Imperial College London

MSc Individual Project

Imperial College London

Department of Computing

Bias Analysis in Chest X-Ray Disease Detection Models

Author:
Soren Antebi

Supervisor:
Dr. Ben Glocker

Second Marker:
Dr. Wenjia Bai

September 18, 2023

Abstract

Pre-trained deep neural networks or "foundation models" promise robust feature extractors for disease detection in transfer learning tasks. They offer an attractive method for model development in the realm of medical imaging, since medical datasets are often (1) limited and (2) imbalanced. However, potential biases might be encoded during pre-training, affecting downstream performance and causing spurious correlations to be formed between unrelated tasks. This is a concern, as deep classifiers have been shown to detect protected patient characteristics, such as biological sex or racial identity, from medical data and might use these features to make decisions when classifying disease. Medical datasets also display annotation bias or label noise, which can arise due to a variety of socio-ecological factors.

We investigate pre-trained chest X-ray classifiers in the TorchXRayVision library by employing subgroup performance analysis and feature exploration frameworks. Features are inspected using the dimensionality reduction techniques, principal component analysis and t-distributed stochastic neighbor embedding. In comparing frozen and unfrozen backbones for model development, we found that unfrozen XRV models do not show compelling evidence that racial or sexual information is explicitly used to make disease predictions. However, some disparities between subgroups could be observed in higher PCA modes, which came with a corresponding decrease in performance.

Subsequently, we test methods for bias mitigation in the presence of label noise by utilizing Domain Adversarial Neural Networks (DANNs). Bias is introduced artificially into targeted subgroups via systematic or random noise, in order to generate a purposefully skewed model. We found that training adversarially is able to recover the unbiased state of most models with a certain performance trade-off. Different mitigation techniques are implemented involving either negated gradients in the backbone or calculating a secondary (confusion) loss function to generate invariant features.

Even though DANNs can correct for bias in a simulated environment, real-world relationships between subgroups are complex and it is difficult to determine where exactly certain biases stem from. Developing unbiased algorithms is only the first step towards achieving fairness in the field of medical artificial intelligence.

Acknowledgements

First, I would like to thank my supervisor Dr. Ben Glocker who has not only provided me with invaluable guidance throughout my project, but has also introduced me to a wonderful community. Secondly, I would like to thank my second marker Dr. Wenjia Bai, who provided essential constructive feedback on my background report in early stages of the project.

A special thanks goes out to Dr. Josiah Wang for his support in difficult times.

Furthermore, I would like to express my gratitude to my friends that surround me with energy and cherished memories. And from the bottom of my heart, I would like to thank my family who have made all this possible.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

As the use of Deep Learning (DL) in healthcare becomes increasingly prevalent, it is important to scrutinize the fairness of medical data, and its impact on model development. From providing simple annotations to computer-aided diagnosis (CAD), DL models are changing the way medical professionals detect and diagnose disease [1]. They have a profound effect on a patient's quality of care and healthcare outcomes [2], making it crucial to generate unbiased algorithms for diagnosis. However, medical datasets face two central issues which interfere with fair model development:

1. they are *imbalanced* and
2. they are *limited*.

Medical datasets often suffer from subgroup imbalance and exhibit disparities in protected-identity characteristics. This includes groups such as racial identity, biological sex, sexual orientation, and age [3, 4]. Consequentially, training models on these datasets can lead to biased predictions and impact performance in underrepresented groups. For instance, deep classifiers trained on the gender imbalanced NIH Chest-XRay14 dataset, consistently show a decreased performance for female patients[1, 5]. Since these differences stem from socio-ecological inequities such as societal prejudice and socioeconomic disparities, applying biased models in the medical industry could amplify already existing healthcare disparities [6, 7].

To complicate matters further, medical data is not readily available; attaining data is a time-consuming and costly process [8, 9]. This is an obstacle for traditional DL model development, which requires substantial amounts of data to train effectively. Without sufficient data, models generalize poorly and become unreliable when data distribution shifts occur [10].

This project explores pre-trained models or "foundation models", which have emerged as a promising solution for both these issues. In particular, we evaluate potential biases in pre-trained chest X-ray disease detection models, as well as methods for bias mitigation involving Domain Adversarial Neural Networks (DANNs).

Pre-trained Models

Foundation models are pre-trained neural networks that promise robust backbones for feature extraction. Conducted on readily-available, diverse datasets, pre-training improves generalization and increases resilience to noise and class imbalance [11]. Since a broad range of features are learned during pre-training, the model can then be fine-tuned without the need for vast amounts of domain-specific (i.e., medical) data. Therefore, foundation models are capable of generating state-of-the-art classifiers, whilst being data-efficient, data-accessible, and robust [10, 12, 8].

However, several questions arise: could foundation models potentially inherit existing biases present in datasets they were pre-trained on? If so, how does this affect downstream tasks? Recent analysis of a published chest X-ray foundation model (Sellergren et al.[12]) has shown that certain protected characteristics might be encoded throughout the pre-training process, subsequently resulting in a decrease in performance [10].

In order to further investigate this, we inspect a set of pre-trained chest X-ray models from the TorchXRayVision (XRV) library.

Label Noise and Bias Mitigation

Decision-making policies in healthcare can be subject to prejudice, causing clinicians to form spurious connections between a patient's protected characteristics and disease outcomes [3]. As a result, historically underserved groups have a higher chance for mis- or underdiagnosis [6, 4]. Recent studies have shown that DL models can rely on these spurious features to form predictions and potentially use protected characteristics as a shortcut to predict health outcomes [7, 3, 13, 14, 15]. Therefore, the validity of collected data for subgroups must be scrutinized.

