box is used to localize the object within the image and is often used in conjunction with a classification label to identify the object. Intersection over Union (IoU) is a metric used to evaluate the accuracy of the bounding box detection. It measures the overlap between the predicted and the ground truth box. The IoU is calculated as follows:

$$IoU = \frac{Area_{Intersection}}{Area_{Union}} = \frac{A_{pred} \cap A_{gt}}{A_{pred} \cup A_{gt}} \quad (2.1)$$

The IoU value ranges from 0 to 1, where 0 indicates no overlap and 1 indicates perfect overlap. A common threshold for considering a detection as a true positive is an IoU of 0.5 or higher.

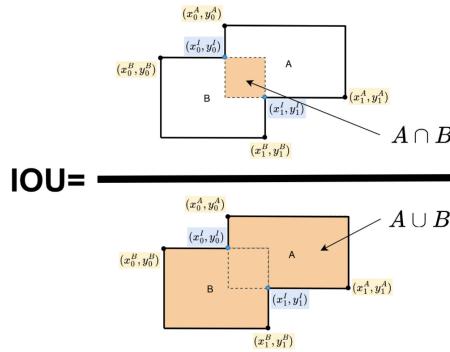


Figure 2.2

---

**Algorithm 1** Intersection over Union (IoU) between two bounding boxes

- 1: **Input:** Boxes  $A = (x_A, y_A, w_A, h_A)$  and  $B = (x_B, y_B, w_B, h_B)$
  - 2: **Compute bottom-right corners:**

$$\begin{aligned} x2_A &= x_A + w_A, & y2_A &= y_A + h_A \\ x2_B &= x_B + w_B, & y2_B &= y_B + h_B \end{aligned}$$
  - 3: **Compute intersection rectangle:**

$$\begin{aligned} x_{int}^1 &= \max(x_A, x_B), & y_{int}^1 &= \max(y_A, y_B) \\ x_{int}^2 &= \min(x2_A, x2_B), & y_{int}^2 &= \min(y2_A, y2_B) \end{aligned}$$
  - 4: **Compute intersection area:**

$$\begin{aligned} w_{int} &= \max(0, x_{int}^2 - x_{int}^1) \\ h_{int} &= \max(0, y_{int}^2 - y_{int}^1) \\ A_{int} &= w_{int} \times h_{int} \end{aligned}$$
  - 5: **Compute areas of each box:**

$$A_A = w_A \times h_A, \quad A_B = w_B \times h_B$$
  - 6: **Compute union area:**

$$A_{union} = A_A + A_B - A_{int}$$
  - 7: **Return:**  $\text{IoU} = \frac{A_{int}}{A_{union}}$
-

## Precision and Recall

Precision and recall are fundamental metrics used to evaluate the performance of object detection models. These metrics provide valuable insights into the model's ability to identify objects of interest within images. Let's define some fundamental concepts before diving into precision and recall:

- **True Positive (TP):** These are instances where the model currently identifies and localizes objects, and the intersection over union (IoU) score between the predicted and the ground truth bounding box is equal or greater than a specified threshold.
- **False Positive (FP):** These are cases where the model incorrectly identifies an object that does not exist in the ground truth or where the predicted box has an IoU score below the defined threshold.
- **False Negative (FN):** These represent instances where the model fails to detect an object that is present in the ground truth. This means that the model has not detected an object that is present in the image, resulting in a missed detection.
- **True Negative (TN):** Not applicable in object detection. It represents correctly rejecting the absence of objects, but in object detection, the goal is to detect objects rather than the absence of them.

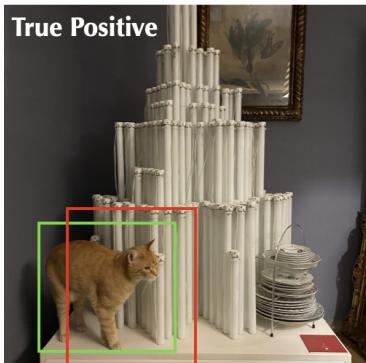


Figure 2.3: The object is there, and the model detects it, with an IoU above threshold.

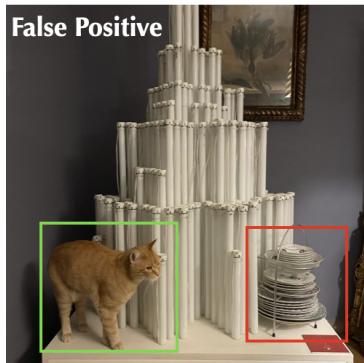


Figure 2.4: The object is there, but the predicted box has an IoU below the threshold.

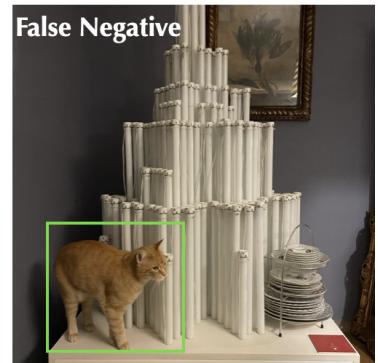


Figure 2.5: The object is there, and the model doesn't detect it. The ground truth object has no prediction.

Figure 2.6: Ground truth box is the green rectangle, while the predicted box is the red rectangle.

