

UNIVERSITÀ DEGLI STUDI DI MILANO-BICOCCA

Department of Physics "Giuseppe Occhialini"

Inter-University Master's Degree Programme
Artificial Intelligence for Science and Technology

Lung Nodule Detection and Explainability in CT Scans

Andrea Borghesi

REGISTRATION NR 916202

SUPERVISOR Prof. Pietro Govoni

CO-SUPERVISOR Dr. Simone Gennai

Academic Year 2024/2025

Contents

1 Introduction **1**
- 1.1 Problem Statement and Motivation . . . 1
- 1.2 Research Challenges . . . 2
- 1.3 Research Questions and Objectives . . . 3
- 1.4 Thesis Structure . . . 4

2 Background **5**
- 2.1 Fundamentals of Object Detection . . . 5
  - 2.1.1 One-Stage vs Two-Stage Detectors . . . 7
  - 2.1.2 Anchor-Based vs. Anchor-Free . . . 10
- 2.2 RetinaNet . . . 12
- 2.3 Faster R-CNN . . . 13
- 2.4 Evaluation Metrics: Average Precision and Average Recall . . . 16
- 2.5 Explainability in Deep Learning . . . 16
  - 2.5.1 Class Activation Maps (CAM) . . . 16
  - 2.5.2 Adaptations for Object Detection . . . 17

3 Data and Preprocessing **19**
- 3.1 Datasets: NLST and DLCSD24 . . . 19
- 3.2 Preprocessing Pipeline . . . 19
- 3.3 Slice Selection Algorithms . . . 19
- 3.4 Data Augmentation Techniques . . . 20

4 Methodology **21**
- 4.1 Object Detection Architectures and Backbones . . . 21
- 4.2 Object Detection Pipeline . . . 21
  - 4.2.1 Loss functions . . . 21
- 4.3 Classification Head Extension . . . 22

4.4 Explainability Methods 22

5 Results **23**
5.1 Object Detection Performance Analysis 23
5.1.1 Dataset Characgteristics and Usage Strategy 23
5.1.2 Model Architectures and Training Details 24
5.2 Explainability Evaluation 25

6 Conclusions **27**
6.1 Future Directions 27

Chapter 1

Introduction

1.1 Problem Statement and Motivation

Lung cancer remains one of the leading causes of cancer-related mortality worldwide, with early detection being crucial for improving patient outcomes and survival rates. Computed Tomography (CT) screening programs, such as the National Lung Screening Trial (NLST), have demonstrated the potential to reduce lung cancer mortality through early identification of pulmonary nodules [1, 3]. However, the manual interpretation of CT scans is a time-intensive process that places significant burden on radiologists, while also being subject to inter-observer variability and potential oversight of small or subtle lesions.

Radiologists are often required to review hundreds of slices per patient, sometimes across dozens of patients in a single day. Under such conditions, cognitive fatigue can accumulate, potentially leading to decreased diagnostic precision, delayed reading times, or even missed findings, especially for low-contrast or small nodules that may appear on only a few slices [28, 29]. This challenge is further amplified in high-volume screening programs, where maintaining consistent accuracy over long hours is difficult even for experienced professionals.

Artificial intelligence (AI) based detection tools have the potential to alleviate this strain by automatically flagging suspicious regions of interest, enabling radiologists to focus attention on the most relevant slices. Such systems do not replace human expertise but can act as a second reader, improving sensitivity to subtle findings, reducing oversight caused by fatigue, and providing explainable visual cues that support decision-making and increase trust in automated recommendations [9]. On this note, it is important to mention that the use of AI in radiology is not

always well-accepted, as it has been shown by Liu et al. [18] that radiologists with a higher workload and lower AI-acceptance are more likely to experience burnout. Regardless, the integration of AI tools into clinical workflows has been shown to enhance diagnostic accuracy and efficiency, ultimately leading to better patient care and outcomes [10, 12]

Nevertheless, existing AI solutions are often limited by computational demands and lack transparency in their predictions, motivating the need for efficient and explainable detection methods specifically tailored to lung nodule identification in CT scans. This need is further reinforced by the recent European Union Artificial Intelligence Act, which establishes a regulatory framework that emphasizes transparency, accountability, and human oversight for AI systems—particularly in high-risk domains such as healthcare [6].

1.2 Research Challenges

Designing AI-based tools for lung nodule detection in CT scans involves navigating a complex landscape of technical and practical challenges that significantly limit the applicability of existing solutions in real-world clinical environments.

The most immediate challenge stems from the computational demands of state-of-the-art approaches. While three-dimensional convolutional neural networks can theoretically exploit the full volumetric context of CT scans to improve detection accuracy, their practical implementation reveals significant limitations. These models typically require substantial GPU memory, often exceeding 16GB, and demand extensive training times that can extend to days or weeks, even on high-end hardware such as A100 GPUs with 40GB of memory [32]. While inference times for individual patient scans may be manageable, the development and iterative improvement of such models becomes prohibitively expensive and time-consuming. This computational burden became particularly acute in our research setting where access to high-end hardware was limited and unreliable. Working with a T4 GPU (16GB), the memory constraints made 3D approaches largely infeasible for our experiments. While we had occasional access to an A100 through shared research infrastructure, the instability of access, environment limitations, and restricted interaction modes (Jupyter-only) severely hindered our exploratory research process. A few preliminary experiments to understand model behavior and optimal hyperparameters consumed weeks of our available computational time, making the iterative development essential for novel research practically impossible. Such computational requirements create a fundamental mismatch with research devel-

opment processes, where rapid experimentation and iteration are crucial for advancing the field. This computational constraint naturally leads to consideration of more efficient 2D approaches, which process individual CT slices rather than entire volumes. This approach significantly reduces memory requirements and training times, allowing for more agile development cycles, at the cost of losing correlations between slices that may be crucial for accurate nodule detection. Regardless, a 2D approach allows for better accessbility to healthcare professionals, as it can be run on standard laptops or workstations without the need for high-end GPUs.

