HIGH RECALL DEEP LEARNING METHODS FOR PEDIATRIC RIB FRACTURE DETECTION

By

Jonathan Burkow

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

Computational Mathematics, Science, and Engineering—Doctor of Philosophy

2024

ABSTRACT

Numerous data-driven, machine-learned models have been developed to localize pathologies in medical images. The work in this dissertation focuses on model development for the application of rib fracture detection in pediatric radiographs. Child abuse leading to physical harm is a severe problem, with over 100,000 confirmed instances in the United States during 2020, and children under 3 having higher susceptibility for fatal outcomes. Rib fractures are the most prevalent fracture site observed in young children with a near 100% positive predictive value for abuse if discovered. However, the detection of rib fractures in pediatric radiographic scans proves an arduous challenge, with even specialized radiologists failing to identify up to two-thirds of present fractures upon initial reviews. There are various reasons for this difficulty: fractures can be obliquely oriented to the imaging detector, obfuscated by other structures, incomplete, subtle, and/or non-displaced.

In this dissertation, we curate a custom dataset of 1,109 pediatric chest radiographs that were labeled by seven board-certified pediatric radiologists, yielding 624 fracture-present images. We analyze fracture prevalence patterns within the dataset, stratifying by age groups and examining factors such as fracture sidedness and rib locations. We investigate several methods for improving the sensitivity performance of two one-stage deep convolutional neural network (CNN) object-detection architectures, RetinaNet and YOLOv5. These include testing different input image filtering techniques, model ensembling approaches that combine predictions from multiple models, and a novel avalanche decision scheme we developed that dynamically adjusts the acceptance threshold during inference based on the number of fractures already detected.

We examine the performance of these networks using several metrics, including F2 score which summarizes precision and recall weighted toward high-sensitivity tasks. Fine-tuning of prior state-of-the-art one-stage models, RetinaNet and YOLOv5, achieves F2 scores of 0.48 ± 0.01 and 0.48 ± 0.04 respectively. Our best performing model used three ensembled YOLOv5 models with multiple image filters and an avalanche decision scheme, achieving an F2 score of 0.73 ± 0.01. When we conducted an expert inter-reader performance evaluation on the same test set of images, it resulted in an F2 score of 0.732. We then take inspiration from our fracture prevalence analysis

to create a domain-specific two-stage detection scheme, reliant on a prevalence-weighted region proposal method. The results demonstrated throughout this dissertation assert that a combination of sensitivity-driving methods yields object detector performance that approaches capabilities of expert radiologists, suggesting these methods may provide a viable approach to help identify these difficult-to-find rib fractures.

This thesis is dedicated to everyone who has influenced my academic and personal growth. To my parents, Richard and Linda Burkow, who have provided so much support and unwavering love. To my sister-in-law, Amy Burkow, for consistently pushing me to be confident in myself and become a better scholar. To my brother, Daniel Burkow, you have always been there to make the mistakes first so I could learn how to be better. In all seriousness, you have always been a benchmark for me to measure against, and your encouragement and guidance means more than you know. And finally, to the love of my life, Tiffany Thornton. You believed in me through all the times I struggled to and picked me up when I needed it. Your love, patience, and compassion are all why I have been able to complete this degree.

ACKNOWLEDGEMENTS

Throughout most of my education from undergrad and below, I felt like I was just going through the motions academically. That changed when I became a part of the Research in Industrial Projects for Students (RIPS) summer research experience hosted at UCLA. The project I worked on, sponsored by the Aerospace Corporation, in addition to the other projects, showed me the potential of applying mathematics and computation in real-world settings. From then on, I knew I wanted to contribute to innovative and exciting applications within industry. The Computational Mathematics, Science, and Engineering department at Michigan State University felt like a perfect fit to advance my academic career with this focus in mind; though, it was still incredibly difficult to consider leaving my Business Analyst job at Discover for. I remember going back and forth on the day of the deadline to accept entry into the program, to the point I considered missing a Nightwish concert I planned to attend that night because of the stress (I did end up going and having a great time). Despite the process of completing this degree proving incredibly challenging, it has been one of the most rewarding experiences and has ultimately changed my life for the better.

I want to thank my graduate student peers and faculty of the CMSE department I have met over the years that I have had the pleasure to talk to and learn from. I particularly want to thank Dr. Bob Termuhlen for being a great classmate and friend through all of our core classes and for giving me countless rides home from campus, especially when we worked past when busses would stop operating.

I also want to thank past and present students of the Medical Imaging and Data Integration (MIDI) lab, for being wonderful resources for any questions I had and for providing suggestions to my work that helped it to become what is shown throughout this dissertation. A massive shoutout to Dr. Muneeza Azmat, who was always available to either vent to or get hype from both when I was there in person and over the phone or Zoom. Though, to be fair, when we were in the office together we probably spent far too much time gossiping or talking about K-Pop that we should have spent working.

I want to thank my committee members, Dr. Adam Alessio, Dr. Vishnu Boddeti, Dr. Arun

Ross, and Dr. Yuying Xie, for all their guidance and helping me become the scholar I am today. I cannot thank my advisor, Adam, enough for everything he has done for me throughout my doctorate journey. Throughout the Covid-19 pandemic, he offered so much kindness and understanding, not only to me but to all members of the lab, constantly checking in to make sure everyone was doing alright. This was particularly impactful for me, as I had moved back home to Arizona and was across the country from everyone. His signing off on me staying home allowed me to be present in the last couple years of my cat Pierre's life, which I am eternally grateful for. I wholeheartedly believe that his mentorship is one of the biggest reasons I was able to complete this journey.

TABLE OF CONTENTS

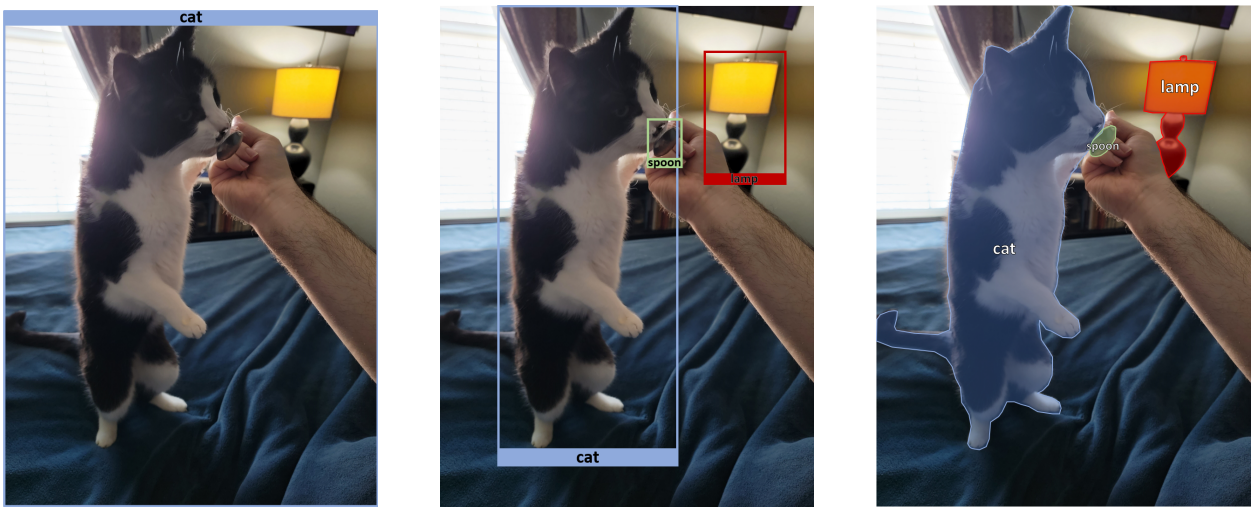
CHAPTER 1

INTRO TO COMPUTER VISION IN MACHINE LEARNING

1.1 Computer Vision

Computer vision is the use of computers to extract high-level information from visual media, such as digital images and video, to provide automated decisions and/or recommendations. Traditional computer vision techniques include edge detection, noise reduction, and image enhancement. Recent advancements in machine learning (ML) from both algorithmic improvements and computational accessibility have garnered massive interest in computer vision tasks. Image recognition, a major sub-task of computer vision, has especially experienced immense growth from the advancements. There are three prevailing tasks within image recognition: image classification, object detection, and segmentation.

Classification attributes a label that describes the overall image or video. Figure 1.1(a) illustrates this, classifying the image as a whole as “cat." For use cases requiring both classification and localization of objects, whole-image classification proves insufficient. Object detection is a task that ascribes a rectangular box around objects of interest and classifying each of them with an associated label. An example is shown in Figure 1.1(b). Rather than the entire image labeled as "cat," the cat object is roughly surrounded with a box and inherits the "cat" label; a spoon and lamp are each also bounded by boxes with their respective labels. Object detection provides more information about what is in the image due to the localization of the objects, thus is closer to what humans do when viewing images. Segmentation outputs a pixel-level representation of where each object is in the image. Figure 1.1(c) shows a segmentation output where the exact pixels of the cat are highlighted, rather than just a box surrounding it. The two most common types of segmentation are semantic and instance segmentation. Semantic segmentation splits the segmentations by the class type; i.e., the output of multiple overlapping objects of the same class in close proximity will be combined into one cluster of pixels. Instance segmentation further refines the output by separating instances of objects of the same class; i.e., a group of people standing together in an image would have boundaries separating each individual into different clusters of pixels with individual labels.



(a) classification (b) detection (c) segmentation

Figure 1.1 Examples of the three main image recognition tasks.

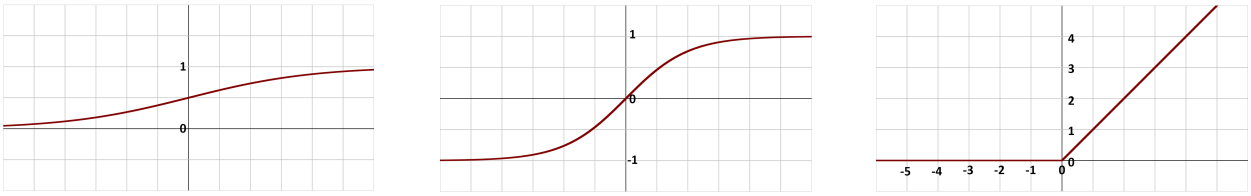
1.2 Artificial Neural Networks

The root for the idea of artificial neural networks was introduced in 1958 by Frank Rosenblatt [1]. Dubbed the perceptron, the idea was influenced by the way the brain sends and processes information signals; information transmits from neurons to neuron via electrical impulses through connections known as synapses. Translated directly, the machine neuron receives an input, processes a calculation through an activation, and directs the result on through the next layer. These bits of information connecting neurons are known as weights. The use of the word “network" implies the existence of many interconnected neurons. Indeed, artificial neural networks can be made up of anywhere from dozens to millions of neurons, with associated weights connecting each, that encode the information learned from the model. The basic structure of an artificial neural network starts with the input layer, then a various number of inner layers known as ‘hidden" layers, all directing toward the final output layer. The size of these neural networks can grow in two ways: increasing the number of hidden layers in the network (known as increasing depth), and increasing the amount of neurons in each layer (increasing width). Generally, all of these layers are called fully connected layers, meaning every neuron in each layer is connected to every neuron in the previous and the following layers, with an associated weight for each connection. The value of each weight represents the quantity of information sent between neurons, i.e., larger weight values represent

avenues of higher information passage through the neurons. Each layer of the network generally also has an associated bias value, which acts as a small shift to the output of the activation function. Over numerous repetitions through the training data, these weight values are updated so that the overall output of the network improves for the given task. This process allows a network to learn complex mappings between given input data and the output.

1.2.1 Activation Layers

As mentioned previously, the activation layer is the final computational step before sending information to subsequent layers in the network. These are various non-linear transformations that take in a weighted sum of the input values from the prior layer with the associated weights. There are various functions that can be used as activation functions; a couple of the most common ones are provided in Figure 1.2. Different functions offer different benefits. For instance, the



(a) sigmoid (b) tanh (c) ReLU

Figure 1.2 Visualizations of the sigmoid, tanh, and ReLU activation functions.

Rectified Linear Unit (ReLU) [2] function maps positive inputs to a linear function and negative inputs to zero, reducing computational overhead during backpropagation (which will be explored in the following section) since its derivative is either 0 or 1 (except at $x = 0$). The sigmoid, or more commonly the logistic function, is useful at the output layer of a binary classification network since the output values lie strictly between $(0, 1)$; we can differentiate class outputs by defining $y \geq 0.5 \rightarrow class_1$ and $y < 0.5 \rightarrow class_0$. The hyperbolic tangent function behaves similarly to the sigmoid, however values $x < 0$ are mapped to negative output values.

1.2.2 Optimization

Before training a neural network from scratch, all weight values dictated by the number and size of all convolutional kernels and fully connected layers are randomly initialized. After propagating

forward through the entire neural network, the network then begins a process called backpropagation. First introduced in 1968 by Rumelhart *et al.* [3], backpropagation is necessary for the neural network to reach an optimal solution by which all the unknown weight parameters from all kernels or neurons throughout the network are updated. In order for both the network to learn and to optimize the weights, a loss function is needed. Loss functions, denoted as $\mathcal{L}(\cdot)$, measure how well-performing the model is after each iteration while training; the ultimate goal is to reduce the loss (as calculated from the loss function) as close to zero as possible. For example, in a classification problem with c classes the most standard loss function is cross entropy,

$$\mathcal{L}(y_i, p_i) = -\sum_{i=1}^{c} y_i \log(p_i), \tag{1.1}$$

where y_i is 1 if item i belongs to class label y_i and p_i is the probability of the model predicting that class. The higher the probability, the lower the loss function, and thus the better performing the model is. Weight values are optimized by backpropagation computing gradients for every weight value throughout the network, starting at the loss function and traversing backward, and updating them all. One of the most common implementations for these gradient computations is gradient descent [4]. A tunable parameter α, also known as the learning rate, modulates the magnitude of changes to the weights and therefore how long it takes for the model to converge during training. The set of all weights θ will be updated during each iteration, t, of backpropagation by

$$\theta_t = \theta_{t-1} - \alpha \nabla_\theta L(\theta_{t-1}). \tag{1.2}$$

where $L(\theta)$ is the loss function with respect to all weights of the network. There are a couple variations of gradient descent that update the model in slightly different ways. The first is batch gradient descent, which waits until the gradients from processing through the entire training set have been calculated and then updates all weights. This offers a decently stable path for convergence, but the computational complexity is proportional to the size of the dataset. On the opposite side of the spectrum, stochastic gradient descent updates weights after calculating the gradient from each training example. This speeds up the computation, but suffers from a more erratic and less stable convergence. Instead, the most commonly implemented optimization technique—and what

is generally abbreviated as SGD—is mini-batch stochastic gradient descent; the training dataset is divided into smaller batches and the weights of the network are updated after gradients for each batch are calculated.

An improved optimization technique coined Adam—from adaptive moment estimation—was introduced in 2017 [5]. As the size of datasets (explored later in Section 2.3 and network complexity increased, the desire for more efficient optimization methods arose. By calculating adaptive learning rates independently for parameters based on estimates of the first two moments (mean, and uncentered variance) of the gradients, Adam is able to update parameters more effectively, especially in situations of parameter sparsity. Some parameters may need to update more quickly or slower than others, which stochastic gradient descent does not handle well since every parameter is given the same learning rate. Overall, Adam is more computationally efficient while requiring less memory, an extremely important attribute with larger and larger datasets.

1.3 Convolutional Neural Networks

Feed-forward, fully connected neural networks are powerful; however, using images as inputs can quickly run into computational bottlenecks. In order to be sent through a traditional multi-layer neural network, an image must be flattened into a one-dimensional vector which becomes exponentially computationally demanding as image sizes increase. For example, given a three-channel RGB image with dimensions 512×512, the resulting vector has a length of $786,432$ that needs to be sent through the network. Convolutional neural networks (CNNs) bypass these limitations and are able to handle image data much more efficiently and effectively, ultimately being able to find patterns within images. The primary components of CNNs that enable network processing of large images is the use of convolutional and pooling layers, which dramatically reduces the number of the unknown, trainable parameters found in conventional fully connected neural networks.

1.3.1 Convolutional Layers

In order for a computer model to learn from an image, it needs a way of traversing across the image to extract meaningful information. Convolutional neural network layers achieve this

through a process known as convolution and using a convolutional kernel: a small, square matrix of initially randomized values. Similar to the associated weights between neurons in a multi-layer fully connected network, the values in these convolutional kernels are the values learned during the training process. Common sizes for convolutional kernels include 3×3, 5×5, and 7×7. Figure 1.3 illustrates the first few steps of applying a kernel to an input. The 3×3 kernel is first placed at the top left of the image, then the Hadamard (element-wise) product is computed on all 9 overlapping cells, and the sum of that product matrix becomes the first value of the output matrix. The entire 3×3 kernel is then shifted to the right one cell, and the process repeated. This is done over and over, until the right-most column of the kernel reaches the right-most column of the image; then, the kernel is moved down a row and back to the left. Colloquially known as the sliding window, this process repeats until the bottom-right kernel value reaches the bottom right of the image. The final output of the kernel applied across the image is known as a feature map. Early on in the convolutional neural network, these kernels extract low-level features such as edges and corners. As the network gets deeper, the kernels start associating these features from previous layers into more abstract things such as shapes and textures.

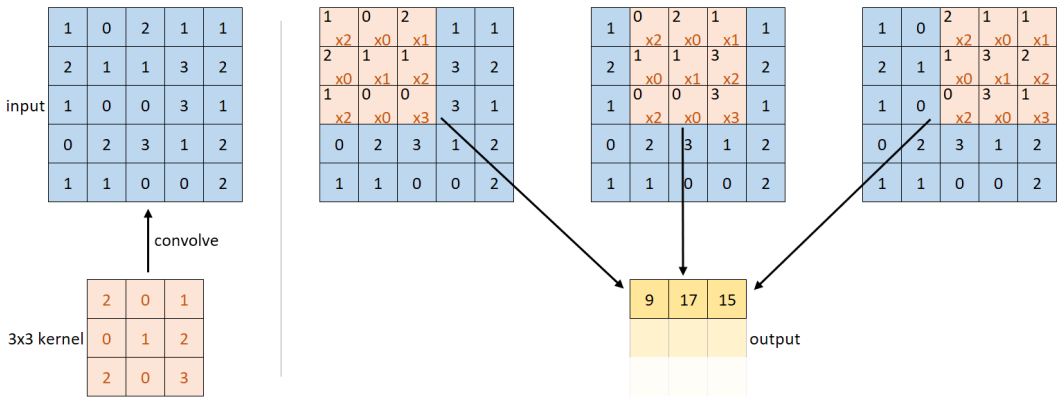


Figure 1.3 Example of the first three steps of convolution.

There are a few options that modify how a convolutional kernel is applied on the input, the first of which is the stride. In the example above, the kernel slid one space to the right after each summed element-wise product; this corresponds to a stride of 1. Increasing stride to 2 would shift the kernel by two spaces after each summed element-wise product with the input. Another

option when customizing convolution is padding, also known as zero-padding, that optionally adds a boundary of zeros around the input. Figure 1.4 illustrates applying a padding of 1, which adds two extra rows and columns to the input before the kernel slides across. By padding with zeros, the kernel is able to encode more information from the edges of the input since it passes over those values more. Padding is also used to counteract the downsampling effect of applying a kernel; by adding a sufficient number of zeros around the image the output can retain the same dimensions as the input, allowing chained convolution layers at the same scale.

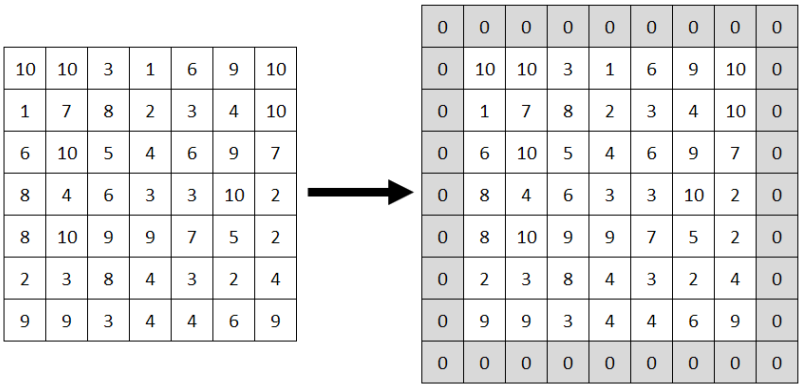


Figure 1.4 Zero-padding an input with one level of zeros.

The last option is called dilation. This alters the kernel by adding spaces between the kernel values for values greater than 1. A 3×3 kernel is unchanged with a dilation of 1; with a dilation of 2, a single space is added between each filter value, as shown in Figure 1.5. Adding dilation can be beneficial where an excessive number of parameters across all convolutional filters creates a performance bottleneck, where using a smaller kernel with dilation gives effectively the same receptive field on the input with significantly fewer parameters to learn per kernel (9 for a 3×3 kernel versus 25 for a 5×5 kernel).

Given an input with height H and width W with C channels, kernel size k, stride s, zero-pad p, and dilation d, the dimensions of the resulting feature map can be calculated as

$$H \times W \times C_{in} \xrightarrow{\text{Conv2D}} \left(\frac{H - (kd - d + 1) + 2p}{s} + 1 \right) \times \left(\frac{W - (kd - d + 1) + 2p}{s} + 1 \right) \times C_{out}. \quad (1.3)$$

Multiple convolutional kernels can be used on the same input within each convolutional layer, adjusting the third dimension of the output layer. For instance, one could create 16 distinct,

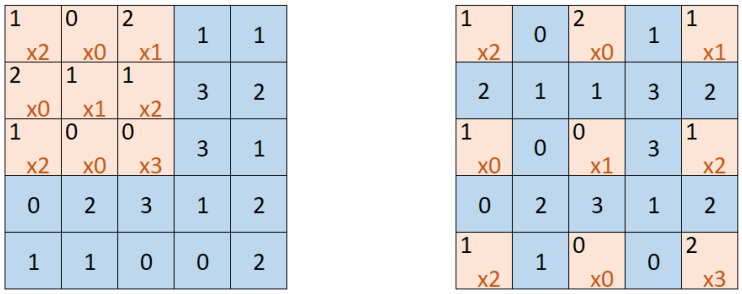
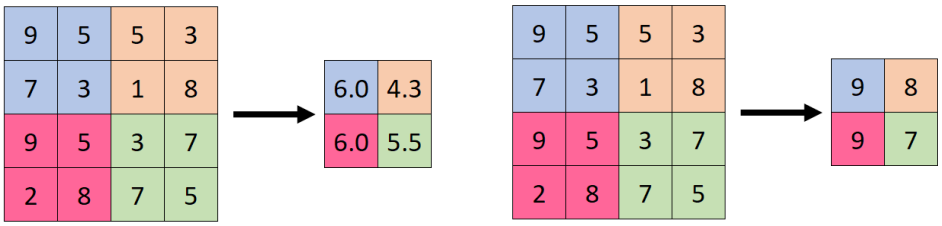


Figure 1.5 (left) a 3×3 kernel with dilation 1. (right) a 3×3 kernel with dilation 2.

randomly generated 3×3 kernels and apply each one of them across the input image. This would change C_{out} in Equation 1.3 to 16, each channel being the resulting values from each respective kernel applied on the input image. At the end of each convolution layer, much like traditional artificial neural networks, an activation function is applied which normalizes the values of the feature map computed with the convolutional kernel.

1.3.2 Pooling Layers



(a) average pooling (b) max pooling

Figure 1.6 The two types of pooling operations.

After one or multiple consecutive convolutional layers, a layer called a pooling layer is added as a downsampling technique to reduce the spatial dimensions for the next stage of the network. One pooling method is known as average pooling, shown in Figure 1.6(a), where small sections of the feature map get averaged into a single value. Another method is max pooling, shown in Figure 1.6(b), where the largest values in similarly sized sections of the feature map are retained. Max pooling is more common due to improved generalization and training performance [6, 7]. Typical sizing for pooling layers is 2×2 with a stride of 2, which reduces each of the spatial dimensions of the input by a factor of 2. Pooling layers serve two purposes; first, reducing the spatial size of the feature map decreases the computational overhead for further convolutional layers. Secondly,

by reducing the spatial size, dominant features learned by the CNN become more invariant to positional changes, i.e., if similar objects are in the same image at different locations, the CNN will be less likely to have issues predicting them as the same objects.

1.3.3 Classification Network Performance

Convolutional neural networks experienced a surge of interest predominantly because of the success of AlexNet [8] in 2012 after it achieved a significant improvement in accuracy score on the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [9] dataset compared to previous, hand-crafted methods. Winners of the ILSVRC competition thereafter continued using CNN-based models, solidifying the transition away from traditional methods to these highly-performing deep learning models for image recognition tasks. Table 1.1 provides the sizes and ImageNet-1K performance for some of the most common CNN architectures over the past decade. The ubiquity

Table 1.1 Performance for various classification architectures over time. Top-5 Error (%) is calculated on the ImageNet-1K test set (except for * for DenseNet where only validation set error was given). The number of layers for ConvNeXt-Base was not provided.

Year	Model	Layers	Parameters	Top-5 Error (%)
1998	LeNet [10]	7	60K	-
2012	AlexNet [8]	8	~62M	16.4
2014	GoogLeNet [11]	22	~6.7M	6.7
2015	VGG-16 [12]	16	~138M	7.3
2016	ResNet-50 [13]	50	~25M	5.25
2016	ResNet-152 [13]	152	~60M	3.57
2017	ResNeXt-101 [14]	101	~44M	5.3
2018	DenseNet-121 [15]	121	~8M	7.71*
2020	EfficientNet-B7 [16]	36	~66M	3.06
2022	ConvNeXt-Base [17]		~89M	3.13

of powerful modern computational hardware has allowed the creation of deeper and more complex CNN architectures, further dominating the object detection task and spearheading advancements in applications such as automated medical diagnosis and autonomous vehicle operation. This can be seen with networks such as VGG-16 with over 138 million parameters, all the way through ConvNeXt-Base with 89 million. Despite AlexNet's success with a top-5 error of 16.4, networks have achieved remarkably high scores in the low 3% ranges. For this image recognition task, human

error has been estimated to be near 5% [9]; ever since ResNet-152 from 2016, there have existed machine learning classifiers capable of out-performing humans on this type of dataset.

As evident in Table 1.1, a notable improvement in classification performance was seen going from the VGG-16 network to the ResNet-50 network. Not only did the error decrease by over 2%, the more important factor was that the number of parameters was drastically reduced from over 138 million to 25 million despite the depth of the network increasing. This was due to the introduction of the residual block (Figure 1.7(b)) allowing more convolutional layers and the reduction in fully connected layers at the end of the network. The residual block adds a “skip" connection from the

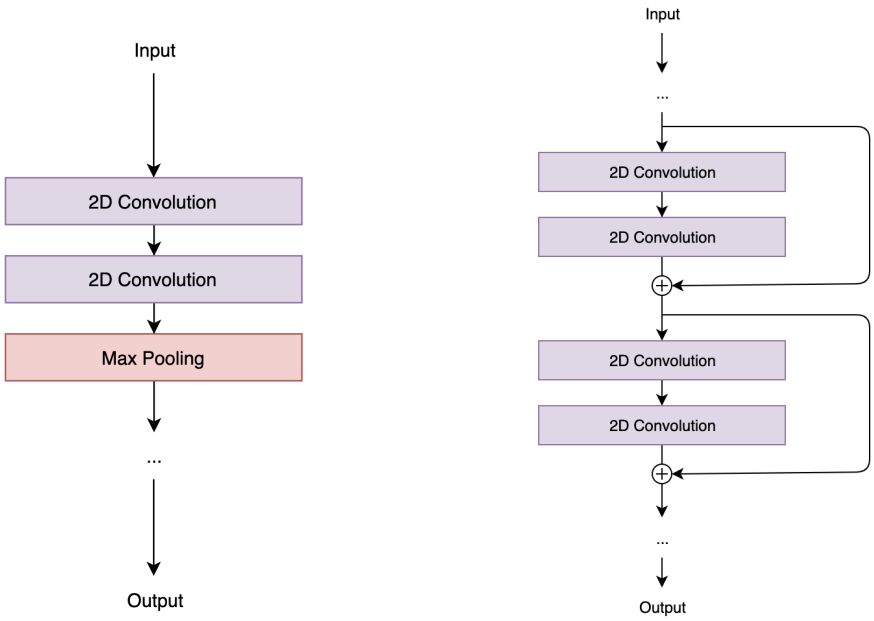


Figure 1.7 (a) standard convolution block as used in VGG-16. (b) residual block as proposed in ResNet. Images from [18].

beginning of a convolution block that concatenates with the output. This ultimately helps training of the neural network by reducing the chance that deeper networks reach a point where training error begins to increase, as well as providing an identity mapping that reduces the chance of a vanishing gradient during backpropagation. The vanishing gradient problem occurs as values from the end of the network are sent backward through the network and get exponentially squished toward zero, therefore preventing the network from learning [19].

1.4 Object Detection

The overwhelming success in classification tasks has extended interest in convolutional neural networks for object detection tasks. This is a natural extension since the same networks used in the classification tasks above can be reused for object detection. When used as the foundation of object detection architectures, these networks are called the “backbone" networks . Traditionally, these networks are pre-trained on massive datasets like the ImageNet [20] dataset mentioned above. It is then common to use these pre-trained weights as the starting point for detection tasks; this is known as transfer learning. With all of the low-level features learned through the backbone architecture through the initial training, detection tasks can begin with a pre-defined starting point and the network can be fine-tuned with the dataset of the given task.

Broadly, there are two ways to set up an object detection network; one-stage and two-stage methods. In a one-stage network, also known as single-shot networks, the entire process from inputting an image to an output with both bounding box and label predictions is handled end-to-end within a single framework. For two-stage networks, there is an initial stage used as a method for proposing regions, with the second stage designed to properly label the predicted regions. Figure 1.8 illustrates the difference between one- and two-stage networks. In the earlier competitions between the two, there was generally an understood trade-off of detection capabilities and inferencing time; i.e., a one-stage network would process images quicker than a two-stage network at the cost of lower detection accuracy.

1.4.1 Two-Stage Detectors

The first well known two-stage detector was introduced in 2014 by Girshick *et al.* , named R-CNN [21]. In their implementation, they employ a selective search algorithm [22] as the first stage to propose $2,000$ regions from the image and sends each of them separately into its second stage (see Figure 1.9(a) for an example). However, due to the first stage being completely detached from the backbone CNN, every region proposal gets sent individually through the classifier stage. The Fast R-CNN improved upon the proposal stage by pooling regions of interest into feature maps before sending them through the classifier [23]. Due to the sharing of computation across the

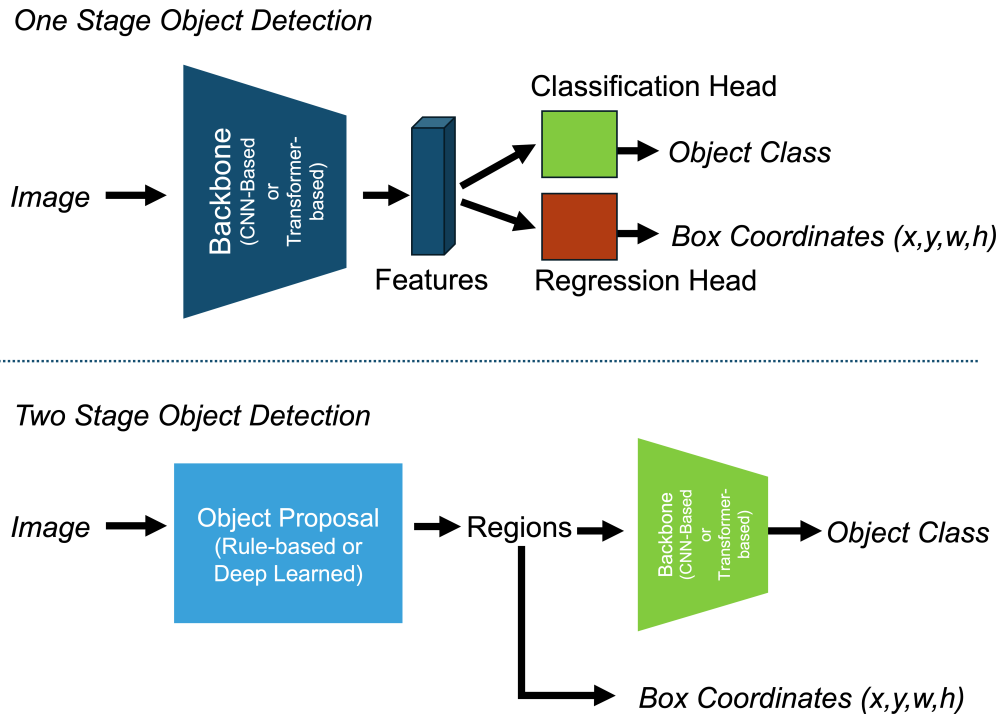


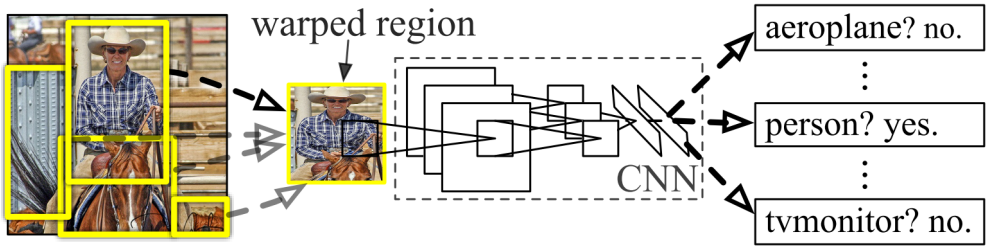
Figure 1.8 Flow chart comparing basic steps of one-stage versus two-stage object detectors

network from the RoI pooling, training time was 18 times quicker, but possibly more important, the time required a model to provide predictions on a test image switched from being measured in seconds per image to images per second.

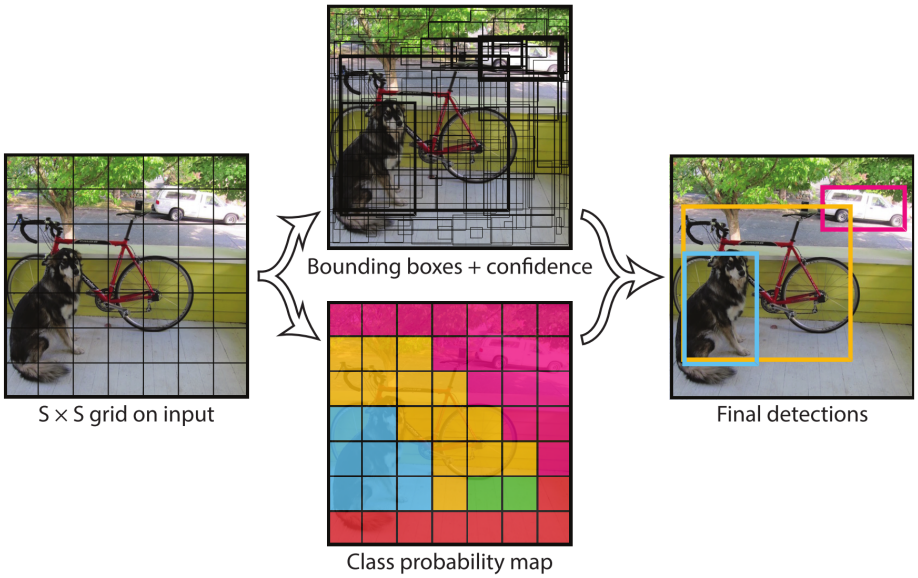
Faster R-CNN introduced a new first stage with a region proposal network (RPN) that was created to output proposal regions for objects [24]. The output from the RPN has a sliding-window process to generate a list of anchor boxes that are then output for each step in the sliding window. Mask R-CNN took this a step further to have a second stage that not only provided the class and bounding box predictions but also output a pixel-level binary mask for each prediction, allowing the capability for full object segmentation [25]

1.4.2 One-Stage Detectors

Examples of one-stage detectors are Single-Shot Detector (SSD) [27], You Only Look Once (YOLO) [26], and RetinaNet [28]. In the YOLO architecture, images are divided into a $S \times S$ grid where each grid cell is responsible for predicting bounding boxes and confidence scores for any objects contained within the cell (Figure 1.9(b)). By sending the image through the network only



(a)



(b)

Figure 1.9 (a) region proposal example for R-CNN [21] (©2014). (b) grid cells detection in YOLO [26] (©2016).

once, the time to train the CNN architectures as well as the time it takes for them to propose bounding boxes on new images is greatly reduced compared to their two-stage counterparts. Originally praised and implemented due to their faster real-time inferencing capabilities, single-stage architectures and their variants have become more common and have been achieving results on par with two-stage methods.