Let's now define the core metrics:

- **Precision:** Is a critical metric in model evaluation as it serves to quantify the accuracy of the positive predictions made by the model. It specifically assesses how well the model distinguishes true objects from false positives. In essence, precision provides insight into the model's ability to make positive predictions that are indeed accurate. A high precision score indicates that the model is skilled at avoiding false positives and provides reliable positive predictions.

$$Precision = \frac{TP}{TP + FP} \quad (2.2)$$

- **Recall:** also known as sensitivity or true positive rate, is another essential metric used in evaluating model performance, especially in object detection tasks. Recall measures the model’s capability to capture all relevant objects in the image. In essence, recall assesses the model’s completeness in identifying objects of interest. A high recall score indicates that the model effectively identifies most of the relevant objects in the data.

$$Recall = \frac{TP}{TP + FN} \quad (2.3)$$

We can define the **Average Precision (AP)** as the area under the precision-recall curve, which is obtained by plotting precision against recall for different confidence thresholds <sup>1</sup>. The AP is a single value that summarizes the model’s performance across different levels of precision and recall. It is calculated as follows:

$$AP = \int_0^1 P_\alpha(box_{gt}, box_{pred}) dR_\alpha \quad (2.4)$$

where  $P_\alpha$  is the precision at a given IoU threshold  $\alpha$  and  $R_\alpha$  is the recall at the same threshold. Precision signifies the accuracy of the model’s positive predictions, while recall quantifies the model’s ability to successfully identify all relevant objects. AP achieves a harmonious balance between false positive and false negatives, providing a comprehensive evaluation of the model’s performance.

## Mean Average Precision (mAP)

Mean Average Precision (mAP) extends the concept of Average Precision (AP) by providing a global summary of a model’s performance across multiple object classes and, in some cases, multiple localization thresholds. While AP evaluates the precision-recall trade-off for a single class — by varying the confidence threshold and computing the area under the resulting precision-recall curve — mAP averages these AP values to reflect the model’s overall detection capabilities.

$$mAP = \frac{1}{N} \sum_{i=1}^N AP_i \quad (2.5)$$

where  $N$  is the number of classes and  $AP_i$  is the Average Precision for class  $i$ .

In practice, several versions of mAP are commonly reported. mAP@0.5 refers to the mean AP calculated at a fixed Intersection over Union (IoU) threshold of 0.5. mAP@[0.5:0.95], computes the average AP across ten IoU thresholds ranging from 0.5 to 0.95 (in increments of 0.05), providing a more rigorous and comprehensive evaluation that accounts for both classification and localization accuracy.

$$mAP@50 = \frac{1}{N} \sum_{k=1}^N AP_K@50 \quad (2.6)$$

---

<sup>1</sup>The confidence threshold is a tunable parameter used to filter out low-confidence predictions in object detection tasks. Each predicted bounding box is assigned a confidence score, typically representing the estimated probability that the detection both contains an object and correctly identifies its class. Varying this threshold affects the trade-off between precision and recall: lower thresholds increase recall but may introduce more false positives, whereas higher thresholds improve precision at the cost of reduced recall.

$$mAP@[50 : 95] = \frac{1}{N(95 - 50)} \sum_{\alpha=50}^{\alpha=95} \sum_{k=1}^N AP_K @ \alpha \quad (2.7)$$

where  $N$  is the number of classes and  $\alpha$  is the IoU threshold.

## 2.4 Transfer learning

Transfer learning is a machine learning technique in which knowledge gained through one task or data set is used to improve the performance of models on another related task and/or on a different data set. In other words, it uses what has been learned in one setting to improve generalization in another. The applications of transfer learning in deep learning are very interesting because models need a big amount of labeled data to learn and gain knowledge, and a lot of times such big datasets are not available.

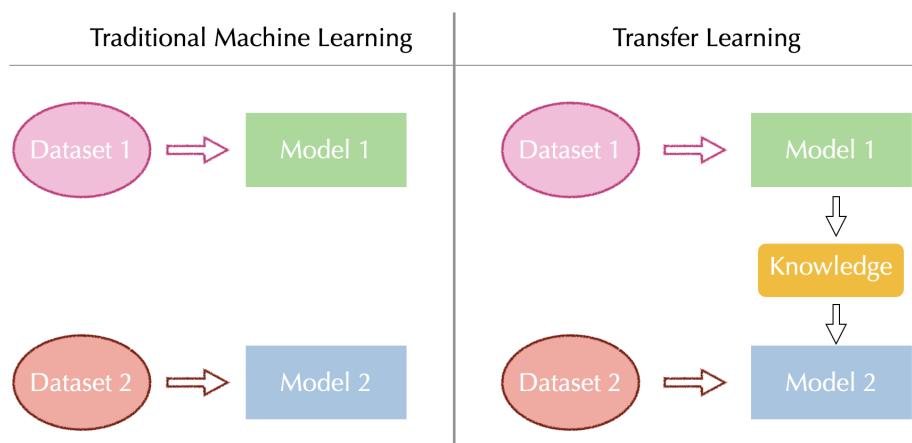


Figure 2.7

Traditional learning processes create a new model for each new activity based on the labeled data available. This is because traditional machine learning algorithms assume that training and test data come from the same feature space, so if the data distribution changes or if the trained model is applied to a new dataset, users need to retrain a newer model from scratch, even if they want to carry out an activity similar to that of the first model. Transfer learning algorithms, on the other hand, take already trained models or networks as a starting point, then apply that model's knowledge, gained in a task or initial source data to a new, but related activity or target data.

Depending on the amount of data available and the similarity between source domain and target domain, there are different transfer learning strategies:

- Feature extraction: freezes most of the model, training only the head. It is useful with very little data and similar domains.
- Partial fine-tuning: only the lowest layers freeze. Suitable for moderately different domains.
- Complete fine-tuning: only the lowest layers freeze. Suitable for moderately different domains.

In the context of medical imaging, transfer learning has shown promising results. Models pre-trained on large natural image datasets (e.g. ImageNet, COCO) can learn general low-level features (such as edges, textures, and shapes) that are also present in medical images. Fine-tuning such models on smaller labeled medical datasets allows for improved performance even with limited annotated data, making transfer learning a valuable tool in the medical AI pipeline.

The following chapters will explore specific cases of transfer learning applied to medical image datasets, comparing various pre-training strategies and model architectures.

## 2.5 Curriculum Learning

Curriculum Learning is a training strategy inspired by the way humans learn, initially tackling simple tasks and then moving on to more complex ones. Introduced by Bengio et al. in 2009 [?], this approach proposes to organize the training of a model according to an ordered sequence of examples, selected based on a difficulty criterion.