Mimicking this behavior, we can train a model where artificial label noise has been introduced into the dataset. By generating a classifier known to be biased, we can then test bias mitigation techniques and determine how well they recover the non-biased state of the model. To achieve this, we utilize a DANN to mitigate subgroup disparities in the presence of different types of label noise.

1.1 Contributions

In this thesis we make many key contributions. In particular:

- **XRV Bias Analysis**

  We analyze bias in foundation models from the XRV library. The models are loaded with pre-trained weights and then fine-tuned on the CheXpert dataset. These are compared to pre-trained ImageNet models also fine-tuned on CheXpert. We show that freezing the backbone during last-layer retraining has a significant adverse effect on disease detection performance.

- **Multitask Learning**

  We implement a multitask head to classify disease, sex and race in a three-part optimization process. The different classification heads use a shared feature representation to facilitate inductive learning.

- **Adversarial Training**

  We develop a DANN using negated gradients/confusion loss operations to generate domain-invariant features. We found that negated gradients target specified parts of the classification head whereas confusion loss impacts all parts.

- **Label Corruption**

  We introduce systematic and random label noise into a dataset, intentionally generating a biased model. We demonstrate that adversarial training (via a DANN) is able to restore performance back to its original state in most cases.

- **Configurability**

  We improve the configurability of model development by introducing command line arguments that can alter hyperparameters before training. This enables us to also choose the type of pre-trained model we wish to implement.

1.2 Code

The code base utilized in this work is built upon contributions and previous research conducted by Dr. Ben Glocker et al. Their corresponding GitHub repository can be found at https://github.com/biomedia-mira/chexploration.

Chapter 2

Background and Context

2.1 Deep Learning

Deep learning is a subset of Machine Learning (ML) that provides efficient methods for feature extraction and data processing. In contrast to conventional ML techniques, DL models are able to automatically map high-dimensional feature representations from raw data [16]. They consist of highly-layered artificial neural networks (ANNs), that utilize backpropagation algorithms to update parameters during training. This sequence of network layers can be referred to as the neural network backbone.

By applying non-linear transformations across multiple layers, the input data becomes increasingly abstract as it progresses through the backbone [7]. Therefore, when the data reaches the penultimate layer, the representations are complex and must be processed by the final classification layer to generate an output prediction [7]. This involves computing a weighted aggregation of the features in the penultimate layer. The basic components and processes are summarized in Fig 2.1 .

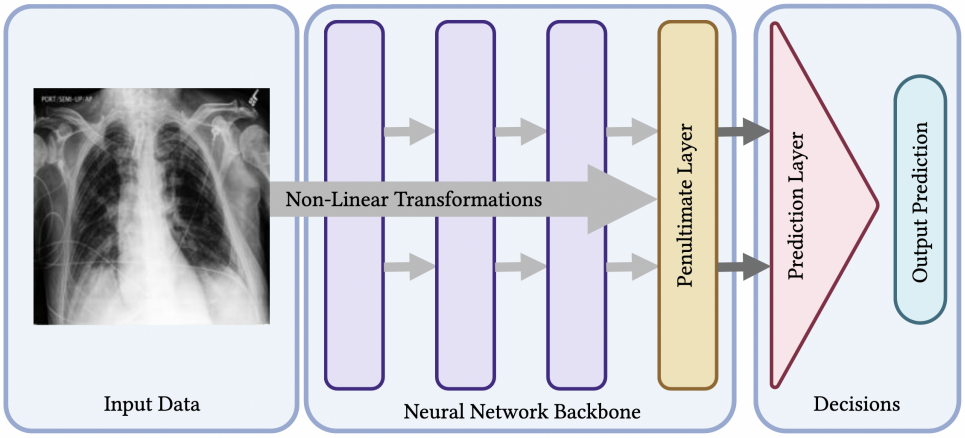


Figure 2.1: **General structure of a deep neural network.** [7]

Due to their state-of-the-art performance and groundbreaking advancements in ML, the use of DL models has become widespread in many domains of science, medicine, business and government [16]. In medical image classification tasks, DL models are not only able to perform comparably to clinical experts, but are even capable of recognizing patterns not apparent to the human eye [17]. For instance, using retinal fundus images, cardiovascular risk factors including age, smoking status, blood pressure and major adverse cardiovascular events could be predicted [7, 18].

Recent studies have also shown that deep classifiers are capable of predicting protected characteristics, such as racial identity, using radiographical data [7, 13]. This raises ethical concerns regarding the use of DL models in medicine. Deep neural networks are able to detect these intricate structures, due to the non-linear transformations and data abstractions that occur in the model backbone.

2.1.1 Convolutional Neural Networks

In deep learning, convolutional neural networks (CNNs) are a class of ANN that offer scalable approaches for common computer vision tasks, such as image classification and object recognition. Their applicability in diverse areas of medicine has inspired a surge of CNN research concerning disease detection, image reconstruction, segmentation and natural language processing [9].

CNNs are designed to process multi-array data and can be applied to a variety of data modalities, from 1D signals and sequences to 3D videos and volumetric images [16]. The typical structure of a CNN consists of three main building blocks: convolutional layers, pooling layers and fully connected layers. The first few stages of a CNN use repeated stacks of convolutional and pooling layers to extract features from input data. Subsequently, the last stage involves fully-connected layers that map these features into a final output [9, 16].