Table 1.2 provides a breakdown various different detection networks, with their mean Average Precision (mAP) performance on either the Pascal VOC 2007 [29] or the Microsoft COCO [30] datasets, in addition to how many images the trained networks can detect on per second (abbreviate FPS for frames per second).

Of the detection architectures in Table 1.2, this dissertation has a large focus on RetinaNet and

Table 1.2 Table of mainstream, state-of-the-art detection architectures of recent years. Performance of the architectures is provided on two datasets: mAP^1 is on the Pascal Visual Object Classes (VOC) 2007 [29] test dataset, mAP^2_{50} is on the Microsoft Common Objects in Context (COCO) [30] test dataset.

Year	Architecture	Backbone	mAP^1	mAP^2_{50}	FPS
2014	R-CNN [21]	VGG-16	66.0		0.08
2015	Fast R-CNN [23]	VGG-16	70.0		0.5
2015	You Only Look Once (YOLO) [26]	Custom	63.4		45
		VGG-16	66.4		21
2016	Faster R-CNN [24]	VGG-16	73.2	42.7	17
2016	Single Shot Detector (SSD) [27]	VGG-16	76.8	46.5	22
2017	Mask R-CNN [25]	ResNet-101		60.3	5
2018	RetinaNet [28]	ResNet-101		61.1	5
2018	YOLOv3 [31]	Darknet-53		57.9	20
2020	EfficientDet [32]	EfficientNet		65.9	35
2020	YOLOv5-l6 [33]	CSP-Darknet53		71.3	63

YOLOv5. RetinaNet was one of the first object detection architectures to incorporate a Feature Pyramid Network (FPN) [34] on top of the backbone network, which can be seen in Figure 1.10(a-b). During convolution, output feature maps are sent directly through to a matching upscaling level at three resolution scales. After these two feature maps are concatenated at each level in the pyramid, anchor boxes are used to scan through the feature maps to send object proposals through to two sub-networks (Figure 1.10(c-d)). There are nine anchor boxes per level that vary in aspect ratio (1:1, 1:2, and 2:1), as well as three scaled sizes per aspect ratio. After anchors are sent through to the sub-networks, one sub-network is responsible for the classification of the object and the second is tasked with the bounding box regression to localize the detection around the object. On one hand, the incorporation of the FPN and classification and regression sub-networks drastically increased RetinaNet's mean average precision (mAP) performance on the MS COCO dataset, but also causing it to have a much slower inference time compared to SSD or YOLO (5 FPS vs 59 and 45, respectively). The other major contribution from the RetinaNet model was a new loss function they dubbed Focal Loss (Equation 1.4. They simplified the standard cross entropy loss function (Equation 1.1) into the $-\log(p_t)$ term by defining $p_t = p$ if $y = 1$ and $p_t = 1 - p$ and then defined

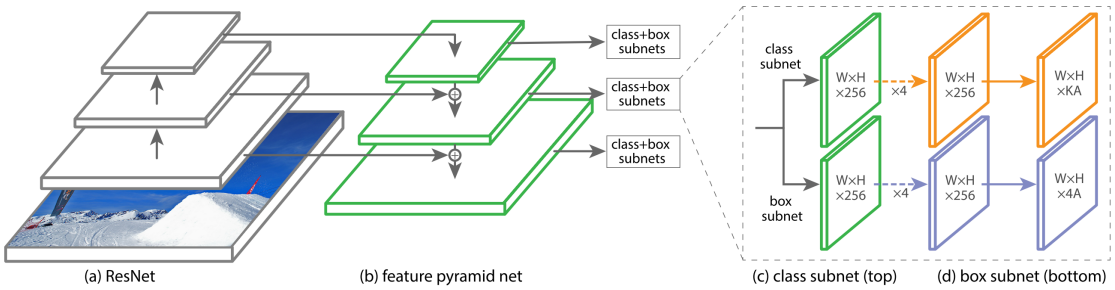


Figure 1.10 Diagram of the RetinaNet detection architecture [28] (©2020).

Focal Loss (FL) as:

$$FL(p_t) = -\alpha_t(1 - p_t)^\gamma \log(p_t), \tag{1.4}$$

where α_t is a class-balancing hyperparameter. The most important term of this new loss function is the $(1 - p_t)^\gamma$ with what they call the "focusing" parameter, γ (which they set to 2). In essence, depending on how confident the model is for a given prediction, this component adjusts how much the loss contributes. For highly confident predictions this will dramatically reduce the effect of the loss; e.g., for $p_t = 0.99$, this term becomes $(1 - p_t)^\gamma = (1 - 0.99)^2 = 0.01^2 = 0.0001$. By introducing this term, the model effectively learns to disregard highly confident predictions and "focus" on improving and correcting more difficult examples.

1.4.3 Challenges for Object Detection

There are numerous challenges encountered in object detection tasks, each causing different effects in the performance of the deep learning models. By far the most common, especially depending on the specific application, is the access and quality of the data. With the increasing size and complexity of models over the years (such as the parameter and depths of backbone networks seen in Table 1.1), there is a proportional need for more and better data to improve training quality. With smaller datasets, there is a chance for the detector to overfit to the training set and not be as capable when generalizing to outside datasets.

There are augmentation techniques that can somewhat mitigate the effects of smaller datasets. These include transformations such as rotating or flipping images horizontally or vertically. Hue, brightness, and saturation modifications modify the values and pixel intensities that can also improve detector performance if given images that are drastically different in terms of exposure or

light-levels. Over time, more complex augmentation strategies have been considered. For instance, YOLOv4[35] introduced the concept of mosaic augmentation, where segments from four different training samples are mixed together at varying sizes into new training samples. This changes the context of the objects being detected, limiting chances for the model to make direct associations from certain contexts to the existence of objects. Even more recently, there has been interest in synthetically generating images as an augmentation technique [36, 37, 38] (a direct application of this will be discussed later in Section 4.9).

Another hurdle for detection tasks, especially with earlier architectures, is object scale variation. Even though humans can easily perceive objects at smaller sizes to be the same, the difference in scale can prove difficult for detectors. The SSD architecture [27] was one of the first one-stage networks to implement sending feature maps at multiple stages during downsampling as outputs straight to the detection and classification end of the network, allowing the network to learn about objects at multiple scales. The Feature Pyramid Network (FPN) [34] discussed above is a more complex implementation of handling the multi-scale problem, since the feature maps from the earlier downsampling stages are concatenated with feature maps of upsampling stages prior to being sent to the regression and classification heads. These led to marked improvements in detecting objects of different sizes.

Object occlusion also hampers localization capabilities for detectors, as the more an object gets obscured by another object the more challenging it becomes to separate them into separate objects, or perhaps even outputting a prediction for the occluded object [39]. Imagine an automated driverless car with vision of an upcoming intersection, but a human behind a sign is not recognized and therefore runs the risk of the car not properly stopping in time if they decide to move onto the street.

1.5 Evaluation Metrics

There are various metrics in use to evaluate the performance of machine learning models across classification, detection, and segmentation tasks. Here, we provide background on a few of the main ones, especially those that will be used throughout this dissertation.

1.5.1 Confusion Matrix

For classification tasks, the overall performance of a model can be visualized in what is known as a confusion matrix. It quantitatively measures how the predicted values from the model compare to their true values. Table 1.3 shows what a confusion matrix looks like for a binary classification task with only two labels, 1 (positive) and 0 (negative). From the values within this matrix, there

Table 1.3 A typical layout for a confusion matrix for a binary classifier.

		True Label	
		1	0
Predicted Label	1	True Positive (TP)	False Positive (FP)
	0	False Negative (FN)	True Negative (TN)

are multiple other metrics that can be derived. The first of which is accuracy,

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN}, \tag{1.5}$$

that provides a very general overview of how well the model performs. It is bounded between 0, representing the model achieving no correct predictions, and 1 where the model is correct on everything. Accuracy can be misleading, however, especially for imbalanced datasets. For example, for a dataset containing forty-five 0-labeled instances and five 1-labeled instances, if the model predicted every instance as a negative, or 0, label, the accuracy would be 0.90. There are additional metrics that provide a more fine-grained perspective of model performance. Precision, otherwise known as the positive predictive value (PPV), is a ratio comparing the number of correctly labeled instances to all of the instances labeled as 1, i.e.,

$$Precision = \frac{TP}{TP + FP}, \tag{1.6}$$

with values approaching 1 representing models that are accurate with the labels they predict as positive. Specificity, or the true negative rate (TNR), captures how many 0-labeled instances compare to the total number of true 0-labeled instances,

$$Specificity = \frac{TN}{TN + FP}, \tag{1.7}$$

where values approaching 1 indicate the percentage of negative classes properly predicted as negative. Recall, otherwise known as sensitivity (note: we use these two terms interchangeably throughout the dissertation), demonstrates how well the model is able to accurately predict the positive classes in the dataset, calculated as

$$Recall = \frac{TP}{TP + FN}. \tag{1.8}$$

All three of these scores (precision, specificity, and recall) are bounded in the range $[0, 1]$ and, as discussed, values closer to 1 represent better model performance. By splitting the performance into these separate metrics, we are able to better grasp the capabilities of machine learning models depending on the perspective we want to approach evaluation.

Note that for object detection tasks, there are slight modifications to the application of these three metrics. Given that object detection carries a localization aspect, a "true negative" cannot be measured as it requires the absence of an object which the detector should not apply a bounding box for. This eliminates specificity as a measure for object detectors. For precision and recall, this can be calculated for all proposed boxes that carry a probability score exceeding the model acceptance threshold; this is how these metrics are used throughout this dissertation.

1.5.2 Receiver Operating Characteristic

While the confusion matrix, and therefore precision, specificity, and recall, are all great ways to measure the performance of classification models, they require the probabilities for each instance output by the model to be assigned to either the positive 1 or negative 0 classes based on a defined threshold, such as 0.50 for a sigmoid output activation. To get a more holistic understanding of how the model performs given all probability scores of the output, the receiver operating characteristic (ROC) curve presents the sensitivity and false positive rate (calculated as $1 - specificity$) for each of the predicted probabilities. This provides insight into the tradeoff of sensitivity and specificity for different operating points for the model (different threshold settings).

From Figure 1.11, it can be seen that model performance tends toward the top-left. A perfect model would appear as a single dot in the top-left most corner, showing the model has a sensitivity

of 1 (every positive class in the dataset was correctly predicted) with a false positive rate of 0 (no items were falsely predicted). The dashed gray line in the diagonal is the ROC to be expected from a completely random classifier, i.e., having a 50/50 chance of scoring a given instance as positive or negative. From the ROC curve, a single measure called the AUC (area-under-the-curve, also ROC-AUC) can be calculated as the name implies: the area underneath the computed ROC curve. Values approaching 1 represent model performance tending towards a perfect classifier, whereas an AUC score of 0.50 matches a random classifier.

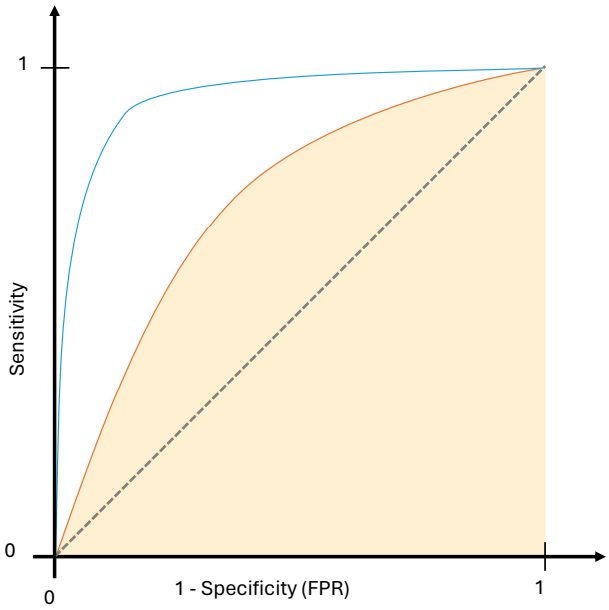


Figure 1.11 An example ROC curve. The dashed diagonal line represents performance expected from a completely random classifier. The shaded region represents the area-under-the-curve (AUC) for the orange curve. Lines tending toward the top-left of the graph represent better model performance.

1.5.3 F2 Score

A common evaluation metric used in classification and detection tasks is F1 score, which is the harmonic mean between precision and recall.

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \tag{1.9}$$

This is a specific implementation of a general F_β score where both precision and recall are considered equally as important in the final calculation. The constant term β can be changed to weight precision or recall β-times more than the other. For instance, if one wants recall to carry β-times as much importance as precision in the calculation, the equation becomes

$$F_\beta = (1 + \beta^2) \cdot \frac{\text{precision} \cdot \text{recall}}{(\beta^2 \cdot \text{precision}) + \text{recall}} \tag{1.10}$$

If instead one desired to weight precision β-times more than recall, the denominator would change to $(\text{precision} + (\beta^2 \cdot \text{recall}))$. For our evaluation, we are placing a larger emphasis on recall over precision, as we want the deep learning models to be liberal with predicting potential fracture locations even if the predictions actually do not contain fractures. In the long term, this tool is intended to serve as a reading aid for radiologists to flag suspicious regions; the final determination of the presence or absence of fractures will be determined by the radiologist aided by the model output. For these reasons, we are predominantly evaluating all models by F2 score and placing twice as much weight on recall as precision, i.e., setting $\beta = 2$ in equation (1.10).

1.5.4 Mean Average Precision (mAP)

As discussed above in Section 1.5.1, precision represents the number of labels predicted as positive that are correctly labeled, whereas recall represents the number of correct labels identified of all possible positive objects. In object detection, the values found for TP, FP, and FN labels can be tweaked in two ways: varying the model confidence for accepting bounding box predictions, or varying the IOU threshold [40]. We can visualize precision and recall performance across all model confidences by computing the precision-recall curve, where the IOU threshold is held constant. Then, to summarize the performance into a single metric, we can calculate the Average Precision (AP) by taking the weighted sum of precision values at equal intervals across recall values. Going further, the mean Average Precision (mAP) includes multiple precision-recall curves across many IOU thresholds and computes the mean of the average precisions from all plots; e.g., if the score is represented as mAP[0.05:0.05:0.95], precision-recall curves are generated for every IOU threshold between 0.05 and 0.95 at intervals of 0.05, and the average precisions across all 19 precision-recall

curves is averaged. This provides a much more broad understanding of the performance of an object detection architecture.

1.5.5 Intersection-over-Union (IOU)

The Jaccard Index, or Intersection-over-Union (IOU) [41], is a measure for quickly and easily scoring the overlap from a predicted region and a ground-truth region. This can be used for both bounding boxes in object detection, and pixel-level predictions from image segmentation. It computes the overlapping region between two regions and divides it by all elements/pixels among the two regions, shown in Equation 1.11,

$$IOU = \frac{|A \cap B|}{|A \cup B|}, \tag{1.11}$$

where $|\cdot|$ is the number of pixels in the region. Bounded in the range $[0, 1]$, higher values represent predicted regions that more closely mirror the true regions. Assuming a proposed box contains within it the entire true box, the IOU score may still be small if the size of the proposed box is much greater than the true box.

1.5.6 Dice Coefficient

The Dice coefficient [42] is a measure for evaluating the similarity between two sets. For image segmentation, it scores how many pixels from a model output overlap with ground-truth pixel annotations. Equation 1.12 gives the mathematical expression to compute the Dice coefficient

$$\text{Dice Coefficient} = 2 \cdot \frac{|A \cap B|}{|A| + |B|} \tag{1.12}$$

where $|\cdot|$ represents the number of pixels in the respective set. The Dice coefficient ranges from $[0, 1]$, with 1 representing pixel-perfect coverage of the ground-truth annotation.

1.5.7 Alternative Free-Response Receiver Operating Characteristic (AFROC)

In traditional classification tasks, the receiver operating characteristic (ROC) curve is an evaluation measure showing the relationship between the true positive rate and false positive rate across a wide range of thresholds. From this, the accompanying area under the curve (AUC) can be computed to summarize ROC curve performance between $[0, 1]$. Unfortunately, this measure is

unsuitable for detection tasks as predictions are localized within the image, rather than the image as a whole. The alternative free-response receiver operating characteristic (AFROC) [43] resolves this issue by incorporating local-level annotation information. Note, for the following equations the word lesion represents a hand-labeled region of interest on a given image (e.g., a cancerous mass in an organ and a bone fracture would both be considered lesions). Two main equations are needed to calculate the AFROC curve, the lesion localization factor (LLF) and non-lesion localization factor (NLF):

$$LLF = \frac{\text{\# correctly marked lesions}}{\text{total \# of lesions}} \tag{1.13}$$

$$NLF = \frac{\text{\# incorrectly marked lesions}}{\text{total \# of images}}. \tag{1.14}$$

Note that $LLF \in [0, 1]$ but $NLF \in [0, \infty)$. To bound both axes, AFROC assumes the likelihood of marking n localized regions inaccurately per image to be determined by the Poisson distribution, $P(n) = \frac{e^{-\lambda}\lambda^n}{n!}$, with λ being the average number of incorrectly marked localizations, i.e., the NLF. Then, the false positive rate can be computed as

$$FPR = 1 - P(0) \tag{1.15}$$

$$= 1 - \frac{e^{-NLF}NLF^0}{0!} \tag{1.16}$$

$$= 1 - e^{-NLF}. \tag{1.17}$$

Therefore the AFROC curve, like the original ROC curve, has both a x- and y-axis restricted to $[0, 1]$. This allows the calculation of the area under the curve once again, with AUC values closer to 1 representing perfect performance.

CHAPTER 2

DEEP LEARNING IN MEDICAL IMAGING

2.1 Significance

Deep learning-based architectures have been gaining popularity within the medical field over the past several years. The capabilities to automatically classify, localize regions of interest, and/or provide pixel-level segmentations of pathologies in images has numerous applications within the medical field that can ultimately lead to improvement of patient outcomes. This is incredibly poignant knowing there is a perpetual shortage of physicians in the country. With a smaller ratio of physicians to population, the ability to effectively and efficiently diagnose patients becomes more and more of a challenge, especially for individuals living in more rural communities with even less access to high quality healthcare. The issue compounds further when specialists are needed, since they are much more difficult to find than general physicians. From the mid 1990's to early 2010's, despite a raw increase in number of radiologists of nearly 40%, the proportion of trained radiologists compared to general physicians dropped by 8.8% [44]. Without enough access to trained and specialized radiologists, there is a two-fold effect on patient outcomes: either patients get care from less specialized doctors that may miss important and possibly life-threatening conditions, or the specialized doctors have less time to interpret the medical images and therefore also risk missing critical regions. While overlooking a fractured bone in a hand or foot from a radiological scan is not likely to cause life threatening issues, missing a malignant tumor in an organ can lead to terminal outcomes, especially if not caught early enough. By leveraging the powers of computers to automate initial evaluation of medical images, the likelihood of detecting more of these lesions increases, and improving the chances for positive patient outcomes.

2.2 Medical Images and Modalities

Compared to natural images, there are various different challenges when attempting to use medical imaging for deep learning. The most obvious is the manner in which images are captured. Figure 2.1 presents example images from different imaging modalities. Each image was generated

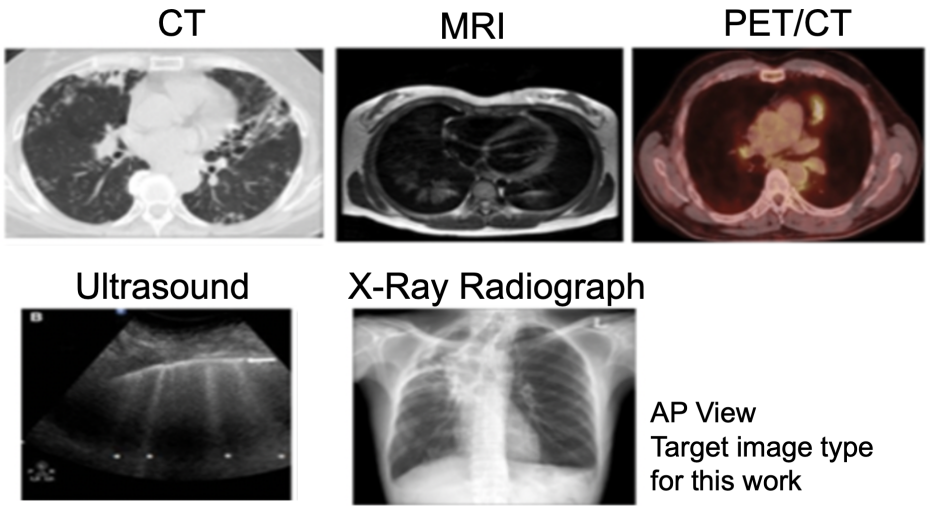


Figure 2.1 Example images from common medical imaging exams of the chest region. Displays the diversity of image presentations and the reality that medical images differ from natural images in multiple respects. Modified with permission under CC 4.0 International from [48].

from very different physical processes as outlined below. The focus of this work will use X-ray radiographs of the chest. In brief, x-rays are generated in a x-ray tube and transmitted throughout a subject. An imaging detector is situated on the other side of the subject and collects the x-rays that are not attenuated by the subject. This creates an image of the x-ray attenuation. Figure 2.2 compares images from common chest radiograph acquisitions.

While the work in this dissertation focuses on X-ray radiographs, there are many other methods. A common medical imaging modality is computed tomography (CT), represented by the top-left image in Figure 2.1. In CT scans, x-rays are sent out through the patient as the x-ray tube rotates around them, capturing many views from all angles around them. There is generally a trade-off of image quality and patient radiation dose, and there have been numerous efforts to retain as much information with lower dosages [45, 46, 47]. Positron emission tomography (PET) (top-right image in Figure 2.1) involves injecting the patient with a glucose-like substance that has been labeled with radioactive isotopes that, once in the tissue, decay and release positrons that annihilate in collisions with nearby electrons, generating two gamma rays that get sent in opposing directions that the detector captures.

For X-ray, CT, and nuclear medicine modalities, there has been a persistent concern over the

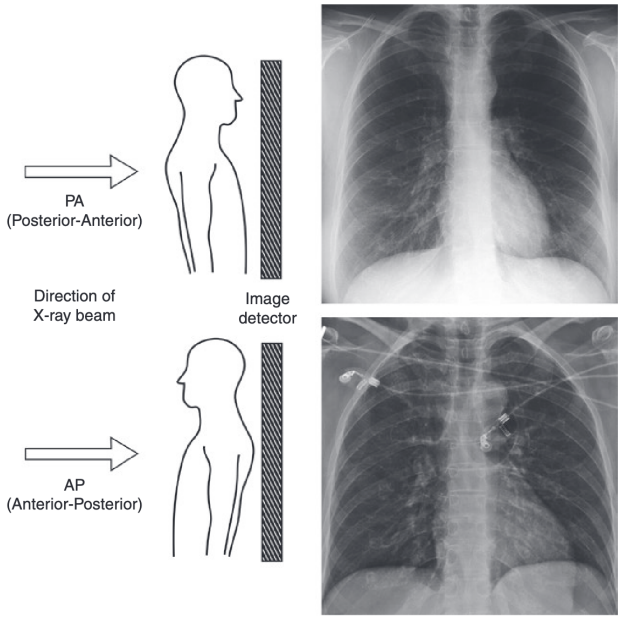


Figure 2.2 Examples of common chest radiograph acquisitions using x-ray and patient positioning for AP and PA views. The work in this dissertation uses AP chest radiographs. Used with permission from [49].

amount of ionizing radiation absorbed by the patient. Each procedure has an approximate effective dose measured in millisieverts (mSv) that changes based on numerous factors including the machine used as well as the region being imaged [50]. Generally, each year an average person receives an effective dose of approximately 3 mSv from natural sources of radiation. A thoracic x-ray imparts approximately 0.1 mSv, whereas a CT scan of the chest imparts approximately 6.1 mSv. There have been efforts to quantify annual effective doses and cumulative effective doses (CED) as ways to monitor the amount of radiation from imaging across age groups and populations. Fazel *et al*. [51] found that the annual effective dose increases as ages in the population grow, with 52 per 1000 patients experiencing between 20-50 mSv annually compared to only 4.9 per 1000 in younger adults aged 18-34. They also found that on a yearly basis, CT scans of the abdomen represent 18.3% of the annual effective dose across all imaging procedures. In many studies, the threshold given for cumulative effective dose is greater than 100 mSv [52]. Frush *et al*. found in their meta-analysis that for pediatric patients, the number of patients exceeding this CED ranged generally represented less then 1% of the patients, but across all ages could be as high as 2.9%. There remain unknown associations between the risk of cancer and the low levels of radiation

imparted in medical imaging. In the face of this unknown knowledge, the general practice is to limit image quality to ensure as-low-as-reasonably-achievable radiation doses are used [53].

While X-ray, CT, and PET involve some level of radiation exposure, there are other imaging procedures that do not involve ionizing radiation. One of these is magnetic resonance imaging (MRI), shown in the top-middle image in Figure 2.1. MRI machines generate a steady magnetic field around the patient, aligning the atoms inside them in a certain way. Then, bursts of radio frequencies are sent toward the patient, and as the atoms relax and re-align with the magnetic field they emit signals that the machine capture. MRI is widely considered to be the superior modality for soft tissue imaging. Lastly, ultrasound (bottom-left image in Figure 2.1) uses sound waves emitted at a higher frequency than we can hear into the patient, and reflections from these sound waves generate the image seen during an ultrasound scan. One of the major benefits of ultrasound is that the image can be generated in real-time so it is very beneficial for patient-facing applications such as fetal development during pregnancy [54].

From the source, medical images are also stored differently. Natural images, especially those taken by cell phones or general consumer cameras, are commonly stored in PNG or JPEG formats which can be easily adapted or directly used in machine learning models. It is common for medical images to be stored in DICOM (Digital Imaging and Communications in Medicine) files in hospital storage systems called PACS (picture archiving and communication system) [55]. These DICOM files contain the pixel values that were reconstructed from the raw detector measurements, as well as metadata that provides information such as the scanner used, the settings of the machine, pixel spacing and slice thickness, but also patient health information (PHI) such as the patient name, age, and imaging center where the image was taken. Whereas natural images are generally three-channel RGB images with 8-bits of information per channel, medical images tend to be monochromatic single-channel images with anywhere from 12-bit to 16-bit values. For a single radiograph, these images might not be the largest—for instance, an uncompressed 2000×3000 16-bit radiograph would be about 12MB in size–compared to images taken from a phone camera which may be in the 4-7MB range for a similar 12MP image. However, consider a 3D CT volume for a single patient,

where each image is 512×512 and there are 500 slices; at a pixel-spacing of 0.1mm, this only represents 50mm of physical space imaged, but is $512 \cdot 512 \cdot 500 \cdot 16/8 = 262$MB in size. A group of 50 patients yields a sizeable 13GB dataset in that case.

2.3 Challenges in Obtaining Medical Imaging Datasets

One of the leading issues with applying deep learning to medical imaging applications is the difficulty in attaining adequate amounts of data. Natural image datasets like the ones mentioned in Chapter 1—PASCAL VOC, ImageNet, and COCO—all have the advantage of being massive sets of images. The 2007 iteration of the PASCAL VOC dataset had $9,963$ images with $24,640$ uniquely annotated objects among 20 classes [29]. At the time of its initial release in 2009, ImageNet consisted of over 3.2 million images with $5,247$ unique labels, where each label has an average of over 600 images each [20]. The standard dataset used for pre-training networks, as well as the ILSVRC classification challenges, used a subset from the ImageNet dataset of $1,000$ classes with over 1.2 million images commonly referred to as ImageNet-1K [9]. ImageNet expanded to contain well over 14 million images with nearly $22,000$ classes; however, these classes are not mutually exclusive like the smaller set and there is heavy class imbalance between labels [56]. While not nearly as large, the Microsoft COCO dataset still had over $328,000$ images with 2.5 million individual annotations of over 90 classes [30]. However, none of these datasets approach the enormity of Google's internal dataset, JFT-300M [57]. This massive image dataset has over 300 million images containing over 375 million labels spread across $18,291$ classes.

For these natural image datasets, bridging the gap from classification tasks to object detection and segmentation is not incredibly difficult. Normal human readers are able to easily discern object classes from one another and either place a box around each one or create pixel-level segmentations of each object. There are various methods these labels can be obtained, such as giving smaller chunks of images to colleagues and friends or crowd-sourcing by providing an online platform for users around the world to submit their labels. Thus, these images are easy to obtain and inexpensive to get labeled. This is unfortunately not the case for medical imaging datasets.

One of the first roadblocks to retrieving medical imaging data is the regulations surrounding

them. In the United States, any information on patients garnered through doctors visits is protected by the Health Insurance Portability and Accountability Act (HIPAA). This includes all types of medical imaging performed, whether it be a radiograph, MRI, or PET scan. In order for medical images to be usable by outside personnel such as machine learning researchers, a request needs to be made to an institutional review board (IRB) and they must grant approval for the data to be released. Then, all potentially identifying patient information needs to be properly anonymized so that any external use cannot possibly be linked back to the actual patient. This is a tedious and time-consuming process, making it quite difficult to acquire data in the first place, let alone large quantities of it. Meanwhile, it is estimated that over 3.6 billion medical diagnostic examinations are completed annually worldwide [58], with approximately 70-80 million chest radiographs and CT scans each year within the United States alone [59, 60].

Even after medical images are granted approval to be used and properly retrieved, there is a chance they cannot be immediately used for computer vision purposes. In some cases, such as for whole-image classification, it is possible that the diagnoses in the charts associated with the images can be extracted; e.g., a patient who received a chest x-ray with pneumonia would have the label extracted simultaneously. However, for localization tasks like object detection and segmentation the problem becomes much more challenging. For each task one wants to apply deep learning to, they must get the hand-labeled annotations on the images which requires specific domain experts. For instance, radiographs containing various types of bone fractures would likely benefit from a musculoskeletal radiologists as opposed to an oncologist who specializes in cancer. Moreover, due to the inherent challenges posed by providing accurate labels on the set of images, the best case scenario for the least noisy labels is to obtain multiple reads on the same images with a final consensus read. This compounds both the requirement in the number of collaborating medical professionals as well as the amount of time it takes to not only acquire the images but to get accurate labels.

Due to these challenges and the increasing interest and demand for applying deep learning methods to medical imaging tasks, there have been a few large-scale imaging datasets released for

public use. One of the most well known chest radiograph datasets is CheXpert [61]. In it, they collected 224,316 chest radiographs from 65,240 unique patients. There are 14 labels, such as pneumonia, pneumothorax, edema, and fracture. In order to obtain ground-truth labels for their data, they used a consensus result from five different board-certified radiologists. DeepLesion [62] is a CT-based dataset released by the NIHCC that comprises over 33,000 images from 4,477 unique patients. They used bookmarked images that contained relevant annotations, such as arrows or diameters of lesions, added by radiologists over time. This dataset serves as a way to measure performance of deep learning models in localizing and detecting lesions of various regions, such as on the liver, lungs, and abdomen.

The data science competition website Kaggle has an extensive collection of datasets for computer vision tasks. Currently, it houses over 2,900 datasets under the "Computer Vision" category. However, when it comes to medical imaging datasets the collection appears to be relatively limited. A search for “x-ray" yields only 48 datasets, while searches for "computed tomography" or "MRI" return 12 and 23 datasets, respectively. Notably, the vast majority of datasets on Kaggle are intended for non-medical applications, demonstrating the divide in publicly available datasets for medical imaging compared to natural imaging tasks.

2.4 Brief Overview of Previous Applications

Given the widespread interest in machine learning and AI, there is an equally wide array of studies applying their automated classification and localization prowess on radiological datasets. Deep learning algorithms have been tasked with classifying breast density, classifying malignancy in lung nodules, identifying various lung disease, and even determining bone age [63]. Below, we provide a few specific examples of classification and detection tasks applied to medical images.

2.4.1 Classification Examples

Deep convolutional neural networks have been able to perform exceptionally well in classifying cases of skin cancer [64, 65]. Esteva *et al.* utilized a GoogleNet Inception v3 architecture pre-trained on ImageNet-1K and fine-tuned on a curated dataset of over 129,000 clinical images showing 2,032 various diseases, with 1,942 test images properly labeled via biopsy. Compared

to expert dermatologists scoring between 65 and 66% on a portion of the validation samples, the deep learning model achieved a result of 72.1 ± 0.9% accuracy. More impressively, the CNN architecture scored above nearly all 21 expert dermatologists who scored a subset of the biopsy-labeled test set, achieving ROC AUC values ranging from 0.91 to 0.96 depending on the type of skin cancer. Similarly, Haenssle *et al.* trained an Inception v4 architecture to classify dermoscopic images of melanocytic lesions as melanoma or benign nevi. Using a test set of 300 images, the CNN achieved a sensitivity of 95%, specificity of 80%, and AUC of 0.95. When tested against 58 dermatologists on a subset of 100 difficult cases, the CNN performed comparably or better than most of the dermatologists, achieving higher specificity at the dermatologists' average sensitivity. In other words, the average sensitivity of the dermatologists was 86.6% with which they achieved a specificity of 71.3% where the Inception V4 network reached 82.5%, yielding a higher ROC AUC of 0.86 compared to the dermatologists' 0.79. When dermatologists were given additional clinical information alongside images, the mean sensitivity increased to 88.9% with a specificity of 75.7%, whereas the CNN at an equivalent sensitivity attained a 82.5% specificity.

There are also many applications of deep learning architectures on more traditional medical imaging modalities such as radiographs, CT, and MRI. Multiple studies have shown the effect of deep classification networks on identifying beast cancer in both mammography and MRI [66, 67, 68]. Hu *et al.* sought to classify 927 unique breast lesions as either benign (N=199) or malignant (N=728). First using an ImageNet pre-trained VGG19 network as a feature extractor and sending the outputs through various SVM classifiers they were able to achieve up to 77.9% sensitivity and 78.5% specificity. Shen *et al.* created a whole-image breast cancer classifier by first training ResNet50 patch classifiers and adding additional convolutional layers. Testing the performance on two datasets, CBIS-DDSM and INbreast, they achieved an ROC AUC score of 0.91 and 0.95 for each, respectively. Whereas these examples used just images for training and evaluation, there has been investigation into the idea of incorporating both image and non-image data for improved classification performance. Holste *et al.* [69] found that on a dataset of over $17,000$ breast MRIs, using just images for a ResNet50 classifier gave an AUC of 0.849 but by adding in a feature vector

extracted from thirty-three non-image inputs, such as patient age and breast density, the AUC increased to 0.898 which could be improved even further through ensembling to 0.903.