Instead of presenting all data simultaneously and in random order, Curriculum Learning involves the progressive selection of subsets of data, which are introduced in a gradual manner during training. This controlled sequence allows the model to first acquire the simplest and most robust regularities, and then refine its predictive ability by tackling more complex cases.

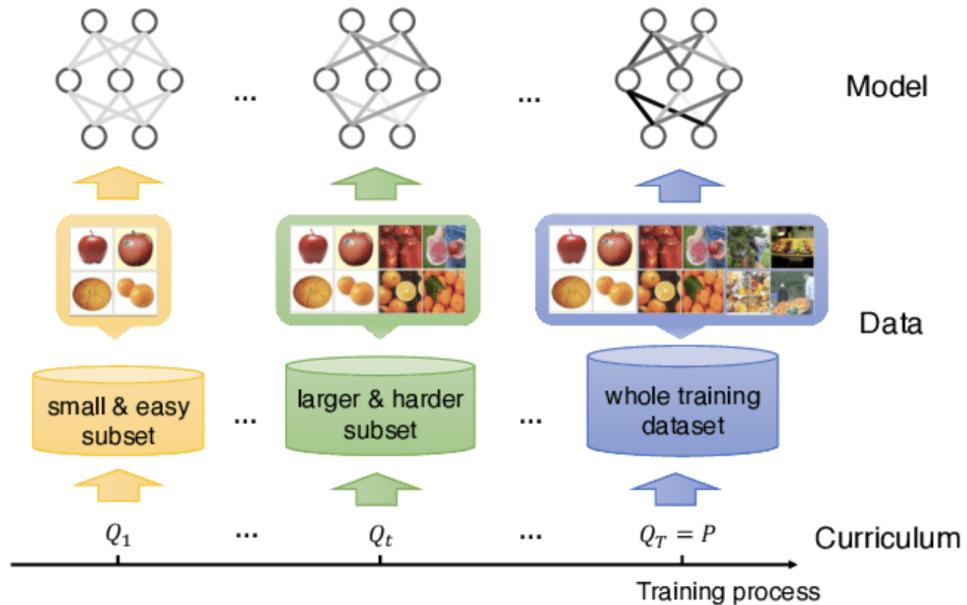


Figure 2.8

An effective curriculum requires defining three basic components:

- Difficulty criterion: is the mechanism through which a difficulty level is assigned to each example. It can be defined explicitly on the basis of known characteristics (eg geometric properties, data quality, object dimensions), or it can emerge automatically, for example by exploiting the loss obtained by the model during the early eras or the uncertainty of predictions.

- Pacing: is the function that regulates the pace at which the most difficult examples are introduced into training. Pacing can be defined a priori (e.g. by adding a certain number of examples to each epoch), or determined dynamically, for example by evaluating the performance of the model on a validation subset.
- Presentation order: consists of the construction of a sequence of data (or subsets) to be presented to the model. This order can be fixed or updated during training, and strongly affects the effectiveness of the curriculum.

This type of approach has proven useful in scenarios where the data has significant variability, contains noisy or unbalanced examples, or when it is desired to accelerate the convergence of optimization. Furthermore, Curriculum Learning can help prevent the model from learning abnormal characteristics or statistical noise too early, thus improving generalization.

## Applications

In the visual field, a curriculum can be constructed in many ways: for example, a classification model can initially be trained on high-quality images and well-defined content, and then tackle more complex examples such as deformed, moving, partially occluded objects or present in unfavorable lighting conditions. In other contexts, such as the regression of physical quantities from images, the difficulty may depend on the structural variability of the visual content or the complexity of the information flow required for accurate prediction.

## Practical considerations

Despite the benefits observed in numerous works, the effectiveness of Curriculum Learning is highly dependent on

- A correct definition of the difficulty;
- An appropriate pacing;
- the coherence of the curriculum with the final task.

In the absence of a clear criterion for distinguishing easy examples from difficult ones, or if pacing is poorly calibrated, the approach can even worsen performance compared to standard training.

However, if well designed, Curriculum Learning can be a powerful tool for controlling training, especially in contexts where data distribution is uneven, the useful signal is weak or learning can benefit from gradual progression.

# Chapter 3

## Hadronic Collisions and Jet Reconstruction

### 3.1 High energy collisions in hadronic accelerators

In a hadronic collider, such as the LHC or the Tevatron, beams of hadrons—primarily protons and antiprotons—are collided at extremely high energies. Unlike electrons, protons are not elementary particles; they possess a complex internal structure composed of quarks and gluons. This internal structure is described by the parton distribution functions (PDFs), denoted in Fig. 3.1 as  $f_i(x_j, \mu_F)$ , where  $i$  indicates the parton type (quark or gluon), and  $j$  refers to the beam. The PDF represents the probability of finding a parton of type  $i$  carrying a fraction  $x_j$  of the proton's momentum at the time of collision.

Therefore, when two protons  $P_1$  and  $P_2$  collide, the interaction does not occur between the protons themselves, but rather between their constituents—the partons—with momenta  $x_j P_j$ . The partons that do not participate in the hard scattering process are referred to as underlying events, as illustrated in the figure.

When high-energy quarks are produced in the final state, they emit additional partons through a cascading process known as parton showering. This process continues until the energy scale drops sufficiently, at which point the partons recombine into observable hadrons in a process called hadronization.

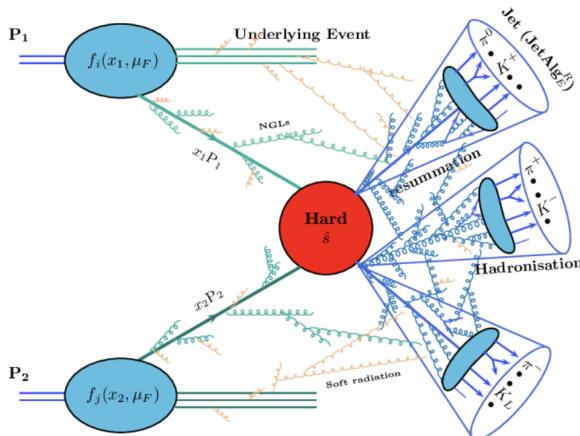


Figure 3.1: Schema di un collisione adronica. I protoni si scontrano e producono una cascata di particelle, che si trasformano in jet di particelle.