Convolutional Layers

Convolutional layers in a CNN perform a combination of linear and non-linear operations for feature extraction. The convolution operation is a specialized type of linear operation that involves convolving a weighted kernel over an input tensor [9]. For each step taken by the kernel, a weighted sum is calculated and mapped to a single field in a feature map. Therefore, CNNs exhibit local connectivity, as each neuron in the generated output sees a small local region, also known as receptive field, from the previous layer.

A convolutional layer can produce multiple feature maps. This is determined by the number of kernels used, which is an arbitrary hyperparameter [9]. After the convolution operation is performed, a non-linear activation function is applied element-wise to the output, allowing complex patterns to be captured. It is important to note that the kernel weights remain the same across the entire input (weight sharing), which is a key feature of convolution operations. This allows for distinct local motifs and patterns to be detected in different parts of the image [16].

Pooling Layers

Pooling layers usually succeed convolutional layers and perform downsampling functions that decrease the dimensions of the input. Specifically, they reduce the height and width, but keep the number of feature maps (depth) unchanged [9]. This is achieved by sliding a kernel across an input tensor and carrying out certain pooling operations, such as max-pooling or average-pooling. Max-pooling takes the maximum value from a local patch of units, whereas average-pooling computes the average value in that window [9, 16]. This allows a larger region of the image to be visualised by the neurons in the next later, while retaining important semantic information.

Additionally, pooling facilitates invariance to spatial distortions and improves computational complexity by reducing the number of learnable parameters [9]. Another commonly used pooling operation is the global average pool, which is an extreme form of downsampling that converts a feature map to a 1×1 array [9].

2.1.2 DenseNet - Densely Connected Convolutional Networks

The ImageNet models we train throughout this project, as well as the pre-trained models from the XRV library utilize a specialized type of CNN: Densely Connected Convolutional Networks (DenseNet). Their state-of-the-art performance and computational efficiency makes them a particularly attractive option for image classification tasks. DenseNets are characterized by their "dense" connectivity patterns that aim to maximize the amount of information shared between each layer of the network [19]. A typical DenseNet structure is illustrated in Fig 2.2.

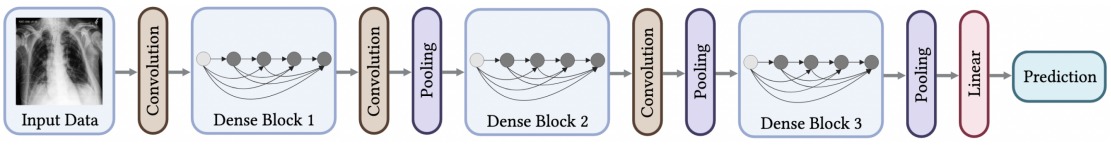


Figure 2.2: **General structure of a DenseNet.** [19]

Instead of having single connections between a layer and its subsequent layer like with traditional CNNs, DenseNets connect all layers directly with each other in a feed-forward manner. In

other words, each layer receives input from all layers that precede it and provides its output feature map representations to all layers that follow [19].

Layer Interconnectivity

As one progresses through the network, feature maps are concatenated. Therefore, in a network with L layers, there are $\frac{L(L+1)}{2}$ connections. This concept of feature reuse allows the model to ignore redundant features, improving information flow and reducing the amount of required parameters [19]. Furthermore, this mitigates the vanishing gradient problem found in other deep CNNs [19]. These groups of interconnected layers are called dense blocks.

Dense Blocks and Transitional Layers

DenseNets are usually split into multiple dense blocks, which allow for essential downsampling in transitional layers [19]. These transition layers involve a batch normalization (BN) step, followed by a 1×1 convolution and a 2×2 average pooling layer. Furthermore, they can be used to reduce the number of feature maps m by using a compression factor θ to generate θm feature maps, typically $\theta = 0.5$.

Within the dense block itself, a sequence of BN → rectified linear unit (ReLU) → 3×3 conv is performed at each layer. Before each BN-ReLU-3×3, a BN-ReLU-1×1 convolution is executed, acting as a bottleneck layer and reducing the number of input feature maps.

Features and Output

The number of features produced per layer is determined by the growth rate k. Due to the interconnective relationship of DenseNet layers, a large growth rate is not needed to obtain state-of-the-art results. When the penultimate layer is reached, a global average pool converts all feature maps into a 1×1×1024 dimensional output, which is put through a 1000D fully-connected layer and a softmax activation function to generate a final output.

In the scope of this thesis, we utilize a DenseNet-121 architecture (Table A.4), which corresponds to the 120 convolutional layers and 1 fully-connected layer present in the network. Other DenseNet architectures such as DenseNet-169 and DenseNet-201 display a similar structure with minor differences in the number of convolutional layers per dense block.

Layers	Output Size	DenseNet-121
Convolution	112 × 112	7 × 7 conv, stride 2
Pooling	56 × 56	3 × 3 max pool, stride 2
Dense Block (1)	56 × 56	$\begin{bmatrix} 1 \times 1 & \text{conv} \\ 3 \times 3 & \text{conv} \end{bmatrix} \times 6$
Transition Layer (1)	56 × 56	1 × 1 conv
	28 × 28	2 × 2 average pool, stride 2
Dense Block (2)	28 × 28	$\begin{bmatrix} 1 \times 1 & \text{conv} \\ 3 \times 3 & \text{conv} \end{bmatrix} \times 12$
Transition Layer (2)	28 × 28	1 × 1 conv
	14 × 14	2 × 2 average pool, stride 2
Dense Block (3)	14 × 14	$\begin{bmatrix} 1 \times 1 & \text{conv} \\ 3 \times 3 & \text{conv} \end{bmatrix} \times 24$
Transition Layer (3)	14 × 14	1 × 1 conv
	7 × 7	2 × 2 average pool, stride 2
Dense Block (4)	7 × 7	$\begin{bmatrix} 1 \times 1 & \text{conv} \\ 3 \times 3 & \text{conv} \end{bmatrix} \times 16$
Classification Layer	1 × 1	7 × 7 global average pool
		1000D fully-connected, softmax