Equally critical, yet often overlooked in the technical literature, is the challenge of explainability in object detection systems. The medical domain presents unique requirements for AI interpretability, driven by regulatory demands, clinical decision-making needs, and the fundamental requirement for physician trust and acceptance. However, most existing explainability methods were developed for image classification tasks [27, 2, 4, 13], where the goal is to explain a single prediction score for an entire image. Object detection fundamentally differs in that it produces structured outputs consisting of multiple bounding boxes, each with associated class probabilities and confidence scores. This structural difference creates significant technical obstacles for applying standard explainability techniques.

1.3 Research Questions and Objectives

The challenges outlined in the previous section give rise to three primary research questions that guide this thesis:

RQ1: How can 2D object detection approaches achieve clinically relevant performance for lung nodule detection within computational constraints? This question addresses the fundamental trade-off between computational efficiency and detection performance. While 3D approaches theoretically offer superior performance by exploiting volumetric context, their computational demands make them impractical for many research and deployment scenarios. We investigate whether 2D slice-based detection can achieve acceptable performance levels while operating within the memory and processing constraints of standard hardware configurations.

RQ2: How can explainability techniques be adapted for the structured outputs of object detection models? Standard explainability techniques are designed for classification tasks with single prediction scores, but object detection

produces complex structured outputs with multiple bounding boxes, confidence scores, and non-differentiable post-processing steps. This question explores how gradient-free Class Activation Map (CAM) methods can be modified to generate meaningful explanations for object detection predictions, addressing the technical challenges posed by structured outputs and non-differentiable operations.

To address these research questions, this thesis pursues the following specific objectives:

O1: Develop a computationally efficient 2D object detection pipeline for lung nodule detection Implement and compare multiple object detection architectures and their variants. Design preprocessing pipelines optimized for 2D slice-based analysis, including slice selection algorithms to maximize information content Achieve detection performance that demonstrates clinical viability while operating within <6GB GPU memory constraints

O2: Adapt CAM methods for object detection tasks Research and adapt CAM methods that can handle structured object detection outputs Implement comparison algorithms that can assess similarity between structured predictions for explainability evaluation Implement end-to-end explainable object detection pipeline that provides both predictions and corresponding explanation maps

O3: Demonstrate integration of detection and classification with explainability Extend the object detection pipeline with a classification head for detected regions Apply explainability methods to both detection and classification components Evaluate the clinical utility of multi-stage explainable predictions

O4: Provide comprehensive experimental validation Evaluate the complete system on the established medical imaging datasets NLST [22], DLCSD24 [31]. Compare performance against relevant baselines and alternative approaches Analyze computational efficiency and practical deployment considerations

1.4 Thesis Structure

Chapter 2

Background

2.1 Fundamentals of Object Detection

Object detection is one of the fundamental task in computer vision that involves localizing and classifying multiple objects within an image. Unlike image classification and other tasks that assign a single label to an individual image, object detection requires predicting a set of bounding boxes, each associated with a class label and a confidence score. The challenging nature of this output stems from the variable cardinality of such sets, as the number of objects in an image can vary significantly, leading to complex structured prediction problems.

To address these challenges in a principled manner, it is useful to describe object detection as a structured prediction problem. This framework not only clarifies the task requirements but also provides a common language for comparing different detection architectures and loss functions.

Let an image be denoted as $x \in \mathbb{R}^{H \times W \times C}$, where H and W are the spatial dimensions and C is the number of channels. The output of an object detector is a finite set of detections:

$$\mathcal{D} = \{(b_i, y_i, s_i)\}_{i=1}^{N},$$

where:

- $b_i = (x_i, y_i, w_i, h_i)$ represents the bounding box in pixel coordinates in COCO format (center position, width, height);
- $y_i \in \{1, \ldots, K\}$ is the predicted class label, with K the total number of object classes;

- $s_i \in [0, 1]$ is the confidence score, often interpreted as the estimated probability that b_i belongs to class y_i.

During training, ground-truth bounding boxes b_i^* are typically transformed into target regression parameters relative to reference boxes (anchors or proposals) as:

$$t_x = \frac{x^* - x_a}{w_a}, \quad t_y = \frac{y^* - y_a}{h_a}, \quad t_w = \log \frac{w^*}{w_a}, \quad t_h = \log \frac{h^*}{h_a},$$

where (x_a, y_a, w_a, h_a) are the anchor box parameters, and (x^*, y^*, w^*, h^*) are the corresponding ground-truth parameters.

The specific optimization objectives used for bounding box regression and classification are detailed in Section 4.2.1, following the description of the detection architectures.

The object detection task can thus be formalized as learning a function:

$$f_\theta : \mathbb{R}^{H \times W \times C} \to \mathcal{P}(\mathbb{R}^4 \times \{1, \ldots, K\} \times [0, 1]),$$

parameterized by θ, that maps an input image to a set of bounding box–label–score triples. The set-valued nature of $\mathcal{P}(\cdot)$ reflects that the number of detections varies between images.

This formalization underpins the taxonomy of detection architectures discussed in the following section, where we distinguish between one-stage and two-stage approaches, anchor-based and anchor-free formulations, and their respective training and inference paradigms.

Feature Pyramid Networks (FPN)

Before diving into the specific architectures, we introduce a widely used component in most object detection models: the Feature Pyramid Network (FPN).

Objects appear at widely varying scales, early CNN backbone layers produce high-resolution but semantically weak features (shallow layers), while deeper layers produce low-resolution but semantically strong features. FPNs fuse these to obtain semantically strong, multi-scale feature maps at multiple resolutions, enabling detectors to handle small and large objects efficiently with a single backbone pass (as opposed to costly image pyramids) [16].

Let $\{C_2, C_3, C_4, C_5\}$ be backbone feature maps with strides $\{4, 8, 16, 32\}$ w.r.t. the input. FPN builds $\{P_2, P_3, P_4, P_5\}$ by a top–down pathway and lateral merges

as shown in Figure 1.