## Jets

In order to reconstruct the primary event that occurred before the parton shower and hadronization, the strategy is to measure the energy of the final-state particles using calorimeters, and then group them based on their transverse momentum and direction. This procedure leads to the formation of what are known as jets. Jets are essentially the result of a clustering algorithm that groups final-state particles with similar transverse momentum and direction, allowing us to trace back to the primary parton-level event that initiated the particle cascade.

## 3.2 Useful variables to describe an hadronic process

In collider physics, the kinematic properties of particles are typically described using variables that are well suited to the cylindrical geometry of the detectors and to the characteristics of the collisions. Among these, the most commonly used are the transverse momentum  $p_T$ , the azimuthal angle  $\phi$ , and the pseudorapidity  $\eta$ .

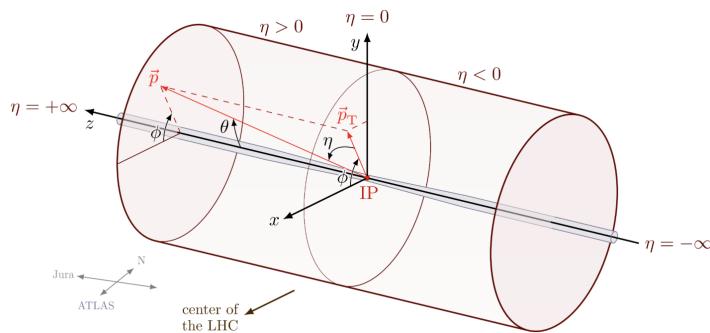


Figure 3.2: Kinematic variables used in collider physics. The transverse momentum  $p_T$  is the component of momentum perpendicular to the beam axis,  $\phi$  is the azimuthal angle, and  $\eta$  is the pseudorapidity.

### $p_T$ - transverse momentum

The transverse momentum  $p_T$  is defined as the component of a particle's momentum that is perpendicular to the beam axis (commonly referred to as the z-axis in collider experiments). Mathematically, it is given by:

$$p_T = \sqrt{p_x^2 + p_y^2} \quad (3.1)$$

where  $p_x$  and  $p_y$  are the components of the momentum in the plane transverse to the beam (also called the transverse plane).

This quantity is especially important in hadron colliders, where the incoming protons travel along the z-axis, but the exact longitudinal momentum of the interacting partons is unknown on an event-by-event basis. However, since the protons travel toward each other along the beam axis, the net transverse momentum of the system before the collision is essentially zero. As a result, conservation of momentum implies that the total transverse momentum of the final-state particles should also be zero, up to detector resolution and effects from invisible particles.

For this reason,  $p_T$  is a Lorentz-invariant quantity under boosts along the beam axis, and provides a frame-independent way to study the dynamics of the collision. It is also widely used in searches for new physics, where an imbalance in the total transverse momentum (often called missing transverse energy, or MET) can signal the presence of non-interacting or undetected particles.

### $\phi$ - azimuthal angle

The azimuthal angle  $\phi$  describes the direction of a particle in the transverse plane, i.e., the plane perpendicular to the beam axis (the  $x$ - $y$  plane if the beam is aligned along the  $z$ -axis). It is defined as the angle between the particle's transverse momentum vector and a fixed reference axis (typically  $x$ -axis) measured counterclockwise:

$$\phi = \tan^{-1} \left( \frac{p_y}{p_x} \right) \quad (3.2)$$

The azimuthal angle takes values in the range  $[-\pi, \pi]$  or  $[0, 2\pi]$ , depending on convention. This variable is invariant under longitudinal boosts.

### $\eta$ - pseudorapidity

The pseudorapidity  $\eta$  is a measure of the angle of a particle relative to the beam axis, defined as:

$$\eta = -\ln \left( \tan \left( \frac{\theta}{2} \right) \right) \quad (3.3)$$

where  $\theta$  is the polar angle of the particle with respect to the beam axis.

## 3.3 Algorithms for jet reconstruction

Jet reconstruction algorithms are essential for identifying and grouping particles produced in high-energy collisions into jets, which represent the hadronic remnants of the original partons. These algorithms can be broadly categorized into two main types: **cone algorithms** and **sequential clustering algorithms**.

### 3.3.1 Cone Algorithms

Cone algorithms assume that particles in jets will show up in conical regions and thus they cluster based on  $\eta - \phi$  space, resulting in jets with rigid circular boundaries. Most of the cone algorithms are iterative cones (IC). In such algorithms, a seed particle  $i$  sets some initial direction, and one sums the momenta of all particles  $j$  within a circle of radius  $R$  around  $i$  in azimuthal angle  $\phi$  and pseudorapidity  $\eta$ , i.e. taking all  $j$  such that

$$\Delta R_{ij}^2 = (y_i - y_j)^2 + (\phi_i - \phi_j)^2 < R^2 \quad (3.4)$$

where  $y_i$  and  $y_j$  are the pseudorapidities of particles  $i$  and  $j$ , respectively, and  $\phi_i$  and  $\phi_j$  are their azimuthal angles. The total momentum of the jet is then calculated as the sum of the momenta of all particles within this cone. The direction of the jet is then recalculated as the weighted average of the momenta of all particles within the cone, and the process is repeated until no more particles can be added to the jet.

### 3.3.2 Sequential Clustering Algorithms

Gli algoritmi a ricombinazione sequenziale utilizzano informazioni geometriche ed energetiche per aggregare le particelle che potenzialmente arrivano dallo stesso partone, ricostruendo così la storia del getto.