Table 2.1: **DenseNet-121 architecture composition.** [19]

2.2 Transfer Learning

2.2.1 A General Overview

Due to the limitations of medical imaging data, one technique that can be employed to improve the performance and efficiency of classification models is transfer learning. Generally, (inductive) transfer learning involves using knowledge from a source domain as the foundation for learning in a target domain [20]. Suppose we have a domain $\mathcal{D}$, defined by a feature space $\mathcal{X}$ and a probability distribution $P(X)$. This has a corresponding task $\mathcal{T}$, defined by a label space $\mathcal{Y}$ and a predictive function $f(\cdot)$ [21]. If we have a source and target domain with their corresponding tasks, where $\mathcal{D}_S \neq \mathcal{D}_T$ (or $\mathcal{D}_S \sim \mathcal{D}_T$) and $\mathcal{T}_S \neq \mathcal{T}_T$ [21], we can express these as:

$$\mathcal{D}_S = \{\mathcal{X}_S, P(X_S)\} \quad \text{and} \quad \mathcal{T}_S = \{\mathcal{Y}_S, f_S(\cdot)\} \tag{2.1}$$

$$\mathcal{D}_T = \{\mathcal{X}_T, P(X_T)\} \quad \text{and} \quad \mathcal{T}_T = \{\mathcal{Y}_T, f_T(\cdot)\} \tag{2.2}$$

The overall aim is to improve the predictive function $f_T(\cdot)$ by using information from $\mathcal{D}_S$ and $\mathcal{T}_S$ [21]. In practice, this typically involves using the general features from a pre-trained backbone and applying them to a given task of interest [9].

As mentioned previously, pre-training is conducted on large-scale, heterogeneous datasets, such as ImageNet. Subsequently, the generated feature vectors can be used as input into a new classification layer and then retrained on a smaller task-specific dataset. This process is data-efficient and therefore transfer learning offers a promising solution for model development in the absence of abundant medical data. Common transfer learning strategies that can be adopted for training are:

- **Fixed feature extraction method**

  This involves freezing the convolutional backbone and only updating the parameters in the classification layer (Fig 2.3). This method can be used to perform supervised information tests. For example, by using a disease detection convolutional base and then training the added classification layer for sex classification or race classification [7].

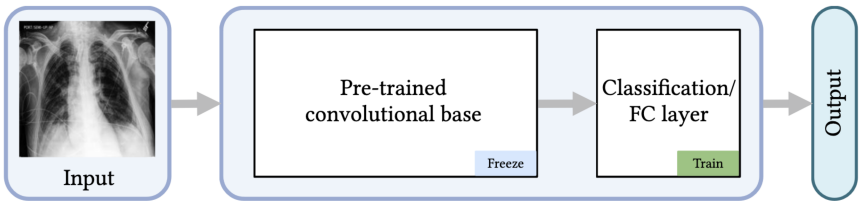


Figure 2.3: **Transfer learning with a fixed feature extractor.** [9]

- **Fine-tuning method**

  This involves unfreezing the layers in the backbone and updating parameters in both the pre-trained feature extractor and the classification head (Fig 2.4). This method is more commonly used in medical imaging tasks [9]. It is possible to choose whether selected layers, or all layers in the convolutional base are unfrozen and fine-tuned.

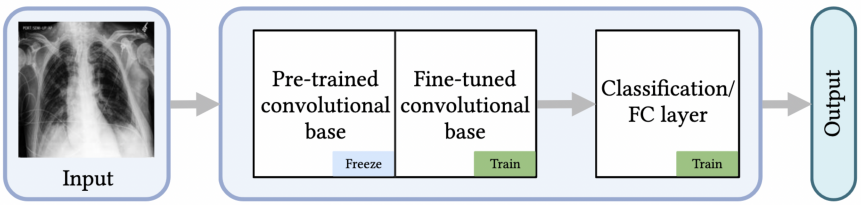


Figure 2.4: **Transfer learning using the fine-tuning method.** [9]

2.2.2 Multitask learning

Multitask learning is a transfer learning approach that involves training multiple classification heads in parallel. Even though each classification head is trained for a different task, they have access to the same backbone and can use shared representations to facilitate learning in closely related tasks [7, 22]. For instance, if two tasks $\mathcal{T}_1$ and $\mathcal{T}_2$ exist, where $\mathcal{T}_1$ is able to detect a certain feature $\mathcal{F}$ more easily, $\mathcal{T}_2$ can use the shared representations from $\mathcal{T}_1$ for improved learning of $\mathcal{F}$ [22]. Therefore, multitask learning is inductive and improves generalization.