The cohort that released the CheXpert dataset from earlier published in-house results of networks on their dataset [61]. They trained a DenseNet121 network on their data and compared the performance to three radiologists in a smaller 500 image subset that they had annotated by five radiologists, making a total of eight radiologists involved in labeling and scoring the test set. Across a handful of the 14 classes, the trained models were able to perform relatively close to the board-certified radiologists with AUCs ranging from 0.85 to 0.97.

2.4.2 Examples of Localization and Detection in Medical Images

There is a wealth of object detection applications on a variety of medical imaging tasks. A review by Mazurowski *et al*. [70] listed 17 studies focusing on detection tasks, compared to 12 on various classification tasks and 13 for segmentation. These detection applications vary wildly by region, such as breast cancers and tumors, pulmonary embolisms, and cerebral hemorrhages. Of the 17, only one was specified as a fracture detector, demonstrating the quite wide range of tasks that object detection can be applied to in the medical field.

When introducing the DeepLesion dataset, the authors trained a Faster R-CNN based model on the dataset for single-lesion and multi-lesion detection. The model obtains a slightly higher detection accuracy when doing multi-class detection compared with single-class, increasing from 59.45% to 64.3% [62]. Prior to conducting the work discussed through the remainder of this dissertation, we compared three different object detection CNNs on their ability to detect lesions. Using a subset from the DeepLesion dataset, we created a training set of $7,500$ lesion-present images and a test set of $1,000$ lesion-present images. Table 2.1 shows the performance of the SSD, YOLOv3, and RetinaNet architectures. Performance varies wildly between the three architectures. YOLOv3, as an example, barely captured any of the lesions present in the test set, but was the most confident in the regions it proposed. RetinaNet performed overall the best, with the highest F1 score of 0.412; even then, it was only able to capture just below 34% of all lesions. Figure 2.3 provides a single slice and the detection performance from each of the models.

Table 2.1 Performance of SSD, YOLOv3, and RetinaNet at detecting lesions on a subset of the DeepLesion [62] dataset.

	SSD	YOLOv3	RetinaNet
Precision	0.657	0.909	0.526
Recall	0.241	0.010	0.339
F1 Score	0.353	0.019	0.412

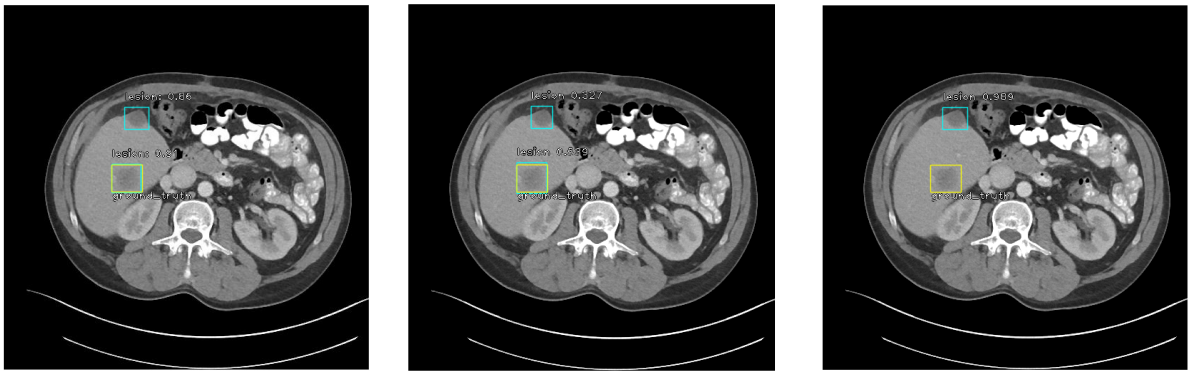


Figure 2.3 Example slice from the DeepLesion test dataset with ground truth (yellow) and predicted (teal) boxes from SSD (left), YOLOv3 (center), and RetinaNet (right).

Numerous deep learning methods for rib fracture detection have been developed in the last 5 years for volumetric CT images [71, 72, 73]. Yao *et al*. implemented a three-stage process for rib fracture detection, beginning with a U-Net for bone segmentation of the CT image, isolating the ribs and removing additional bony structures such as scapulae, and classifying whether a fracture was present via a 3D DenseNet [74]. A similar approach was taken by Zhang *et al*. utilizing a nnU-Net to segment areas of ribs that may contain a fracture and a secondary stage with a DenseNet to classify the segmented region [75]. MICCAI hosted the RibFrac challenge in 2020 that invited methods to detect the location and classify fractures into four clinical categories on CT images [76]. The three leading methods from this challenge used RetinaNet or variations of masked R-CNNs. Fewer methods have been developed for rib fracture detection on 2D radiographs.

There have been substantial efforts to apply deep learning to chest x-ray images, thanks partially to large publicly available data sets of common chest pathologies [61, 77]. These efforts largely focus on classification of the image and on lung diseases such as fibrosis, pneumonia, and COVID-19 [78]. There have been successful efforts for improved wrist fracture detection on radiographs [79], but

there has been less effort focused on rib fracture assessment using radiographs [80]. Gao *et al*. performed rib fracture detection on radiographs with their proposed CCE-Net where multiple feature extraction modules were fused together as inputs to a two-stage detection network demonstrating improved performance compared to competing R-CNN and YOLOv4 architectures [81]. At the time of writing this dissertation, we are not aware of any other published methods attempting to automatically detect the location of rib fractures on pediatric chest x-rays.

2.5 Challenges of Pediatric Rib Fracture Detection in Medical Imaging

Detection of rib fractures on pediatric chest radiographs is challenging for a host of reasons [82]. Depending on the child's orientation to the imaging detector, a region containing a fracture on the rib can be obscured by the same rib. For example, on an AP-view radiograph a fracture occurring at posterior arc of the rib can be covered by the posterior part of the rib, making it particularly difficult to identify especially if the fracture is not recent. The age of fractures also make identification difficult. Acute rib fractures may be undetectable on chest radiographs and in some cases only become evident after callus formation (new bone growth) develops 10 to 14 days into the healing process [83]. There are also age-related variations in the rib structure; in children, each rib can grow by nearly 10% per year with middle ribs growing the fastest [84]. Lastly, overlying anatomy and artifacts, such as monitor leads, support devices, and clothing, can create further perceptional differences that can hinder outcomes of the interpretations. It is therefore not surprising that missed rib fractures in children are quite common and that the sensitivity for detection of any rib fracture on pediatric chest radiography by experts is only about 31% [85, 86]. In practice, it is desired to have multiple expert radiologists read images and use either a combination of annotations or a separate consensus read to act as ground truth. As we reveal later in Section 4.2, even intra-reader variability on these difficult pediatric datasets can still leave a lot of fractures missed.

As was discussed previously, the a major challenge to overcome in object detection is access to large enough datasets, especially for medical image applications. This challenge grows even more when looking specifically at pediatric datasets, considering children are a vulnerable population and also generally have less medical imaging encounters than older patients. Ghosh *et al*. presented

results for a similar task as rib fracture detection; specifically, they perform classification of sub-patches of the radiograph, which provides coarse localization of objects. For their study, they were also challenged by dataset volume and were only able to procure a total of 415 chest radiographs of children under the age of 2 including chest radiographs, which were imaged over the course of 18 years [87].

CHAPTER 3

PEDIATRIC RIB FRACTURES - DATA AND PATTERNS

3.1 Background

Child physical abuse is a serious problem with over 108,000 confirmed cases in the US in 2020, leading to an estimated 1,750 deaths (2.38 for every 100,000 children) from abuse and neglect [88]. This issue becomes even more severe when focusing on children under the age of 3, where over 70% of infant deaths were from children 3 years and younger [89]. Bone fractures are the second most common presentation for abused children, after soft-tissue injuries [90]. The most common fracture site in abused children is the ribcage, accounting for over 70% of fractures [91]. In young children, studies have shown rib fractures are a result of child abuse 80-100% of the time [92, 93] and is 23.7 times more likely from abuse than from an accident [94]. Barsness *et al*. [95] found that out of 78 children, 62 were under the age of 3 but represented over 94% of the total number of rib fractures. After removing accidental or disease-related fractures, they calculated a 100% positive predictive value for rib fracture occurrence stemming from abuse. In short, rib fractures are extremely important to detect as they are a sentinel injury for physical abuse in young children that portend poor outcomes; a single rib fracture in children associated with a 2.5 times increase in mortality rate [96].

Detection of rib fractures in pediatric radiographs is difficult. Expert, specialized radiologists performing their first reads on x-rays can miss up to two-thirds of all present rib fractures [97]. The skeletal structure of small children is not the same as adults; rather than being rigid, bones are much more elastic and flexible. This causes breaks to appear differently as opposed to adults where a rib fracture is usually a much more defined break. Additionally, small children may not have their chest imaged immediately after injury considering they are unable to express the cause for discomfort. If rib fractures were indeed present, it would likely not coincide with the timing of the imaging and they may present in various stages of healing. In addition to the base challenge of reading these exams, there are decreasing numbers and availability of radiologists to interpret these studies. While the raw number of trained radiologists has increased 39.2% from 1995 to 2011, the

ratio of radiologists to general physicians has decreased from 4.0% to 3.7% in the same time [44]. With the ever-increasing application of deep learning to the medical field, the implementation of computer vision models as a first-read augmentation technique could improve both detection of rib fractures and speed of interpretation of radiologists, especially for non-expert readers.

Moreover, a driving motivation for our work is to improve the sensitivity of fracture detection to ensure that few to no fractures are missed during interpretation. The pediatric rib fracture detection task warrants methods that are high sensitivity, even if they have reduced precision, because of the high positive-predictive value of rib fractures as an indicator of child abuse and the downstream risk of missing a fracture is considerable [95]. Automatic object detectors with high sensitivity (identical to high recall) have the potential to augment radiologist performance by flagging multiple suspicious regions in the image leading to higher sensitivity interpretations.

3.2 Dataset Curation and Pre-Processing

Our pediatric rib fracture dataset was collected in collaboration with Seattle Children's Hospital and approved by their institutional review board (STUDY00000853) with informed consent waived due to the study design. Data were collected for this minimal risk retrospective analysis and all methods and experiments in this study were carried out in accordance with relevant guidelines and regulations of the institution (Seattle Children's Hospital). In this convenience sample, we first searched the medical record for patients with chest radiographs with confirmed rib fractures and identified 624 cases in the period of Aug 15, 2006 through Aug 15, 2017. Gender and age statistics for these fracture-present studies were extracted. An age- and gender-matched sample of chest radiographs with no rib fractures was created. Chest radiographic images were extracted from the medical record and fully anonymized. These images had (height by width) dimensions on average of 2348 ± 685 by 2134 ± 500 pixels with pixel spacing of 0.128±0.023 mm. All images were quantized from 12- or 16-bit integer to 8-bit integer precision and analyzed with their original pixel spacing.

The images we have are chest radiographs in an anterior-posterior perspective (front-to-back). The files were provided in Digital Imaging and Communications in Medicine (DICOM) format,the

standard file type for storing medical imaging data and related information. There is four-tiered hierarchy that governs the organization of DICOM files. First is the patient receiving the imaging procedure. The second is the study, or the specific imaging procedure being performed on the patient (e.g. radiograph, PET, CT, MRI). Third is series, which could be different regions being images for the given study, such as extremities, head, abdomen, etc. The last are the instances, which are individual slices or images performed for that series. In a CT scan, for instance, each instance corresponds to a single slice of a 3D CT volume; a single series could contain hundreds of instances for a full-body, high resolution CT scan.

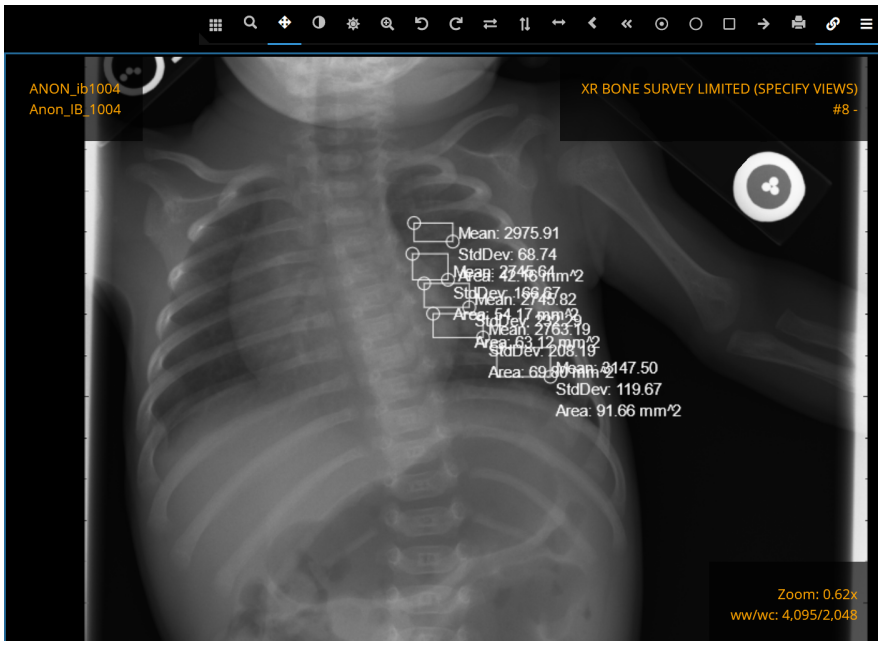


Figure 3.1 The Orthanc web viewer for providing annotations.

In order to perform object detection, we needed a way to get ground-truth annotations for all images. Despite anonymizing each image by removing any and all protected health information, we still did not want to distribute all DICOM files themselves. Another prohibitive factor is the size of the dataset, which would require long data transfers to and from colleagues. Orthanc [98], an open-source, locally-hosted web server, allows us to locally upload all DICOM images and colleagues gain access to them through a browser. An example of the web viewer interface colleagues used for annotating is shown in Figure 3.1.

In total, the dataset contains 1,109 unique patients, of which 624 are fracture present and 485 are fracture absent. There are 385 (34.7%) female and 724 (65.3%) male patients. The average age of patients is 281.74 ± 769.42 (range 0-7300; median 84; IQR 224) days. In order to perform and evaluate object detection, we obtained hand-drawn ground-truth annotations for all images. In short, the fracture present cases were each interpreted by one of seven board-certified pediatric radiologists with 5-20 years of experience. During interpretation, the radiologists had access to all of the available radiographic views of the chest (usually supine anterior-posterior (AP) although occasionally other views were available). They were instructed to draw bounding boxes as closely around each detected fracture as possible on the AP view only; the object detection methods discussed below were only applied to the AP view image. From our total batch of 624 fracture-present images that had a single interpretation each, a subset of 338 were given a second examination by a second board-certified pediatric radiologist to enable estimation of inter-reader variability. This inter-reader variability estimate served as a performance baseline for evaluating the proposed methods.

We created custom scripts using RestAPI to pull the bounding box locations drawn by the radiologists to our workflow. Individual JSON files were made for each patient that saved the associated top left x and y and bottom right x and y values of each bounding box.

3.3 U-Net Segmentation and Cropping

An inherent issue with medical images is that there tends to be a large amount of unnecessary regions surrounding the object or region of interest. Because all of this space provides no additional information but requires larger image sizes and more space, we wanted get as closely around the thoracic region as possible. To achieve this, we utilized a U-Net style model [99] we trained and evaluated to segment chest radiographs into multiple anatomic regions. A custom MATLAB tool was developed to enable annotation of target labels. Three trained users drew boundaries around the lungs, spine, and bottom of the thorax. The tool smoothed all drawn lines and used these user-drawn boundaries to divide the chest into seven non-overlapping regions: left and right lung, left and right "subdiaphragm" (the thorax below the superior boundary of diaphragm), spine,

mediastinum, and background. In total, 469 radiographs were manually labeled for segmentation. This is an extension of the work presented by Holste *et al.* [100].

We adapted a U-Net3+ architecture which includes full-scale skip connections [101] and report Dice coefficients (discussed in Section 1.5.6) on the test set. After each inference from the U-Net3+ model, the proposed segmentation maps are refined with basic morphological operations to remove small spurious disconnected regions from background and close all foreground regions. We also utilized per-image pixel spacing to add 5mm of physical space to all four sides of the segmented region in order to account for possible fracture bounding boxes along the edge.

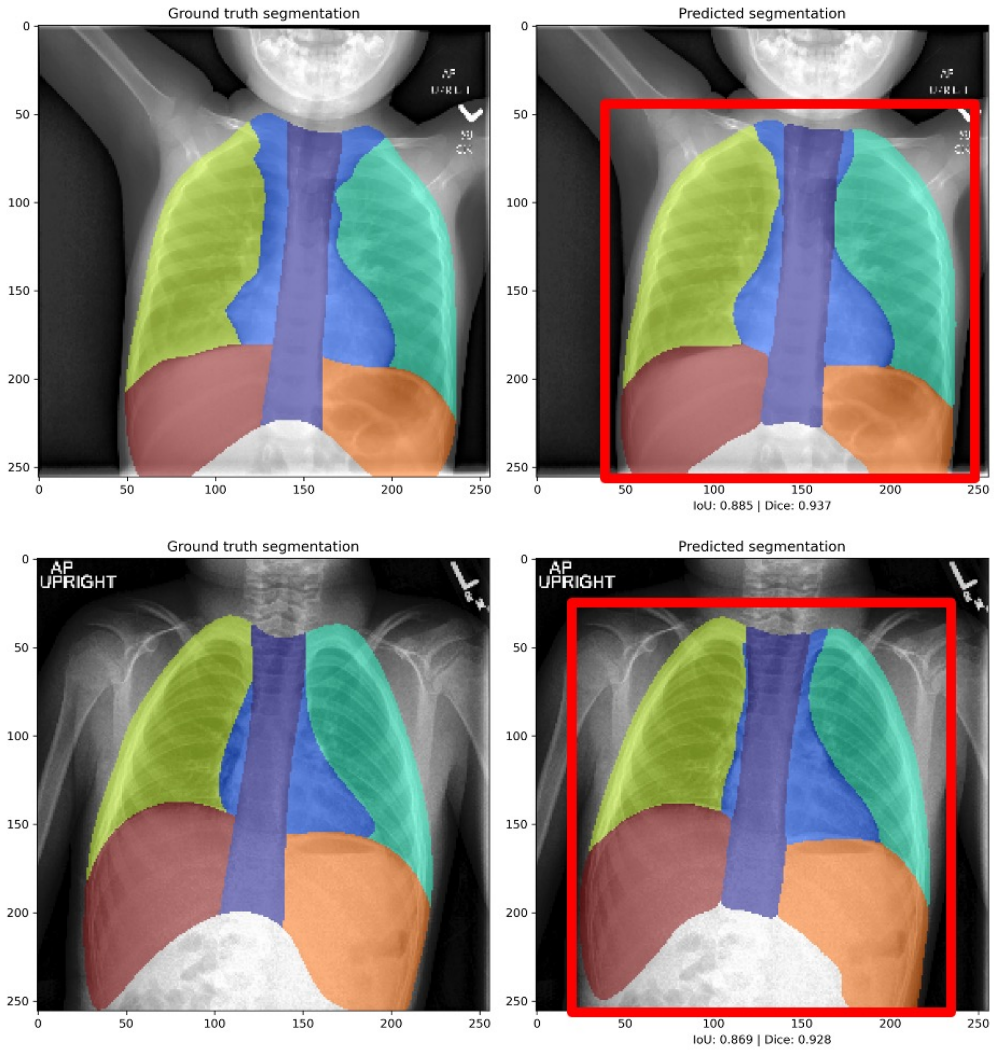


Figure 3.2 Representative results from multi-class segmentation showing manually labeled images (left) and U-Net results (right) with final automatically-generated cropped region represented by red box.

Of the 469 labeled images, the U-Net3+ model was trained with 422 and tested on the remaining 47 images. Representative segmentations from the test set are presented in Figure 3.2. The mean Dice coefficient for each of the seven regions exceeded 0.88 and visual assessment confirmed that all final cropped images in the test set contained only the thoracic cavity.

3.4 Fracture Prevalence

Before we could take a deeper dive into the ground truth data and identify any potential underlying patterns, we need to understand the previously established literature on rib fracture patterns in pediatric patients. Males tend to make up a majority of cases of non-accidental rib fractures in children, with percentages ranging anywhere from the mid 50's up to 68% of patients [102, 95, 103, 104, 105, 106]. Various studies have looked at different ranges of children, such as 0-12 months [102], 0-24 months [103], and 0-36 months [107], while a few had extended ranges up through 18 years of age [95, 108, 109, 110]. The common discovery among these studies is that rib fractures tend to be most common in children under 2 years of age, with an especially high rate in infants under the age of 12 months.

It has been repeatedly found that up to 80% of patients present with multiple rib fractures [103, 108, 110]. This is an especially salient statistic as multiple rib fractures has high specificity for child abuse [111, 112]. There tends to be an association with which ribs fractures frequently occur. More specifically, prior studies have shown the fifth through eighth ribs to be the most likely to present fractures [105, 103, 102]. However, there does not appear to be a definitive location along the ribs where fractures are most common. Aboughalia *et al.* , Barber *et al.* , and Barsness *et al.* all noted higher rates of fracture in the posterior arc, whereas Kriss *et al.* saw higher lateral fractures [113, 102, 95, 103]. Sidedness is another common pattern described, with studies showing anywhere between 1.4-2 times higher likelihood of fractures on the patient's left side as opposed to their right, accounting for 59-67% of all fractures.

3.4.1 Stratifying Data into Age Groups

Starting with our 624 unique patients containing at least one rib fracture, 17 patients were above the age of 2 years (ranging from 2-21 years old) and were removed, resulting in 607 patients under

the age of 2 that we use for the rest of this section. This data consists of 399 (65.73%) males and 208 (34.27%) females with an average age of 120 ± 105 days.

As previously mentioned, most of the established literature on the patterns of rib fractures in children point out the higher tendency of presentation in children under 18 months of age. Multiple studies investigated further by breaking up the cohort into different age groups. Quiroz *et al.* used groups of 0-12 months and 12-48 months for younger patients [110]; this was similar to Worlock *et al.* that used splits of 0-18 months and 19-60 months [106] in their study. The splits from Loder and Feinberg stayed slightly younger at 0-12 months and 12-24 months, with older patients being in much larger groups of 3-12 years and 13-20 years [109]. These helped influence the range of ages we split our data into to further investigate fracture prevalence.

Table 3.1 Patient sex and age summary for the age-separated groups of the fracture-present data set used for investigating fracture prevalence.

Age Group	Patients	Male	Female	Mean Age (days)
0-3 months	116	70 (60.34%)	46 (39.66%)	40 ± 16
3-6 months	112	67 (59.82%)	45 (40.18%)	105 ± 22
6-12 months	58	50 (86.21%)	8 (13.79%)	222 ± 42
12-24 months	16	8 (50.00%)	8 (50.00%)	435 ± 102
Age Unknown	305	204 (66.89%)	101 (33.11%)	–

Because of the size of our dataset, we were able to create relatively smaller age groups (0-3 months, 3-6 months, 6-12 months, and 12-24 months of age) than prior literature. It should be noted that while the largest group is of patients with unknown ages, these patients were pulled alongside the other examples in our dataset and likely correlate with the age ranges shown. For groups with known ages, the largest were the 0-3 month and 3-6 month groups with 116 and 112 patients, respectively, accounting for over a third of our overall data. The calculated mean ages for each group are estimates, as values in the DICOM headers with week, month, and year markers were given approximate day equivalents (i.e., if a patient's age was marked as "2M" the output would be 56 days).

Much like previous studies, our overall dataset exhibits the same male majority with over 65% of our patients being male. This gender skew is even higher in the 6-12 month age group, where

over 86% of patients were male. Only in the 12-24 month group with the fewest number of overall patients were the gender ratios even. While there may not be a definitive reason provided for why this slant exists, it is reassuring that our data aligns with previous investigations.

Table 3.2 shows the rib fracture statistics per age group. Looking first at patients with a known age, the group with the highest number of annotated fractures were patients 3-6 months old, with 520 total fractures averaging 4.64 per image. Patients aged 6-12 months saw the highest count of multiple fractures averaging 5.64 per patient and exhibited a staggering inter-quartile range of 10 whereas the next highest IQR was 5 (3-6 months and unknown age).

The number of left and right side fractures was calculated by finding the centroid of the thoracic cavity and dividing the fractures based on the side from the centroid they belonged to. The overall dataset has a slight left-side skew to fracture occurrence, 55.6% compared to 44.4%. However, patients over the age of 6 months had a right-sidedness, especially in the cases of patients older than 1 year of age (72.7%). Patients with unknown ages had the highest left-sidedness with over 60%.

Table 3.2 Fracture statistics for each age group.

Age Group	Fractures (per patient)	Median	IQR	Left Fractures	Right Fractures
All Patients	2770 (4.56)	3	4	1540 (55.6%)	1230 (44.4%)
0-3 months	402 (3.47)	3	4	227 (56.5%)	175 (43.5%)
3-6 months	520 (4.64)	3.5	5	267 (51.3%)	253 (48.7%)
6-12 months	327 (5.64)	3	10	137 (41.9%)	190 (58.1%)
12-24 months	22 (1.38)	1	1	6 (27.3%)	16 (72.7%)
Age Unknown	1499 (4.91)	4	5	903 (60.2%)	596 (39.8%)

3.4.2 Translating Fractures onto Reference Images

To get a visual sense of where fractures are more likely to present, we developed a method to combine all ground truth annotations onto one image. For a given reference image, we utilized its associated segmentation mask from the pre-processing cropping stage to locate the centroid of the rib cage. We then chose an inner boundary of the segmentation to contain 85% of the width and height pixels to create what we call the interquartile range (IQR) of the segmentation. This can be

(a) (b)

Figure 3.3 (a) color-separated segmentation mask from the U-Net trained in Section 3.3. (b) stacked segmentation mask with centroid (blue dot) and IQR box (teal).

illustrated by Figure 3.3(b) with the centroid on the mask shown as a blue dot and the inner 85% IQR box of the mask in teal. This IQR range and the previously computed centroid are used to create offset ratios for each of the bounding boxes in each image. These ratios were computed as follows:

$$R_x = \frac{B_x - C_x}{IQR_x}, \tag{3.1}$$

$$R_y = \frac{B_y - C_y}{IQR_y} \tag{3.2}$$

where R_x and R_y are x and y ratios from the centroid, respectively, B_x and B_y are the center pixels of the ground truth bounding box, C_x and C_y are the centroid of the image's segmentation mask, and IQR_x and IQR_y are the interquartile ranges of the x and y pixels of the segmentation mask, respectively. Once these ratios are computed for each bounding box on their respective images, we then choose one of the images to serve as a reference image to place all of the bounding boxes onto. Once that reference image is chosen and the centroid and IQR for it are calculated, the new locations of the bounding boxes on the reference image can be calculated by solving for a new B_x and B_y, i.e.,

$$B_x^{new} = R_x \cdot IQR_x^{\text{ref}} + C_x^{\text{ref}}, \tag{3.3}$$

$$B_y^{new} = R_y \cdot IQR_y^{\text{ref}} + C_y^{\text{ref}}, \tag{3.4}$$

Figure 3.4 shows every hand-annotated rib fracture in our dataset superimposed onto a single reference image. These images are anterior-posterior (AP) views, thus the patient's left side is on the right side of the image. The heatmap colors were generated by creating a circle of a fixed radius around the center of every fracture location after being mapped to the reference image and stacking them onto a new array. A greater number of overlapping fractures is indicated by the brighter colors on the image as shown in the accompanying colorbar.

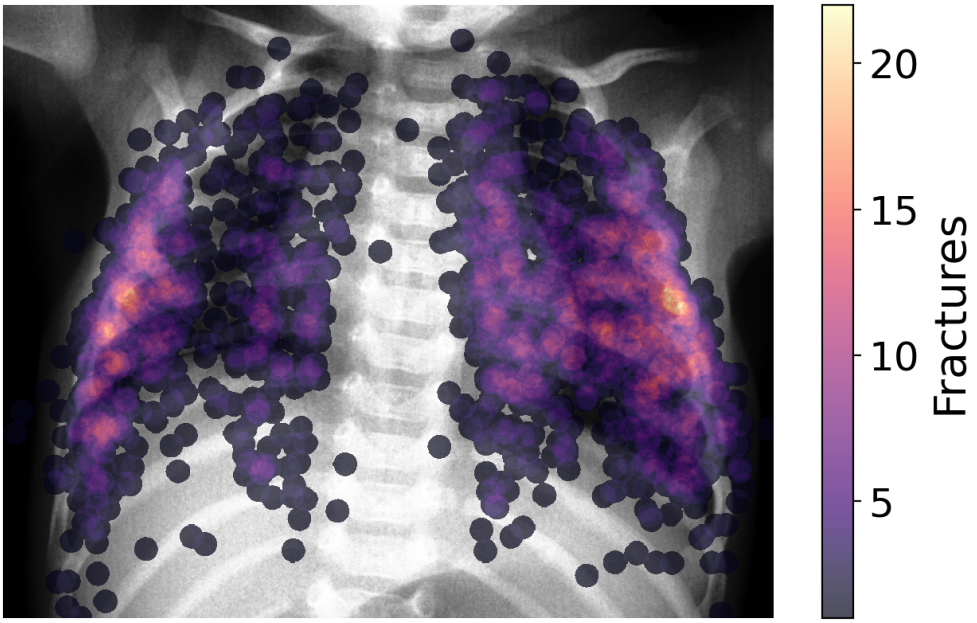


Figure 3.4 Fracture prevalence map of all fractures in the dataset combined onto a single reference image.

When looking at all fractures mapped onto a single image, the location of fractures are spread fairly evenly across the ribs. There does not appear to be a specific region over another or specific ribs for fracture prevalence. The lateral regions of the ribs do appear to have slightly higher concentrations of fractures opposed to the anterior or posterior parts of the ribs, i.e., the regions closer to the sternum (anterior) or spine (posterior). Due to limited knowledge from our annotation process, we are not able to specify whether fractures are on the anterior or posterior of the rib; going forward, we will be labeling fractures in these areas as either the inner or horizontal parts of the rib.

After splitting the fractures into their respective age groups and mapping onto a reference image

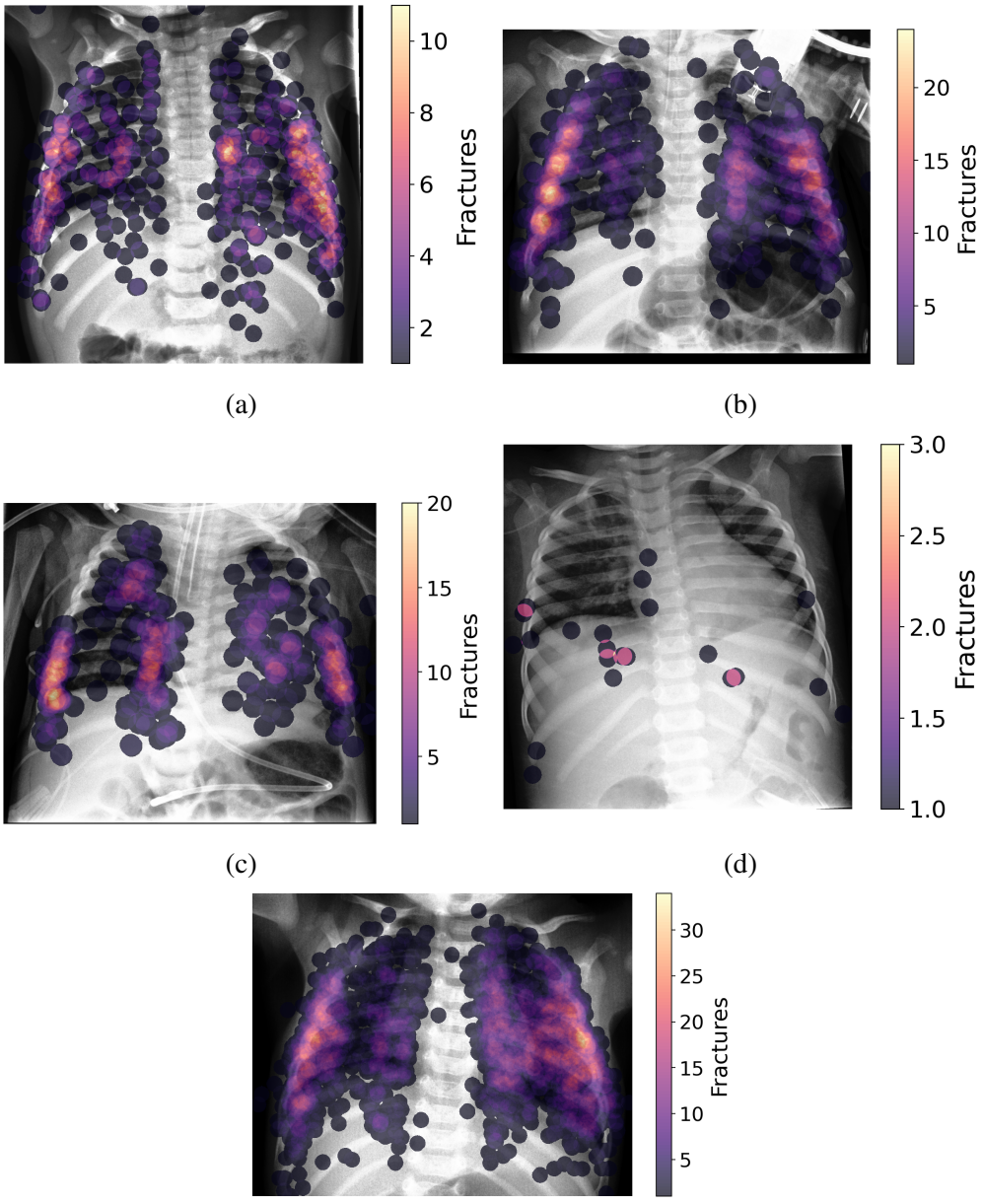


Figure 3.5 Fracture prevalence maps for all age groups: (a) 0-3 months, (b) 3-6 months, (c) 6-12 months, (d) 12-24 months, (e) age unknown.

within each set, there are more noticeable regions of prevalence. In Figure 3.5(a) with patients aged 0-3 months, there seems to be a left-sidedness to the fractures with a heavy emphasis on the lateral regions of the ribs. The 3-6 month group also shows a higher prevalence for the lateral region, but instead on patient right side. An interesting separation of fracture clusters appears in the right side of the 6-12 month group with highest overlap along the lateral but also high overlap on the inner

ribs. It is in the 3-6 month and 6-12 month groups (Figures 3.5(b) and (c)) that we see a pattern of specific ribs that matches the established literature, where the highest concentration of overlapping ribs appear on the fifth through ninth ribs.