#### $k_T$ Algorithm

The  $k_T$  algorithm is a sequential clustering algorithm that uses the concept of distance between particles to group them into jets. The distance between two particles  $i$  and  $j$  is defined as:

$$d_{ij} = \min(p_{T,i}^2, p_{T,j}^2) \frac{\Delta R_{ij}^2}{R^2} \quad (3.5)$$

while the distance between a particle  $i$  and the beam axis is defined as:

$$d_{iB} = p_{T,i}^2 \quad (3.6)$$

For each particle, the algorithm calculates the minimum distance between  $d_{ij}$  and  $d_{iB}$ .

$$d_{min} = \min(d_{ij}, d_{iB}) \quad (3.7)$$

If  $d_{min}$  is smaller than a predefined threshold, the particle is added to a jet; otherwise, it is considered a standalone particle. The algorithm proceeds iteratively, recalculating the distances after each clustering step until all particles are grouped into jets or no more particles can be clustered.

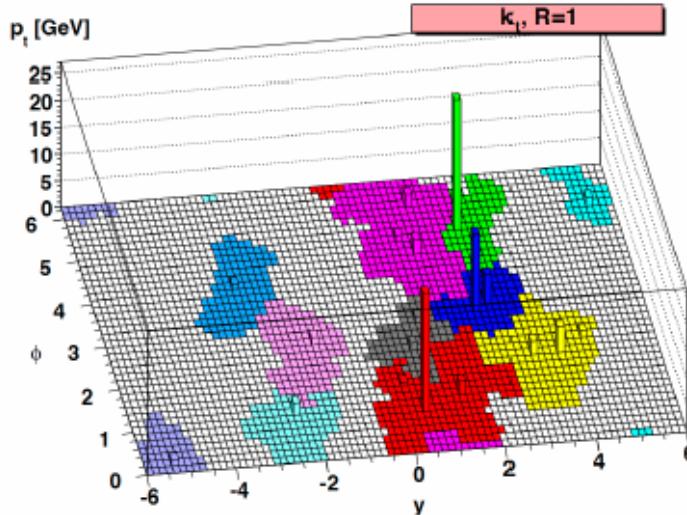


Figure 3.3: Schematic representation of the  $k_T$  algorithm. The algorithm iteratively clusters particles based on their transverse momentum and distance in  $\eta-\phi$  space, forming jets.

#### Anti- $k_T$ Algorithm

The anti- $k_T$  algorithm is a variant of the  $k_T$  algorithm that uses a different distance measure to cluster particles. It is designed to produce jets with more uniform shapes and

is particularly effective in high-energy collisions where jets can be closely spaced. The distance between two particles  $i$  and  $j$  is defined as:

$$d_{ij} = \min\left(\frac{1}{p_{T,i}^2}, \frac{1}{p_{T,j}^2}\right) \frac{\Delta R_{ij}^2}{R^2} \quad (3.8)$$

while the distance between a particle  $i$  and the beam axis is defined as:

$$d_{iB} = \frac{1}{p_{T,i}^{2\beta}} \quad (3.9)$$

The algorithm proceeds similarly to the  $k_T$  algorithm, calculating the minimum distance between particles and the beam axis, and clustering them into jets based on this distance. The key difference is that the anti- $k_T$  algorithm tends to produce jets with more uniform shapes, making it more suitable for high-energy collisions where jets can be closely spaced.

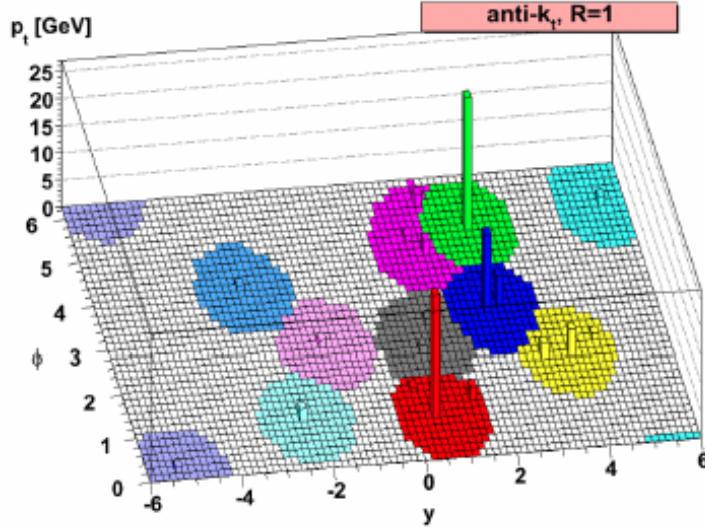


Figure 3.4: Schematic representation of the anti- $k_T$  algorithm. The algorithm iteratively clusters particles based on their transverse momentum and distance in  $\eta-\phi$  space, forming jets with more uniform shapes.

### Cambridge/Aachen Algorithm

The Cambridge/Aachen algorithm is another sequential antik<sub>T</sub> clustering algorithm that uses a different distance measure to cluster particles. It is designed to produce jets with more uniform shapes and is particularly effective in high-energy collisions where jets can be closely spaced. The distance between two particles  $i$  and  $j$  is defined as:

$$d_{ij} = \Delta R_{ij}^2 \quad (3.10)$$

while the distance between a particle  $i$  and the beam axis is defined as:

$$d_{iB} = 1 \quad (3.11)$$

Physically, the differences between the three algorithms are contained in the momentum weighting. For the  $k_T$  algorithm, the weighting is done to preferentially merge

constituents with low transverse momentum with respect their nearest neighbours. For the antik<sub>T</sub>, the weighting is done so as to preferentially merge constituents with high transverse momentum with respect to their nearest neighbours. The C-A algorithm relies only on distance weighting with no  $k_T$  weighting at all.