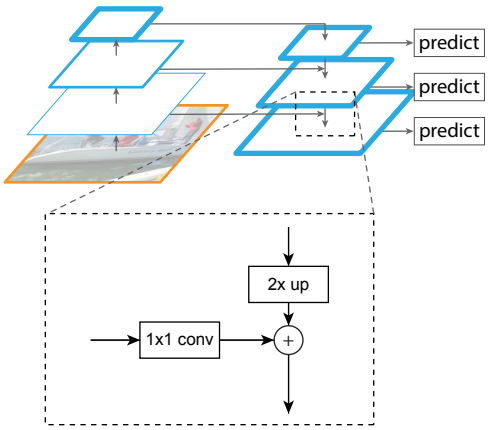

Figure 1: A building block illustrating the lateral connection and the top-down pathway, merged by addition [16]

2.1.1 One-Stage vs Two-Stage Detectors

Object detectors can be broadly categorized into *one-stage* and *two-stage* paradigms. Both ultimately predict a variable-sized set of detections $\mathcal{D} = \{(b_i, y_i, s_i)\}$, but they differ in how they structure intermediate computations and where they allocate capacity for localization vs. classification.

Two-stage detectors.

Two-stage methods decompose detection into proposal generation followed by proposal classification and box refinement. Let g_ϕ denote a proposal mechanism (e.g., a region proposal network operating on a shared backbone), which maps an image x to a set of candidate regions $\mathcal{P}$:

$$\mathcal{P} = g_\phi(x) = \{p_j\}_{j=1}^{M}.$$

A second function f_θ then classifies and refines these regions to produce final detections:

$$\mathcal{D} = f_\theta(x, \mathcal{P}).$$

Operationally, the pipeline is:

1. Backbone (optionally with a feature pyramid) extracts multi-scale features from x.
2. A proposal generator produces a sparse set of regions $\mathcal{P}$ (typically with non-maximum suppression to reduce redundancy).
3. Per-region features are pooled/aligned from the shared feature maps and passed to a detection head to output class scores and refined boxes.

The idea underlying two-stage detectors is to incentivise accuracy by first generating a manageable number of promising candidates, which can then be processed and refined with more complex per-region heads. This also allows for a more efficient use of computational resources, as the second stage can focus on a smaller set of regions rather than processing the entire image densely. But unfortunately, this approach introduces a fair amount of complexity requiring careful design.

One-stage detectors.

One-stage methods simplify the detection pipeline by predicting detections directly from the shared feature maps without an explicit proposal stage. This design is motivated by the desire to maximize throughput and reduce latency, particularly in real-time applications. This type of detection is often referred to as *dense predictions* because it involves predicting detections at every spatial location of the feature maps, often having to take into account hundreds of thousands of locations per image. Conceptually, it implements a direct mapping

$$\mathcal{D} = h_\psi(x),$$

where h_ψ produces, for each spatial location (and possibly for multiple predefined reference shapes), joint classification scores and box parameters. The pipeline is:

1. Backbone (optionally with a feature pyramid) extracts multi-scale features from x.
2. Lightweight prediction heads applied densely over feature maps output (b, y, s) candidates.
3. A single-stage post-processing (e.g., top-k filtering and non-maximum suppression) yields $\mathcal{D}$.

This design maximizes throughput and simplifies the graph, but operates under severe class imbalance, due to the fact that most locations tend to be related to a generic background class, and thus the model has to learn to distinguish between a very large number of background locations and a comparatively small number of foreground locations. This imbalance is often addressed with specialized loss functions, such as the Focal Loss [17], which down-weights easy-to-classify examples and focuses training on hard negatives. The precision of the predictions often relies on effective assignment and post-processing techniques. An alternative way of addressing this imbalance, ironically, is to use a two-stage approach, where the first stage takes care of separating the foreground (potential candidates) from the background.

Comparative characteristics. One might ask "Which paradigm is better then?" and the answer would be, as usual, "it depends". One-stage detectors are generally faster and more resource-efficient, and that is because they were intended to address the latency and throughput requirements of real-time applications. This comes at the cost of some accuracy that two-stage detectors can achieve with their more careful filtering mechanisms and (usually) more complex per-region heads that come at the cost of increased compatuational requirements and latency. And similarly one-stage detectors, they were designed to be accurate rather than fast.

Ideally one would like to have the best of both worlds, and we can find in literature a number of approaches that tries to reach two-stage performance with one-stage efficiency, such as the RetinaNet and YOLO [17, 24].

We can summarize the main differences between one-stage and two-stage detectors as follows:

- **Computation.** Two-stage: cost scales with the number of proposals M (feature pooling and per-RoI head). One-stage: cost scales with the number of feature locations (and reference shapes) processed densely.
- **Capacity allocation.** Two-stage allocates more parameters per candidate via the second-stage head; one-stage relies on shallow, shared heads and benefits from strong multi-scale features.
- **Class imbalance.** One-stage training observes extreme foreground/background imbalance due to dense supervision; two-stage partially mitigates this by filtering with proposals and sampling strategies.

- **Latency/throughput.** One-stage typically achieves lower latency and higher FPS; two-stage often offers higher accuracy at increased computational cost.
- **Post-processing.** Two-stage commonly applies suppression both after proposal generation and after final classification; one-stage applies it once on dense predictions (often per class).

2.1.2 Anchor-Based vs. Anchor-Free

Object detectors differ not only by staging (one- vs. two-stage) but also by how they parameterize candidate boxes. The two prevailing choices are *anchor-based* (predefined reference boxes) and *anchor-free* (direct box prediction without references). This design is largely orthogonal to the staging paradigm.