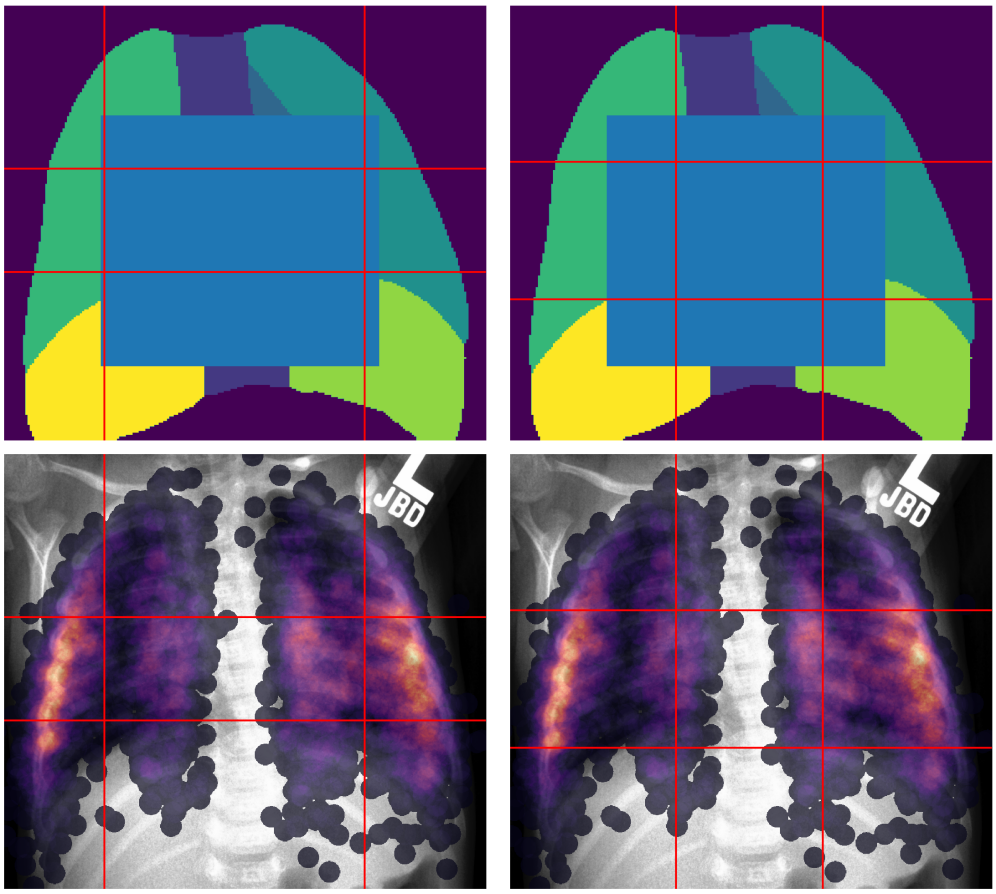
Despite there being a comparative lack of patients and therefore fractures in the 12-24 month group, there is a definite right-sidedness in these patients, going against the studied behavior of left-sidedness. Overall, the age unknown group in Figure 3.4(e) presents very similarly to Figure 3.5 with all fractures, likely because approximately half of all fractures in our dataset exist in this group. Though, there is still a slight bias toward the patient's left side.

3.4.3 Three-by-Three Split Grid Prevalence Maps

While the prevalence maps provide an overall presentation of the patterns of fractures across the stratified patient groups, we wanted to investigate further to see if we could more objectively correlate findings with the previous studies, especially in terms of the most common ribs fractured and the lateral versus inner rib (again, we note that the literature is able to differentiate posterior versus anterior for the inner parts of the rib while we have to combine both). To achieve this, we integrated an image-splitting mechanism to break each of the fracture prevalence maps into a three-by-three grid, where we could then calculate the presence of fractures individually within each.

Figure 3.6 visualizes two of the ways the prevalence images could be split. In (a), the top horizontal split was found by calculating the centroid of the segmentations of the left and right lungs and setting it as the higher of the two, i.e., $\left[\min(C_y^{\text{right lung}}, C_y^{\text{left lung}})\right]$. For the lower horizontal split, the highest pixel location based on the left and right subdiaphragms is used. The left vertical index is based on the right lung's centroid value, $C_x^{\text{right lung}}$, and the right vertical index is from the left lung's centroid, $C_x^{\text{left lung}}$. In the bottom image in Figure 3.6, it does appear to separate the lateral regions of the ribs from the central region of the rib cage. However, this creates a very wide central region with comparatively small left and right thirds, leading to quite large center top and center bottom segments and much smaller middle left and middle right segments.

Method (b) is much simpler. From the combined segmentation mask of the thoracic cavity,



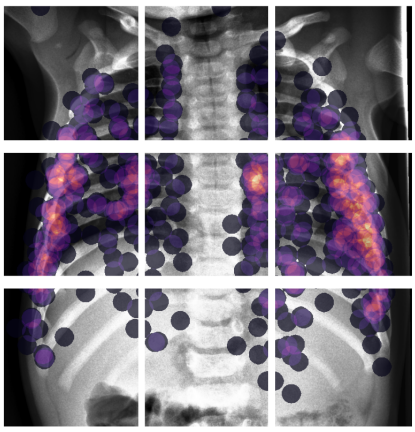
(a) (b)

Figure 3.6 Splitting techniques for creating a 3x3 grid for fracture prevalence images. (a) uses the left and right lung centroids and top of the subdiaphragms for indices, (b) takes the height and width indices of the thoracic segmentation mask and splits them into three even sizes.

the height and width indices are extracted and the left vertical split index is the value one-third through the index array, and the right vertical is the value two-thirds through the array. This process is replicated for the two horizontal splitting indices. We proceeded with this method both for its general simplicity and its consistency across images.

Looking across the five age group's respective three-by-three grids, we can more easily extract certain patterns in the prevalence of fractures. In every case, the highest horizontal third is the middle third, ranging from 64.78% at the lowest across all age unknown cases to 77.27% in the 12-24 month group. This middle region accounts for the ribs previously mentioned to have the highest likelihood of fracture presentation: ribs 5-8.

Multiple studies mentioned the left-sidedness in fracture presentation, and this is shown in most



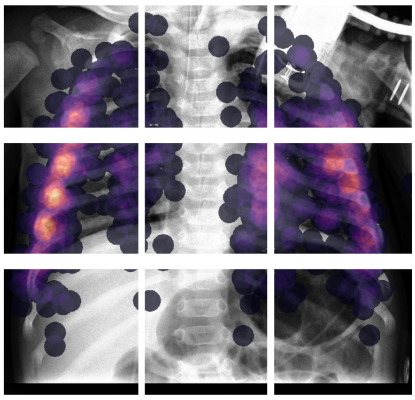
Fractures

	R		L	SI Total
S	26	28	20	74
	96	51	124	271
I	17	11	29	57
LR Total	139	90	173	402

% Fractures

	R		L	SI Total
S	6.47%	6.97%	4.98%	18.41%
	23.88%	12.69%	30.85%	67.41%
I	4.23%	2.74%	7.21%	14.18%
LR Total	34.58%	22.39%	43.03%	100.00%

Figure 3.7 Three-by-three grid of fracture prevalence for the 0-3 month age group.



Fractures

	R		L	SI Total
S	60	22	42	124
	147	59	143	349
I	21	7	19	47
LR Total	228	88	204	520

% Fractures

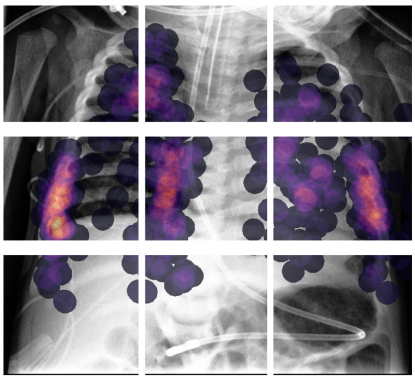
	R		L	SI Total
S	11.54%	4.23%	8.08%	23.85%
	28.27%	11.35%	27.50%	67.12%
I	4.04%	1.35%	3.65%	9.04%
LR Total	43.85%	16.92%	39.23%	100.00%

Figure 3.8 Three-by-three grid of fracture prevalence for the 3-6 month age group.

of the groups with 43.03% of fractures in the 0-3 month group, 39.76% in the 6-12 month group, and 50.63% in the age unknown group. The largest outlier is the 12-24 month group with 50% fracture in the left vertical third, with the 3-6 month group also showing 4.62% more fractures in the left third than the right vertical third.

3.5 Summary

This chapter outlines the custom data collection, labeling, and summarization efforts to support the proposed rib fracture detection methods presented in the following chapters. The major work presented here includes the development of a system to collect expert labels from pediatric radiologists. This chapter also presents a method to map labeled fractures to a shared coordinate system to enable analysis of fracture prevalence patterns. The results found when investigating



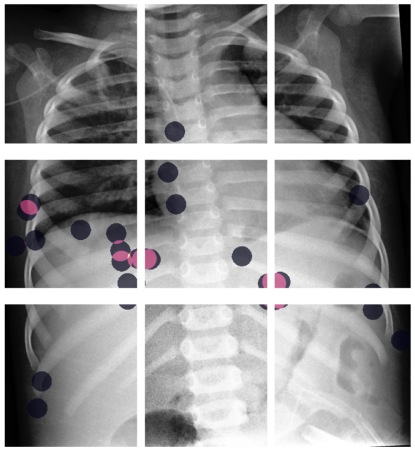
Fractures

	R		L	SI Total
S	15	29	20	64
	68	65	100	233
I	11	9	10	30
LR Total	94	103	130	327

% Fractures

	R		L	SI Total
S	4.59%	8.87%	6.12%	19.57%
	20.80%	19.88%	30.58%	71.25%
I	3.36%	2.75%	3.06%	9.17%
LR Total	28.75%	31.50%	39.76%	100.00%

Figure 3.9 Three-by-three grid of fracture prevalence for the 6-12 month age group.



Fractures

	R		L	SI Total
S	0	1	0	1
	9	5	3	17
I	2	0	2	4
LR Total	11	6	5	22

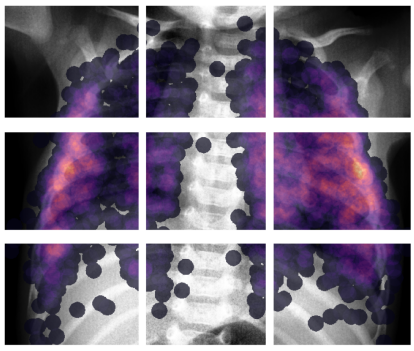
% Fractures

	R		L	SI Total
S	0.00%	4.55%	0.00%	4.55%
	40.91%	22.73%	13.64%	77.27%
I	9.09%	0.00%	9.09%	18.18%
LR Total	50.00%	27.27%	22.73%	100.00%

Figure 3.10 Three-by-three grid of fracture prevalence for the 12-24 month age group.

fracture prevalence corroborates previous literature on the study of patterns of fractures in physically abused children. Furthermore, insights gained from these prevalence patterns, as well as a direct integration of the segmentation-guided shared coordinate system, are used later in Chapter 5. Work demonstrated in this chapter contributed to the following academic product:

- Gongol, Burkow, Junewick, Simms, Alessio, "*Pediatric rib fracture prevalence patterns on chest radiographs*." In preparation for Pediatric Radiology, 2024.



Fractures

	R		L	SI Total
S	70	99	114	283
	308	145	518	971
I	86	32	127	245
LR Total	464	276	759	1499

% Fractures

	R		L	SI Total
S	4.67%	6.60%	7.61%	18.88%
	20.55%	9.67%	34.56%	64.78%
I	5.74%	2.13%	8.47%	16.34%
LR Total	30.95%	18.41%	50.63%	100.00%

Figure 3.11 Three-by-three grid of fracture prevalence for the age unknown group.

CHAPTER 4

IMPROVING ONE-STAGE DETECTORS

4.1 Training and Evaluation Setup

To perform detection on our radiographs, we chose to evaluate two state-of-the-art single-stage detection architectures, RetinaNet and YOLOv5. Both models used were adapted from public GitHub repositories, from user yhenon for RetinaNet [114] and from Ultralytics for YOLOv5 [33]. These are the hyperparameters and training setups we used for all runs.

For RetinaNet, we used a ResNet-50 backbone with pre-trained weights on ImageNet-1K. We trained all RetinaNet models on our dataset using a NVIDIA V100S for a maximum of 300 epochs at a batch size of 8 using the Adam optimizer. The learning rate was set initially to 0.0001 and was decreased by one-tenth if validation performance did not improve within 4 epochs. The dataset was augmented with a 50% chance of applying any of the following transformations to each image: shift/scale/rotate, horizontal flip, random brightness, random contrast, or Gaussian blur. Training would cease via an early stopping clause if performance on the validation set had not improved in 30 epochs.

We used the large YOLOv5-L6 model pre-trained on the COCO dataset prior to training on our dataset. Similarly, all YOLOv5 models were trained on a NVIDIA V100S for a max of 300 epochs and batch size of 8. A stochastic gradient descent (SGD) optimizer was used with momentum 0.937 and weight decay of 0.0005. Learning rate was initially 0.01 and decreased linearly each epoch. There was also an early stopping feature that stopped training if no improvement in validation was observed after 100 epochs.

Twenty percent of the total dataset were withheld as the fixed test set (N=222 images), with half randomly drawn from fracture-present images and the other half randomly drawn from fracture-absent images. While we know this split is heavily skewed toward the presence of fracture containing images compared to real-world data, we want to ensure the architectures get enough information to aid in discerning fracture present from absent. The remaining 80% of data was then used to create the training (70%; N=776) + validation (10%; N=111) sets. All evaluations were performed after

training with a 10-fold cross-validation strategy in order to examine the range in model performance; the ten separate training and validation sets were randomly drawn with replacement between each set. It is worth noting that during the sampling process, unique patients were sampled rather than bounding boxes to ensure that no images would overlap between training, validation, and test sets.

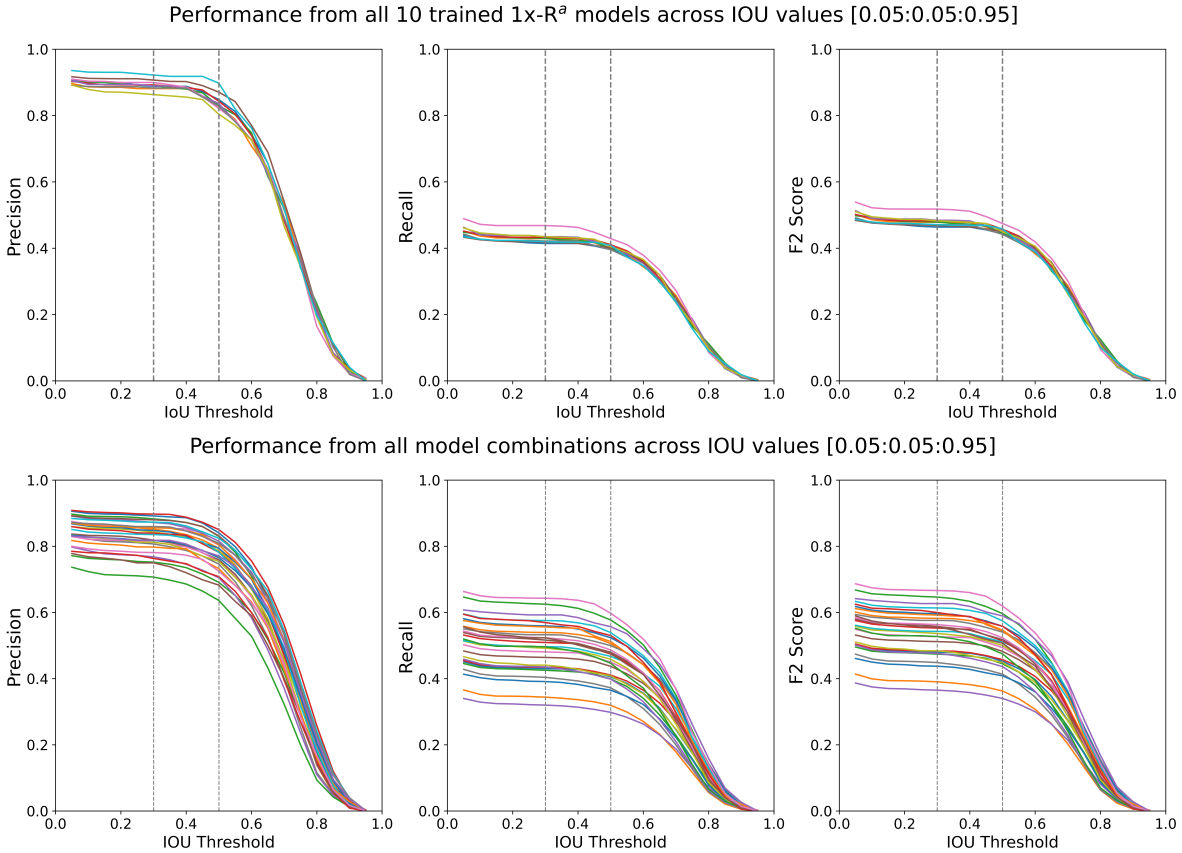


Figure 4.1 Summary of performance metrics versus IOU setting, which dictates whether a proposed region matches an expert labeled region. The top row presents results from single-model RetinaNet and the bottom row presents results from a suite of all different models. For all models, performance is very similar for IOU thresholds ranging from 0.1 to 0.4. We selected an IOU setting of 0.3 as a compromise that is slightly higher in this range requiring more overlap for concordance. It should be noted that unlike many object detection tasks in the computer vision literature, rib fractures generally do not have clear margins leading to clear, unambiguously defined bounds. For this reason, IOU thresholds below the conventional 0.5 setting are warranted.

Object detection performance was evaluated in terms of recall, precision, and F2 score on the fixed test set (as discussed in Section 1.5.3, we use F2 score to weight recall twice as much as precision). An intersection-over-union (IOU) threshold of 0.30 was applied across all model and ensemble evaluations to identify concordance between model predictions and labeled annotations.

Figure 4.1 provides rationale for the selection of this IOU threshold. We used F2 score rather than F1 to give sensitivity (i.e., recall) twice the importance of precision, considering this task warrants high sensitivity performance as discussed above. Max F2 scores are also provided for each combination by finding the highest F2 score achieved across all potential decision thresholds. Furthermore, to summarize average performance across a range of settings, we calculated mean average precision (mAP) by computing the areas under multiple precision-recall curves generated at IOU thresholds ranging from 0.25 to 0.75 in 0.05 increments. Note that mAP was not be calculated for the avalanche decision schemes since precision-recall curves are not analogous between fixed and dynamic decision thresholds.

For single-model calculations, we evaluated performance metrics for all ten trained models (trained on each of the ten folds) and report the average ± standard deviation across these models. For two-, three-, and six-model ensembles, twenty ensemble combinations were arbitrarily selected from the multitude of different ways to combine 2, 3, or 6 models from the 10-fold data; In other words, twenty model combinations were taken from the 10 choose 2 (45), 10 choose 3 (120), and 10 choose 6 (210) possible combinations, respectively, that were then evaluated and averaged. This random selection of twenty models was also applied to the three-model ensembles that incorporate multiple image processing techniques, which yield over 1,000 total possible combinations, as well as the hybrid-model, hybrid-input ensembles where there are as many as 1,000,000 total possible combinations.

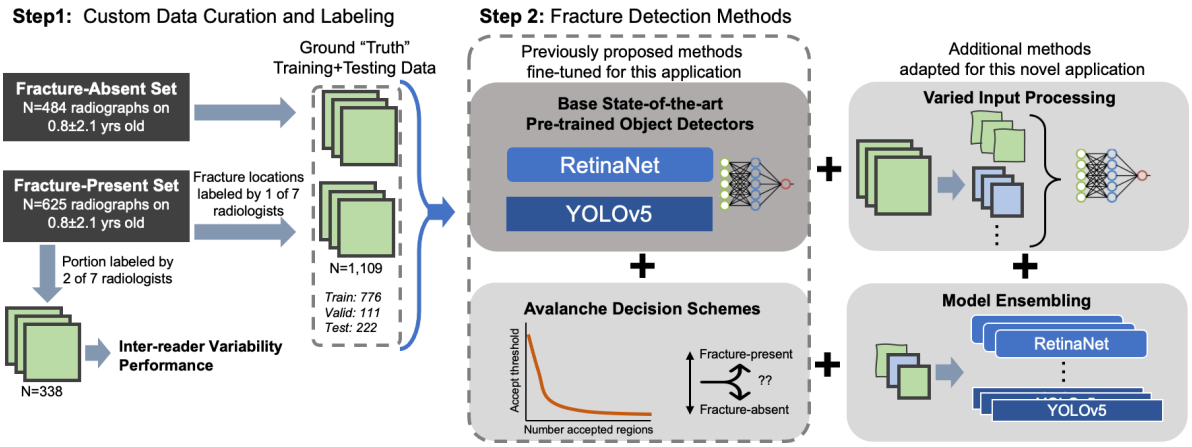


Figure 4.2 Overall workflow of our one-stage detector work.

4.2 Inter-reader Variability

In order to determine a baseline level of expert human performance, we explored inter-reader variability among the expert radiologists. Of the total 625 fracture-present images, 338 were read by two board-certified radiologists. We calculated inter-reader performance for two different data sets: 1) images from the fixed test set (which contains 222 images, although only 111 of these are fracture-present and therefore interpreted by radiologists), and 2) images from the set of 338 fracture-present images that have been read by two radiologists. For clarity, this inter-reader study was performed on fracture-present only images, while the deep learning training and testing was performed on present and absent images.

On the test set, the first reader marked 536 total rib fractures for an average of 4.83 ± 3.30 (range 1-14; median 4; IQR 5) fractures per image. The second reader marked 486 fractures overall, averaging 4.38 ± 3.74 (range 1-27; median 4; IQR 4) fractures per image. Setting the first reader's annotations as "ground truth" between the two, fractures were scored as true positive (second reader box matches a reader 1 box), false positive (reader 2 box has no corresponding reader 1 box), or false negative (reader 1 box has no matching reader 2 box), dictated by an intersection-over-union (IOU) threshold of 0.30.

Three-hundred eighty-five fractures were counted as true positive matches, with 101 false positives and 151 false negatives. This led to the second reader scoring a precision of 0.792, recall of 0.718, and F2 score of 0.732. Essentially, the second reader "detected" just under 72% of the rib fractures discovered by the first reader. With these scores, the second reader's boxes overlapped reader 1 on average by 84% with a mean intersection-over-union of 0.63 across the 111 images. For clarity, overlapping represents the percentage of reader 1's annotated box pixels that are covered by the pixels from reader 2's matching box.

Inter-reader performance metrics remained very similar when looking at all 338 multi-read images. Reader 1 marked $1,719$ fractures, averaging 5.09 ± 4.30 (range 1-22; median 4; IQR 5) fractures per image and reader 2 marked $1,567$ fractures for an average of 4.64 ± 4.08 (range 1-27; median 4; IQR 4) fractures per image. Percent overlap and IOU remained essentially identical at

84% and 0.62. Precision, recall, and F2 score all decreased slightly to 0.777, 0.709, and 0.721. If we were to assume the first reader caught all fractures during their reads, the second reader was able to find 71% of the fractures in their reads. This again leaves over one-quarter of all fractures undetected between expert radiologists.

4.3 Base Network Results

Representative test set images with ground truth annotations and model predictions are presented in Figure 4.3.

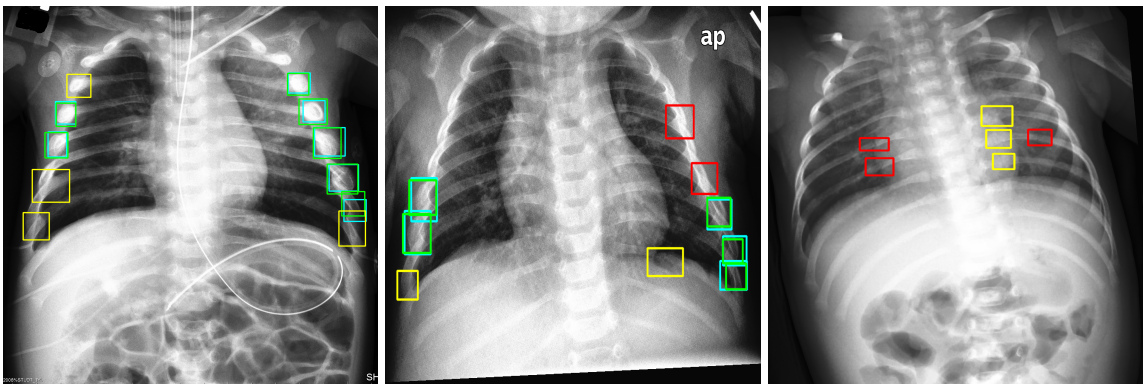


Figure 4.3 Test set images with ground truth (teal, red) and model predictions (green, yellow), with true positives (green), false positives (yellow), and false negatives (red). Predictions from the 6x-YOLOv5 ensemble trained on histogram equalized input images with a $\gamma = 0.20$ avalanche scheme, achieving 0.536 ± 0.044 precision, 0.795 ± 0.022 recall, and 0.723 ± 0.010 F2 score.

Base network performance was evaluated for single-model performance of both RetinaNet or YOLOv5 using histogram equalization image pre-processing (method (**a**) from Fig. 4.4). These results are presented in the Standard rows in Table 4.2 (and below with the nomenclature 1x-R$^{\mathbf{a}}$ and 1x-Y$^{\mathbf{a}}$). RetinaNet achieved 0.892 ± 0.015 precision, 0.430 ± 0.014 recall, and 0.480 ± 0.014 F2 score, whereas YOLOv5 scored 0.897 ± 0.032 precision, 0.434 ± 0.040 recall, and 0.484 ± 0.037 F2 score. When compared to expert-level human performance, both networks had marked higher values in precision but lower recall and therefore F2 scores. If either network were to predict a region for a potential rib fracture, they were essentially 90% likely to be correct in that prediction. However, both networks detected less than half of all rib fractures in the test set.

Changing from the histogram equalized inputs to the two alternative image processing methods

((**b**) binary and (**c**) blended) slightly alters performance for both networks in different ways. RetinaNet with binary inputs (1x-R$^{\mathbf{b}}$) incurs a small decrease in precision to 0.852 ± 0.015 (-4.5%) and large decreases in recall and F2 score at 0.344 ± 0.027 (-18.75%) and 0.390 ± 0.028 (-20%), respectively. Blended inputs (1x-R$^{\mathbf{c}}$) also cause a 1.01-1.25% drop in all three measures compared with method (**a**). YOLOv5 sees a similar drop in performance using the binary inputs, with 1x-Y$^{\mathbf{b}}$ achieving 0.872 ± 0.060 (-2.79%) precision, 0.320 ± 0.049 (-26.27%) recall, and 0.365 ± 0.050 (-24.59) F2 score. However, the blended inputs of method (**c**) for YOLOv5 led to the best single-model recall and F2 performance of all variants: 1x-Y$^{\mathbf{c}}$ scored 0.880 ± 0.024 (-1.9%) precision, 0.464 ± 0.043 (+6.91%) recall, and 0.512 ± 0.041 (+5.79%) F2 score. As presented later, the YOLOv5 architectures using the varied inputs (ensembles of models using input processing a, b, and c) had improvement over the other processing methods.

4.4 Image Processing

We wanted to explore how varying the type of processing performed on the input images changed the performance of the trained models. All types of processing were applied following the segmentation and cropping via the U-Net discussed previously. For the approaches discussed above, we were processing our dataset by taking a single-channel array of histogram equalized images and stacking identical arrays in order to create an RGB image to input into the network. We thought that these additional channels could offer new ways to incorporate different information that could potentially improve model performance. Figure 4.4 provides a visualization of the different types of processing and how they are combined to create input images for training and evaluation. Method **a** applies histogram equalization to the single-channel, grayscale image array after which the array is replicated three times to provide the three channel input to the object detector models.

The two additional variations go an additional step by utilizing the pixel-level segmentation information from the U-Net from the previous pre-processing stage. After cropping the image around the segmented thoracic region, all background pixels are masked out to generate a masked foreground image containing only anatomical structures. In method **b**, adaptive thresholding is applied using a Gaussian weighting method to determine threshold values in a given neighborhood

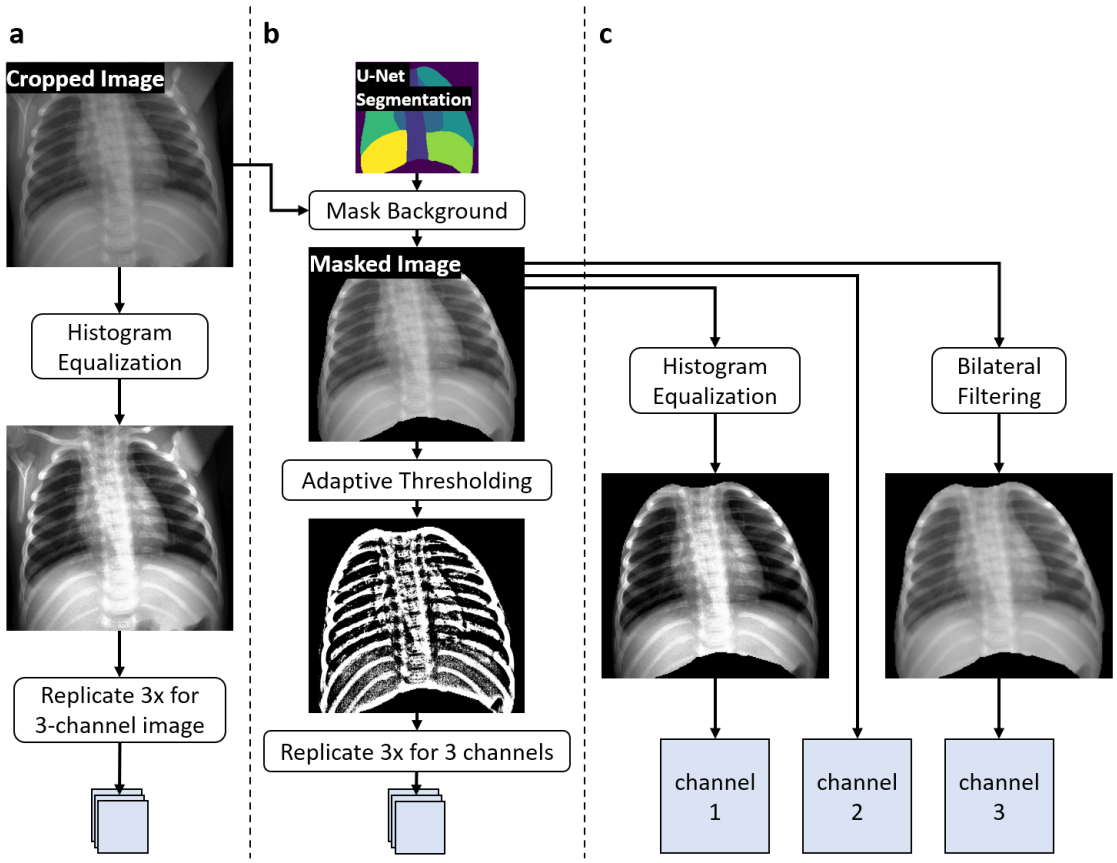


Figure 4.4 The three types of image processing used in the varied-input-processing ensemble models. **a**) normal-cropped histogram equalized 3x-stacked images; **b**) segmentation-masked adaptive thresholding 3x-stacked images; **c**) segmentation-masked raw, histogram equalized, and bilateral low-pass filtered blend.

of pixels. This transforms the image from grayscale to binary, with 1 (white) representing pixels above the threshold and 0 (black) below, providing a rough segmentation of just the ribs. This binary mask is then stacked three times as the final image.

In method **c**, the masked image goes through two separate filtering operations, inspired by Heidari et al. [115]: histogram equalization (like method **a**) for increased contrast and bilateral low-pass filtering for edge-preserving noise reduction. The low-pass filter uses mid-line σ-space and range values of 150, with a 9 pixel neighborhood diameter. The original masked image, histogram equalized masked image, and bilateral filtered masked image are then stacked as the three channels for the detector input, which we label as "blended" input.

4.5 Ensembling

We also investigated the impact of model ensembles on rib fracture detection which is commonly one of the first ways to approach improving the performance of machine learning networks. Model ensembles with deep neural networks have shown better generalizability as well as improved performance on tasks with smaller datasets [116, 117, 118]. To survey this, we tested the following types of ensembles:

(1) Same-Model Ensemble: The simplest form of ensembling is the combination the proposal results of multiple identical models each with different training runs with either differently initialized seeds similar to the deep ensembles analyzed by Lakshminarayanan et al. [119], or with different combinations of training data.

(2) Hybrid-Model Ensemble: This is a slight variation to the same-model ensemble, combining an equal number of training runs of both deep learning architectures we tested; for example, combining one run of RetinaNet with one run of YOLOv5.

(3) Varied-Input-Processing Ensemble: The final type of ensembling models incorporated all three of the different image pre-processing operations as summarized in Figure 4.4, requiring at minimum three trained models trained on each of the input processing variations.

Prior to final evaluation, proposed bounding boxes from all members of each ensemble were aggregated together and overlapped boxes then removed via non-maximum suppression (NMS) with an intersection-over-union (IOU) threshold of 0.55. This threshold was set based on initial validation experiments, but not fully optimized across all model variants. There is an interesting aspect differentiating predictions from RetinaNet and YOLOv5: RetinaNet offers many more box predictions than YOLOv5. We were interested to see whether that aspect led to better performance for one architecture over the other when incorporating ensembling.

Figure 4.5 provides a visual of a handful of detector F2 score performance along with their respective coefficient of variation. As the ensembles of models become more complex in terms of both number of ensemble members and sources of model diversity, F2 scores increase while the variation in F2 scores decreases. The hybrid-model, varied-input ensemble achieves both the

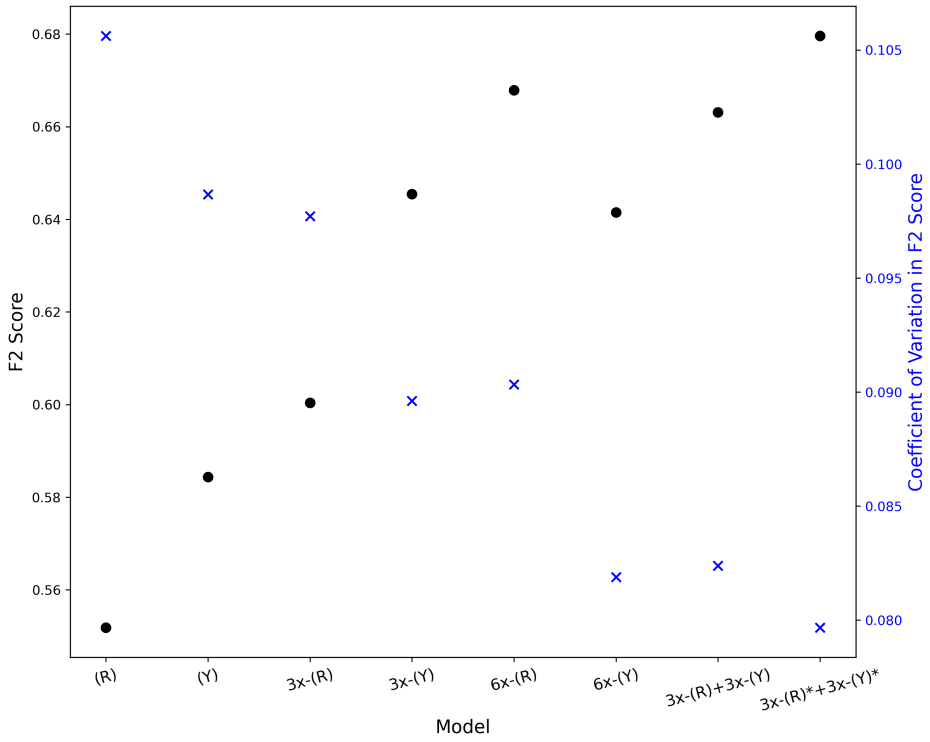


Figure 4.5 F2 score (black dot) and accompanying coefficient of variation (blue x) for each model and ensemble. As ensembles become more intricate and diverse, F2 scores improve while overall variation in performance decreases.

highest F2 score and the lowest coefficient of variation for F2. In other words, our best-performing ensemble exhibits both the highest performance and smallest uncertainty, perhaps indicating greater generalization ability that could carry over its performance to new images.