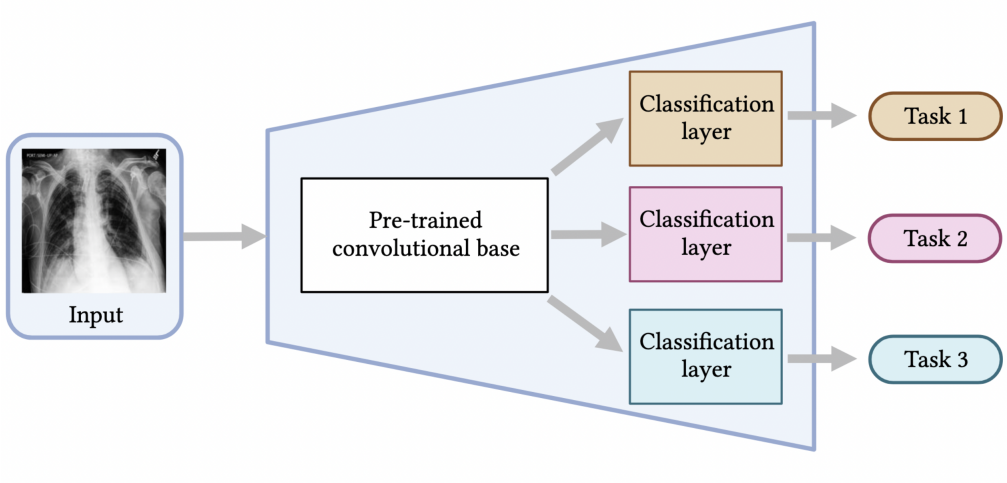


Figure 2.5: **General structure of a multitask model with 3 classification heads.**

In this project, we employ a multitask model trained to detect disease, racial identity and biological sex in order to inspect the relationships between these tasks. Furthermore, by adjusting the functionality of the multitask model, but keeping the general structure intact, this can be adapted to perform adversarial training using a DANN.

2.3 Foundation Models

The first set of experiments in this project focuses on foundation models. Foundation models are pre-trained models that can be used as a convolutional base for transfer learning. As a result, they can be adapted for more domain-specific tasks, without the need for large quantities of data. This makes them an attractive method for developing medical imaging models. However, while foundation models do function as robust feature extractors, biases could be encoded during pre-training stages that affect downstream predictions.

2.3.1 Bias Analysis

In a study conducted by Glocker et al. [10], the potential biases and disease detection performance of a recently published CXR foundation model are investigated ([12]). The CXR foundation model was pre-trained using a two-step process, involving an initial pre-training on natural images followed by supervised contrastive learning on a large chest X-ray dataset.

Feature embeddings were extracted from the model and used as inputs to three types of classification heads – one containing a single linear layer (denoted as CXR-Linear), and two containing three (CXR-MLP-3) and five (CXR-MLP-5) hidden layers. These were compared to a baseline model with a single layer classification head and a DenseNet-121 backbone loaded with ImageNet weights. All models were then trained and validated on the comprehensive chest X-ray dataset, CheXpert.

Model Inspection

Firstly, to examine whether generated feature spaces have persisting biases from pre-training, principal component analysis (PCA) and t-distributed stochastic neighbor embedding (t-SNE) is incorporated into their bias analysis framework. They observe larger variations across subgroups

for the CXR foundation model compared to the baseline, suggesting that certain protected characteristics are encoded more strongly. In particular, they detect significant differences between the Asian and Black patients, as well as significant differences between Male and Female subgroups.

Subgroup Performance

Following this, a disease detection evaluation shows consistent degradation in performance for all models trained on top of the CXR foundation model. This indicates that the encoded biases from pretraining have an adverse effect on downstream performance, highlighting potential risks for the use of foundation models in medicine.

However, one drawback of this study is that the inspected foundation model is not publicly available, meaning that it was not possible to update its backbone parameters during training. Therefore, they essentially adopt a fixed feature extraction method for training. This is more likely to propagate biases through the network, since fine-tuning to override biases in the backbone cannot be incorporated. Furthermore, only one particular type of foundation model was investigated, making it difficult to draw any general conclusions about how bias is encoded.

Therefore, in this project, we utilize a number of open-source chest X-ray foundation models from the XRV library, allowing us to analyze the subgroup performance of models with frozen and unfrozen backbones.

2.3.2 Algorithmic Encoding

DL models can rely on spurious features in data to make decisions [15]. A feature is considered spurious, when it is used to predict a target label, but has little connection to the true labeling function [15]. For instance, models trained to detect pneumonia were able to predict the specific hospital that the training data was acquired from and ended up confounding these features with those relevant to identifying the pathology [23].

In a more concerning example, a study performed by Gichoya et al. shows that deep neural networks are able to detect protected characteristics, such as racial identity, from chest X-ray images with high accuracy [13]. Therefore, if a dataset exhibits spurious correlations between these protected characteristics and disease, a model might use these features as a shortcut to predict health outcomes [13, 7].

Supervised Prediction Layer Information Test

Gichoya et al. propose the supervised prediction layer information test (SPLIT) to determine whether a model trained for disease detection might have acquired features to detect racial information [7]. Based on transfer learning, this method involves developing a feature extractor for an initial task and then retraining its prediction layer for a different task. This allows us to observe whether shared features might be used for the decision making process in the new task.

In this case, Gichoya et al. first train a feature extractor for predicting disease and then retrain its classification head for race classification. During last-layer retraining, the neural network backbone is frozen, only allowing parameters in the classification head to be updated. Therefore, the new classification head should only have access to the previously learned disease detection features, despite being trained for race classification. However, if the accuracy for predicting race is still relatively high, one might conclude that these two tasks are spuriously correlated or at least have some relation to one another [7, 13].

However, Glocker et al. argue that this method is not sufficient to show how protected characteristics are encoded. It can not be used to discern whether tasks are related on both an output- and feature-level versus just a feature-level. Instead, they propose multitask learning using a shared backbone to more explicitly capture the relationship between these tasks [7].