Anchor-based detectors

Let $\mathcal{A} = \{a_k\}_{k=1}^{K}$ be a set of predefined anchors tiled over feature maps. Each anchor $a_k = (x_a, y_a, w_a, h_a)$ serves as a reference against which a detector predicts offsets (t_x, t_y, t_w, t_h) to obtain a box:

$$x = x_a + w_a t_x \quad y = y_a + h_a t_y \quad w = w_a e^{t_w} \quad h = h_a e^{t_h}$$

Anchors are generated at multiple scales and aspect ratios per spatial location and per FPN level ℓ (with stride s_ℓ), e.g. scales $\{s_\ell \cdot \alpha\}$ and ratios $r \in \{1{:}2,\ 1{:}1,\ 2{:}1\}$. The number of anchors generated for each image tends to be large, but since we're considering the spatial locations of the feature maps, and not the original image, this number is still manageable, although it can still be in the order of hundreds of thousands, depending on the number of feature maps, stride, number of scales, aspect ratios and size of the image.

Training uses *IoU-based assignment*: for ground-truth boxes $\{b_i^*\}$, compute $M_{ik} = \mathrm{IoU}(b_i^*, a_k)$; for some given $\tau_{\text{pos}}, \tau_{\text{neg}} \in [0, 1]$ mark a_k positive if $M_{ik} \geq \tau_{\text{pos}}$ for some i, negative if $M_{ik} \leq \tau_{\text{neg}}$, and ignore otherwise.

We now have the anchors partitioned into $\mathcal{A}^+$ (positive), $\mathcal{A}^-$ (negative), and $\mathcal{A}^0$ (ignored). Positives receive a foreground class label y^* and regression targets t_k; negatives receive the background label; ignored anchors are excluded from loss

computation. Training thus optimizes

$$\mathcal{L} = \frac{1}{N_{\text{cls}}} \sum_{k \in \mathcal{A}^+ \cup \mathcal{A}^-} \ell_{\text{cls}}(\hat{c}_k, c_k) \; + \; \lambda \frac{1}{N_{\text{reg}}} \sum_{k \in \mathcal{A}^+} \ell_{\text{reg}}(\hat{t}_k, t_k),$$

where ℓ_{cls} is the classification/objectness loss and ℓ_{reg} (e.g., Smooth-L_1 or IoU-based) is applied *only* to positives. Negatives contribute *only* to classification as background; ignored anchors contribute to neither term.

Suppressing redundancy. As seen in this section, anchor-based detectors generate a large number of anchors, many of which are near-duplicates (they overlap with the same ground-truth boxes). Ideally we would like only one anchor, hence one detection, per ground-truth, therefore we need a post-processing procedure to remove these redundant detections.

This is typically done in two steps: we first keep the top-k detections per level/class, where k is an hyperparameter (pre-NMS step), and then we apply Non-Maximum Suppression (NMS) to remove overlaps, producing the final set $\mathcal{D}$ [23]. Given a set of competing detections $\mathcal{D} = \{(b_i, y_i, s_i)\}$, NMS iteratively selects the detection with the highest score s_i and removes all other detections that overlap with it above a threshold τ_{nms} (typically 0.5). The process continues until no detections remain above the threshold. As a result, NMS produces a final set of detections $\mathcal{D} = \{(b_i, y_i, s_i)\}$ that are non-overlapping and have the highest scores among the competing detections. This procedure is clearly non-differentiable, and thus it is not possible to backpropagate through it, hence we always backpropagate on the pre-NMS detections.

Anchor-free detectors.

Anchor-free methods remove the predefined anchor set $\mathcal{A}$: each feature-map location p is supervised directly. A common approach is the one used in FCOS, which assigns each location $p = (x_p, y_p)$ to at most one ground-truth box b^* if $p \in b^*$ (optionally restricted to a center region and a size range matched to level ℓ). After assignment, the location p is labeled positive with class y^* and regression targets given by distances to the four sides of b^*:

$$\ell_p = x_p - x^*_{\text{left}}, \quad t_p = y_p - y^*_{\text{top}}, \quad r_p = x^*_{\text{right}} - x_p, \quad b_p = y^*_{\text{bottom}} - y_p,$$

decoded at inference as $[x_p - \ell_p,\ y_p - t_p,\ x_p + r_p,\ y_p + b_p]$. Locations not assigned to any box act as background in the classification loss and do not contribute to regression. Many designs also predict a per-location quality term (e.g. "centerness") to down-weight ambiguous border locations. Alternative anchor-free formulations include keypoint-based representations (e.g. object centers or corners) with local offsets [30, 15, 5].

Anchor-free models avoid anchor tuning and can better handle unusual aspect ratios, but rely on effective assignment rules (inside-box, center sampling, size ranges) and robust post-processing.

Comparison and practical notes.

- **Design knobs** Anchor-based: choose scales/ratios, IoU thresholds, per-level priors. Anchor-free: choose assignment region, size ranges per level, and optional quality terms.
- **Label assignment** Anchor-based uses IoU with thresholds; anchor-free uses geometric inclusion at locations (often with center sampling). Advanced heuristics (e.g., ATSS/OTA) can be applied to either family [33, 7].
- **Computational profile** Similar dense heads; anchor-free can reduce memory and labels by removing large anchor sets.

2.2 RetinaNet

RetinaNet is a one-stage, anchor-based detector that combines a backbone with a feature pyramid and two lightweight, densely-applied heads (classification and regression). Its key contribution is the *focal loss*, which mitigates extreme foreground/background imbalance in dense prediction [17].

Architecture

A convolutional backbone (e.g., ResNet) feeds a feature pyramid $\{P_\ell\}$ (typically P_3–P_7). At each pyramid level ℓ with spatial size $H_\ell \times W_\ell$, and stride s_ℓ, a set of A anchors per location is tiled. Two subnetworks are applied at each level of the pyramid, producing per-anchor outputs:

- *Classification head:* a small CNN produces per-class logits $\hat{z}_{\ell,ij,a,c}$, using independent sigmoids (no softmax) over $c \in \{1, \ldots, K\}$.