4.6 Avalanche Decision Scheme

We developed a novel inferencing strategy we coined the avalanche decision scheme [120] that makes the decision threshold for fracture-present model predictions a function of the number of already detected bounding box proposals. In other words, the decision threshold is not fixed, but rather changes depending on the number of high probability proposed regions. This approach is motivated by the reality that if a subject has one fracture they are very likely to have more than one fracture, and a subject with two fractures is very likely to have three, and so on.

In order to determine how to adjust the decision threshold as a function of number of cleared proposals, we used the training dataset to calculate the probabilities of there being more proposals

given that at least X fractures are currently present in the images, i.e., $P(X > 1|X \geq 1), \ldots, P(X > 4|X \geq 4)$ as presented in the third column of Table 4.1. Then, for a given starting model threshold α_0, if the model predicted 1 bounding box with a probability greater than α_0, we scale down the threshold to $\alpha_1 = \alpha_0 \cdot (1 - P(X > 1|X \geq 1))$ and the number of bounding box predictions that clear this threshold will be re-evaluated. If now three proposals have probabilities greater than α_1, we scale the threshold down to $\alpha_3 = \alpha_1 \cdot (1 - P(X > 2|X \geq 2) \cdot (1 - P(X > 3|X \geq 3)$. These likelihoods are presented in Table 4.1.

Table 4.1 Posterior likelihood for the training dataset, with each row summarizing probabilities of having more fractures, X (a random variable), when at least x=1, 2, 3, and 4 fractures are present (a deterministic variable).

X ≥ x Fractures	N	$\mathbb{P}$(X>x|X≥ x)
X ≥ 1	444	73.6%
X ≥ 2	327	81.3%
X ≥ 3	266	76.7%
X ≥ 4	204	77.0%

We explored an alternative calculation where if one proposal was found in an image at the given starting threshold α_0, the next threshold to re-evaluate proposals would be $\alpha_1 = \alpha_0 \cdot P(X > 1|X \geq 1)$. This is a more conservative reduction in confidence thresholds, since with the prior calculation each successive threshold α_{i+1} would be approximately $16 - 24\%$ of α_i. With the new calculation, each α_{i+1} would be $76 - 84\%$ of α_i. To further investigate variants of this more moderate version, we tested cases where the decrease between each successive α_i threshold was a constant rate γ, ranging from $10 - 30\%$. For example, if three rib fractures were proposed for a given starting threshold α_0, the new threshold will reduce to $\alpha_3 = \alpha_2 \cdot \gamma = \alpha_0 \cdot \gamma^3$.

Figure 4.6 provides a visual of the standard, posterior, and conservative schemes as well as three representative schemes with constant γ reduction. This illustrates the posterior avalanche scheme has the tendency to severely lower the decision threshold as soon as one, or especially two, rib fractures are proposed by the architectures that exceed the starting threshold α_0.

In brief, we explored different relationships for decision thresholds vs apparent number of

accepted fractures as presented in Figure 4.6. For the standard approach, the decision threshold is constant no matter how many proposed regions clear this threshold level, and typically the threshold is set of 0.5 (all proposals with greater than 50% are accepted). For the avalanche approaches, the decision threshold decreases as more proposed regions are accepted.

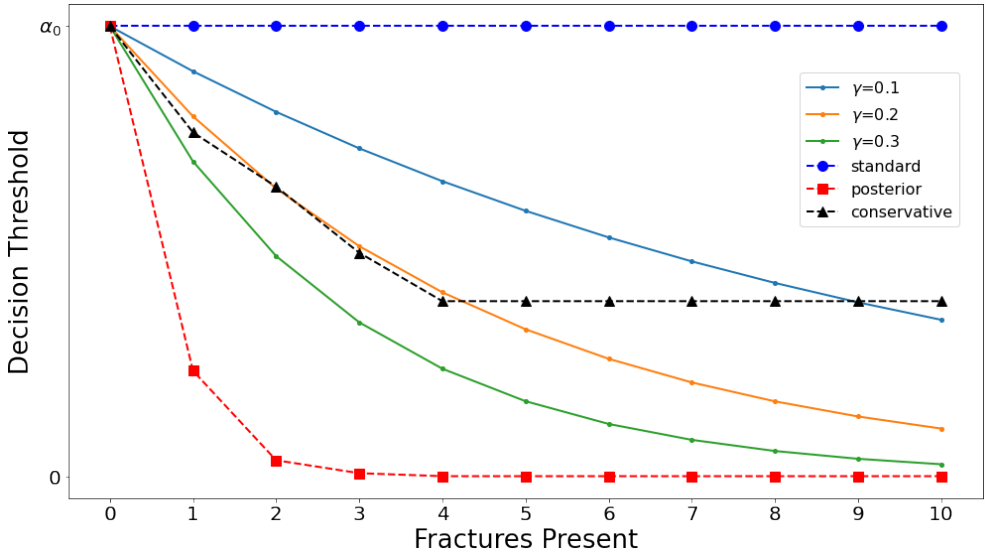


Figure 4.6 Plot of relative decision threshold for bounding box acceptance as a function of the number of accepted proposals.

Figure 4.7 shows F2 score performance of a single RetinaNet model as the initial decision threshold α_0 changes, using the “standard" approach of a constant threshold to get all proposals, the posterior distribution avalanche scheme, the conservative scheme, and constant γ reduction schemes with $\gamma \in [0.10, 0.15, 0.20, 0.25, 0.30]$. This figure influenced the decision of the γ values used in the avalanche scheme tests in Table 4.2.

Based on the starting threshold α_0 that obtained the highest F2 score for $\gamma = 0.15$ and $\gamma = 0.20$ in Figure 4.7, we chose $\alpha_0 = 0.55$ and $\alpha_0 = 0.75$ for the constant reduction avalanche schemes, respectively, to test alongside the posterior and conservative decision schemes.

We improved on our previous work by implementing a non-maximum suppression (NMS) step, a common approach to filter out regions proposals with a large overlap of each other. This NMS step is applied after the avalanche decision schemes have been applied on the given trained model predictions. This is particularly effective on networks like RetinaNet where the number of bounding

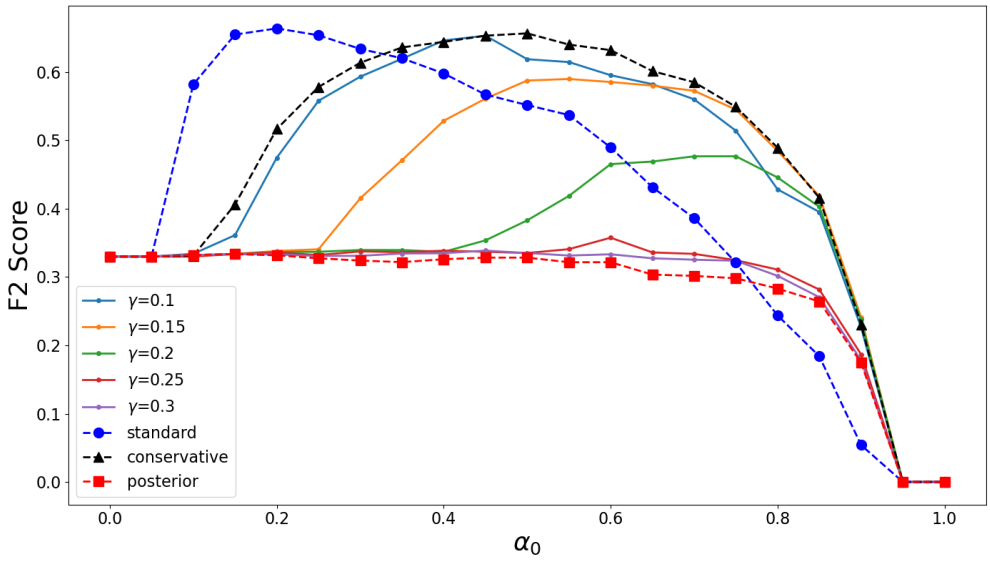


Figure 4.7 F2 scores for various avalanche decision schemes (as described in section 4.6) applied on a single RetinaNet model across all possible starting decision thresholds α_0. The "standard" approach uses a constant threshold set at α_0 for accepting proposed boxes. Remaining lines represent different avalanche schemes where the threshold starts at α_0 and decreases as a function of accepted boxes by a) fixed percentages, b) conservative reductions based on posterior likelihood values in table 4.1, b) and posterior reductions based on (1-posterior likelihood values).

box proposals per image is significantly higher than the more reserved models such as YOLOv5.

Table 4.2 compares how applying the avalanche schemes affect performance for single RetinaNet and YOLOv5 models trained on the histogram equalized inputs. The posterior scheme with RetinaNet drops precision to 0.141 ± 0.015, a significant 84.19% drop, whereas recall saw a large 102.8% increase to 0.872 ± 0.013. This, however, leads to an F2 score of 0.427 ± 0.026 which is 11% lower than standard. Instead, the best performing avalanche scheme for RetinaNet is the conservative scheme, where the 40.6% reduction in precision and 69.8% increase in recall sees the F2 score increase to 0.679 ± 0.010 (+41.5%).

Interestingly, YOLOv5 experiences a relatively minor drop in precision but marked improvement in recall, and therefore F2 score, with the avalanche decision schemes. Precision drops by 7-15% across the schemes while recall increases between 35-49%. This means the lowest performing YOLOv5 model with the $\gamma = 0.15$ scheme scoring 0.615 ± 0.050 is still 27.1% better than standard; the best performance comes from the posterior scheme with an F2 score of 0.652 ± 0.051 (+34.7%).

Table 4.2 RetinaNet and YOLOv5 single-model results comparing performance with the standard fixed decision threshold and applying the various avalanche schemes. γ represents the constant rate reduction between each decision threshold in the avalanche scheme.

Model	Scheme	Precision	Recall	F2
RetinaNet	Standard	**0.892** ± 0.015	0.430 ± 0.014	0.480 ± 0.014
	Posterior	0.141 ± 0.015	**0.872** ± 0.013	0.427 ± 0.026
	Conservative	0.530 ± 0.023	0.730 ± 0.015	**0.679** ± 0.010
	$\gamma = 0.15$	0.304 ± 0.028	0.766 ± 0.015	0.586 ± 0.019
	$\gamma = 0.20$	0.256 ± 0.023	0.770 ± 0.024	0.548 ± 0.015
YOLOv5	Standard	**0.897** ± 0.032	0.434 ± 0.040	0.484 ± 0.037
	Posterior	0.759 ± 0.164	**0.647** ± 0.101	**0.652** ± 0.051
	Conservative	0.831 ± 0.101	0.590 ± 0.075	0.622 ± 0.053
	$\gamma = 0.15$	0.814 ± 0.120	0.587 ± 0.077	0.615 ± 0.050
	$\gamma = 0.20$	0.816 ± 0.114	0.593 ± 0.087	0.621 ± 0.060

Considering the large number of different avalanche schemes, the remaining results will only present schemes corresponding to best F2 score performance for each model and/or ensemble.

4.7 Combining Avalanching, Input Processing, and Ensembling

This section outlines some of the results of the combination of avalanche decision schemes, varied input processing, and ensembling. The nomenclature for these combined models is presented in figure 4.8.

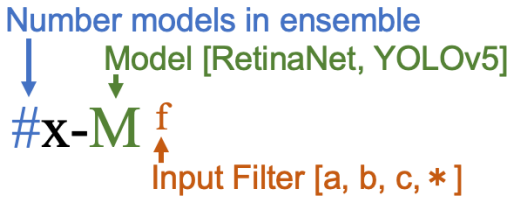


Figure 4.8 Explanation of model nomenclature for ensembling combined with different input processing techniques. The selection of input processing [a,b,c] is described in Figure 4.4 and varied input processing [*] uses one from each input processing type. For example, results presented for method 3x-R* would be for an ensemble of three RetinaNet models (trained on different folds) using varied input processing.

When applying the novel avalanche decision scheme to the base networks as presented above in Table 4.2, we see anticipated results that recall increases while precision decreases. The level of these changes varies depending on the avalanche scheme. Posterior with RetinaNet provides

the highest recall but very low precision. Conservative decision threshold reduction with YOLOv5 provides a reasonable compromise of recall and precision.

Performance of combining methods are presented in Table 4.3 and Table 4.4, with the former including evaluations only with the standard decision scheme (i.e., fixed acceptance threshold of 0.50 for all model predictions) and the latter including the best avalanche scheme for each model and/or ensemble. Table 4.3 also includes inter-reader variability performance at the top for comparison between expert human readers and deep learning models. For full results of all models and ensembles, see Supplementary Tables S1 and S2.

As the avalanche schemes are applied to the ensembled prediction results, we see the anticipated trend: precision drops as recall improves. With the ensembles that incorporate the various image processing techniques, the avalanche schemes provide the best overall results. The precision and recall of the 3x-RetinaNet and 3x-YOLOv5 approach the values found in the inter-reader variability study we discussed earlier.

Table 4.3 Performance results of selected models with standard decision threshold. I.R.V. represents inter-reader variability performance between two radiologists. Bolded values represent the top two scores for each metric. Superscripts **a**, **b**, and **c** represent the type of input processing to train the models as shown in Fig. 4.4. Ensembles with * have hybrid inputs, i.e., each ensemble member was trained on a different input processing method.

Models	Precision	Recall	F2	Max F2	mAP
I.R.V. (Test set)	0.792	0.718	0.732	-	-
I.R.V.	0.777	0.709	0.721	-	-
1x-R^{a}	**0.892** ± 0.015	0.430 ± 0.014	0.480 ± 0.014	0.630 ± 0.011	0.480 ± 0.008
1x-Y^{c}	**0.880** ± 0.024	0.464 ± 0.043	0.512 ± 0.041	0.644 ± 0.046	0.555 ± 0.016
3x-R*	0.812 ± 0.011	0.523 ± 0.014	0.563 ± 0.013	0.649 ± 0.006	0.493 ± 0.008
3x-Y^{c}	0.814 ± 0.017	0.599 ± 0.021	0.633 ± 0.018	**0.694** ± 0.008	**0.559** ± 0.006
6x-R^{c}	0.762 ± 0.008	0.571 ± 0.009	0.601 ± 0.008	0.666 ± 0.004	0.499 ± 0.004
6x-Y^{c}	0.756 ± 0.010	**0.653** ± 0.008	**0.671** ± 0.007	**0.699** ± 0.006	**0.555** ± 0.004
3x-R*+3x-Y*	0.752 ± 0.019	**0.625** ± 0.021	**0.647** ± 0.017	0.686 ± 0.008	0.522 ± 0.004

Using the standard decision scheme, the four three-model ensembles, 3x-R^{c}, 3x-R*, 3x-Y^{c}, and 3x-Y*, performed very similarly with regards to precision, scoring within 0.6% of one another.

Compared to their single-model versions, ensemble methods resulted in improved recall and thus F2 scores. While the best single-model recall was 0.464 ± 0.043, the worst performing three-model ensemble (3x-R*) achieved 0.523 ± 0.014 (+12.7%) and the best performing ensemble (3x-$\mathbf{Y^c}$) reached 0.599 ± 0.021 (+29.1%). This led to an F2 score of 0.633 ± 0.018 with the 3x-$\mathbf{Y^c}$ model. The mean average precision (mAP) trended upward as ensemble size increased and had similar trends as the F2 score, demonstrating that these models have similar rankings in performance across a range of inference hyper-parameters.

After applying avalanche decision schemes, we see the expected decrease in precision and increase in recall. The 3x-$\mathbf{Y^c}$ ensemble with the $\gamma = 0.20$ decision scheme had the superior performance among three-model ensembles achieving 0.725 ± 0.012 F2 score, which is within 1% of expert human-level performance. One interesting thing to note is that the three-model ensembles with standard decision schemes have lower F2 scores than single-models with avalanche schemes at the trade-off of maintaining much higher precision values that exceed the inter-reader performance.

Table 4.4 Performance of models from Table 4.3 with their best corresponding avalanche decision scheme result with respect to F2 score. Bolded values represent the top two scores for each metric. Superscripts *a*, *b*, and *c* represent the type of input processing to train the models as shown in Fig. 4.4. Ensembles with * have hybrid inputs, i.e., each ensemble member was trained on a different input processing method.

Models	Avalanche	Precision	Recall	F2	Max F2
1x-$\mathbf{R^a}$	Conservative	0.530 ± 0.023	0.730 ± 0.015	0.679 ± 0.010	0.897 ± 0.054
1x-$\mathbf{Y^c}$	Posterior	$\mathbf{0.645} \pm 0.130$	0.724 ± 0.085	0.695 ± 0.041	0.812 ± 0.056
3x-R*	Conservative	0.340 ± 0.013	0.809 ± 0.009	0.634 ± 0.011	$\mathbf{0.898} \pm 0.026$
3x-$\mathbf{Y^c}$	$\gamma = 0.20$	$\mathbf{0.573} \pm 0.058$	0.780 ± 0.030	$\mathbf{0.725} \pm 0.012$	0.812 ± 0.036
6x-$\mathbf{R^c}$	Conservative	0.311 ± 0.005	$\mathbf{0.816} \pm 0.005$	0.616 ± 0.005	$\mathbf{0.912} \pm 0.025$
6x-$\mathbf{Y^a}$	$\gamma = 0.20$	0.536 ± 0.044	0.795 ± 0.022	$\mathbf{0.723} \pm 0.010$	0.797 ± 0.016
3x-R*+3x-Y*	Conservative	0.314 ± 0.015	$\mathbf{0.841} \pm 0.014$	0.630 ± 0.011	0.833 ± 0.052

Six-model ensembles with standard decision schemes have lower precision scores than prior model and ensemble sizes, though still being on-par with expert-level performance. Once again, YOLOv5 with the blended, method (**c**) input images achieved the highest F2 score at 0.671 ± 0.007, a 6% improvement over its three-model variant. Incorporating avalanche schemes with 3x-$\mathbf{Y^c}$ utilizing

the γ avalanche scheme with a fixed rate of 0.20 provided the highest F2 score of 0.725 ± 0.012. The most complex 3x-R*+3x-Y* model was unable to achieve the highest performance in any metric, and in fact even had a slightly lower mAP score than the 1x-Y^C models, though its recall and F2 performance were still second-highest among the standard decision threshold models/ensembles.

4.7.1 Max F2 Score between Standard and Avalanche Schemes

In order to get a better understanding of how well the avalanche schemes perform compared to the standard inferencing technique, we plotted the F2 scores of a handful of test cases across all possible decision threshold values in Figure 4.9. For the avalanche methods, the x-axis of these plots represents the starting threshold (a_0) that is then potentially reduced if proposed regions have probabilities greater than a_0. We chose one model from the single-model group: the single

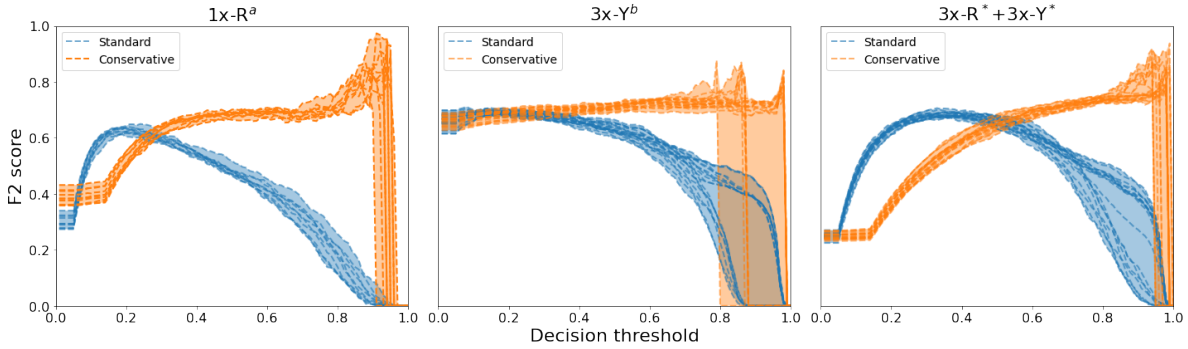


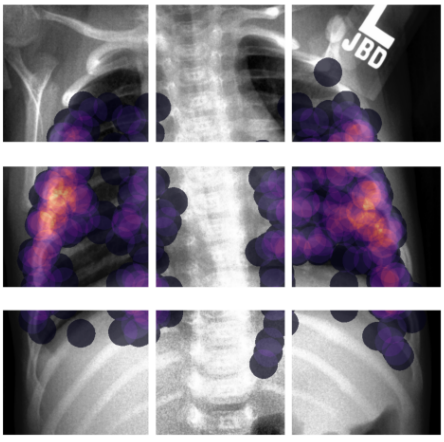
Figure 4.9 F2 scores across all possible confidence thresholds for 1x-$R^{\mathbf{a}}$ models, 3x-$Y^{\mathbf{c}}$ ensembles, and the hybrid, 3x-R* + 3x-Y* ensemble. Each dashed line represents performance from one model or combination of ensembles. The best performing avalanche scheme ('Conservative') is compared to the 'standard' decision scheme. Generally, the avalanche scheme performed better than conventional inferencing, except for very low decision thresholds in the 1x-R and 3x-Y cases. In the 3x-R*+3x-Y* ensemble, performance shifts around a threshold of 0.5. In every case, the avalanche decision scheme reaches higher max F2 scores than the standard technique.

RetinaNet using the histogram equalized images. Then we chose the best three-member ensemble with the 3x-$Y^{\mathbf{c}}$ ensemble, and the most diverse ensemble with the six-member 3x-R*+3x-Y* ensemble. Each of their best corresponding avalanche decision schemes was plotted along with their traditional decision scheme performances. In the 1x-$R^{\mathbf{a}}$ and 3x-$Y^{\mathbf{c}}$ cases, the avalanche scheme performs better than the standard decision scheme across a majority of the possible decision

thresholds. For the 3x-R*+3x-Y* ensembles, the standard scheme outperforms the avalanche scheme for thresholds less than around 0.50. In all three cases, not only do the avalanche schemes generally perform better than their standard decision scheme counterparts, but the maximum F2 scores were also higher than what the standard decision versions could attain. This maximum F2 performance can also be seen in the last column of tables 4.3 and 4.4. For each model and/or ensemble, the Max F2 value of its corresponding avalanche scheme is significantly higher than the standard schemes, many reaching a score above 0.9.

4.8 Regions Where Deep Networks Fail

In order to gain a better understanding of where the architectures are struggling, we are going to apply the same shared coordinate mapping that we used in our fracture prevalence analysis. For this, we will be looking at a single RetinaNet (1x-R^a) and a single YOLOv5 (1x-Y^c) network, which were the two higher performing single-model runs in our testing. In this analysis, all of the IOU thresholds were held at 0.30 and all acceptance thresholds for the bounding box probabilities was held at 0.50. First, we will look at RetinaNet's performance, split into three figures: Figure 4.10 shows the true positive, matching predictions, Figure 4.11 shows all false positives, where the model provided predictions without a ground truth fracture, and Figure 4.12 shows the false negatives where the model missed fractures from the test set.



Fractures

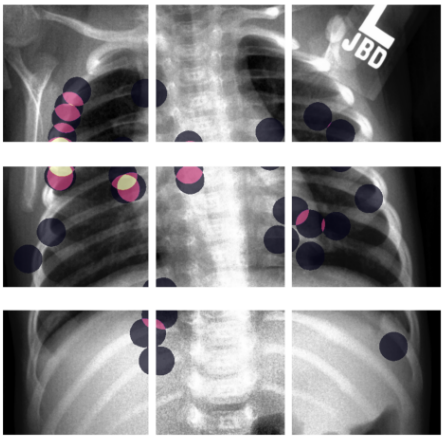
	R		L	SI Total
S	12	1	17	30
	75	17	80	172
I	9	5	6	20
LR Total	96	23	103	222

% Fractures

	R		L	SI Total
S	5.41%	0.45%	7.66%	13.51%
	33.78%	7.66%	36.04%	77.48%
I	4.05%	2.25%	2.70%	9.01%
LR Total	43.24%	10.36%	46.40%	100.00%

Figure 4.10 True positive fractures captured by 1x-R^a.

Out of a total possibly 536 labeled rib fractures in the test set, the 1x-R^a model correctly caught 222 (41.4%). It seems to have predicted predominantly in the central rib region, where 77% of its predictions are located. The RetinaNet model did not over-predict regions that did not contain fractures; only 27 fractures total were counted as false positives. It over-predicted in the central region, which may explain why such a high percentage of the true positive fractures were captured in this central band. Interestingly, it predicted nearly twice as many fractures on patient-right sides compared to central or left, whereas our fracture prevalence analysis revealed there to be a slight left-sidedness to fracture presence. Shifting finally to all fractures missed by the network, we see



Fractures

	R		L	SI Total
S	5	1	2	8
	7	3	5	15
I	1	2	1	4
LR Total	13	6	8	27

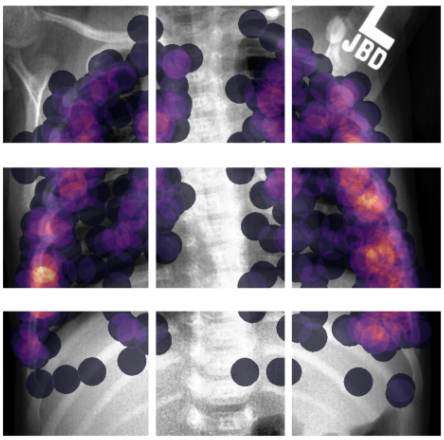
% Fractures

	R		L	SI Total
S	18.52%	3.70%	7.41%	29.63%
	25.93%	11.11%	18.52%	55.56%
I	3.70%	7.41%	3.70%	14.81%
LR Total	48.15%	22.22%	29.63%	100.00%

Figure 4.11 False positive fractures predicted by 1x-R^a.

that despite its tendency of predicting in the middle horizontal region, there were still over 53% of all missed fractures residing there. Of the three vertical zones, the most missed fractures were in the patient left side.

Let us now look at the YOLOv5 network performance. Figure 4.13 shows the true positive performance, where it captured almost the same number of fracture as RetinaNet with 218 (40.7%) and overall performs almost identically to the RetinaNet model. Looking at the false positive performance in Figure 4.14, we see that YOLOv5 falls into the same issue as RetinaNet: predicting more fracture regions on the patient right side compared to patient left. It appears that both RetinaNet and YOLOv5 have a tendency to falsely predict fractures in the top-left region (patient



Fractures

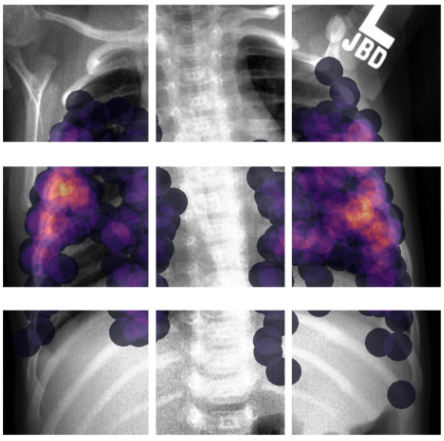
	R		L	SI Total
S	33	34	32	99
	67	19	82	168
I	18	3	26	47
LR Total	118	56	140	314

% Fractures

	R		L	SI Total
S	10.51%	10.83%	10.19%	31.53%
	21.34%	6.05%	26.11%	53.50%
I	5.73%	0.96%	8.28%	14.97%
LR Total	37.58%	17.83%	44.59%	100.00%

Figure 4.12 False negative fractures missed by 1x-R^a.

right on ribs 1-4), with RetinaNet proposing 5 and YOLOv5 proposing 6 in that region. Once



Fractures

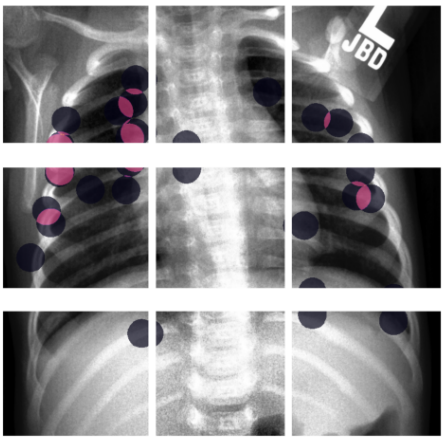
	R		L	SI Total
S	10	0	17	27
	76	17	83	176
I	7	4	4	15
LR Total	93	21	104	218

% Fractures

	R		L	SI Total
S	4.59%	0.00%	7.80%	12.39%
	34.86%	7.80%	38.07%	80.73%
I	3.21%	1.83%	1.83%	6.88%
LR Total	42.66%	9.63%	47.71%	100.00%

Figure 4.13 True positive fractures captured by 1x-Y^c.

again, YOLOv5 missed slightly more fractures in the patient-left vertical region (43.59% compared to 38.46% on patient-right). Overall, both the single RetinaNet and YOLOv5 models perform similarly, and therefore miss fractures in very similar ways. Over half of all missed fractures appear in the middle-third region, which we know is the horizontal region with the most fractures, and both also seem to be missing a higher percentage on the patient-left side.



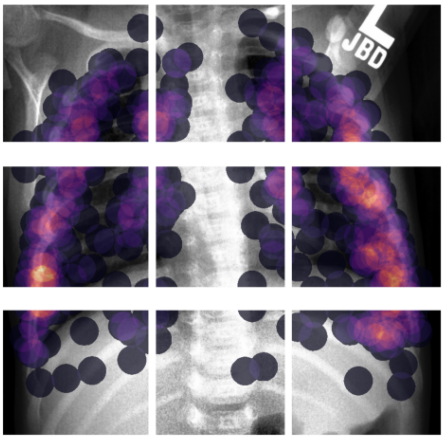
Fractures

	R		L	SI Total
S	6	1	2	9
	7	1	4	12
I	1	0	2	3
LR Total	14	2	8	24

% Fractures

	R		L	SI Total
S	25.00%	4.17%	8.33%	37.50%
	29.17%	4.17%	16.67%	50.00%
I	4.17%	0.00%	8.33%	12.50%
LR Total	58.33%	8.33%	33.33%	100.00%

Figure 4.14 False positive fractures predicted by 1x-Y^c.



Fractures

	R		L	SI Total
S	35	32	31	98
	66	20	77	163
I	19	4	28	51
LR Total	120	56	136	312

% Fractures

	R		L	SI Total
S	11.22%	10.26%	9.94%	31.41%
	21.15%	6.41%	24.68%	52.24%
I	6.09%	1.28%	8.97%	16.35%
LR Total	38.46%	17.95%	43.59%	100.00%

Figure 4.15 False negative fractures missed by 1x-Y^c.

4.9 GAN Synthetic Fracture Generation via Fracture Prevalence

As mentioned previously, data scarcity is an unavoidable issue with medical imaging datasets. Synthetically generated images have seen increasing interest as a method to address the limited data problem. Generative Adversarial Networks (GANs) were first introduced in 2014 by Goodfellow *et al.* [121] which demonstrated a concept where one neural network, called the generator, creates images to try and “fool" a second neural network, the discriminator, into believing the synthetic image is, in fact, real. Variations on the GAN architectures have been proposed over the years, such as pix2pix [122], StyleGAN [123], and CycleGAN [124]. While these were introduced using

natural images, various other studies sought to tailor them toward generating synthetic medical images. Sorin *et al*. [125] conducted a review of over 30 different studies that implemented a GAN for some radiological applications, ranging from image reconstruction and denoising, transferring between modalities, data augmentation, and image segmentation. However, despite there being 33 studies overall, only one of them used chest radiographs—for image segmentation—whereas nearly every one of them dealt with MRI or CT scans regardless of application.

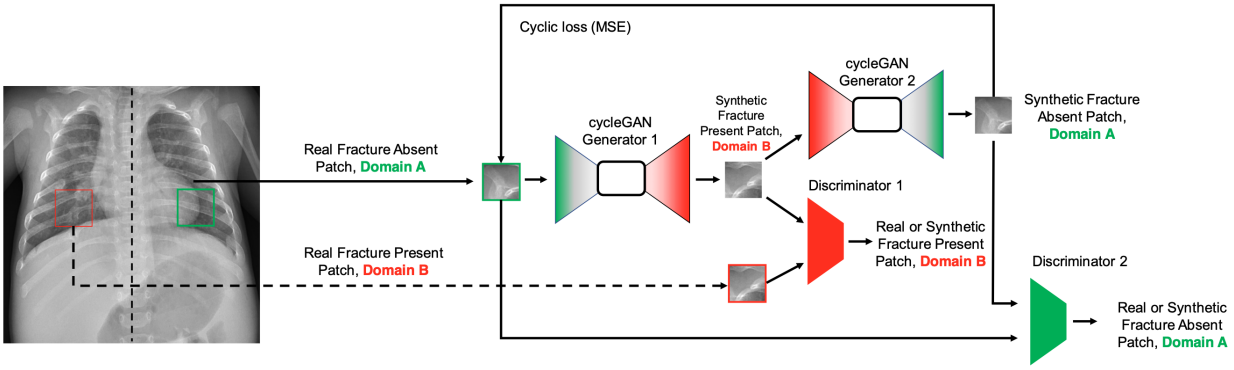


Figure 4.16 Schematic of near-pair patch GAN training process. Two sets of generator/discriminator pairs are trained simultaneously using near pair patches. Generator 1 converts real fracture-absent patches to fracture-present patches. Discriminator 1 distinguishes between synthetic fracture-present and real fracture-present patches. Generator 2 removes pathology to create synthetic fracture-absent patches. Discriminator 2 distinguishes between synthetic and real fracture-absent patches. Figure from [126].

The task of generating synthetic chest radiographs is not only a difficult problem, but also uncommon and is thus quite important given the limiting constraints of data. Our colleague, Ethan Tu, in the Medical Imaging and Data Integration (MIDI) Lab sought to synthetically generate rib fractures in chest radiographs with the goal of improving object detector performance compared to only real rib fracture containing images [126]. To achieve this, he used a CycleGAN trained on localized patches. A CycleGAN is advantageous because it does not require identically-paired images from the two domains; these types of paired images would be impossible to obtain for our task of needing fracture-absent and fracture-present images. Additionally, this method relies on localized patches of rib regions. Our goal is to generate a localized portion of a rib with a synthetic fracture. This localized portion is then inserted back into the full radiograph image. This

greatly reduces the computational requirements for generating synthetic fractures and constrains the problem to localized scenes. Finally, an innovation of Tu's work is to use near-pair patches, where the images are from similarly localized scenes with one having a present fracture and one being fracture-absent.

We utilized a subset of the data used in this chapter, consisting of 704 pediatric patients with 515 fracture-present and 189 fracture-absent patients. Near-pair patches were found by first isolating the hand-labeled rib fractures then manually finding a matching fracture-free rib from the same patient, yielding a closely similar rib region where one has a fracture and one does not. Each of these localized patches were of size 128×128. A second GAN was trained using another set of paired patches, however this time the fracture-absent patches were randomly selected from fracture-absent patients in the dataset. A third and final GAN was trained using the Fréchet Inception Distance (FID) [127] which is a metric used to evaluate the realness of image generated by GANs. A schematic of this model is presented in figure 4.16. Each of these trained GANs will be abbreviated as NPPGAN, RandGAN, and FIDGAN, respectively.