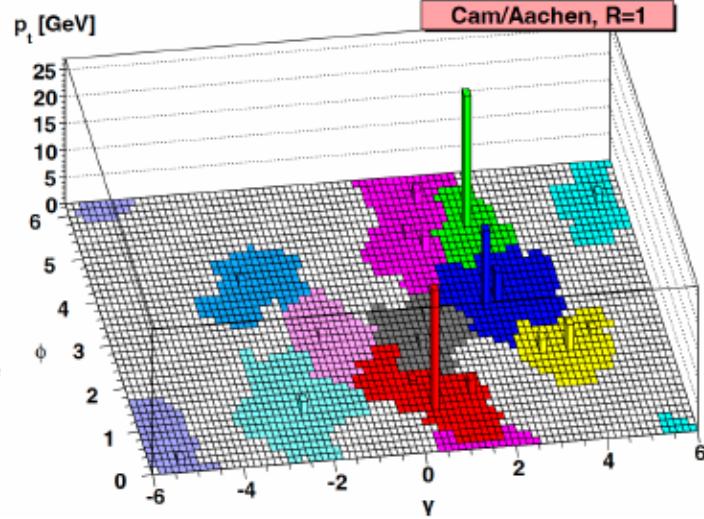


Figure 3.5

# Chapter 4

## Synthetic Particle Dataset

### 4.1 Blob Generation

In primo luogo, i dati sono stati letti e convertiti in una struttura uniforme, assicurando che ogni evento avesse lo stesso numero massimo di particelle considerate. Le particelle di ciascun evento sono state quindi convertite in oggetti cinematici (pseudo-jet) e raggruppate in jet mediante l'algoritmo di clustering anti- $k_T$  con raggio  $R = 1.0$ . Per ciascun jet sono stati identificati i costituenti e ne sono state estratte le coordinate ( $p_T, \eta, \phi$ ). Successivamente, ogni jet è stato ricentrato nello spazio  $(\eta, \phi)$  in modo che il baricentro di momento trasverso coincidesse con il centro dell'immagine. A questo stadio sono state applicate operazioni di data augmentation, come riflessioni e rotazioni, per aumentare la varietà delle configurazioni spaziali e ridurre la dipendenza del modello da particolari orientazioni geometriche. Una volta preprocessati, i costituenti di ciascun jet sono stati trasformati in un'immagine bidimensionale mediante discretizzazione (pixelation) dello spazio  $(\eta, \phi)$ , generando una mappa in cui l'intensità di ciascun pixel è proporzionale al momento trasverso depositato in quella cella. Le immagini ottenute da un gruppo prefissato di jet sono state sommate pixel per pixel, e la somma è stata trasformata tramite la funzione  $\log 1 + x$  per comprimere la gamma dinamica dei valori, evitando che pochi pixel ad altissimo  $p_T$  dominassero la scala dell'immagine. Il risultato di questa operazione è il blob primario, un'immagine composita che rappresenta la distribuzione media di energia trasversa di un insieme di jet.

Nel dataset JetClass sono presenti dieci diverse classi di processi fisici. In questo studio, l'analisi è stata ristretta a tre classi di interesse:

- eventi di produzione di bosone di Higgs con decadimento  $H \rightarrow b\bar{b}$ ;
- eventi  $t\bar{t}$  (produzione di coppie di quark top);
- eventi QCD generici, che costituiscono il fondo principale per i processi sopra citati.

---

**Algorithm 2** Generation of the *primary blob* from the JetClass dataset

---

- 1: **Input:** ROOT files from the JetClass dataset
  - 2: **Output:** Primary blob in image format
  - 3: **Event loading and preparation:**
    - Read ROOT files and extract  $(p_T, \eta, \phi)$  of each particle
    - Pad events to a fixed number of particles
  - 4: **Jet clustering:**
    - Convert particles into PseudoJet objects
    - Apply the anti- $k_t$  algorithm with  $R = 1.0$
    - Select jets and their constituents
  - 5: **Pre-processing:**
    - Center jets so that the  $p_T$ -weighted centroid is at the origin
    - Apply reflections and rotations (data augmentation)
  - 6: **Image conversion:**
    - Discretize the  $(\eta, \phi)$  space into a pixel grid
    - Assign each pixel the total  $p_T$  in its area
  - 7: **Primary blob construction:**
    - Sum the images from a fixed number  $N$  of jets
    - Apply the  $\log(1 + x)$  transformation
    - Save the resulting image as the *primary blob*
-

# Chapter 5

## Medical Dataset

### 5.1 NLST - National Lung Screening Trial Dataset

The aggressive and heterogeneous nature of lung cancer has thwarted efforts to reduce mortality from this cancer through the use of screening. The advent of low-dose helical computed tomography (CT) altered the landscape of lung-cancer screening, with studies indicating that low-dose CT detects many tumors at early stages. The National Lung Screening Trial (NLST) was conducted to determine whether screening with low-dose CT could reduce mortality from lung cancer. The NLST was a large, multicenter, randomized controlled trial that enrolled over 53,000 participants at high risk for lung cancer. Participants were randomly assigned to receive either low-dose CT scans or standard chest X-rays. The primary outcome was lung cancer mortality, with secondary outcomes including the detection of early-stage lung cancer and the impact of screening on overall mortality.

Radiologists at the screening centers reviewed the images obtained at each of the three annual screening exams to check for signs of lung cancer.

The image review was made without reference to any historical images. The radiologist recorded information about all visible abnormalities and assigned a preliminary screening result. A positive screening result (suspicious for lung cancer) was assigned if any non-calcified nodules or masses  $\geq 4\text{mm}$  in diameter were noted or if any other abnormalities were judged suspicious for lung cancer by the radiologist.

#### 5.1.1 Dataset Pruning

The NLST dataset contains chest CT scans where each scan comprises multiple ( $\sim 143$ ) axial slices. Notably, not all slices include tumor lesions. The dataset was then pruned by filtering slices based on a provided CSV file that identifies slices containing clinically relevant findings. After pruning, approximately 9,000 slices remain with malignant lesions with  $\geq 4\text{mm}$  in diameter. However, benign tumor slices lack bounding box annotations, which presents a limitation for detection tasks.