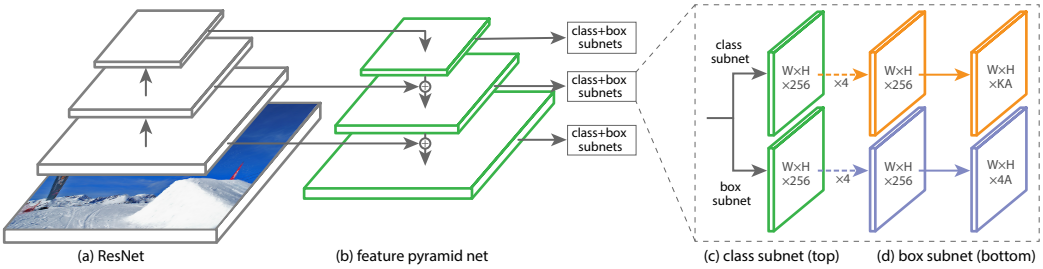

Figure 2: RetinaNet architecture overview [17]

- *Regression head:* a parallel CNN predicts box deltas $\hat{t}_{\ell,ij,a} = (\hat{t}_x, \hat{t}_y, \hat{t}_w, \hat{t}_h)$.

Once we obtain the structured outputs from the network, we apply the post-processing steps discussed in Section 2.1.2 to discard redundant detections.

Focal Loss. As previously mentioned, RetinaNet introduced the *focal loss* to address the extreme class imbalance in dense predictions. It essentially down-weights easy to classify examples and focuses training on hard negatives (Figure 3), which is particularly useful in medical imaging where the background class can dominate the training set. To do so it adds a modulating factor $(1 - p_t)^\gamma$ to the cross-entropy loss, with a focus parameter γ. The focal loss is hence defined as follows:

$$\mathrm{FL}(p_t) = -\,\alpha\,(1 - p_t)^\gamma\,\log(p_t),$$

with typical α=0.25, γ=2.

2.3 Faster R-CNN

Faster R-CNN is a two-stage, anchor-based detector that couples a Region Proposal Network (RPN) with a per-region classifier and regressor [25]. A shared backbone (we consider one using FPN) feeds both stages.

Region Proposal Network (RPN).

At each feature-map location and for A anchors $a_k = (x_a, y_a, w_a, h_a)$, the RPN predicts:

- objectness logits $\hat{o}_k$ (binary);

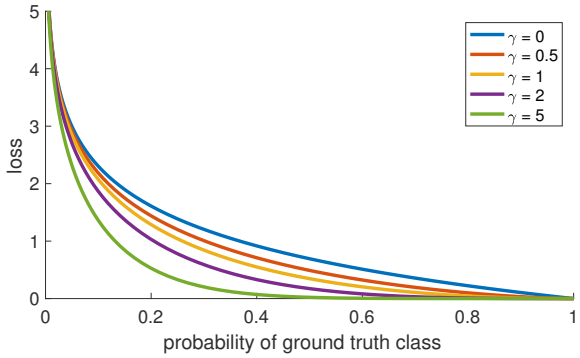

Figure 3: Focal loss at different γ values [17]. When $\gamma = 0$, it is equivalent to cross-entropy loss. As γ increases, the loss focuses more on hard examples, down-weighting easy ones.

- box deltas $\hat{t}_k = (\hat{t}_x, \hat{t}_y, \hat{t}_w, \hat{t}_h)$.

At this stage we do not have any class information, but rather we unify all classes into a single *object* against a *background* class, hence the binary classification. We call *objectness* the probability that an anchor contains an object, regardless of its initial class. Such probability is typically obtained by applying a sigmoid activation to the logits $\hat{o}_k$.
Anchors are assigned to ground-truth boxes by IoU thresholds: positives $\mathcal{A}^+$, negatives $\mathcal{A}^-$, ignored $\mathcal{A}^0$. Positives receive class 1 and regression targets t_k (standard parameterization),

$$t_x = \frac{x^* - x_a}{w_a}, \quad t_y = \frac{y^* - y_a}{h_a}, \quad t_w = \log \frac{w^*}{w_a}, \quad t_h = \log \frac{h^*}{h_a}.$$

At the end of the RPN, we filter the proposals by objectness score $\hat{o}_k$ and apply non-maximum suppression (NMS) to obtain a sparse set of proposals $\mathcal{P}$, which are then passed to the second stage.

We notice how the RPN is a fully convolutional network, therefore the size of input has not to be fixed, although it will increase the number of anchors and thus the computational cost.

Second-stage head. The second stage process starts with RoIAlign, a procedure that pools features from the shared backbone for each proposal into a fixed-size tensor F_j (e.g., $7\times7\times C$) without quantization errors [11]. This is done by

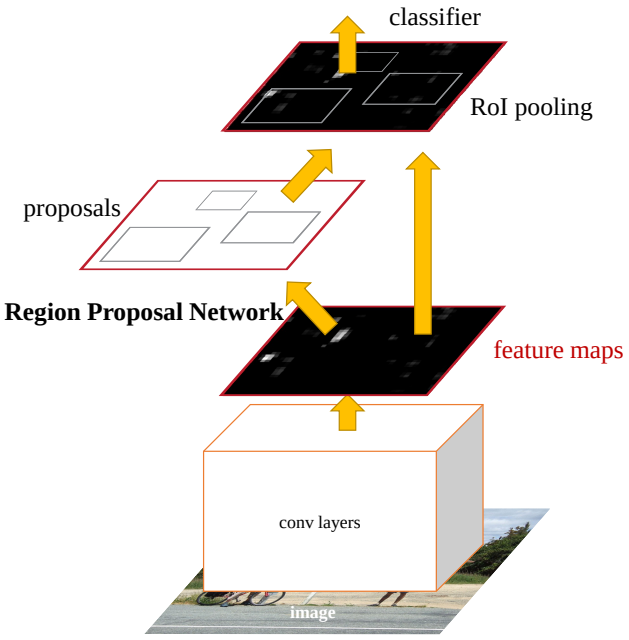

Figure 4: Faster RCNN architecture overview [25]

bilinear interpolation of the feature maps at the proposal locations, which avoids the quantization issues that arise with max pooling, that was a common issue in the original R-CNN architectures using RoIPooling instead.