Table 4.5 The various training sets used to train YOLOv5 to investigate the effect of radiographs with synthetically generated rib fractures on detection performance.

Real Images	Synthetic Images	Total Training Images
0	500	500
50	0	50
50	500	550
250	0	250
250	500	750
500	0	500
500	500	1000

We evaluated the performance of adding the images with synthetically generated rib fractures to the training sets for a single YOLOv5 network, i.e., 1x-Y^a. This was due to YOLO's efficient implementation, making it quick to train on various dataset sizes of real and real + synthetic training sets, and its tendency to be more constrained in the number of predictions provided as opposed to RetinaNet. Table 4.5 shows the various training set sizes and number of validation and test images

used to evaluate YOLOv5. Each model was trained on a NVIDIA V100S like before, with a batch size of 8 and at most 300 epochs with an early stopping protocol if validation performance did not improve within 100 epochs. Unlike previously where 10 separate models were trained on different randomizations of the training set to generate error bars for results, error bars for this work were calculated by performing $5,000$ iterations of stratified bootstrapping on the test set results.

Table 4.6 Results from adding synthetically generated rib fractures from the random-pair patch CycleGan, i.e., RandGAN, on the detection performance of a single YOLOv5 network.

Training Dataset	Precision	Recall	F2 Score
0 Real 500 Synthetic	0.125 ± 0.128	0.004 ± 0.004	0.005 ± 0.005
50 Real 0 Synthetic	0.735 ± 0.070	**0.214** ± 0.043	**0.249** ± 0.048
50 Real 500 Synthetic	**0.815** ± 0.091	0.169 ± 0.044	0.200 ± 0.050
250 Real 0 Synthetic	**0.820** ± 0.035	0.452 ± 0.048	0.496 ± 0.047
250 Real 500 Synthetic	0.816 ± 0.039	**0.456** ± 0.053	**0.499** ± 0.052
500 Real 0 Synthetic	**0.901** ± 0.030	0.448 ± 0.051	0.498 ± 0.051
500 Real 500 Synthetic	0.860 ± 0.033	**0.493** ± 0.048	**0.539** ± 0.047

Table 4.7 Results from adding synthetically generated rib fractures from the near-pair patch CycleGan, i.e., NPPGAN, on the detection performance of a single YOLOv5 network.

Training Dataset	Precision	Recall	F2 Score
0 Real 500 Synthetic	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000
50 Real 0 Synthetic	0.779 ± 0.059	**0.254** ± 0.035	**0.294** ± 0.038
50 Real 500 Synthetic	**0.922** ± 0.046	0.179 ± 0.035	0.213 ± 0.040
250 Real 0 Synthetic	0.894 ± 0.047	**0.468** ± 0.048	**0.517** ± 0.048
250 Real 500 Synthetic	**0.909** ± 0.033	0.449 ± 0.050	0.499 ± 0.050
500 Real 0 Synthetic	0.825 ± 0.035	**0.536** ± 0.044	**0.576** ± 0.042
500 Real 500 Synthetic	**0.874** ± 0.036	0.524 ± 0.047	0.569 ± 0.046

Tables 4.6, 4.7, and 4.8 provide the results after fine-tuning YOLOv5 for each the RandGAN, NPPGAN, and FIDGAN synthetically generated fracture images, respectively. Training purely with synthetically generated fractures, regardless of GAN, leads to abysmal performance. Interestingly, only the RandGAN provided any metrics on the test set where NPP and FID both gave scores of 0 across all metrics. Looking at Table 4.6 with RandGAN, precision increases 10.8% with a small 50

Table 4.8 Results from adding synthetically generated rib fractures from the near-pair patch CycleGan with added FID scoring, i.e., FIDGAN, on the detection performance of a single YOLOv5 network.

Training Dataset	Precision	Recall	F2 Score
0 Real 500 Synthetic	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000
50 Real 0 Synthetic	0.764 ± 0.051	**0.327** ± 0.048	**0.369** ± 0.051
50 Real 500 Synthetic	**0.981** ± 0.019	0.201 ± 0.031	0.239 ± 0.036
250 Real 0 Synthetic	0.883 ± 0.031	**0.434** ± 0.042	**0.482** ± 0.042
250 Real 500 Synthetic	**0.923** ± 0.027	0.408 ± 0.049	0.458 ± 0.050
500 Real 0 Synthetic	**0.991** ± 0.009	0.412 ± 0.041	0.466 ± 0.042
500 Real 500 Synthetic	0.855 ± 0.034	**0.488** ± 0.041	**0.534** ± 0.040

real image dataset while decreasing for both 250 (−0.49%) and 500 (−4.55%) real image datasets. This trend flips when looking at recall, where with 50 real images there is a 21% decrease, but 250 and 500 real images experience an increase of 0.88% and 10%, respectively.

NPPGAN results (Table 4.7) are quite different. In every case, adding synthetically generated fracture images leads to precision improvement while recall decreases. This consequently reduces F2 score in each case as well. The same cannot be said for results in Table 4.8 for the FIDGAN images. Like the RandGAN images, precision values see marked increases for both 50 real images (+28.4%) and 250 real images (+4.53%), with a decrease for 500 images (−13.72%). Recall also decreases for both 50 images (−38.53%) and 250 images (−5.99%), but then experiences a significant improvement with 500 images (+18.45%). This leads to the largest improvement in F2 scores across all training combinations, increasing 14.59% from 0.466 ± 0.042 to 0.534 ± 0.040.

It must be noted that detector performance on these sets of training data cannot be compared directly between the three GANs. There are two factors for this: first, the real-only images randomly chosen for each GAN were not drawn equally—which was unfortunately overlooked—and second, the images with synthetically generated fractures were also different across the GANs. This is why the real-only performances vary in each of the tables, and why performance changes discussed above were largely in terms of percent increases or decreases.

The performance from these sets of images from the RandGAN, NPPGAN, and FIDGAN all

show there can be value added by being able to synthetically generate rib fractures to improve an object detector's ability in correctly identifying and localizing fractures, but more can be done. A new set of synthetically generated images from each of the GANs was created, this time with each synthetic bounding box region and all images kept the same across the three generative models. To compare with the results from above, a similar number of 500 images with synthetically generated rib fractures was curated as well as 500 real images that were also held consistent when training with the three sets of GAN images. In other words, the training set consisted of 500 real images, 389 of which are fracture-present—with a mean 4.65 ± 4.07 fractures per image—and 61 fracture-absent, and 500 synthetically modified images with an average of 3.01 ± 1.45 fractures per image. The validation set is 62 real images with 31 each fracture-present (mean 5.58 ± 4.84 fractures per image) and fracture-absent. Lastly, the test set size is the same 120 real images also split evenly between fracture-present (mean 3.68 ± 3.43 fractures per image) and absent.

The performance of of YOLOv5 with the new synthetic images from the three GANs is shown in Table 4.9. Unlike previously, these metrics can be directly compared to one another. Starting with a training set of 500 real-only images, the YOLOv5 model performs similarly to what we saw for base performance for the entire dataset, being just shy of 90% accurate when it predicts a box but only catching 42% of the rib fractures in the test set. In each case, adding the images with synthetic rib fractures decreases precision performance while improving recall. This is especially the case with the FIDGAN results, where precision is barely reduced (−0.78%) and recall improves to 47% of fractures (+11.6%).

Table 4.9 Performance of 1x-YOLOv5 using 500 real and 500 updated synthetic images to train. The first row (-) is a 1x-YOLOv5 trained on just 500 real images.

GAN	Precision	Recall	F2 Score
-	0.895 ± 0.034	0.422 ± 0.052	0.471 ± 0.053
Rand GAN	0.826 ± 0.042	0.439 ± 0.045	0.484 ± 0.045
NPP GAN	0.860 ± 0.040	0.437 ± 0.053	0.484 ± 0.053
FID GAN	**0.888 ± 0.036**	**0.471 ± 0.052**	**0.520 ± 0.051**

Qualitatively, the generated synthetic fractures appear realistic, but only around one-third of the

generated fractures were clear and convincing. This could be an indication of the GANs output being inconsistent, but there is also a chance that our lack of expertise prevents us from knowing for sure that a fracture was generated in a more difficult patch. Regardless, because of the results in Table 4.9, it is clear that the synthetically generated fracture images as a data augmentation technique provides useful information during training which could be quite beneficial for improved detector performance in the future.

4.10 Summary

Throughout this chapter, we have detailed the training setup and data splitting techniques we used to trained a plethora of different models. We carried out an expert inter-reader performance study to serve as a baseline understanding of experienced radiologists in order to compare the performance of the deep learning methods. We show that through various techniques such as model ensembling, varied image processing techniques, and our novel “avalanche" decision scheme we are able to drive the recall performance of these deep learning models upward.

We also visualized the performance of standalone RetinaNet and YOLOv5 models using the same method as our fracture prevalence study from Section 3.4. This demonstrated that both models in their base configurations perform very similarly, and tend to predict regions and miss fractures in similar ways. We also provided a summary of the work carried out by our colleague, Ethan Tu, with his work on synthetically generated fracture regions as a method of data augmentation that, when combined with our object detector work, presented a promising initial look into the feasibility of using these synthetic images to improve the capability of deep learning detectors in capturing more of fractures. Work demonstrated throughout this chapter contributed to the following academic products:

- Gadgeel, Burkow, Perez, Junewick, Zbojniewicz, Otjen, Alessio, “*Evaluation of inter-reader reproducibility for detection and labeling of pediatric rib fractures on radiographs*,” International Pediatric Radiology Congress, Rome [Virtual], Oct 11-15, 2021.

- Burkow, Holste, Otjen, Perez, Junewick, Alessio, "*Avalanche Decision Schemes to Improve*

Pediatric Rib Fracture Detection," Proc. SPIE 12033, Medical Imaging 2022: Computer-Aided Diagnosis, 120332A, Apr. 2022.

- Tu, Burkow, Otjen, Perez, Junewick, Alessio, "*Synthetic Pathology Generation with Near-Pair Cyclic GANs for Object Detectors*," IEEE Nucl. Sci. Symp. and Med. Imaging Conf., Milan, Nov. 2022.

- Burkow, Holste, Otjen, Perez, Junewick, Zbojniewicz, Romberg, Menashe, Frost, Alessio, "*High sensitivity methods for automated rib fracture detection in pediatric radiographs*," Sci Rep 14, 8372 (2024). https://doi.org/10.1038/s41598-024-59077-5.

- Tu, Burkow, Tsai, Perez, Junewick, Alessio, "*Near-Pair Patch Generative Adversarial Network for Data Augmentation of Focal Pathology Object Detection Models*," In review for Journal of Medical Imaging, 2024.

CHAPTER 5

DOMAIN-SPECIFIC TWO-STAGE DETECTOR

5.1 Exhaustive Search

To get a preliminary idea of how a two-stage network would perform on our data, we manually created a patch-based version of our dataset using an exhaustive search process. All images were rescaled based on the median pixel spacing of 0.125 mm so that physical spatiality was homogeneous among the images. To ensure a consistent resolution for patch generation, the

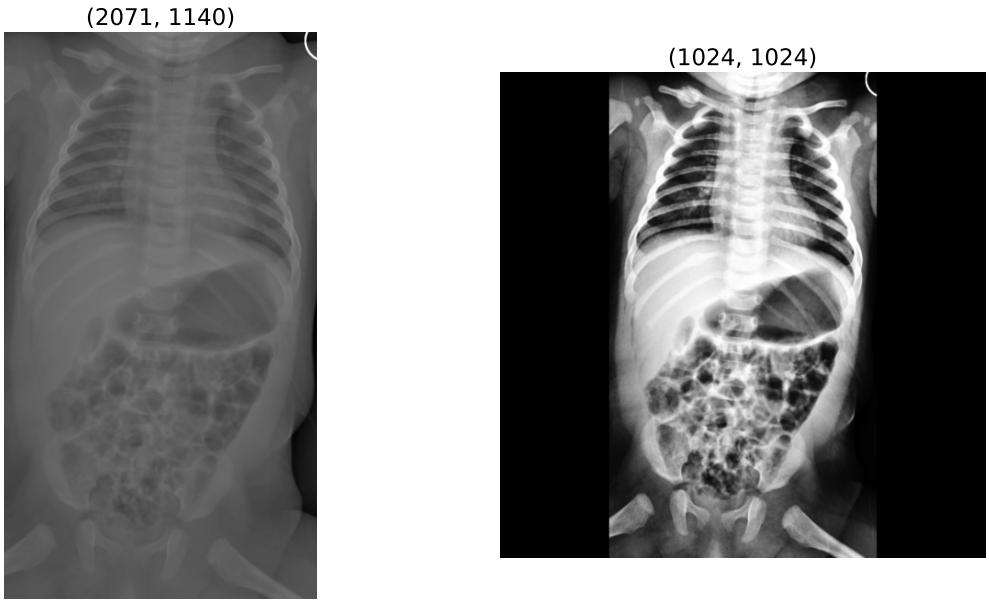


Figure 5.1 Representation of the pre-processing stage of rescaling an image array to a fixed (1024, 1024) resolution with original image aspect ratio preserved and histogram equalization applied.

matching pixel-spacing arrays were then rescaled to a square resolution of 1024×1024, with the aspect ratios of the original images preserved by padding with zeros in either the height or width dimension as necessary. During this process, each image received the same histogram equalization as applied previously prior to the rescaling so that equalization is applied evenly across the entire image rather than calculating it per patch later. An example of this process is shown in Figure 5.1. All of the associated annotated bounding boxes underwent a similar rescaling process, first adjusting for the pixel spacing and subsequently for the resolution rescaling.

From these now rescaled, square images, patches of size 128×128 were generated with a stride

of 64 pixels, resulting in a 50% overlap between adjacent patches in each direction and yielding 225 patches per image. Figure 5.2 illustrates a single image represented by all of its patches. Patches of this size were chosen to minimize the chance of introducing too much information within a single patch; generally, a patch will at most show parts from two ribs. Overall, from the entire dataset of $1,109$ patients this produced a total of 140,400 patches, among which 7,921 (5.6%) contained fractures, while the remaining 132,479 were fracture-absent.

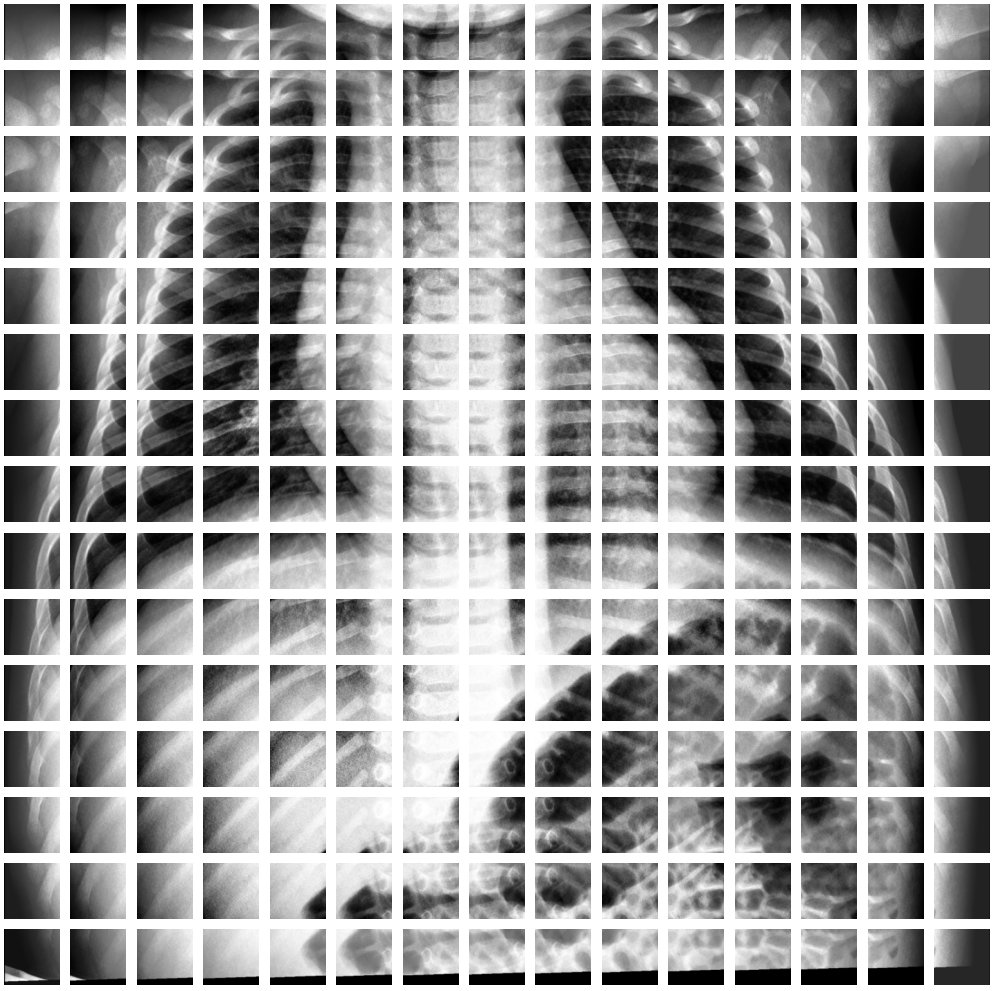


Figure 5.2 A grid view showing all 225 patches associated with a single image.

5.1.1 Training Setup

The dataset was split into training, validation, and test sets based on 70%, 10%, and 20% ratios, respectively. The training set consists of $98,282$ patches, with $5,545$ fracture-present and $92,737$ fracture-absent patches. The validation set includes $14,039$ patches, with 792 fracture-present and

13, 247 fracture-absent patches. Finally, the test set contains 28, 079 patches, of which 1, 584 were fracture-present and 26, 495 were fracture-absent.

Two classification architectures were trained and evaluated: ResNet-50 and DenseNet-121. Both ResNet-50 and DenseNet-121 were trained to a maximum of 100 epochs with batch sizes of 64 with an Adam optimizer. All layers with trainable layers were unfrozen with the output of the final prediction layer reduced down to a binary class output of either fracture-absent or fracture-present. Given the massive class imbalance of the dataset, a simple weighting of about 16.7 (ratio of fracture-absent to fracture-present patches in the training set) was applied for the positive class on the binary cross entropy loss function. Using a single A100 NVIDIA graphics card, completing all epochs of training for ResNet-50 and DenseNet-121 took 481 minutes (8.02 hours) and 375 minutes (6.25 hours), respectively.

5.1.2 Results for Exhaustive Search

Table 5.1 displays how ResNet-50 and DenseNet-121 perform under the normal binary classification acceptance threshold (i.e., $p_i >= 0.50 \implies y_i = 1$). Accuracy appears relatively high for both, reaching nearly 78% for ResNet-50 and 82% for DenseNet-121, though it is worth remembering the test dataset contained 1, 584 fracture-present and 26, 495 fracture-absent patches; simply scoring every patch as fracture-absent would yield an accuracy of 0.944. Looking at precision and recall scores provides additional clarity on how the models are scoring the patches. Precision for both are below 0.2, indicating that both have a high likelihood of assigning false positives. A trade-off of this is that the classifiers both identify nearly three-fourths of the fracture-present patches in the test set.

Table 5.1 Performance of ResNet-50 and DenseNet-121 on classifying fracture-present versus absent patches found via exhaustive search. Threshold of 0.50 used for setting labels as 1 and 0.

	Accuracy	Precision	Recall	F2 Score
ResNet-50	0.779	0.169	0.745	0.442
DenseNet-121	0.819	0.198	0.724	0.473

To gain a more holistic understanding of both networks across all acceptance thresholds, we

look at the precision-recall and ROC curves (shown in the left and right images in Figure 5.3, respectively). In both cases, DenseNet-121 outperforms the ResNet-50 models, achieving an average precision of 0.414 and a ROC-AUC of 0.906. They both perform respectably in terms of ROC, achieving AUC scores well above a random classifier, but the precision-recall curve demonstrates how much both networks struggle at both capturing the fracture patches and being confident in their predictions.

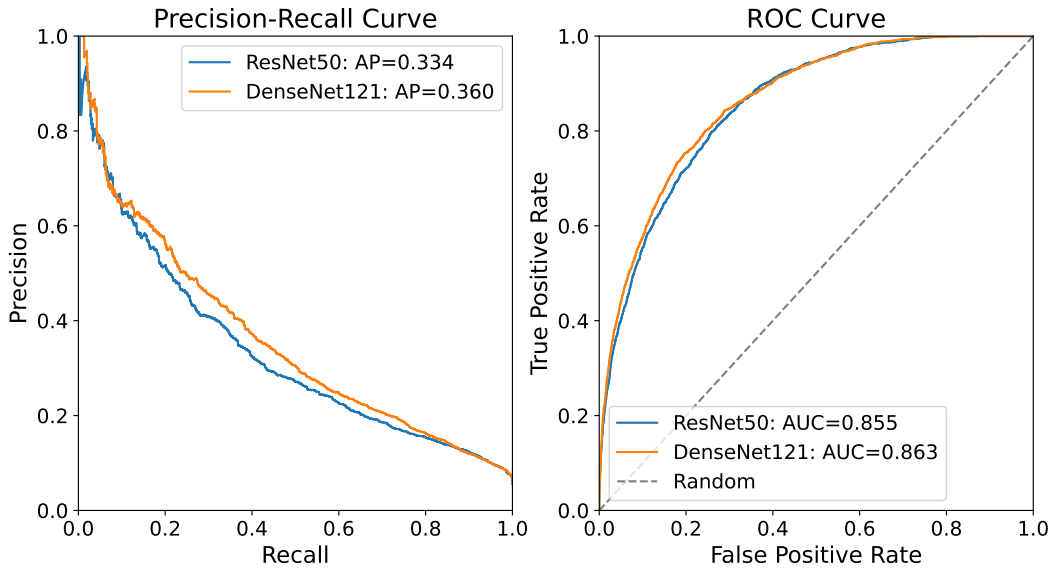


Figure 5.3 Precision-Recall and ROC Curves for ResNet-50 and DenseNet-121 performance on the exhaustive search patch test set.

5.2 Fracture Prevalence-inspired Two-Stage Detection

After an initial investigation at classification performance on an exhaustive search region extraction approach from our data, we shift our attention toward creating a domain-specific two-stage implementation. Ultimately, the goal is to utilize the fracture prevalence-style mapping system we discussed in Section 3.4 as an a priori region proposal strategy.

5.2.1 First stage: Fracture Prevalence and U-Net Region Proposal

Through our analysis earlier on the fracture prevalence patterns of our dataset, we were able to ascertain a few key points: fractures are spread relatively evenly across the rib cage, a higher proportion are present on the central ribs and lateral margins, and there is a slight patient left-

sidedness. Knowing these, we believe that directly using this in the method for region proposal could lead to improved fracture detection by eliminating erroneous boxes that the one-stage detectors we tested in Chapter 4 automatically generate through anchor boxes.

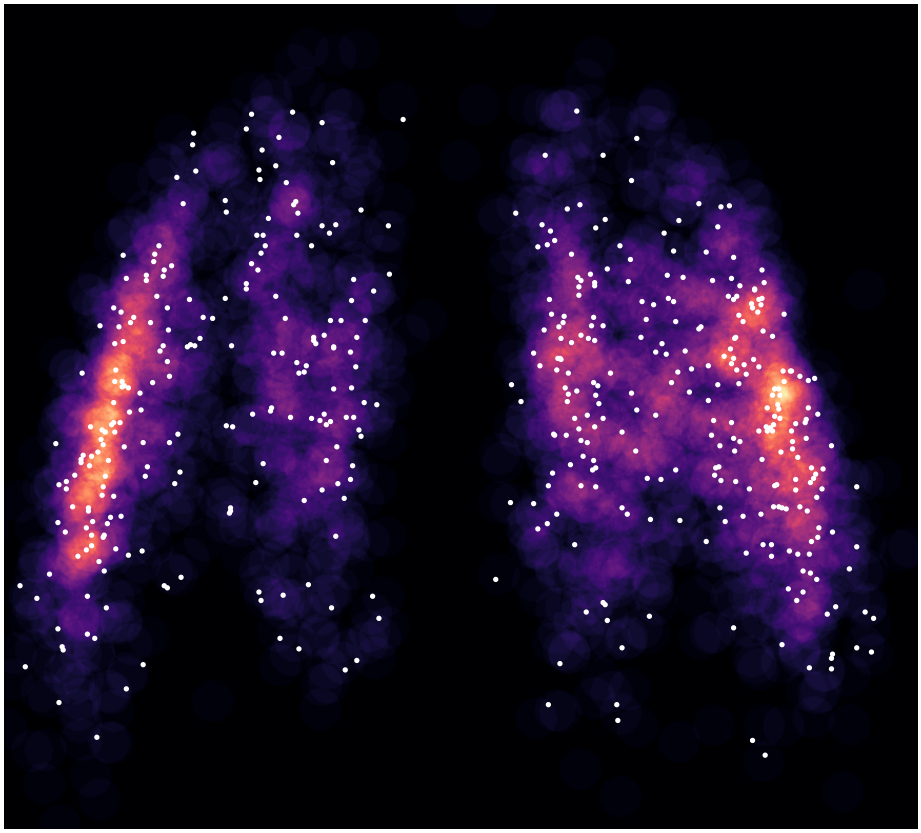


Figure 5.4 Randomly drawn set of 200 points using the fracture prevalence map intensities as weighting for the random drawing.

Figure 5.4 illustrates an example of using the fracture prevalence array as a weighted random location generator for 200 randomly generated points. To achieve this, we took the set of all (R_x, R_y) x- and y-ratio offsets calculated from each individual segmentation mask's centroid from the patients in the training and validation sets from Section 4.1—to eliminate the chance of test set bounding box locations directly influencing the random location generation—and mapped them onto a reference image from the test set.

The values of the 2D fracture prevalence map was flattened into a 1D vector and used as weights for generating a series of random points from within the reference image's segmentation

mask. After the indices from the 1D vector are randomly drawn, the indices get converted back into 2D indices that become the center coordinates for bounding boxes (white dots in Figure 5.4).

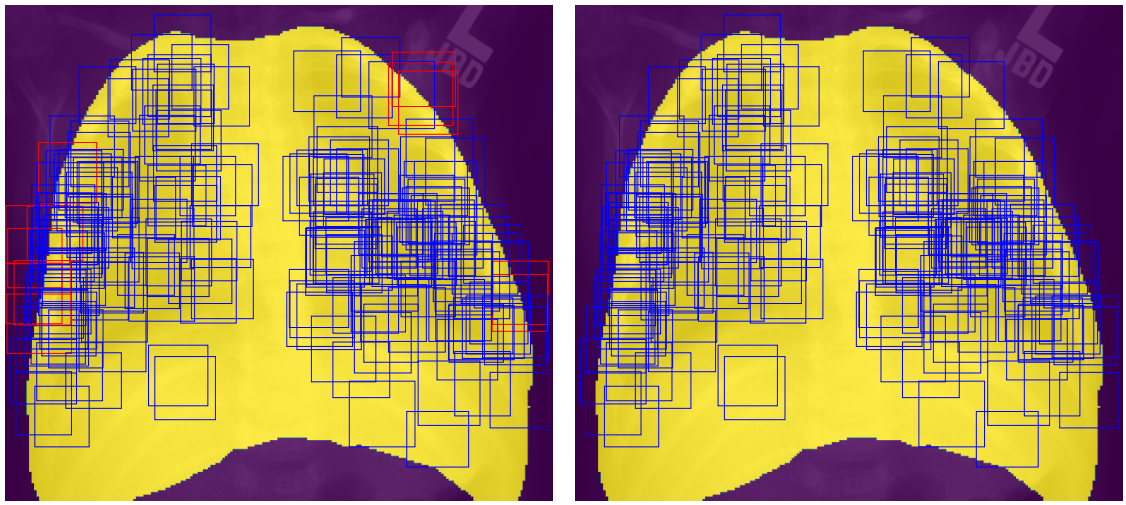


Figure 5.5 Example image with 200 bounding boxes randomly drawn from the fracture prevalence map. The yellow area represents the segmentation mask output from the trained U-Net. (left) all 200 boxes, where red represents boxes that do not have at least 65% of pixels within the segmentation mask, (right) red boxes removed, yielding the final set of 188 bounding box regions for the image.

Figure 5.5 shows what the bounding boxes generated from those center coordinates look like on the reference image. The center coordinates were all drawn from the set of indices present inside the yellow segmentation mask. In the left image, there are red boxes that represent boxes that do not have at least 65% pixel-overlap with the segmentation mask; we decide to remove them from the set since being on the periphery of the mask yields a box with a large portion being non-rib containing. On this example with 200 randomly generated boxes, 12 boxes are below this 65% threshold and are removed leaving a set of 188 bounding box regions for the given image.

5.2.2 Second stage: Patch Fracture Classifier

The second-stage classifier portion of our method will again be using ResNet-50 and DenseNet-121 like explored in the preliminary exhaustive search work. However, due to the large difference in sizes and patch content between the bounding boxes proposed in the fracture prevalence region proposal stage, we cannot directly utilize the trained networks from before. Thus, we create a new, label and segmentation-guided random patch dataset.

With each of the 1, 109 patients in our overall dataset, we use its associated segmentation mask to sample 10, 000 random points within the mask. Each random point is a potential location for the center of a bounding box, and temporary bounding box coordinates are created for all points. The random drawing of these points were done uniformly and not weighted based on the fracture prevalence map like above. The width and height of the bounding box are randomly drawn from the mean ± standard deviation of the widths and heights of all labeled fractures from our radiologist reads. Each generated box is then compared to the set of all ground truth boxes within the given image using an IOU threshold of 0.40 if it is a fracture-present patient image.

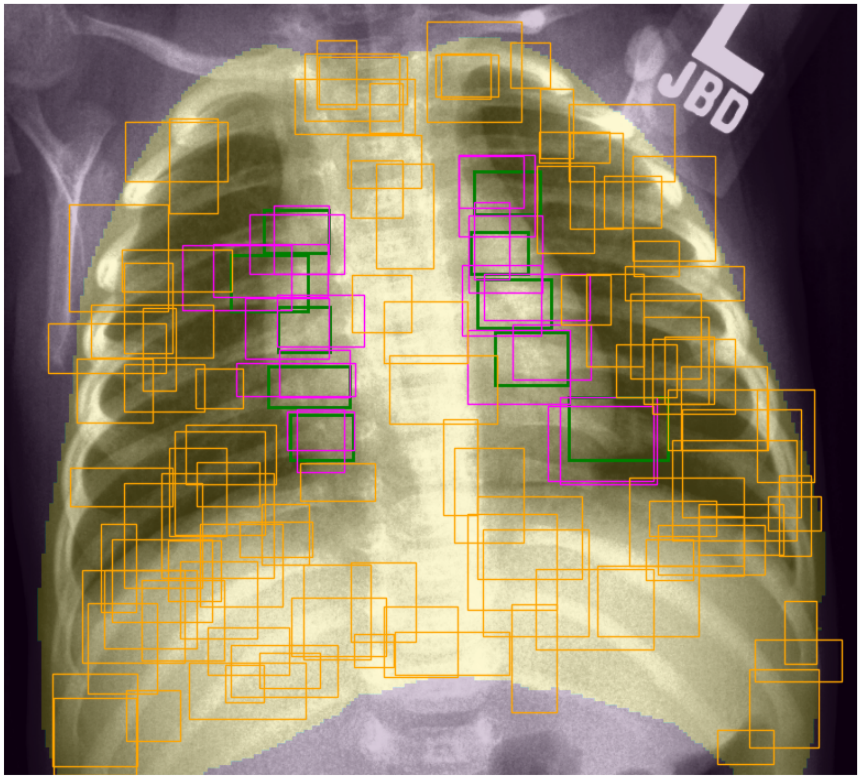


Figure 5.6 Example output of proposals for an image using the label and segmentation-guided region generator. Green boxes represent ground truth fracture labels. Magenta boxes are boxes generated from ground truth labels. Orange boxes are boxes randomly sampled from the segmentation mask (yellow pixels).

Every box is marked as whether they have a 65% pixel overlap with the segmentation mask, similar to the region proposal stage. Then, for boxes that have a non-zero IOU value with a ground truth, an additional threshold of 60% overlap with ground truth was added to ensure generated

bounding boxes were large enough to contain enough relevant information. For all generated boxes that both exceeded the IOU and overlap thresholds, two bounding boxes were chosen for each associated ground truth label in the image. In other words, if an image contained 6 labeled annotations, there will be 12 generated boxes with each actual fracture being represented by two boxes of varying size and aspect ratio. From the remaining possible locations with no overlap with any ground truth fractures, 100 of the boxes were randomly selected. Figure 5.6 illustrates this label and segmentation-guided random patch proposal for a single image.

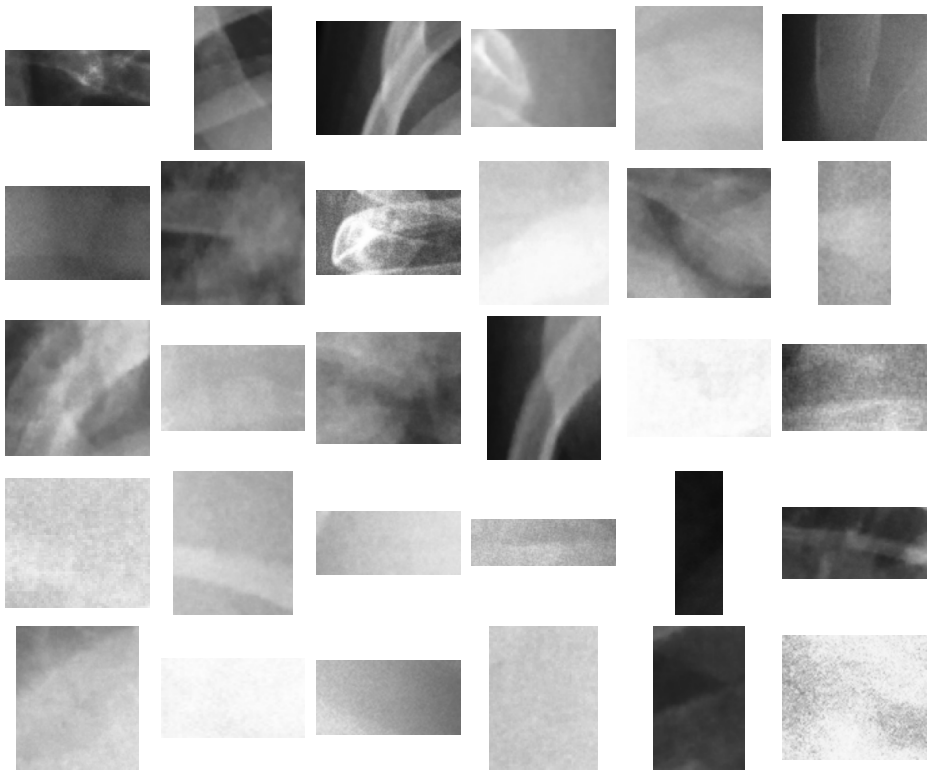


Figure 5.7 An example of 30 random patches generated for this dataset. 15 of these are fracture-present, and 15 are fracture-absent. This illustrates the difficulty of classifying patches as fracture-present or absent.