The annotations are in the form of a CSV file that contains the following columns:

- **PID:** Unique identifier for each patient.
- **Slice Number:** The specific slice within the CT scan, which is also the name of the file.
- **CT filter:** Indicates the type of filter applied to the CT scan.

- **x, y, width, height:** Coordinates of the bounding box around the lesion, x and y represent the top-left corner of the bounding box, while width and height represent its dimensions.

### 5.1.2 Extraction of the pixel array

The CT images are in the dicom format, which is a standard format for medical imaging data. Each CT scan consists of multiple slices, and each slice is represented as a 2D image. The pixel values in these images are typically stored in Hounsfield units (HU), which represent the radiodensity of the tissue. A dicom file contains:

- **Pixel Array:** The pixel array is a 2D array of pixel values that represent the intensity of the image. Each pixel value corresponds to a specific location in the image and is typically stored as a 16-bit integer.
- **Metadata:** The metadata contains information about the image, such as the patient's name, the acquisition date, the image orientation, pixel spacing, and slice thickness. This information is essential for interpreting the image correctly.

### 5.1.3 Image Metadata

The metadata of a CT image contains important information about the image acquisition parameters and the physical characteristics of the image. The following are some key metadata fields commonly found in CT images:

- **Pixel Spacing (mm):** The physical distance between adjacent pixels in the x and y directions, typically measured in millimeters.
- **Slice Thickness (mm):** The thickness of each slice in the z direction, also measured in millimeters.
- **Field of View (FOV) (cm):** The physical dimension of the area captured in the image, typically measured in centimeters.
- **Image Orientation:** The orientation of the image, which can be specified using a combination of row and column vectors.
- **Image Position:** The position of the image in 3D space, typically specified using x, y, and z coordinates.

## 5.2 DLCS - Duke Lung Cancer Screening Dataset

### 5.2.1 Extraction of the 2D slices

### 5.2.2 Dataset Pruning

## 5.3 Pre processing the images

Medical image preprocessing is essential to train accurate tumor detection models. Raw scans often contain noise, inconsistent resolutions, or irrelevant regions that can mislead

algorithms. Our pipeline standardizes the data through key steps: resampling ensures uniform resolution, windowing highlights tumor-relevant contrasts, and masking removes healthy tissue to focus analysis. These optimizations allow the model to learn meaningful patterns, improving its ability to detect tumors reliably across diverse clinical cases.

## Resampling

Since the CT images were acquired using different machines and protocols, the pixel spacing (i.e., the physical dimension per pixel or voxel) is not consistent across all scans, which in turn affects the apparent size of lesions. To standardize, we apply resampling to ensure that all voxels represent the same physical size in centimeters. Consequently, the pixel dimensions of the images may change. To determine a suitable uniform voxel size, the pixel spacing was extracted from each image and computed the mean (x and y spacing is equal), resulting in:

$$(\text{pixel spacing})_x = (\text{pixel spacing})_y = 0.623\text{mm}$$

Then, using this target spacing, all images are resampled accordingly. It can be achieved using SimpleITK, a medical imaging library, which provides a robust implementation for resampling. For example:

---

**Algorithm 3** Resampling of DICOM Slices

---

```
1: Input:
    • Directories  $D$  containing DICOM slices
    • Target pixel spacing in  $x$  and  $y$ :  $t_x = t_y = 0.623$  mm
2: Output: Resampled PNG slices stored in output directory  $D_{out}$ 
3:  $P \leftarrow \text{ListFiles}(D, ".dcm")$ 
4: for each file  $p \in P$  do
5:    $I \leftarrow \text{ReadImage}(p)$ 
6:    $(s_x, s_y, s_z) \leftarrow \text{GetSpacing}(I)$ 
7:    $(n_x, n_y, n_z) \leftarrow \text{GetSize}(I)$ 
8:    $t_z \leftarrow s_z$ 
9:    $T \leftarrow (t_x, t_y, t_z)$ 
10:   $N[0] \leftarrow \text{round}(n_x \cdot s_x / t_x)$ 
11:   $N[1] \leftarrow \text{round}(n_y \cdot s_y / t_y)$ 
12:   $N[2] \leftarrow 1$ 
13:   $\text{resampler} \leftarrow \text{ResampleImageFilter}()$ 
14:   $\text{resampler}.SetSize(N)$ 
15:   $\text{resampler}.SetOutputSpacing(T)$ 
16:   $\text{resampler}.SetOutputOrigin(\text{GetOrigin}(I))$ 
17:   $\text{resampler}.SetOutputDirection(\text{GetDirection}(I))$ 
18:   $\text{resampler}.SetInterpolator(sitkBSplineResampler)$ 
19:   $I_r \leftarrow \text{resampler}.Execute(I)$ 
20:   $I_{2D} \leftarrow \text{Extract}(I_r, (N[0], N[1], 0), (0, 0, 0))$ 
21:   $A \leftarrow \text{GetArrayFromImage}(I_{2D})$ 
22:   $\text{SavePNG}(A, D_{out}, \text{basename}(p))$ 
23: end for
```

---

## Padding

After resizing, all images now share a uniform pixel spacing, but their pixel dimensions differ. Since YOLOv8—the model employed in this thesis—requires square images of identical pixel size as input, a padding step is necessary.