A per-RoI head predicts:

- class logits $\hat{z}_{j,c}$ for $c \in \{1, \ldots, K\}$ (softmax over $K+1$ including background);
- refined box deltas $\hat{u}_{j,c}$ (class-specific) or $\hat{u}_j$ (class-agnostic).

With labels (y_j^*, u_j^*) from matching p_j to ground truth, the loss is

$$\mathcal{L}_{\text{RoI}} = \frac{1}{N_{\text{roi}}} \sum_j \ell_{\text{ce}}(\hat{z}_j, y_j^*) + \lambda_{\text{roi}} \frac{1}{N_{\text{pos}}} \sum_{j \in \mathcal{P}^+} \ell_{\text{reg}}(\hat{u}_{j,(y_j^*)}, u_j^*),$$

where regression applies only to positives $\mathcal{P}^+$.

Training of Faster R-CNN is performed end-to-end, with both the RPN and the second-stage detection head sharing the same backbone features. To keep the optimization stable, a sampling strategy is used at both stages to balance the large number of negative examples against the relatively few positives: mini-

batches are constructed with a fixed foreground-background ratio for anchors in the RPN and for RoIs in the second stage. The assignment of positives and negatives relies on IoU thresholds – for the RPN, anchors above a high threshold are treated as positives and those below a low threshold as negatives, while intermediate cases may be ignored; for the RoI head, proposals with sufficiently high IoU to a ground-truth box are considered foreground, and the rest background. During backpropagation, gradients propagate through the RoIAlign operation and further into the shared backbone, but not through the non-differentiable proposal selection and NMS steps, which are treated as fixed post-processing operations. Instead we backpropagate through the pre-NMS detections, with analogous loss functions as the ones used in the RoI heads that uses the notion of objectness and anchor's coordinates, allowing the model to learn to produce better proposals and detections.

2.4 Evaluation Metrics: Average Precision and Average Recall

Explain the need to formally define evaluation metrics as COCO's definitions are shaky, seems like everyone's using them but no one ever explains them.

2.5 Explainability in Deep Learning

Quick introduction of explainability in deep learning, highlighting how models are essentially black boxes and the need to understand, although partially, their inner workings. This need is even more pronounced in the medical field, where explainability is a requirement for clinical acceptance and regulatory compliance. Highlight how explainability methods, especially the ones that we are going to cover, are desinged to provide *insights* into the model's decision-making process, it does not make it fully explainable, but rather makes it a gray-box model, which is still better than a black box. Trainsition to CAMs for computer vision tasks

2.5.1 Class Activation Maps (CAM)

What are CAMs, how they work, and their (usual) limitations. Describe Grad-CAM as a gradient-based method and then explain why it is problematic for object detection tasks due to their sensitivity to the structured outputs and the non-differentiable post-processing steps.

2.5.2 Adaptations for Object Detection

Discuss gradient-free CAM methods ScoreCAM, SS-CAM and EigenCAM, highlighting how they can be adapted to work with object detection outputs. there's a bit of math involved here as we got to pinpoint what fails in their original implementations and how we can fix it.

Chapter 3

Data and Preprocessing

3.1 Datasets: NLST and DLCSD24

Briefly describe the datasets and their characteristics. mention that one does not have benign annotations, while the other does, nonetheless the first one can be yet used to train a detection model (as long as it detects nodules, regardless of their malignancy). regardless, we might also use the first one as a pretrain for the second one.

3.2 Preprocessing Pipeline

Resampling, and HU clipping, explaining why we're clipping and why those ranges and not others, this is supported by the HU table. We might include here also discarded preprocessing steps, such as outlier removal (although it was a post-process step), and lung masking through morphological operations.

3.3 Slice Selection Algorithms

Explain the need for slice selection algorithms in this 2D setting, as some might not contain relevan information (nodule mass). explain the statistical approach, the sliding window approach and finally the simple thresholding approach.

3.4 Data Augmentation Techniques

Discuss the data augmentation techniques used to enhance model robustness, such as random rotations, flips, and intensity variations. Explain how these techniques help mitigate overfitting and improve generalization to unseen data.

Chapter 4

Methodology

4.1 Object Detection Architectures and Backbones

Provide an overview of the object detection used (they're already described in the background chapter), and mention how we also experiment with different backbones.

4.2 Object Detection Pipeline

Describe the overall object detection pipeline, including the preprocessing steps, the training process, and the evaluation metrics used. I might include here curriculum learning

4.2.1 Loss functions

A central metric for bounding box quality is the *Intersection over Union* (IoU):

$$\mathrm{IoU}(b, b^*) = \frac{\mathrm{area}(b \cap b^*)}{\mathrm{area}(b \cup b^*)},$$

which measures the degree of spatial overlap between a predicted box b and a ground-truth box b^*. Variants such as Generalized IoU (GIoU), Distance IoU (DIoU), and Complete IoU (CIoU) incorporate additional geometric terms to improve optimization stability, particularly when boxes do not overlap [26, 34, 35].

Bounding box regression is also commonly evaluated in coordinate space using Minkowski distances. The general p-norm between two vectors $u, v \in \mathbb{R}^n$ is defined

as:

$$\|u - v\|_p = \left(\sum_{j=1}^{n} |u_j - v_j|^p \right)^{\frac{1}{p}}.$$

Special cases include the Manhattan distance (L_1 norm, $p = 1$):

$$\|u - v\|_1 = \sum_{j=1}^{n} |u_j - v_j|,$$

and the Euclidean distance (L_2 norm, $p = 2$):

$$\|u - v\|_2 = \sqrt{\sum_{j=1}^{n} (u_j - v_j)^2}.$$

In practice, modern detectors often adopt the Smooth-L_1 loss [8] for bounding box regression, which combines the robustness of L_1 for large errors with the stability of L_2 for small errors:

$$\text{SmoothL1}(x) = \begin{cases} 0.5\, x^2, & \text{if } |x| < 1, \\ |x| - 0.5, & \text{otherwise.} \end{cases}$$

This formulation reduces sensitivity to outliers while maintaining differentiability at the origin, making it suitable for gradient-based optimization.