Figure 5.7 shows a small 30-patch subset from this random patch generation, illustrating the difficulty in classifying these patches. This process yields an overall dataset size of $118,223$ patches, of which $5,049$ (4.3%) are fracture-present and $113,174$ are fracture-absent. This is an even smaller ratio of fracture-present to absent patches, however the patches containing fractures in this set are specifically based on the ground-truth labels. Rather than doing a generic 70/10/20

train/val/test split like we did in the exhaustive section, we instead used one of the cross-validation folds with the same split of training, validation, and test patients used throughout Chapter 4.

5.2.3 Results for Two-Stage Detection

As before, both ResNet-50 and DenseNet-121 were trained to a maximum of 100 epochs with batch sizes of 64 with an Adam optimizer. Using a single V100S NVIDIA graphics card, completing all epochs of training for ResNet-50 and DenseNet-121 took 549 minutes (9.15 hours) and 524 minutes (8.73 hours), respectively.

We tested three different class-balancing weightings for the binary cross-entropy function, with a maximum of approximately 21.6 (the ratio of fracture-absent to fracture-present patches in the training set) and two smaller ratios of this value, one 0.66× (around 14.3) and one 0.33× (around 7.1). The precision-recall and ROC curves for ResNet-50 and DenseNet-121 are provided in Figures 5.8 and 5.9.

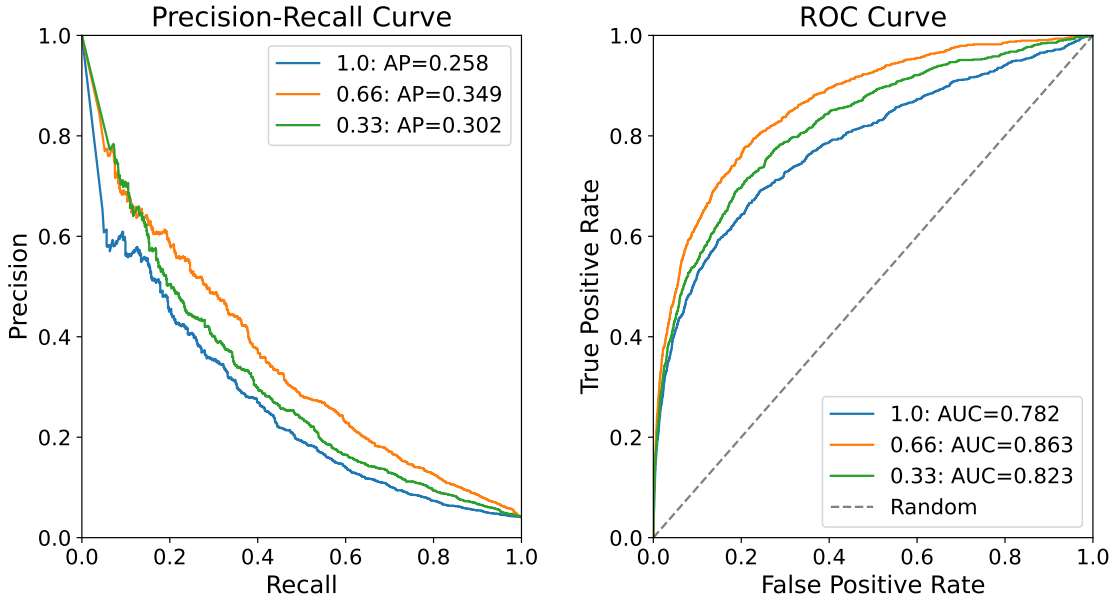


Figure 5.8 Precision-Recall and ROC Curves for ResNet-50 with three different positive class-balancing weights.

We can see that the performance for both of these classification networks is very similar on the label + segmentation-guided data set. DenseNet-121 slightly out-performs ResNet-50 in both average precision as well as ROC-AUC. For ResNet-50, the best performing model used a 0.66×

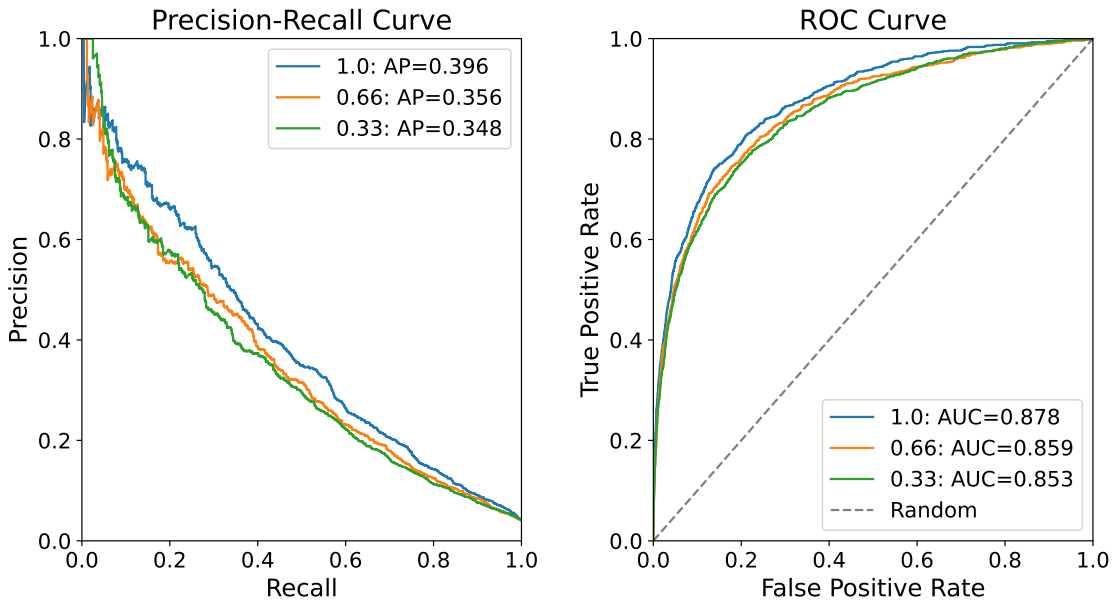


Figure 5.9 Precision-Recall and ROC Curves for DenseNet-121 with three different positive class-balancing weights.

positive class-balancing weight of 14.3, whereas DenseNet-121 saw the highest performance with the full 21.6 weighting value. Both models perform slightly better than their trained versions on the exhaustive search set; for instance, ResNet-50 improved AP from 0.334 to 0.349 and ROC-AUC from 0.855 to 0.863.

Table 5.2 shows the performance of each of the three weighted ResNet-50 and DenseNet-121 models with a static acceptance threshold of 0.50. Once again, both networks seem to perform worse than if every patch was simply negatively-labeled as '0', though with much closer performance than seen in the exhaustive search test. Precision and recall values at this acceptance threshold are relatively low but more consistent with one-another. Interestingly, ResNet-50 had higher precision and lower recall than DenseNet-121. However, due to our concern for higher recall, DenseNet-121 performs significantly better in terms of F2 score.

Finally, we explore how these classifiers perform when implementing the fracture prevalence-weighted region proposal stage on the test set of images. We use the best performing version of each model: ResNet-50 with a positive class-weighting of 14.3 and DenseNet-121 with a class-weighting of 21.6. We test a range of box proposals for each image (i.e., 50, 100, 150, 200, 250,

Table 5.2 Performance of ResNet-50 and DenseNet-121 on classifying fracture-present versus absent patches found through a label and segmentation-guided random proposal process. Threshold of 0.50 used for setting labels as fracture-present and fracture-absent.

Model	Weighting Ratio	Accuracy	Precision	Recall	F2 Score
ResNet-50	1.0	0.947	0.345	0.317	0.322
	0.66	0.957	0.473	0.315	0.337
	0.33	0.952	0.391	0.309	0.322
DenseNet-121	1.0	0.944	0.361	0.483	0.452
	0.66	0.935	0.315	0.497	0.446
	0.33	0.948	0.373	0.391	0.387

300) to gauge performance as the number of proposals increases. For 50 patches per image, the overall number of patches for the test set is $11,099$; for 300 boxes per image, there are $66,481$ boxes. The predictions were passed through a non-max suppression stage with an IOU threshold of 0.50 to remove overlapping predictions. For reference, this reduced the 50 boxes per image set down to around $9,700$ and the 300 boxes per image set down to just under $37,800$.

Figure 5.10 shows how both ResNet-50 and DenseNet-121 perform across these various proposal counts. As we would expect, increasing the number of regions proposed improves the recall performance of both methods. However, due to the networks' tendencies to falsely over-predict, the precision performance decreases as well. This leads to an almost unchanging F2 score across amounts of boxes proposed. Interestingly, we see the flipped precision and recall performances between the two networks leading to near-identical F2 performance between 100 and 250 proposals per image.

Table 5.3 Performance of using fine-tuned ResNet-50 and DenseNet-121 on detecting fractures in the test set using 200 region proposals per image from the fracture prevalence-weighted proposal stage. Error bars were calculated through a 1000 iteration bootstrapping of the test set.

	Precision	Recall	F2 Score
ResNet-50	0.097 ± 0.009	0.384 ± 0.028	0.241 ± 0.019
DenseNet-121	0.076 ± 0.006	0.507 ± 0.028	0.236 ± 0.016

Table 5.3 shows the performance of ResNet-50 and DenseNet-121 models compared to ground truth when using the same evaluation parameters as used in Chapter 4: a confidence acceptance

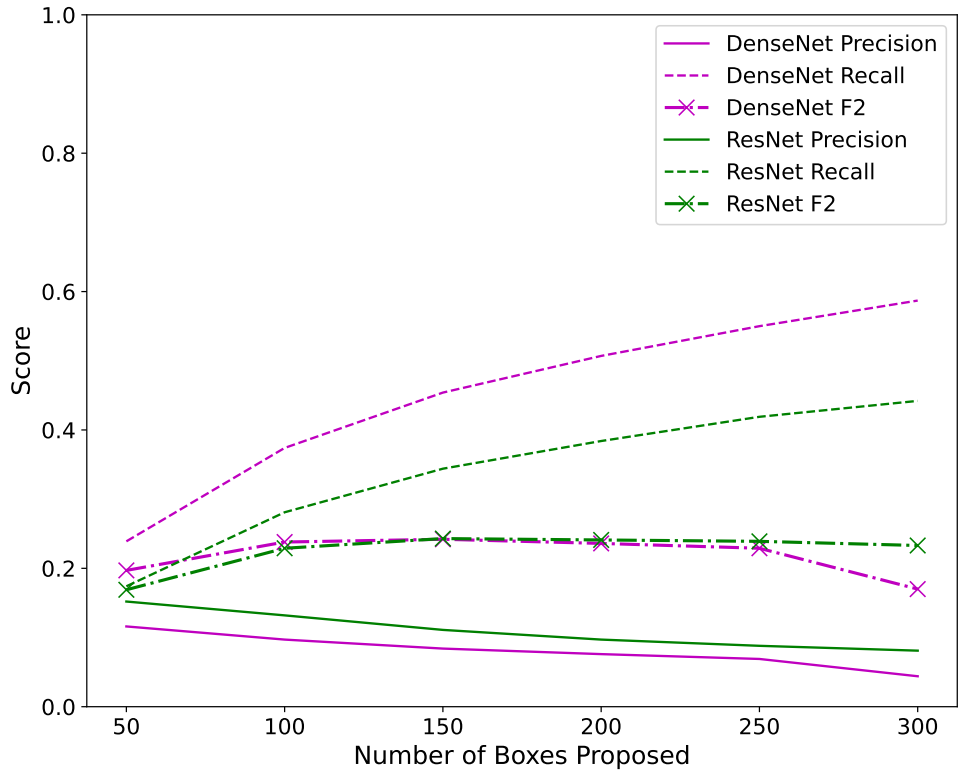


Figure 5.10 Performance of ResNet-50 and DenseNet-121 in terms of Precision, Recall, and F2 score across various numbers of proposed boxes per image from the fracture prevalence region proposal stage.

threshold of 0.50 and an IOU threshold of 0.30. As a reminder, the fine-tuned RetinaNet achieved 0.892 ± 0.015, 0.430 ± 0.014, and 0.480 ± 0.014 scores for precision, recall, and F2, respectively; YOLOv5 scored 0.897 ± 0.032, 0.434 ± 0.040, and 0.484 ± 0.037.

There are a couple reasons to explain such low performance. First, as mentioned previously these classifier models have a high likelihood to produce false positive results on the patches, leading to the demonstrably low precision score (e.g., DenseNet-121 only 271 generated bounding boxes matching as true positive matches compared to $3,319$ false positives). We do see DenseNet covering a higher number of fractures from the test set (50% opposed to 43% for RetinaNet and YOLOv5) though with much lower precision. This leads to much lower F2 score values compared to the base one-stage networks.

Taking inspiration from the ensembling efforts on the one-stage architectures in the previous chapter, we looked into two methods of combining the performance of the ResNet-50 and DenseNet-121 classifiers to possibly improve detection performance on the test set. The first technique is a very rough maximum probability selection, i.e., following inferencing with both models whichever model provided a higher probability score for each box was the score kept. Since these scores can only possibly increase, we expect that more boxes will have scores exceeding the 0.50 model confidence and therefore lead to more false positives as well as more accepted proposals matching ground truth. The second method can be considered a late-model fusion. The outputs from both models are averaged just prior to applying sigmoid. This offers more nuance, since the amplitude of discrepancies between the models can lead boxes that may have been marked as fracture-present for one model to be labeled fracture-absent, and vice-versa.

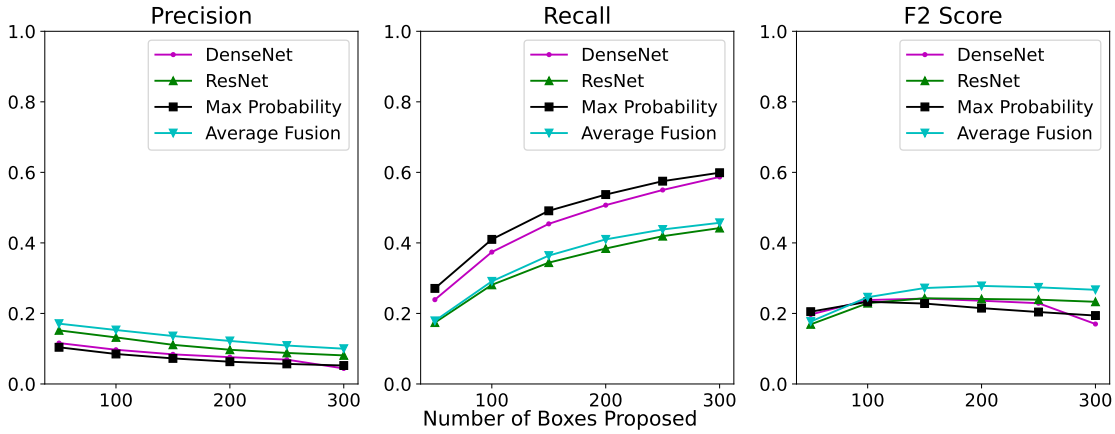


Figure 5.11 Performance of ResNet-50 and DenseNet-121 as well as a max probability and average fusion, in terms of Precision, Recall, and F2 scores, across increasing numbers of proposed boxes per image from the fracture prevalence-weighted region proposal stage.

Figure 5.11 shows the performance of the individual ResNet-50 and DenseNet-121 classifiers from Figure 5.10 as well as the two combination efforts. As expected, with the max probability method we see an improvement in recall across all amounts of box proposals with a decrease in precision. This leads to F2 score performance between 100 and 250 proposed boxes that is below any other method. With the average fusion method, we see that recall does not experience much of an increase in score; in fact, it gains a marginal improvement over the standalone ResNet-50.

However, this method led to the highest precision values across all region proposal counts. These two aspects combine to an overall best performing F2 score for our two-stage work.

Table 5.4 compares the results from our inter-reader study on the matching test set, the best performing one-stage detector including all of our recall-improving techniques, and the late-model fusion two-stage detector. It is clear that the severely low precision from both DenseNet-121 and ResNet-50 on this task is leading to a considerable drop in F2 score performance compared to radiologists and our best one-stage network.

Table 5.4 Performance of the best results across our work: expert radiologists, a 3x-YOLOv5 ensemble using a γ avalanche scheme, and our fracture prevalence region proposal (FPRP) with average late-model fusion with 200 boxes per image.

Avalanche	Precision	Recall	F2 Score	
Expert Radiologists	-	0.792	0.718	0.732
One-Stage: 3x-Y^{c}	$\gamma = 0.20$	0.573 ± 0.058	0.780 ± 0.030	0.725 ± 0.012
Two-Stage: FPRP + Fusion	-	0.122 ± 0.011	0.412 ± 0.028	0.278 ± 0.019

5.3 Summary and Future Work

Within this chapter, we present new work developing a custom, domain-inspired two-stage rib fracture detection network based on a weighted region proposal guided by fracture prevalence in our dataset. Fine-tuning a classifier network such as ResNet-50 and DenseNet-121 on a large set of fracture-present and fracture-absent patches reveals a high tendency of over-predicting fractures when considering a standard acceptance threshold of 0.50. When considering the best performing version of the two-stage work, using an averaged late-model fusion, the recall score achieved nearly matches the base RetinaNet and YOLOv5 networks; however, the precision is drastically lower with only a 12% likelihood of a predicted box correctly containing a rib fracture.

Future work can be done on both the region proposal stage as well as the classifier stage. Despite being weighted by our fracture prevalence map, regions are still drawn randomly. This can lead to newly drawn regions significantly overlapping existing boxes, both reducing the chance of representing enough of the fractures within the dataset as well as increasing the chance for incorrectly labeled boxes. Additionally, a more sophisticated version of the region proposer could

include a way of segmenting out ribs themselves so that regions drawn from the prevalence map are centered on a rib instead of predominantly the intercostal regions.

The poor capabilities of the classifier networks is a main contributor to the lackluster detection performance of the two-stage work. There could be further effort to tune hyperparameters to yield the best possible results, such as more thorough positive class-balancing weight or varied probability acceptance thresholds. Given the sheer difficulty of classifying small image patches and the incredible class imbalance, perhaps different loss function (e.g. focal loss [28]) would lead to better performance. There may also be more impressive ways of implementing multi-model fusion to combine results as opposed to averaging the pre-sigmoid outputs. Finally, since we have separately segmented out regions from our U-Net3+ architecture, incorporating thoracic-region location information alongside the image patches in a multi-modal fashion could provide additional context to the classifiers to yield better performance across the metrics.

CHAPTER 6

CONCLUSION

Throughout this dissertation, we presented multiple approaches to improve the performance for automatic pediatric rib fracture detection. We acknowledge that there are various limitations throughout the work in this project. For example, we demonstrated the value of ensembling models to increase recall; there could be multiple ways to combine regions from ensembled models. In this effort, we only explored combining model results with non-maximum suppression (NMS) of all proposed boxes. This is an affirmative approach that inherently will only result in equal or better recall (coupled with equal or worse precision). Future work could explore additional methods for merging model results.

The data used in our training and testing was limited. We used a relatively small dataset from a single institution; future efforts are needed to replicate these results and ensure generalizability in larger, more diverse datasets. Moreover, our 624 fracture-present images were labeled with single reads and therefore our fracture-present labels are noisy and likely contain errors. This is especially likely considering the challenge of detecting pediatric fractures and given that our inter-reader variability sub-study showed that along with other measures, the recall between radiologists was just under 71% demonstrating over one-quarter of all fractures were missed by the second reader. Future work using consensus interpretation is needed to improve our labels. Finally, the performance was evaluated on a test set with half fracture-present cases and half fracture-absent cases. While the prevalence of rib fractures in real-world clinical settings will vary depending on the site and nature of the practice (out-patient versus emergency room setting, etc.), this 50% prevalence test set contains a higher likelihood of fractures than would be encountered in practice. Future work is needed to determine performance in realistic clinical settings with more thorough comparisons to expert human performance.

The hypothetical end-goal of this work is to be implemented as a semi-autonomous procedure, i.e., the work demonstrated here would constitute the automated portion providing an initial read of potential regions of interest with a follow-up by a trained radiologist to use the model predictions

as guidance for their review. Given the performance of the detection networks compared to expert radiologists, we are unsure how effective the model predictions would actually be in a clinical setting. For example, on images where the detector outputs numerous box proposals, e.g. greater than 10, would it possibly take longer for the radiologist to thoroughly vet each of the regions proposed to reach a conclusion of actual rib fracture regions as opposed to performing their own interpretation without the AI influence. Therefore, future work is needed to evaluate the proposed methods as an in-line augmentation strategy with a radiologist serving as the arbitrator of model predictions. The importance of being able to trust the object detector proposals extends much further when considering the possibility of implementing an entirely automated system to serve as a sole diagnostic decider, i.e., an image receiving any number of predicted rib fractures directly generating an alert of possible child abuse. This raises ethical concerns, as the accusation of child abuse and neglect is not minor and has potentially massive implications for anyone who may be involved. Even if the object detectors we trained demonstrated performance far exceeding the approximately 0.7 precision and recall of the radiologist inter-reader sub-study, a lack of expert human input could lead to doubt regarding the actual presence of abuse on the patient given the black-box nature of these types of deep learning architectures.

Foundation models [128] are some of the most cutting-edge models and have begun taking over the AI and machine learning spheres by storm in recent years. The idea behind these models is that they are trained on such a large corpus of multi-modal information that they do not need to be specifically fine-tuned to be able to provide output for a given task. Prominent examples include the BERT [129] and GPT [130] language models and the DALL-E [131] and Stable Diffusion [132] text-to-image models. One especially interesting foundation model is the Segment Anything project by Meta AI [133] that provides zero-shot segmentations of any images uploaded. We wanted to see whether this model could provide any sort of comparison to the task-specific, fine-tuned work we have discussed throughout this dissertation. In its demo, three different tasks can be performed: hovering over a region and clicking within it to segment it, drawing a box to segment what it contains, and an automatic “segment every object" task they call “everything."

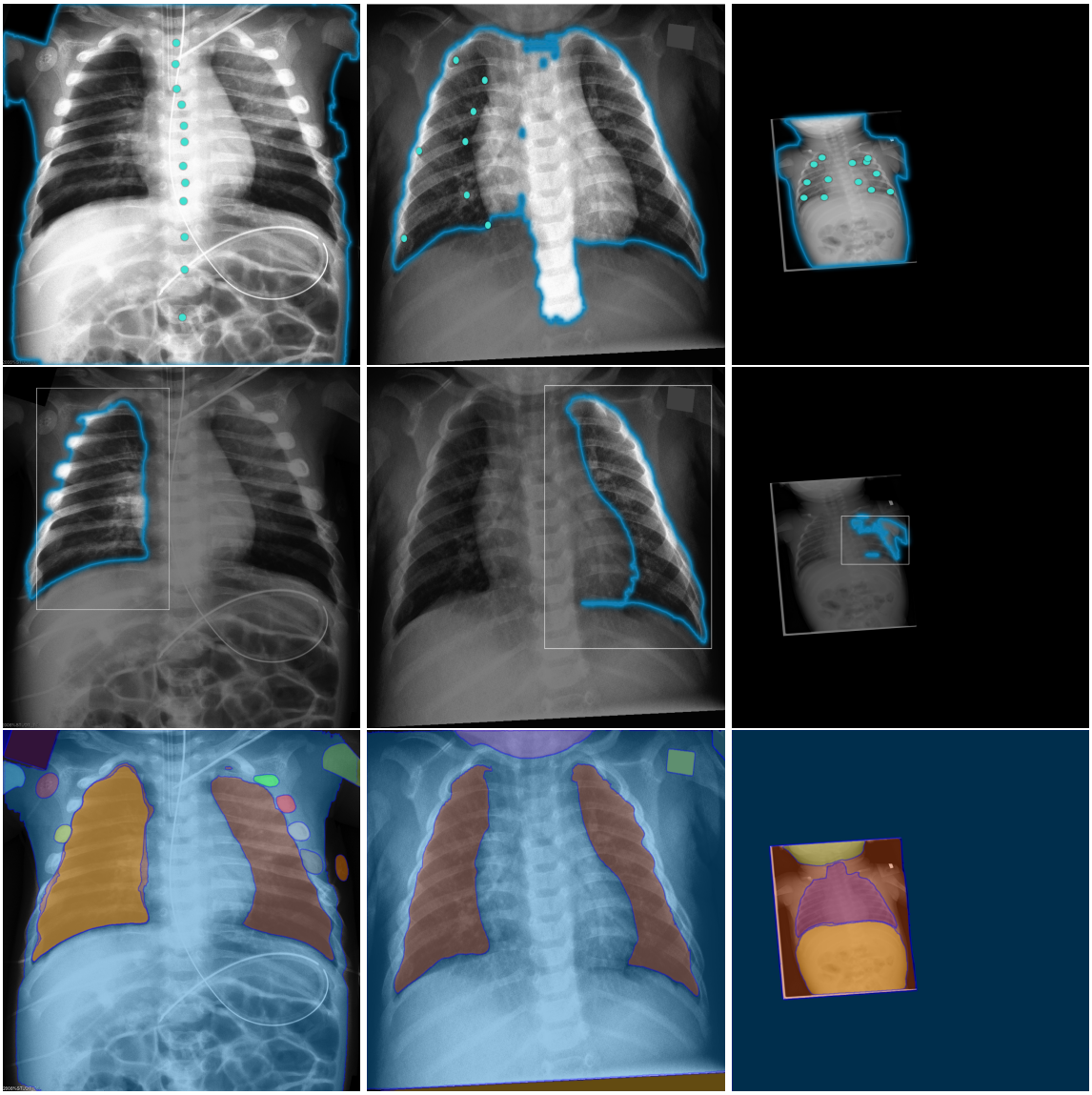


Figure 6.1 Examples of outputs from the Segment Anything model [133] on three different patients in our dataset. The top row is a hover-click option where hovering will show the region it would segment based on that location and placing a dot locks in that part of the segmentation. The center shows segmenting based on a user-drawn box. The bottom row uses their “everything" option to segment every object it finds.

With the hover-click option, we see quite erratic performance. For the first patient, we attempted to have it segment the spine and put a dot on each of the spinal vertebrae; it instead just segmented out the entire patient outline. For the second patient, we wanted to segment out the patient's right lung but it segmented both lungs as well as a protruding region down the spine. Like the first patient, the last patient just resulted in outlining the patient regardless of where dots were placed within the lungs. The box-segmentation task was slightly better. In the first patient we see a mostly segmented region of the right lung. We see similar results in the second patient, where the segmentation clearly avoids the heart. It struggles the most in the last patient, where it really has no idea what to segment within the box. Finally, using their "everything" option which attempts to find and segment every object within the image, we see quite varying results between the three patients. The first patient is the only to have some of the lateral arcs of the ribs being highlighted as separate objects. In the second patient, the only segmented regions are very roughly around the lungs as well as the skull (disregarding the added anomalous square in the top right of the image). When using an extreme case of an image where most of the image is blank, we see the model created a segmentation of the skull, the top part of the torso, and the bottom torso. Across the example patient images, we see that the segmentation performance of the thoracic cavity is no where near as capable as the custom fine-tuned model we trained and discuss in Section 3.3 that we use for tight cropping, masking for the image processing techniques, and in the region proposal two-stage work. Additionally, with each of these patients containing rib fractures (see Figure 4.3 for locations of fractures in each image), we only see overlap with the first patient with the lateral arcs of the ribs being segmented out. However, this is clearly due to those regions presenting as much brighter, separating them from the surrounding rib pixels rather than any indication that they may be rib fractures. Once again, we see that, for now, a task as difficult as the detection of rib fractures on pediatric radiographs, there is a need for high performing, customized, fine-tuned detector models. That said, if current performance trends in AI continue, future foundation models may have a role for this challenging task.

In conclusion, we demonstrated multiple methods that improve the sensitivity (i.e., recall)

performance of state-of-the-art object detectors on a custom curated dataset of pediatric chest radiographs. We present developments for two-stage and one-stage objects detectors. The one-stage methods performed much better than two-stage methods with innovations that include our novel avalanche decision scheme as well as three methods of pre-processing the images. Additionally, various ensembling approaches combined with the aforementioned techniques were investigated. These techniques provided reduced precision with higher recall resulting in improvements in F2 score by extension. Simple ensembles, such as same-model three- and six-model ensembles, offered straightforward improvements over single-model detectors. This is likely due to the enhanced generalizability by training each ensemble member on different cross-validation folds of the training and validation data sets. Interestingly, many of the best performing models utilized the blended method (**c**) of pre-processing, where each channel of the input images was processed differently. The method with the highest F2 score was an ensemble of three YOLOv5 models using the input (**c**) pre-processing and with the $\gamma = 0.20$ avalanche scheme. This model achieved an F2 score of 0.725 $\pm$ 0.012, which was only approximately 1% below the inter-reader variability of expert radiologists of 0.732 and with a recall score exceeding the experts at 0.780 $\pm$ 0.030 versus 0.718. Overall, this work demonstrates promising methods for high sensitivity rib fracture detection that could serve as automatic approaches to augment the performance of radiologists performing pediatric chest radiograph interpretation.

BIBLIOGRAPHY

[1] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain," *Psychological Review*, vol. 65, no. 6, pp. 386–408, 1958.

[2] G. E. Hinton, "Rectified Linear Units Improve Restricted Boltzmann Machines Vinod Nair," 2010.

[3] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, pp. 533–536, Oct. 1986.

[4] H. Robbins and S. Monro, "A Stochastic Approximation Method," *The Annals of Mathematical Statistics*, vol. 22, pp. 400–407, Sept. 1951.

[5] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," Jan. 2017.

[6] D. Scherer, A. Müller, and S. Behnke, "Evaluation of Pooling Operations in Convolutional Architectures for Object Recognition," in *Artificial Neural Networks – ICANN 2010* (K. Diamantaras, W. Duch, and L. S. Iliadis, eds.), Lecture Notes in Computer Science, (Berlin, Heidelberg), pp. 92–101, Springer, 2010.

[7] Y.-L. Boureau, J. Ponce, and Y. LeCun, "A Theoretical Analysis of Feature Pooling in Visual Recognition," in *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pp. 111–118, 2010.

[8] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," *Advances in Neural Information Processing Systems*, vol. 25, 2012.

[9] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *arXiv:1409.0575 [cs]*, Jan. 2015.

[10] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, pp. 2278–2324, Nov. 1998.

[11] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going Deeper with Convolutions," *arXiv:1409.4842 [cs]*, Sept. 2014.

[12] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *arXiv:1409.1556 [cs]*, Apr. 2015.

[13] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, June 2016.

[14] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated Residual Transformations for Deep Neural Networks," *arXiv:1611.05431 [cs]*, Apr. 2017.

[15] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely Connected Convolutional Networks," *arXiv:1608.06993 [cs]*, Jan. 2018.

[16] M. Tan and Q. V. Le, "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks," *arXiv:1905.11946 [cs, stat]*, Sept. 2020.

[17] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie, "A ConvNet for the 2020s," Mar. 2022.

[18] S. Sarin, "VGGNet vs ResNet (The Vanishing Gradient Problem)." https://towardsdatascience.com/vggnet-vs-resnet-924e9573ca5c, Apr. 2020.

[19] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training Recurrent Neural Networks," *arXiv:1211.5063 [cs]*, Feb. 2013.

[20] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, June 2009.

[21] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 580–587, June 2014.

[22] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders, "Selective Search for Object Recognition," *International Journal of Computer Vision*, vol. 104, pp. 154–171, Sept. 2013.

[23] R. Girshick, "Fast R-CNN," *arXiv:1504.08083 [cs]*, Sept. 2015.

[24] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," *arXiv:1506.01497 [cs]*, Jan. 2016.

[25] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," *arXiv:1703.06870 [cs]*, Jan. 2018.

[26] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 779–788, June 2016.

[27] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single Shot MultiBox Detector," *arXiv:1512.02325 [cs]*, vol. 9905, pp. 21–37, 2016.

[28] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal Loss for Dense Object Detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, pp. 318–327, Feb. 2020.

[29] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The Pascal Visual Object Classes (VOC) Challenge," *International Journal of Computer Vision*, vol. 88, pp. 303–338, June 2010.

[30] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár, "Microsoft COCO: Common Objects in Context," *arXiv:1405.0312 [cs]*, Feb. 2015.

[31] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," *arXiv:1804.02767 [cs]*, Apr. 2018.

[32] M. Tan, R. Pang, and Q. V. Le, "EfficientDet: Scalable and Efficient Object Detection," *arXiv:1911.09070 [cs, eess]*, July 2020.

[33] G. Jocher, A. Chaurasia, A. Stoken, J. Borovec, NanoCode012, Y. Kwon, K. Michael, TaoXie, J. Fang, imyhxy, Lorna, Z. Yifu, C. Wong, A. V, D. Montes, Z. Wang, C. Fati, J. Nadar, Laughing, UnglvKitDe, V. Sonck, tkianai, yxNONG, P. Skalski, A. Hogan, D. Nair, M. Strobel, and M. Jain, "Ultralytics/yolov5: V7.0 - YOLOv5 SOTA realtime instance segmentation," Nov. 2022.

[34] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature Pyramid Networks for Object Detection," *arXiv:1612.03144 [cs]*, Apr. 2017.

[35] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," *arXiv:2004.10934 [cs, eess]*, Apr. 2020.

[36] M. Frid-Adar, E. Klang, M. Amitai, J. Goldberger, and H. Greenspan, "Synthetic data augmentation using gan for improved liver lesion classification," in *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)*, pp. 289–293, 2018.

[37] H. Rashid, M. A. Tanveer, and H. Aqeel Khan, "Skin lesion classification using gan based data augmentation," in *2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 916–919, 2019.

[38] C. Bowles, L. Chen, R. Guerrero, P. Bentley, R. Gunn, A. Hammers, D. A. Dickie, M. V. Hernández, J. Wardlaw, and D. Rueckert, "Gan augmentation: Augmenting training data using generative adversarial networks," 2018.

[39] K. Saleh, S. Szénási, and Z. Vámossy, "Occlusion handling in generic object detection: A review," in *2021 IEEE 19th World Symposium on Applied Machine Intelligence and Informatics (SAMI)*, pp. 000477–000484, 2021.

[40] T. C. Arlen, "Understanding the mAP Evaluation Metric for Object Detection," Mar. 2018.

[41] H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, and S. Savarese, "Generalized Intersection over Union: A Metric and A Loss for Bounding Box Regression," *arXiv:1902.09630 [cs]*, Apr. 2019.

[42] L. R. Dice, "Measures of the Amount of Ecologic Association Between Species," *Ecology*, vol. 26, no. 3, pp. 297–302, 1945.

[43] D. P. Chakraborty and L. H. Winter, "Free-response methodology: Alternate analysis and a new observer-performance experiment.," *Radiology*, vol. 174, pp. 873–881, Mar. 1990.

[44] A. B. Rosenkrantz, D. R. Hughes, and R. Duszak, "The U.S. Radiologist Workforce: An Analysis of Temporal and Geographic Variation by Using Large National Datasets," *Radiology*, vol. 279, pp. 175–184, Apr. 2016.

[45] D. P. Naidich, C. H. Marshall, C. Gribbin, R. S. Arams, and D. I. McCauley, "Low-dose CT of the lungs: Preliminary observations.," *Radiology*, vol. 175, pp. 729–731, June 1990.

[46] C. Rampinelli, D. Origgi, and M. Bellomi, "Low-dose CT: Technique, reading methods and image interpretation," *Cancer Imaging*, vol. 12, pp. 548–556, Feb. 2013.