2.4 Adversarial Training

Adversarial training is a bias mitigation technique intended to develop fair representations of data, whilst maintaining high accuracy for predictions [24]. This is usually achieved by employing a secondary model that acts as an adversary to a primary task. Since the primary model (i.e., label

predictor) and the adversary share a network backbone, the adversary is trained to assess whether learned feature representations from the primary task are predictive of the adversarial domain [7].

In order to visualize this, let us consider the feature representation R which is generated from an input X. Furthermore, we have an adversary that tries to predict S from R and a predictor that tries to predict Y from R. The objective is to use R to predict the target Y but minimize its dependence on S. Therefore, this can be described by a minimax optimization problem, where the general aim is to minimize the loss of the predictor whilst maximizing the loss of the adversary [24]. Omitting the cost of decoding, we get the general loss function

$$L(\theta, \phi) = D_{\theta,\phi}(S, R) + E_{\theta}(Y, R) \tag{2.3}$$

where D represents the dependence between R and S, and E represents the error for predicting Y from R [24]. In D, θ encompasses the parameters used for encoding R. In E, θ is extended to include the parameters of the predictor as well. The adversary parameters are represented by ϕ. Therefore, the minimax function can be expressed as:

$$\min_{\theta} \max_{\phi} L(\theta, \phi) \tag{2.4}$$

2.4.1 DANN

Domain Adversarial Neural Networks (DANNs) implement adversarial training to combine domain-invariance and discriminativeness into one training process. The overall goal is to train a feature extractor that produces domain-invariant features, such that a label predictor can predict for source and target domains without prior training on the target domain [25]. Thus, the typical structure of a DANN can be divided into the following three optimization tasks (Fig 2.6):

1. *Optimization of the Label Predictor*
2. *Optimization of the Domain Classifier*
3. *Optimization of the Feature Extractor*

The parameters in both the label predictor and domain classifier are optimized to minimize loss in their respective domains. However, the representation parameters adopt a minimax optimization operation that minimizes the loss in the label predictor and maximizes the loss in the domain classifier.

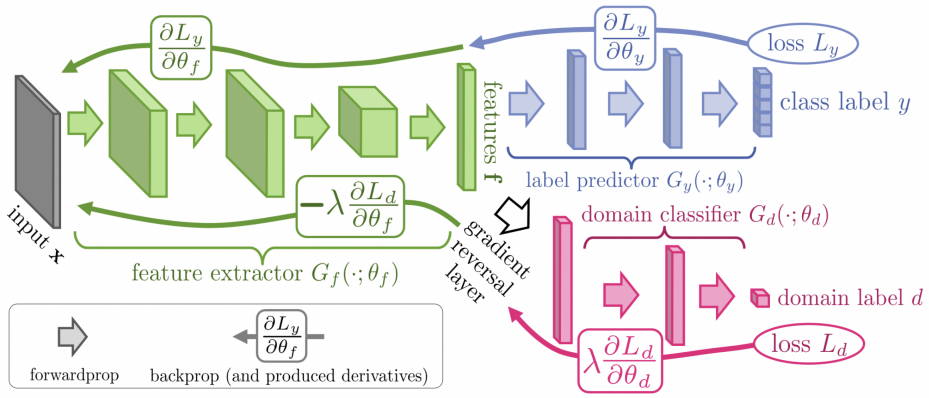


Figure 2.6: **Standard DANN architecture breakdown** Figure from [25]

Therefore, the feature extractor generates domain-invariant features, that impede the domain classifier's ability to discern source from target domains, whilst maintaining the label predictor's ability to form predictions for the source domain [25, 26]. This is achieved by employing a gradient reversal layer (GRL) between the feature extractor and the domain classifier. During forward propagation, the GRL acts as an identity layer keeping inputs unchanged, whereas during back-propagation, the GRL reverses all gradients generated from the domain classifier [25]. In contrast, the label predictor forms a standard feed-forward network with the feature extractor.

This is illustrated in Fig 2.6, where $G_f(\cdot;\theta_f)$ represents the feature extractor with parameters θ_f, $G_y(\cdot;\theta_y)$ represents the label predictor with parameters θ_y and $G_d(\cdot;\theta_d)$ represents the domain classifier with parameters θ_d. The overall optimization task can be described as [25]:

$$E(\theta_f,\theta_y,\theta_d) = \sum_{\substack{i=1\ldots N \\ d_i=0}} L_y^i(\theta_f,\theta_y) - \lambda \sum_{i=1\ldots N} L_d^i(\theta_f,\theta_d) \tag{2.5}$$

where

$$(\hat{\theta}_f,\hat{\theta}_y) = \underset{\theta_f,\theta_y}{\operatorname{argmin}}\, E(\theta_f,\theta_y,\hat{\theta}_d) \tag{2.6}$$

$$\hat{\theta}_d = \underset{\theta_d}{\operatorname{argmax}}\, E(\hat{\theta}_f,\hat{\theta}_y,\theta_d) \tag{2.7}$$

L_y and L_d are the loss functions for the label predictor and domain classifier respectively, where L_d is multiplied by the hyperparameter λ which determines the adversary strength during training.

2.5 Fairness

In medical imaging, algorithmic fairness and developing "fair" models is a growing topic of discussion. Since medical datasets are historically imbalanced, developing unbiased models for healthcare purposes becomes a challenge. This poses an ethical dilemma for the use of artificial intelligence (AI) in medicine, as implementing models that are potentially biased could perpetuate already existing healthcare disparities and negatively affect patients' quality of care.