4.3 Classification Head Extension

Discuss the classification network, how we extracted a classification dataset from the detection dataset using the bounding boxes.

4.4 Explainability Methods

Describe how are we using the adapted CAM methods, which layers we're targeting and an example of expected output. Explain how we're going to compare the explainability methods using the segmentation game.

Chapter 5

Results

5.1 Object Detection Performance Analysis

Our experimental approach involved a two-stage strategy designed to address both the technical challenges of developing robust object detection models and the practical constraints of limited computational resources.

5.1.1 Dataset Characgteristics and Usage Strategy

The National Lung Screening Trial (NLST) dataset served as our initial development and pretraining dataset. NLST contains individual CT slices with annotations for malignant nodules, characterized by relatively large and visually evident nodules that are easier to detect. The individual slice format eliminated the need for complex 3D-to-2D conversion pipelines, allowing us to focus initially on core object detection methodology without the additional complexity of volume processing. In contrast, the Duke Lung Cancer Screening (DLCS) dataset represents a more challenging and clinically realistic scenario. DLCS provides full CT volumes with annotations for both benign and malignant nodules, many of which are smaller and more subtle than those in NLST. The volumetric format required implementation of our complete preprocessing pipeline, including the slice selection and pruning algorithms developed to optimize 2D detection performance from 3D volumes discussed in Chapter 3. This dataset was used for final model evaluation, providing a more rigorous test of our methods in a realistic clinical context.

To assess the optimal training approach for our challenging DLCS evaluation scenario, we designed an ablation study comparing two training strategies:

- **NLST Pretraining**: Models were pretrained on the NLST dataset before

being fine-tuned on the DLCS dataset. This approach leverages the larger, more visually distinct nodules in NLST to provide a strong initial feature representation.

- **Direct Training on DLCS**: Models were trained directly on the DLCS dataset from ImageNet-pretrained weights, without prior exposure to NLST. This strategy aims to adapt the model directly to the more challenging and clinically relevant nodules in DLCS.

This comparison allows us to evaluate whether the progressive difficulty approach provides significant benefits in terms of detection performance over direct training on the more complex DLCS dataset.

The transfer learning strategy from NLST to DLCS tries to address the large domain gap between a CT dataset and ImageNet, where the latter is primarily focused on natural images. Nonetheless it is important to note that pretraining on ImageNet is often a standard practice in computer vision as it provides a strong initial feature representation, especially on the first layers of the network that tend to capture low-level features such as edges, textures, and basic shapes, all of which are easily transferable across different domains.

5.1.2 Model Architectures and Training Details

We mainly implemented two object detection architectures: Faster R-CNN and RetinaNet, both of which are widely used in the field. Each architecture was tested on several backbones, namely ResNet50, MobileNetV2, and EfficientNetV2S. The experimental framework was built entirely from scratch, allowing us to customize each component of the object detection pipeline, including the backbone selection and anchor box sizes, along with the usual training hyperparameters such as learning rate and batch size.

The implementation of these two architectures was mainly provided by the `torchvision` library [20] . Unfortunately, it does not provide an implementation of these architectures with EfficientNetV2S as a backbone, so we had to implement it ourselves and attach it to an FPN head. The same applied for the MobileNetV2 backbone although for the RetinaNet architecture only.

Each experiment used mixed precision training, which allows most computation to be performed using 16-bit floating-point (FP16) precision, while maintaining 32-bit floating-point (FP32) precision for critical operations that require numerical stability such as loss computation and gradient accumulation [21]. This ap-

proach significantly reduces GPU memory usage and accelerates training without compromising model accuracy. The implementation of mixed precision training was provided by the `torch.cuda.amp` module, which automatically manages the scaling of gradients and the conversion between FP16 and FP32 as needed.

The following hyperparameters were kept constant for all experiments to ensure a fair comparison. Due to the pretrained nature of every experiment, the learning rate was set to a low value of 0.0001 to avoid catastrophic forgetting, and the batch size was set to 8 to fit the GPU memory constraints. The training was performed for a maximum of 30 epochs, with early stopping based on the validation metrics to prevent overfitting and cut down on training time with a patience of 10 epochs. For the same reason the validation set was used to monitor the training process and decided to run a validation step every 3 epochs. The optimzer used was AdamW, which is a variant of the Adam optimizer that decouples weight decay from the optimization step, providing better generalization performance in many cases [19, 14].

As for the scheduler used, we opted for a Cosine Annealing scheduler, which graduelly reduces the learning rate over the course of training, allowing for a more fine-tuned convergence towards the end of the training process. We also experimented with Cosine Annealing with Warm Restarts, which periodically resets the learning rate to a higher value, but it often led to failed training runs due to the mixed precision training induced instability, so we decided to stick with the simpler version. Lastly, for the DLCS dataset, the 3-channels contain the previous, current, and next slice of the analyzed nodule, allowing the model to have a better context of the nodule's surroundings, effectively simulating a 3D input while still using a 2D object detection architecture.

The experiments were run using `parallel` to allow queuing multiple training runs on the same GPU, which is particularly useful for hyperparameter tuning and ablation studies. This approach allows us to efficiently utilize the available GPU and time resources.

5.2 Explainability Evaluation

Discuss the performance of the explainability methods, comparing them based on the segmentation game, performing some statistical analysis on the results.

Chapter 6

Conclusions

Summary of the work, and discuss how the research question and objectives were addressed in this work.

6.1 Future Directions

Potential future directions, such as exploring lightweight attention based methods, segmentation-based approaches, and comparing the adapted CAM methods with other explainability techniques ad-hoc designed for object detection tasks.

Bibliography

[1] Denise R. Aberle, Alden M. Adams, Christine D. Berg, William C. Black, Jonathan D. Clapp, Roger M. Fagerstrom, Ilana F. Gareen, Constantine Gatsonis, Pamela M. Marcus, and Jennifer D. Sicks. Reduced lung-cancer mortality with low-dose computed tomographic screening. *New England Journal of Medicine*, 365(5):395–409, 2011.