[47] S. Rob, T. Bryant, I. Wilson, and B. K. Somani, "Ultra-low-dose, low-dose, and standard-dose CT of the kidney, ureters, and bladder: Is there a difference? Results from a systematic review of the literature," *Clinical Radiology*, vol. 72, pp. 11–15, Jan. 2017.

[48] S. A. Khan, Y. Gulzar, S. Turaev, and Y. S. Peng, "A modified hsift descriptor for medical image classification of anatomy objects," *Symmetry*, vol. 13, no. 11, 2021.

[49] J. Kissane, J. A. Neutze, and H. Singh, *Conventional Radiology*, pp. 15–20. Cham: Springer International Publishing, 2020.

[50] R. S. o. N. A. R. a. A. C. of Radiology (ACR), "Radiation Dose." https://www.radiologyinfo.org/en/info/safety-xray, Nov. 2022.

[51] R. Fazel, H. M. Krumholz, Y. Wang, J. S. Ross, J. Chen, H. H. Ting, N. D. Shah, K. Nasir, A. J. Einstein, and B. K. Nallamothu, "Exposure to Low-Dose Ionizing Radiation from Medical Imaging Procedures," *New England Journal of Medicine*, vol. 361, pp. 849–857, Aug. 2009.

[52] D. Frush, "The cumulative radiation dose paradigm in pediatric imaging," *The British Journal of Radiology*, vol. 94, p. 20210478, Oct. 2021.

[53] A. Yeung, "The 'as low as reasonably achievable' (alara) principle: a brief historical overview and a bibliometric analysis of the most cited publications," *Radioprotection*, 2019.

[54] E. Bercovich and M. C. Javitt, "Medical Imaging: From Roentgen to the Digital Revolution, and Beyond," *Rambam Maimonides Medical Journal*, vol. 9, p. e0034, Oct. 2018.

[55] M. Larobina and L. Murino, "Medical Image File Formats," *Journal of Digital Imaging*, vol. 27, pp. 200–206, Apr. 2014.

[56] T. Ridnik, E. Ben-Baruch, A. Noy, and L. Zelnik-Manor, "ImageNet-21K Pretraining for the Masses," Aug. 2021.

[57] C. Sun, A. Shrivastava, S. Singh, and A. Gupta, "Revisiting Unreasonable Effectiveness of Data in Deep Learning Era," Aug. 2017.

[58] World Health Organization, *Communicating Radiation Risks in Paediatric Imaging: Information to Support Health Care Discussions about Benefit and Risk*. Geneva: World Health Organization, 2016.

[59] L. Iyeke, R. Moss, R. Hall, J. Wang, L. Sandhu, B. Appold, E. Kalontar, D. Menoudakos, M. Ramnarine, S. P. LaVine, S. Ahn, and M. Richman, "Reducing Unnecessary 'Admission' Chest X-rays: An Initiative to Minimize Low-Value Care," *Cureus*, vol. 14, p. e29817, Oct. 2022.

[60] H. H. Publishing, "Radiation risk from medical imaging." https://www.health.harvard.edu/cancer/radiation-risk-from-medical-imaging, Sept. 2021.

[61] J. Irvin, P. Rajpurkar, M. Ko, Y. Yu, S. Ciurea-Ilcus, C. Chute, H. Marklund, B. Haghgoo, R. Ball, K. Shpanskaya, J. Seekins, D. A. Mong, S. S. Halabi, J. K. Sandberg, R. Jones, D. B. Larson, C. P. Langlotz, B. N. Patel, M. P. Lungren, and A. Y. Ng, "CheXpert: A Large Chest Radiograph Dataset with Uncertainty Labels and Expert Comparison," Jan. 2019.

[62] K. Yan, X. Wang, L. Lu, and R. M. Summers, "DeepLesion Automated Deep Mining, Categorization and Detection of Significant Radiology Image Findings using Large-Scale Clinical Lesion Annotations," *arXiv:1710.01766 [cs]*, Oct. 2017.

[63] M. P. McBee, O. A. Awan, A. T. Colucci, C. W. Ghobadi, N. Kadom, A. P. Kansagra, S. Tridandapani, and W. F. Auffermann, "Deep Learning in Radiology," *Academic Radiology*, vol. 25, pp. 1472–1480, Nov. 2018.

[64] A. Esteva, B. Kuprel, R. A. Novoa, J. Ko, S. M. Swetter, H. M. Blau, and S. Thrun, "Dermatologist-level classification of skin cancer with deep neural networks," *Nature*, vol. 542, pp. 115–118, Feb. 2017.

[65] H. A. Haenssle, C. Fink, R. Schneiderbauer, F. Toberer, T. Buhl, A. Blum, A. Kalloo, A. B. H. Hassen, L. Thomas, A. Enk, L. Uhlmann, C. Alt, M. Arenbergerova, R. Bakos, A. Baltzer, I. Bertlich, A. Blum, T. Bokor-Billmann, J. Bowling, N. Braghiroli, R. Braun, K. Buder-Bakhaya, T. Buhl, H. Cabo, L. Cabrijan, N. Cevic, A. Classen, D. Deltgen, C. Fink, I. Georgieva, L.-E. Hakim-Meibodi, S. Hanner, F. Hartmann, J. Hartmann, G. Haus, E. Hoxha, R. Karls, H. Koga, J. Kreusch, A. Lallas, P. Majenka, A. Marghoob, C. Massone, L. Mekokishvili, D. Mestel, V. Meyer, A. Neuberger, K. Nielsen, M. Oliviero, R. Pampena, J. Paoli, E. Pawlik, B. Rao, A. Rendon, T. Russo, A. Sadek, K. Samhaber, R. Schneiderbauer, A. Schweizer, F. Toberer, L. Trennheuser, L. Vlahova, A. Wald, J. Winkler, P. Wölbing, and I. Zalaudek, "Man against machine: Diagnostic performance of a deep learning convolutional neural network for dermoscopic melanoma recognition in comparison to 58 dermatologists," *Annals of Oncology*, vol. 29, pp. 1836–1842, Aug. 2018.

[66] L. Shen, L. R. Margolies, J. H. Rothstein, E. Fluder, R. McBride, and W. Sieh, "Deep Learning to Improve Breast Cancer Detection on Screening Mammography," *Scientific Reports*, vol. 9, p. 12495, Aug. 2019.

[67] Q. Hu, H. M. Whitney, and M. L. Giger, "A deep learning methodology for improved breast cancer diagnosis using multiparametric MRI," *Scientific Reports*, vol. 10, p. 10536, June 2020.

[68] S. M. McKinney, M. Sieniek, V. Godbole, J. Godwin, N. Antropova, H. Ashrafian, T. Back, M. Chesus, G. S. Corrado, A. Darzi, M. Etemadi, F. Garcia-Vicente, F. J. Gilbert, M. Halling-Brown, D. Hassabis, S. Jansen, A. Karthikesalingam, C. J. Kelly, D. King, J. R. Ledsam, D. Melnick, H. Mostofi, L. Peng, J. J. Reicher, B. Romera-Paredes, R. Sidebottom, M. Suleyman, D. Tse, K. C. Young, J. De Fauw, and S. Shetty, "International evaluation of an AI system for breast cancer screening," *Nature*, vol. 577, pp. 89–94, Jan. 2020.

[69] G. Holste, S. C. Partridge, H. Rahbar, D. Biswas, C. I. Lee, and A. M. Alessio, "End-to-End Learning of Fused Image and Non-Image Features for Improved Breast Cancer Classification from MRI," in *2021 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, (Montreal, BC, Canada), pp. 3287–3296, IEEE, Oct. 2021.

[70] M. A. Mazurowski, M. Buda, A. Saha, and M. R. Bashir, "Deep learning in radiology: An overview of the concepts and a survey of the state of the art with focus on MRI," *Journal of Magnetic Resonance Imaging*, vol. 49, no. 4, pp. 939–954, 2019.

[71] B. Zhang, C. Jia, R. Wu, B. Lv, B. Li, F. Li, G. Du, Z. Sun, and X. Li, "Improving rib fracture detection accuracy and reading efficiency with deep learning-based detection software: A clinical evaluation," *The British Journal of Radiology*, vol. 94, p. 20200870, Feb. 2021.

[72] M. Kaiume, S. Suzuki, K. Yasaka, H. Sugawara, Y. Shen, Y. Katada, T. Ishikawa, R. Fukui, and O. Abe, "Rib fracture detection in computed tomography images using deep convolutional neural networks," *Medicine*, vol. 100, p. e26024, May 2021.

[73] J. E. Burns, J. Yao, and R. M. Summers, "Artificial Intelligence in Musculoskeletal Imaging: A Paradigm Shift," *Journal of Bone and Mineral Research*, vol. 35, no. 1, pp. 28–35, 2020.

[74] L. Yao, X. Guan, X. Song, Y. Tan, C. Wang, C. Jin, M. Chen, H. Wang, and M. Zhang, "Rib fracture detection system based on deep learning," *Scientific Reports*, vol. 11, p. 23513, Dec. 2021.

[75] J. Zhang, Z. Li, S. Yan, H. Cao, J. Liu, and D. Wei, "An Algorithm for Automatic Rib Fracture Recognition Combined with nnU-Net and DenseNet," *Evidence-Based Complementary and Alternative Medicine*, vol. 2022, p. e5841451, Feb. 2022.

[76] L. Jin, J. Yang, K. Kuang, B. Ni, Y. Gao, Y. Sun, P. Gao, W. Ma, M. Tan, H. Kang, J. Chen, and M. Li, "Deep-learning-assisted detection and segmentation of rib fractures from CT scans: Development and validation of FracNet," *eBioMedicine*, vol. 62, p. 103106, Dec. 2020.

[77] X. Wang, Y. Peng, L. Lu, Z. Lu, M. Bagheri, and R. M. Summers, "ChestX-Ray8: Hospital-Scale Chest X-Ray Database and Benchmarks on Weakly-Supervised Classification and Localization of Common Thorax Diseases," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3462–3471, IEEE Computer Society, July 2017.

[78] Y.-X. Tang, Y.-B. Tang, Y. Peng, K. Yan, M. Bagheri, B. A. Redd, C. J. Brandon, Z. Lu, M. Han, J. Xiao, and R. M. Summers, "Automated abnormality classification of chest radiographs using deep convolutional neural networks," *npj Digital Medicine*, vol. 3, pp. 1–8, May 2020.

[79] R. Lindsey, A. Daluiski, S. Chopra, A. Lachapelle, M. Mozer, S. Sicular, D. Hanel, M. Gardner, A. Gupta, R. Hotchkiss, and H. Potter, "Deep neural network improves fracture detection by clinicians," *Proceedings of the National Academy of Sciences*, vol. 115, pp. 11591–11596, Nov. 2018.

[80] S. Anis, K. W. Lai, J. H. Chuah, S. M. Ali, H. Mohafez, M. Hadizadeh, D. Yan, and Z.-C. Ong, "An Overview of Deep Learning Approaches in Chest Radiograph," *IEEE Access*, vol. 8, pp. 182347–182354, 2020.

[81] Y. Gao, H. Liu, L. Jiang, C. Yang, X. Yin, J.-L. Coatrieux, and Y. Chen, "CCE-Net: A rib fracture diagnosis network based on contralateral, contextual, and edge enhanced modules," *Biomedical Signal Processing and Control*, vol. 75, p. 103620, May 2022.

[82] P. K. Kleinman, S. C. Marks, M. R. Spevak, and J. M. Richmond, "Fractures of the rib head in abused infants.," *Radiology*, vol. 185, pp. 119–123, Oct. 1992.

[83] T. R. Sanchez, H. Nguyen, W. Palacios, M. Doherty, and K. Coulter, "Retrospective evaluation and dating of non-accidental rib fractures in infants," *Clinical Radiology*, vol. 68, pp. e467–e471, Aug. 2013.

[84] R. M. Schwend, J. A. Schmidt, J. L. Reigrut, L. C. Blakemore, and B. A. Akbarnia, "Patterns of Rib Growth in the Human Child," *Spine Deformity*, vol. 3, pp. 297–302, July 2015.

[85] S. L. Wootton-Gorges, R. Stein-Wexler, J. W. Walton, A. J. Rosas, K. P. Coulter, and K. K. Rogers, "Comparison of computed tomography and chest radiography in the detection of rib fractures in abused infants," *Child Abuse & Neglect*, vol. 32, pp. 659–663, June 2008.

[86] J. S. Kondis, J. Muenzer, and J. D. Luhmann, "Missed Fractures in Infants Presenting to the Emergency Department With Fussiness," *Pediatric Emergency Care*, vol. 33, p. 538, Aug. 2017.

[87] A. Ghosh, D. Patton, S. Bose, M. K. Henry, M. Ouyang, H. Huang, A. Vossough, R. Sze, S. Sotardi, and M. Francavilla, "A Patch-Based Deep Learning Approach for Detecting Rib Fractures on Frontal Radiographs in Young Children," *Journal of Digital Imaging*, Mar. 2023.

[88] C. Kelly, C. Street, and M. E. S. Building, "Child Maltreatment 2020," *Child Maltreatment*, p. 313, 2020.

[89] C. Kelly, C. Street, and M. E. S. Building, "Child Maltreatment 2019," *Child Maltreatment*, p. 306, 2019.

[90] P. McMahon, W. Grossman, M. Gaffney, and C. Stanitski, "Soft-tissue injury as an indication of child abuse.," *JBJS*, vol. 77, p. 1179, Aug. 1995.

[91] A. M. Kemp, F. Dunstan, S. Harrison, S. Morris, M. Mann, K. Rolfe, S. Datta, D. P. Thomas, J. R. Sibert, and S. Maguire, "Patterns of skeletal fractures in child abuse: Systematic review," *BMJ*, vol. 337, p. a1518, Oct. 2008.

[92] S. E. Darling, S. L. Done, S. D. Friedman, and K. W. Feldman, "Frequency of intrathoracic injuries in children younger than 3 years with rib fractures," *Pediatric Radiology*, vol. 44, pp. 1230–1236, Oct. 2014.

[93] B. Bulloch, C. J. Schubert, P. D. Brophy, N. Johnson, M. H. Reed, and R. A. Shapiro, "Cause and Clinical Characteristics of Rib Fractures in Infants," *Pediatrics*, vol. 105, pp. e48–e48, Apr. 2000.

[94] N. K. Pandya, K. Baldwin, H. Wolfgruber, C. W. Christian, D. S. Drummond, and H. S. Hosalkar, "Child Abuse and Orthopaedic Injury Patterns: Analysis at a Level I Pediatric Trauma Center," *Journal of Pediatric Orthopaedics*, vol. 29, p. 618, Sept. 2009.

[95] K. A. Barsness, E.-S. Cha, D. D. Bensard, C. M. Calkins, D. A. Partrick, F. M. Karrer, and J. D. Strain, "The Positive Predictive Value of Rib Fractures as an Indicator of Nonaccidental Trauma in Children," *Journal of Trauma and Acute Care Surgery*, vol. 54, pp. 1107–1110, June 2003.

[96] G. Rosenberg, A. K. Bryant, K. A. Davis, and K. M. Schuster, "No breakpoint for mortality in pediatric rib fractures," *Journal of Trauma and Acute Care Surgery*, vol. 80, p. 427, Mar. 2016.

[97] D. F. Merten, M. A. Radkowski, and J. C. Leonidas, "The abused child: A radiological reappraisal.," *Radiology*, vol. 146, pp. 377–381, Feb. 1983.

[98] S. Jodogne, "The Orthanc Ecosystem for Medical Imaging," *Journal of Digital Imaging*, vol. 31, pp. 341–352, June 2018.

[99] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," May 2015.

[100] G. Holste, R. Sullivan, M. Bindschadler, N. Nagy, and A. Alessio, "Multi-class semantic segmentation of pediatric chest radiographs," in *Medical Imaging 2020: Image Processing*, vol. 11313, pp. 323–330, SPIE, Mar. 2020.

[101] H. Huang, L. Lin, R. Tong, H. Hu, Q. Zhang, Y. Iwamoto, X. Han, Y.-W. Chen, and J. Wu, "UNet 3+: A Full-Scale Connected UNet for Medical Image Segmentation," Apr. 2020.

[102] I. Barber, J. M. Perez-Rossello, C. R. Wilson, and P. K. Kleinman, "The yield of high-detail radiographic skeletal surveys in suspected infant abuse," *Pediatric Radiology*, vol. 45, pp. 69–80, Jan. 2015.

[103] S. Kriss, A. Thompson, G. Bertocci, M. Currie, and V. Martich, "Characteristics of rib fractures in young abused children," *Pediatric Radiology*, vol. 50, pp. 726–733, May 2020.

[104] V. K. Kundal, P. R. Debnath, A. K. Meena, S. Shah, P. Kumar, S. S. Sahu, and A. Sen, "Pediatric Thoracoabdominal Trauma: Experience from a Tertiary Care Center," *Journal of Indian Association of Pediatric Surgeons*, vol. 24, no. 4, pp. 264–270, 2019.

[105] A. Tsai, J. Perez-Rossello, S. A. Connolly, K. Kawai, and P. K. Kleinman, "Temporal Pattern of Radiographic Findings of Costochondral Junction Rib Fractures on Serial Skeletal Surveys in Suspected Infant Abuse," *American Journal of Roentgenology*, vol. 216, pp. 1649–1658, June 2021.

[106] P. Worlock, M. Stower, and P. Barbor, "Patterns of fractures in accidental and non-accidental injury in children: A comparative study.," *Br Med J (Clin Res Ed)*, vol. 293, pp. 100–102, July 1986.

[107] C. W. Paine, O. Fakeye, C. W. Christian, and J. N. Wood, "Prevalence of Abuse Among Young Children with Rib Fractures: A Systematic Review," *Pediatric emergency care*, vol. 35, pp. 96–103, Feb. 2019.

[108] H. Carty and A. Pierce, "Non-accidental injury: A retrospective analysis of a large cohort," *European Radiology*, vol. 12, pp. 2919–2925, Dec. 2002.

[109] R. T. Loder and J. R. Feinberg, "Orthopaedic Injuries in Children With Nonaccidental Trauma: Demographics and Incidence From the 2000 Kids' Inpatient Database," *Journal of Pediatric Orthopaedics*, vol. 27, p. 421, June 2007.

[110] H. J. Quiroz, J. J. Yoo, L. C. Casey, B. A. Willobee, A. R. Ferrantella, C. M. Thorson, E. A. Perez, and J. E. Sola, "Can we increase detection? A nationwide analysis of age-related fractures in child abuse," *Journal of Pediatric Surgery*, vol. 56, pp. 153–158, Jan. 2021.

[111] E. G. Flaherty, J. M. Perez-Rossello, M. A. Levine, W. L. Hennrikus, and the AMERICAN ACADEMY OF PEDIATRICS COMMITTEE ON CHILD ABUSE AND NEGLECT, SECTION ON RADIOLOGY, SECTION ON ENDOCRINOLOGY, SECTION ON ORTHOPAEDICS, the SOCIETY FOR PEDIATRIC RADIOLOGY, C. W. Christian, J. E. Crawford-Jakubiak, E. G. Flaherty, J. M. Leventhal, J. L. Lukefahr, R. D. Sege, C. I. Cassady, D. I. Bulas, J. A. Cassese, A. R. Mehollin-Ray, M.-G. Mercado-Deane, S. S. Milla, I. N. Sills, C. A. Bloch, S. J. Casella, J. M. Lee, J. L. Lynch, K. A. Wintergerst, R. M. Schwend, J. E. Gordon, N. Y. Otsuka, E. M. Raney, B. A. Shaw, B. G. Smith, L. Wells, and P. W. Esposito, "Evaluating Children With Fractures for Child Physical Abuse," *Pediatrics*, vol. 133, pp. e477–e489, Feb. 2014.

[112] P. Jayakumar, M. Barry, and M. Ramachandran, "Orthopaedic aspects of paediatric non-accidental injury," *The Journal of Bone and Joint Surgery. British volume*, vol. 92-B, pp. 189–195, Feb. 2010.

[113] H. A. Aboughalia, A.-V. Ngo, S. J. Menashe, H. H. Kim, and R. S. Iyer, "Pediatric rib pathologies: Clinicoimaging scenarios and approach to diagnosis," *Pediatric Radiology*, vol. 51, pp. 1783–1797, Sept. 2021.

[114] Y. Henon, "pytorch-retinanet." `https://github.com/yhenon/pytorch-retinanet`, 2021.

[115] M. Heidari, S. Mirniaharikandehei, A. Z. Khuzani, G. Danala, Y. Qiu, and B. Zheng, "Improving the performance of CNN to predict the likelihood of COVID-19 using chest

X-ray images with preprocessing algorithms," *International Journal of Medical Informatics*, vol. 144, p. 104284, Dec. 2020.

[116] M. A. Ganaie, M. Hu, M. Tanveer*, and P. N. Suganthan*, "Ensemble deep learning: A review," *arXiv:2104.02395 [cs]*, Apr. 2021.

[117] S. Qummar, F. G. Khan, S. Shah, A. Khan, S. Shamshirband, Z. U. Rehman, I. Ahmed Khan, and W. Jadoon, "A Deep Learning Ensemble Approach for Diabetic Retinopathy Detection," *IEEE Access*, vol. 7, pp. 150530–150539, 2019.

[118] V. Gavrishchaka, Z. Yang, R. Miao, and O. Senyukova, "Advantages of Hybrid Deep Learning Frameworks in Applications with Limited Data," *International Journal of Machine Learning and Computing*, vol. 8, no. 6, p. 11, 2018.

[119] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles," *arXiv:1612.01474 [cs, stat]*, Nov. 2017.

[120] J. Burkow, G. Holste, J. Otjen, F. Perez, J. Junewick, and A. Alessio, "Avalanche decision schemes to improve pediatric rib fracture detection," in *Medical Imaging 2022: Computer-Aided Diagnosis*, vol. 12033, pp. 611–618, SPIE, Apr. 2022.

[121] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative Adversarial Networks," *arXiv:1406.2661 [cs, stat]*, June 2014.

[122] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-Image Translation with Conditional Adversarial Networks," Nov. 2018.

[123] T. Karras, S. Laine, and T. Aila, "A Style-Based Generator Architecture for Generative Adversarial Networks," *arXiv:1812.04948 [cs, stat]*, Mar. 2019.

[124] Q.-Q. Zhou, J. Wang, W. Tang, Z.-C. Hu, Z.-Y. Xia, X.-S. Li, R. Zhang, X. Yin, B. Zhang, and H. Zhang, "Automatic Detection and Classification of Rib Fractures on Thoracic CT Using Convolutional Neural Network: Accuracy and Feasibility," *Korean Journal of Radiology*, vol. 21, pp. 869–879, July 2020.

[125] V. Sorin, Y. Barash, E. Konen, and E. Klang, "Creating Artificial Images for Radiology Applications Using Generative Adversarial Networks (GANs) – A Systematic Review," *Academic Radiology*, vol. 27, pp. 1175–1185, Aug. 2020.

[126] E. Tu, *MACHINE LEARNED DATA AUGMENTATION TECHNIQUES FOR IMPROVING PATHOLOGY OBJECT DETECTION*. PhD thesis, Michigan State University, 2023.

[127] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium," Jan. 2018.

[128] R. Bommasani, D. A. Hudson, E. Adeli, R. Altman, S. Arora, S. von Arx, M. S. Bernstein, J. Bohg, A. Bosselut, E. Brunskill, E. Brynjolfsson, S. Buch, D. Card, R. Castellon, N. Chatterji, A. Chen, K. Creel, J. Q. Davis, D. Demszky, C. Donahue, M. Doumbouya,

E. Durmus, S. Ermon, J. Etchemendy, K. Ethayarajh, L. Fei-Fei, C. Finn, T. Gale, L. Gillespie, K. Goel, N. Goodman, S. Grossman, N. Guha, T. Hashimoto, P. Henderson, J. Hewitt, D. E. Ho, J. Hong, K. Hsu, J. Huang, T. Icard, S. Jain, D. Jurafsky, P. Kalluri, S. Karamcheti, G. Keeling, F. Khani, O. Khattab, P. W. Koh, M. Krass, R. Krishna, R. Kuditipudi, A. Kumar, F. Ladhak, M. Lee, T. Lee, J. Leskovec, I. Levent, X. L. Li, X. Li, T. Ma, A. Malik, C. D. Manning, S. Mirchandani, E. Mitchell, Z. Munyikwa, S. Nair, A. Narayan, D. Narayanan, B. Newman, A. Nie, J. C. Niebles, H. Nilforoshan, J. Nyarko, G. Ogut, L. Orr, I. Papadimitriou, J. S. Park, C. Piech, E. Portelance, C. Potts, A. Raghunathan, R. Reich, H. Ren, F. Rong, Y. Roohani, C. Ruiz, J. Ryan, C. Ré, D. Sadigh, S. Sagawa, K. Santhanam, A. Shih, K. Srinivasan, A. Tamkin, R. Taori, A. W. Thomas, F. Tramèr, R. E. Wang, W. Wang, B. Wu, J. Wu, Y. Wu, S. M. Xie, M. Yasunaga, J. You, M. Zaharia, M. Zhang, T. Zhang, X. Zhang, Y. Zhang, L. Zheng, K. Zhou, and P. Liang, "On the Opportunities and Risks of Foundation Models," July 2022.

[129] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," May 2019.

[130] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language Models are Few-Shot Learners," July 2020.

[131] A. Ramesh, M. Pavlov, G. Goh, S. Gray, C. Voss, A. Radford, M. Chen, and I. Sutskever, "Zero-Shot Text-to-Image Generation," Feb. 2021.

[132] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-Resolution Image Synthesis with Latent Diffusion Models," Apr. 2022.

[133] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. Girshick, "Segment Anything," Apr. 2023.

APPENDIX

SUPPLEMENTAL TABLES

The following tables are the full versions of Table 4.3 and Table 4.4 discussed in Section 4.7, containing all variations of ensembles and image processing methods.

Table A.1 Full set of results from Table 4.3. Precision, recall, and F2 values across all models were measured using an IOU threshold of 0.30 with ground truth. mAP values were calculated using IOU values from 0.25 to 0.75. Bolded values represent the top two scores for each ensemble size and metric. Superscripts *a*, *b*, and *c* represent the type of input processing to train the models as described in Section 4.4. Ensembles with * have hybrid inputs, i.e., each ensemble member was trained on a different input processing method.

Models	Precision	Recall	F2	Max F2	mAP
1x-R^{a}	**0.892** ± 0.015	0.430 ± 0.014	0.480 ± 0.014	0.630 ± 0.011	0.480 ± 0.008
1x-R^{b}	0.852 ± 0.015	0.344 ± 0.027	0.390 ± 0.028	0.570 ± 0.015	0.395 ± 0.010
1x-R^{c}	0.883 ± 0.013	0.425 ± 0.027	0.474 ± 0.027	**0.635** ± 0.012	0.473 ± 0.009
1x-Y^{a}	**0.897** ± 0.032	**0.434** ± 0.040	**0.484** ± 0.037	0.590 ± 0.062	**0.563** ± 0.009
1x-Y^{b}	0.872 ± 0.060	0.320 ± 0.049	0.365 ± 0.050	0.539 ± 0.077	0.462 ± 0.035
1x-Y^{c}	0.880 ± 0.024	**0.464** ± 0.043	**0.512** ± 0.041	**0.644** ± 0.046	**0.555** ± 0.016
2x-R^{a}	**0.859** ± 0.013	0.488 ± 0.014	0.534 ± 0.013	0.651 ± 0.011	0.495 ± 0.005
2x-R^{b}	0.803 ± 0.014	0.408 ± 0.026	0.453 ± 0.025	0.587 ± 0.011	0.412 ± 0.009
2x-R^{c}	0.847 ± 0.012	0.487 ± 0.025	0.532 ± 0.023	0.652 ± 0.010	0.488 ± 0.007
2x-Y^{a}	**0.862** ± 0.032	**0.523** ± 0.027	**0.567** ± 0.024	**0.672** ± 0.022	**0.571** ± 0.015
2x-Y^{b}	0.820 ± 0.062	0.414 ± 0.045	0.459 ± 0.042	0.610 ± 0.020	0.468 ± 0.025
2x-Y^{c}	0.835 ± 0.017	**0.558** ± 0.024	**0.597** ± 0.021	**0.690** ± 0.012	**0.562** ± 0.012
3x-R^{a}	**0.839** ± 0.012	0.516 ± 0.012	0.559 ± 0.011	0.658 ± 0.007	0.500 ± 0.005
3x-R^{b}	0.769 ± 0.013	0.452 ± 0.015	0.492 ± 0.014	0.598 ± 0.006	0.415 ± 0.008
3x-R^{c}	**0.817** ± 0.015	0.527 ± 0.015	0.567 ± 0.014	0.661 ± 0.006	0.494 ± 0.005
3x-R*	0.812 ± 0.011	0.523 ± 0.014	0.563 ± 0.013	0.649 ± 0.006	0.493 ± 0.008
3x-Y^{a}	0.843 ± 0.029	0.548 ± 0.024	**0.589** ± 0.020	**0.688** ± 0.016	**0.572** ± 0.016
3x-Y^{b}	0.803 ± 0.050	0.451 ± 0.035	0.494 ± 0.030	0.623 ± 0.010	0.467 ± 0.017
3x-Y^{c}	0.814 ± 0.017	**0.599** ± 0.021	**0.633** ± 0.018	**0.694** ± 0.008	**0.559** ± 0.006
3x-Y*	**0.817** ± 0.037	**0.561** ± 0.033	0.598 ± 0.027	0.685 ± 0.011	0.543 ± 0.014
6x-R^{a}	**0.798** ± 0.008	0.558 ± 0.007	0.593 ± 0.006	0.659 ± 0.006	0.507 ± 0.005
6x-R^{b}	0.710 ± 0.007	0.501 ± 0.008	0.532 ± 0.008	0.603 ± 0.004	0.423 ± 0.007
6x-R^{c}	0.762 ± 0.008	0.571 ± 0.009	0.601 ± 0.008	0.666 ± 0.004	0.499 ± 0.004
6x-Y^{a}	**0.785** ± 0.017	0.609 ± 0.015	0.638 ± 0.012	**0.705** ± 0.003	**0.568** ± 0.008
6x-Y^{b}	0.725 ± 0.036	0.533 ± 0.021	0.562 ± 0.016	0.635 ± 0.007	0.470 ± 0.172
6x-Y^{c}	0.756 ± 0.010	**0.653** ± 0.008	**0.671** ± 0.007	**0.699** ± 0.006	**0.555** ± 0.004
3x-R*+3x-Y*	0.752 ± 0.019	**0.625** ± 0.021	**0.647** ± 0.017	0.686 ± 0.008	0.522 ± 0.004

Table A.2 Full set of results from Table 4.4. Precision, recall, and F2 values across all models were measured using an IOU threshold of 0.30 with ground truth. mAP values were calculated using IOU values from 0.25 to 0.75. Bolded values represent the top two scores for each ensemble size and metric. Superscripts a, b, and c represent the type of input processing to train the models as described in Section 4.4. Ensembles with * have hybrid inputs, i.e., each ensemble member was trained on a different input processing method.

Models	Avalanche	Precision	Recall	F2	Max F2
1x-R$^{\mathbf{a}}$	Conservative	0.530 ± 0.023	**0.730** ± 0.015	0.679 ± 0.010	0.897 ± 0.054
1x-R$^{\mathbf{b}}$	Conservative	0.469 ± 0.030	0.695 ± 0.016	0.634 ± 0.018	**0.910** ± 0.067
1x-R$^{\mathbf{c}}$	Conservative	0.529 ± 0.026	**0.745** ± 0.012	**0.688** ± 0.012	**0.908** ± 0.046
1x-Y$^{\mathbf{a}}$	Posterior	**0.759** ± 0.164	0.647 ± 0.101	0.652 ± 0.051	0.777 ± 0.045
1x-Y$^{\mathbf{b}}$	Posterior	0.581 ± 0.206	0.673 ± 0.141	0.620 ± 0.074	0.776 ± 0.090
1x-Y$^{\mathbf{c}}$	Posterior	**0.645** ± 0.130	0.724 ± 0.085	**0.695** ± 0.041	0.812 ± 0.056
2x-R$^{\mathbf{a}}$	Conservative	0.438 ± 0.011	**0.768** ± 0.012	0.667 ± 0.009	**0.909** ± 0.033
2x-R$^{\mathbf{b}}$	Conservative	0.376 ± 0.017	0.743 ± 0.011	0.621 ± 0.013	0.883 ± 0.040
2x-R$^{\mathbf{c}}$	Conservative	0.433 ± 0.021	**0.779** ± 0.010	0.671 ± 0.012	**0.909** ± **0.031**
2x-Y$^{\mathbf{a}}$	Posterior	**0.634** ± 0.160	0.736 ± 0.083	**0.697** ± 0.028	0.786 ± 0.034
2x-Y$^{\mathbf{b}}$	Conservative	0.582 ± 0.103	0.676 ± 0.060	0.648 ± 0.033	0.797 ± 0.081
2x-Y$^{\mathbf{c}}$	$\gamma = 0.20$	**0.642** ± 0.065	0.746 ± 0.038	**0.720** ± 0.018	0.807 ± 0.044
3x-R$^{\mathbf{a}}$	Conservative	0.394 ± 0.009	0.791 ± 0.010	0.658 ± 0.007	0.895 ± 0.041
3x-R$^{\mathbf{b}}$	Conservative	0.331 ± 0.010	0.767 ± 0.012	0.607 ± 0.010	0.886 ± 0.034
3x-R$^{\mathbf{c}}$	Conservative	0.385 ± 0.012	**0.797** ± 0.008	0.656 ± 0.008	**0.916** ± 0.028
3x-R*	Conservative	0.340 ± 0.013	**0.809** ± 0.009	0.634 ± 0.011	**0.898** ± 0.026
3x-Y$^{\mathbf{a}}$	Posterior	0.558 ± 0.119	0.776 ± 0.056	0.710 ± 0.019	0.786 ± 0.029
3x-Y$^{\mathbf{b}}$	Conservative	0.523 ± 0.080	0.728 ± 0.043	0.670 ± 0.016	0.823 ± 0.089
3x-Y$^{\mathbf{c}}$	$\gamma = 0.20$	**0.573** ± 0.058	0.780 ± 0.030	**0.725** ± 0.012	0.812 ± 0.036
3x-Y*	Conservative	**0.590** ± 0.069	0.758 ± 0.028	**0.714** ± 0.014	0.802 ± 0.040
6x-R$^{\mathbf{a}}$	Conservative	0.320 ± 0.004	0.814 ± 0.006	0.622 ± 0.006	**0.867** ± 0.026
6x-R$^{\mathbf{b}}$	Conservative	0.258 ± 0.005	0.802 ± 0.007	0.564 ± 0.007	0.850 ± 0.023
6x-R$^{\mathbf{c}}$	Conservative	0.311 ± 0.005	**0.816** ± 0.005	0.616 ± 0.005	**0.912** ± 0.025
6x-Y$^{\mathbf{a}}$	$\gamma = 0.20$	**0.536** ± 0.044	0.795 ± 0.022	**0.723** ± 0.010	0.797 ± 0.016
6x-Y$^{\mathbf{b}}$	Conservative	0.405 ± 0.028	0.791 ± 0.010	0.664 ± 0.013	0.851 ± 0.030
6x-Y$^{\mathbf{c}}$	Conservative	**0.508** ± 0.020	0.797 ± 0.010	**0.715** ± 0.005	0.817 ± 0.026
3x-R*+3x-Y*	Conservative	0.314 ± 0.015	**0.841** ± 0.014	0.630 ± 0.011	0.833 ± 0.052