2.5.1 Social Determinants of Health

Differences in medical conditions are found across protected identity subgroups, such as race, sex, age and gender. By investigating the implicit causes of these disparities, we can observe a much wider net of risk factors that influence health outcomes [27]. Prominent examples are: socioeconomic status, cognitive bias, geographical location, and education. These factors, associated with the social determinants of health, can be categorized into five main domains:

- **Economic stability**: Economic stability and socioeconomic status are major factors that contribute to existing health inequalities. Higher levels of income are associated with better health outcomes [4]. Not only does economic freedom enable access to better healthcare, it allows for healthier lifestyle behaviors and improved cognitive health. As a result, economically stable groups have a lower risk for developing non-communicable diseases such as cardiovascular disease, diabetes, and major depressive disorder [4, 28].

- **Access to education**: Many families are not able to afford the cost of education or send their children to private schools. Furthermore, coming from a low socioeconomic background can expose a child to social and environmental stressors that negatively impact their performance in school [28]. As a result, it becomes more difficult for them to find higher paying jobs and attain economic stability [28].

- **Access to healthcare**: Access to healthcare, while often determined by a person's socioeconomic status can also be influenced by other factors such as geographical location (e.g. if the patient lives too far away from their provider, or if there are limited providers in their area) [29]. These limitations cause issues in preventative care, diagnosis and treatment [4].

- **Environment**: Environmental factors have a significant impact on the health and safety of a person. For instance, growing up in violent environment can have a detrimental affect on emotional health. Alternatively, being exposed to toxic environments, such as buildings with contaminated water, lead, or carcinogens such as asbestos can stunt development and exacerbate disease [28].

- **Social context**: Lastly, social aspects have a particular influence on mental health and health outcomes [28]. Chronic exposure to social stressors can lead to the development of hypertension (high blood pressure) and hyperlipidemia (high cholesterol) [29].

While socioeconomic status plays a central role in tying the social determinants of health together, it is important to note that not all differences between groups stem from this cause. For instance, in the case of racial/ethnic groups, racial segregation could be the sole stressor which negatively impacts emotional health, and interferes with healthcare access and economic stability [29]. Due to the complex causal relationships between these social determinants, treating algorithmic fairness as a straightforward technical problem for developing unbiased models, is not a fully adequate approach to this issue [27]. Therefore, *in addition* to developing neutral algorithms, further steps such as improving transparency at the point of care or using diverse sources for predicting health outcomes, must be taken [27].

Furthermore, it is important to consider underlying biases that may not necessarily be explicitly reflected in the data. Often arising from outdated or prejudiced decision-making policies, these biases can result in certain groups being underdiagnosed. For example, due to having a lower risk adjustment for osteoporosis, Black women are less likely to have bone density screenings after hip fractures compared to their White counterparts [30]. As a result of reduced screenings, osteoporosis might be underdiagnosed in this subgroup, leading to annotation bias in the medical data.

Thus, even if a dataset might be balanced in terms of subgroup representation, the factual data might be considered noisy or low-quality. In the second part of this project, we investigate how DL models respond to this type of noise, and how effective DANNs are at mitigating bias generated by this noise.

2.6 Summary

To summarize, there are key limitations for machine learning models in medicine which need to be addressed. Specifically, we describe the limitations and inequities in medical datasets that pose challenges for fair model development. Since we will be training with chest radiographical data, we then introduce architectures for image classification. In particular, we analyze the DenseNet-121 architecture which is a type of CNN that allows for efficient learning through feature reuse. All the models we investigate implement DenseNet-121 as a network backbone. Following this, we inspect foundation models – pre-trained feature extractors that use transfer learning techniques for domain adaptation. However, as one of the central problem statements in this thesis, we emphasize that biases can be encoded during pre-training. We suggest utilizing a DANN to mitigate this bias, which is accomplished by generating domain-invariant features that confuse the domain classifier. Finally, we discuss how social determinants of health impact bias development and propose exploiting an intentionally biased model to test the effectiveness of DANNs.

Chapter 3

Methodology

This project can be divided into two sets of experiments:

1. **Bias analysis of foundation models**
    - Investigate pre-trained models from the TorchXRayVision (XRV) library by assessing and comparing subgroup performance for the fixed feature extraction method vs fine-tuning method.
    - For each of these methods, inspect features in the penultimate layer for biased distributions using dimensionality reduction techniques Principle Component Analysis (PCA) and t-Distributed Stochastic Neighbor Embedding (t-SNE).
    - Implement a multitask model and analyze the relationship between classification heads using above model inspection frameworks.
2. **DANN and bias mitigation with label noise**
    - Adopt a DANN utilizing XRV foundation models as the convolutional base and observe differences in penultimate features/performance compared to models trained normally.
    - Introduce systematic/random label noise into certain subgroups of the dataset, so that we intentionally create a biased model.
    - Assert whether DANN works as a bias mitigation technique, by observing if subgroup performance is rescued.

3.1 Models

3.1.1 TorchXRayVision

In this project, we utilize foundation models from the TorchXRayVision library. This library contains a collection of PyTorch compatible models, that are pre-trained on various chest X-ray datasets. The goal of XRV is to function as a tool for easy model implementation and development, where the models can be used for baseline comparisons, as well as feature extractors for transfer learning [31]. Furthermore, XRV provides useful configurability and accessibility for many datasets, making model evaluation a simple task.