[2] Aditya Chattopadhay, Anirban Sarkar, Prantik Howlader, and Vineeth N Balasubramanian. Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, March 2018.

[3] Harry J de Koning, Carlijn M van Der Aalst, Pim A de Jong, Ernst T Scholten, Kristiaan Nackaerts, Marjolein A Heuvelmans, Jan-Willem J Lammers, Carla Weenink, Uraujh Yousaf-Khan, Nanda Horeweg, et al. Reduced lung-cancer mortality with volume ct screening in a randomized trial. *New England journal of medicine*, 382(6):503–513, 2020.

[4] Rachel Lea Draelos and Lawrence Carin. Use hirescam instead of grad-cam for faithful explanations of convolutional neural networks, 2021.

[5] Kaiwen Duan, Song Bai, Lingxi Xie, Honggang Qi, Qingming Huang, and Qi Tian. Centernet: Keypoint triplets for object detection, 2019.

[6] European Parliament and Council of the European Union. Regulation (eu) 2024/1689 of the european parliament and of the council laying down harmonised rules on artificial intelligence. Official Journal of the European Union (OJ L 2024/1689), adopted 13 June 2024, published 12 July 2024, entered into force 1 August 2024, 2024. High-risk AI systems in healthcare are regulated under this act.

[7] Zheng Ge, Songtao Liu, Zeming Li, Osamu Yoshie, and Jian Sun. Ota: Optimal transport assignment for object detection, 2021.

[8] Ross Girshick. Fast r-cnn, 2015.

[9] Ella Glikson and Anita Williams Woolley. Human trust in artificial intelligence: Review of empirical research. *Academy of Management Annals*, 14(2):627–660, 2020.

[10] Ali Guermazi, Chadi Tannoury, Andrew J Kompel, Akira M Murakami, Alexis Ducarouge, André Gillibert, Xinning Li, Antoine Tournier, Youmna Lahoud, Mohamed Jarraya, et al. Improving radiographic fracture recognition performance and efficiency using artificial intelligence. *Radiology*, 302(3):627–636, 2022.

[11] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn, 2018.

[12] Elizabeth Huynh, Ahmed Hosny, Christian Guthier, Danielle S Bitterman, Steven F Petit, Daphne A Haas-Kogan, Benjamin Kann, Hugo JWL Aerts, and Raymond H Mak. Artificial intelligence in radiation oncology. *Nature Reviews Clinical Oncology*, 17(12):771–781, 2020.

[13] Peng-Tao Jiang, Chang-Bin Zhang, Qibin Hou, Ming-Ming Cheng, and Yunchao Wei. Layercam: Exploring hierarchical class activation maps for localization. *IEEE Transactions on Image Processing*, 30:5875–5888, 2021.

[14] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[15] Hei Law and Jia Deng. Cornernet: Detecting objects as paired keypoints, 2019.

[16] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection, 2017.

[17] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection, 2018.

[18] Hui Liu, Ning Ding, Xinying Li, Yunli Chen, Hao Sun, Yuanyuan Huang, Chen Liu, Pengpeng Ye, Zhengyu Jin, Heling Bao, et al. Artificial intelligence and radiologist burnout. *JAMA network open*, 7(11):e2448714–e2448714, 2024.

[19] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2019.

[20] TorchVision maintainers and contributors. Torchvision: Pytorch's computer vision library. `https://github.com/pytorch/vision`, 2016.

[21] Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, and Hao Wu. Mixed precision training, 2018.

[22] National Cancer Institute. National lung screening trial (nlst) - low-dose ct imaging dataset. `https://wiki.cancerimagingarchive.net/display/NLST`, 2002. Data collection: 2002–2009. Accessed via The Cancer Imaging Archive (TCIA), subset of NLST. Available under CC BY 4.0 License.

[23] Alexander Neubeck and Luc Van Gool. Efficient non-maximum suppression. In *18th international conference on pattern recognition (ICPR'06)*, volume 3, pages 850–855. IEEE, 2006.

[24] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection, 2016.

[25] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks, 2016.

[26] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. Generalized intersection over union: A metric and a loss for bounding box regression, 2019.

[27] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. *International Journal of Computer Vision*, 128(2):336–359, October 2019.

[28] Nadia Stec, Danielle Arje, Alan R Moody, Elizabeth A Krupinski, and Pascal N Tyrrell. A systematic review of fatigue in radiology: is it a problem? *American Journal of Roentgenology*, 210(4):799–806, 2018.

[29] Sian Taylor-Phillips and Chris Stinton. Fatigue in radiology: a fertile area for future research. *The British journal of radiology*, 92(1099):20190043, 2019.

[30] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection, 2019.

[31] Fakrul Islam Tushar, Avivah Wang, Lavsen Dahal, Michael R. Harowicz, Kyle J. Lafata, Tina D. Tailor, and Joseph Y. Lo. Ai in lung health: Benchmarking detection and diagnostic models across multiple ct scan datasets, 2025.

[32] Hui Wu, Matrix Yao, Albert Hu, Gaofeng Sun, Xiaokun Yu, and Jian Tang. A systematic analysis for state-of-the-art 3d lung nodule proposals generation, 2018.

[33] Shifeng Zhang, Cheng Chi, Yongqiang Yao, Zhen Lei, and Stan Z. Li. Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection, 2020.

[34] Zhaohui Zheng, Ping Wang, Wei Liu, Jinze Li, Rongguang Ye, and Dongwei Ren. Distance-iou loss: Faster and better learning for bounding box regression, 2019.

[35] Zhaohui Zheng, Ping Wang, Dongwei Ren, Wei Liu, Rongguang Ye, Qinghua Hu, and Wangmeng Zuo. Enhancing geometric factors in model learning and inference for object detection and instance segmentation, 2021.

Acknowledgements