The padding is applied so that the original image is placed in the top-left corner of a black canvas, whose side length corresponds to the nearest multiple of 32 greater than or equal to the largest resampled image dimension:

- NLST: 704x704 pixel, con uno spacing di 0.623 mm x 0.623 mm
- DLCS: 736x736 pixel, con uno spacing di 0.7 mm x 0.7 mm

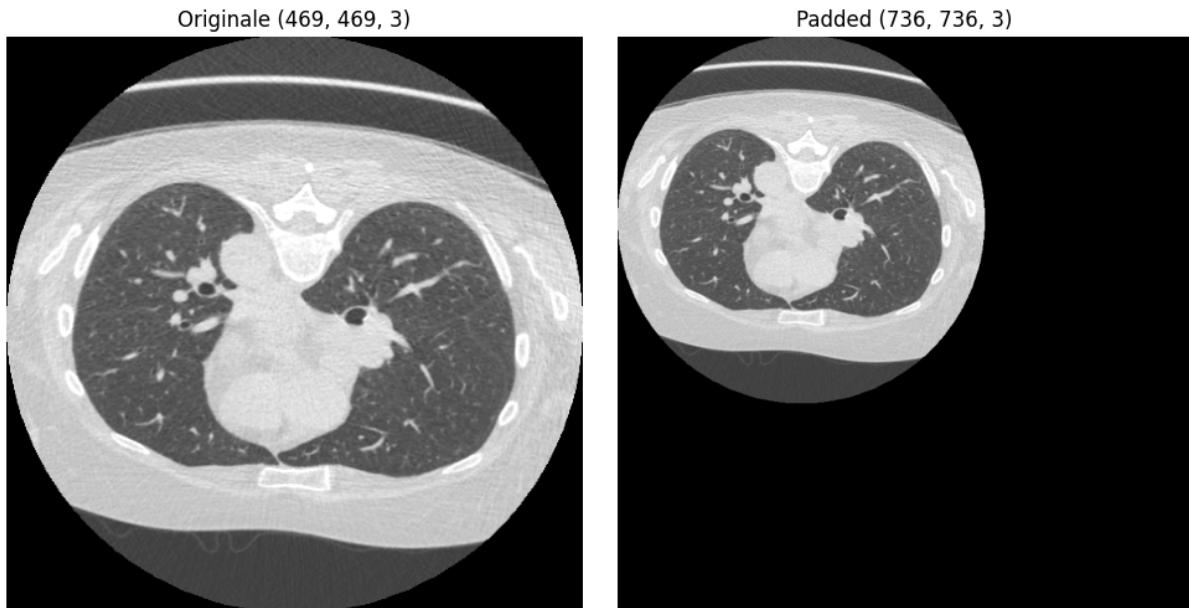


Figure 5.1

## Windowing

As described in Chapter 1, the range of intensity values in a CT scan depends on the tissue absorption coefficients, typically spanning approximately from -1000 to +3000 Hounsfield Units (HU). Because this range does not have fixed minimum or maximum limits, directly visualizing the full range is not practical, especially when converting images to formats like PNG, which support only a limited number of intensity levels. To effectively visualize relevant anatomical structures, we apply a process called windowing, which restricts the displayed intensity values to a specific range that highlights the tissues of interest. In this case, to emphasize lung structures, we set the window center at -600 HU and the window width at 1700 HU, values that are well established in the literature and illustrated in the accompanying figure. This means that only intensity values within the window boundaries—calculated as:

$$\text{window minimum} = -600 - \frac{1700}{2} = -1450$$

$$\text{window maximum} = -600 + \frac{1700}{2} = +250$$

—are displayed linearly. Any values outside this range are clipped to the nearest boundary value to avoid distortion. This windowing technique is applied consistently to all images from both datasets, resulting in enhanced visualization of pulmonary structures, as shown below:



(a) Standard PNG conversion without windowing (0–255 mapping).



(b) PNG conversion with windowing (center = -600 HU, width = 1700 HU), enhancing lung structures.

Figure 5.2: Comparison between standard intensity mapping and windowing applied to CT slices. The windowing technique improves contrast by focusing on relevant intensity ranges.

The following figure illustrates the step-by-step processing of a representative CT slice. On the left, the original image is shown with its native dimensions and pixel spacing. In the middle, the image after resampling is presented, including updated spatial resolution parameters. Finally, on the right, the windowing technique is applied to enhance the visualization of pulmonary structures by adjusting the intensity range. Dimensions and spacing values are displayed above each image for clarity.

## RBG Windowing

### 5.3.1 Masking

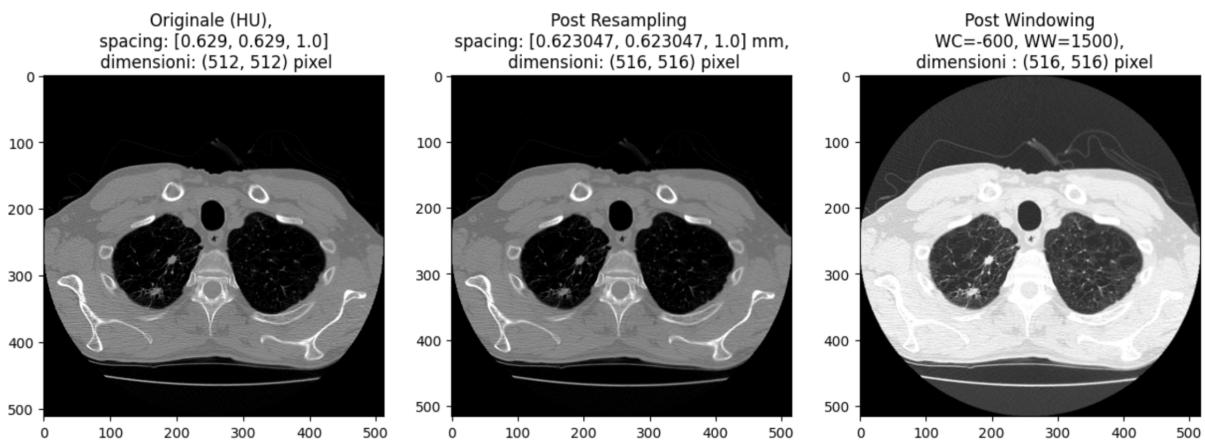


Figure 5.3

# **Chapter 6**

## **Dataset creation and Training Strategies**

### **6.1 YOLO dataset**

### **6.2 Curriculum Learning**

# **Chapter 7**

## **Training Results**

**7.1 Medical Images**

**7.2 Jet Images - Curriculum Learning**

**7.3 Transfer Learning in a data scarcity regime**

**7.4 Transfer learning in a normale data regime**

# Conclusion