While the majority of models in the library are DenseNets, a few have residual neural network (ResNet) architectures. However, we investigate the core classifiers which utilize a DenseNet-121 architecture. Each model can be loaded with their corresponding weights as follows:

```
model = xrv.models.DenseNet(weights="densenet121-res224-all")
```

These represent the CNN architecture, resolution of the input and dataset it was trained on respectively. The datasets used for training are: **RSNA** Pneumonia Detection Challenge (*-rsna*), **NIH** Chest X-ray14 (*-nih*), **PadChest** (*-pc*), **CheXpert** (*-chex*), and **MIMIC-CXR** (*-mimic_ **). There exist two types of models which were trained on MIMIC-CXR: mimic_nb and mimic_ch. This naming convention is dependent on which auto-labeller was used, where the term "nb" stands for the NIH NegBio labeller and "ch" stands for the CheXpert labeller. Lastly, the

"All" (*-all*) model is pre-trained on each of the previously listed datasets (rsna, nih, pc, chex, mimic_ch) as well as the Google NIH [32] and OpenI [33] datasets.

Every model is trained for an output size of 18 pathology classes. These include: Atelectasis, Cardiomegaly, Consolidation, Edema, Effusion, Emphysema, Enlarged Cardio, Fibrosis, Fracture, Hernia, Infiltration, Lung Lesion, Lung Opacity, Mass, Nodule, Pleural Thickening, Pneumonia, and Pneumothorax. However, not all output classes are represented in every dataset, so models will return NaN as a prediction for unknown pathologies; the exception being the `densenet121-res224-all` model which predicts all pathologies. Models and their corresponding output classes are depicted in Table 3.1.

Dataset	Weights	Pathologies	Citation
RSNA	`densenet121-res224-rsna`	Lung Opacity, Pneumonia	Shih [34]
NIH	`densenet121-res224-nih`	Atelectasis, Cardiomegaly, Consolidation, Edema, Effusion, Emphysema, Fibrosis, Hernia, Infiltration, Mass, Nodule, Pleural Thickening, Pneumonia, Pneumothorax	Wang [35]
PadChest	`densenet121-res224-pc`	Atelectasis, Cardiomegaly, Consolidation, Edema, Effusion, Emphysema, Fibrosis, Fracture, Hernia, Infiltration, Mass, Nodule, Pleural Thickening, Pneumonia, Pneumothorax	Bustos [36]
CheXpert	`densenet121-res224-chex`	Atelectasis, Cardiomegaly, Consolidation, Edema, Effusion, Enlarged Cardio, Fracture, Lung Lesion, Lung Opacity, Pneumonia, Pneumothorax	Irvin [37]
MIMIC	`densenet121-res224-mimic_nb`, `densenet121-res224-mimic_ch`	Atelectasis, Cardiomegaly, Consolidation, Edema, Effusion, Enlarged Cardio, Fracture, Lung Lesion, Lung Opacity, Pneumonia, Pneumothorax	Johnson [38]

Table 3.1: **Datasets used for developing XRV models and the pathologies they predict.**

During pre-training, data augmentation is used to improve generalization of the models. This involves rotation (up to 45°), translation (up to 15%) and resizing of the images [39]. XRV models are trained to receive 224 × 224 images as input with pixel values normalized to [-1024, 1024].

Though XRV offers many procedures for dataset manipulation and reconfiguration, we primarily use this library for its feature extractors. Applying transfer learning techniques, we implement them as a backbone for supervised information tests, multitask learning and DANNs. Fine-tuning is performed using a sampled CheXpert set that involves retraining prediction heads for disease detection, race classification or sex classification. The details of how sampling is performed is explained in section 4.2.

3.1.2 ImageNet

ImageNet is an extensive, diverse database of annotated images, consisting of over 14M images with 22,000 categories. Typically, a subset of this database is used for model development, referred to as the ImageNet-1K dataset or ILSVRC 2012. This dataset consists of 1000 object classes and 1,281,167 training images. Its validation and test sets consist of 50,000 and 100,000 images respectively. As a baseline, we utilize a DenseNet-121 model loaded with ImageNet-1K weights and fine-tuned on CheXpert. Therefore, the ImageNet models act as a control group for observing differences in feature vectors, classification performance and bias development compared to XRV models. This is implemented using the torchvision package, a part of the PyTorch framework:

```
model = models.densenet121(pretrained=True)
```

Furthermore, they are used as a convolutional base for multitask learning and adversarial training. The ImageNet models expect a 224 × 224 input as well, with 3 channels for color.

3.2 Exploration of feature representations

In order to detect bias between subgroups, dimensionality reduction techniques can be employed that help visualize the relationship between datapoints. This involves mapping high-dimensional feature representations from the penultimate layer of a model, to low-dimensional spaces that act as summaries of those features. We use principal component analysis (PCA) and t-distributed stochastic neighbor embedding (t-SNE) to accomplish this. Both of these methods function in an unsupervised manner, meaning that patterns can be recognized without prior knowledge of the data.

3.2.1 PCA

In PCA, the low-dimensional representations of data are organized into modes, or principal components (PCs). They capture the main axes of variation in the data, with PC1 representing the largest variance between projected units [7]. All PCA modes are geometrically orthogonal, meaning that all subsequent PCs are chosen such that they are uncorrelated with each other [40]. As a result, 2 PCs can be projected onto a 2-D space, where the PC with the highest variance (i.e., lower mode) goes along the first axis. These plots can be used to find potential clusters in the data and separations in distributions for certain subgroups (Fig. 3.1). Therefore, in classification tasks, separations used to make predictions are usually located within the first few modes of PCA.

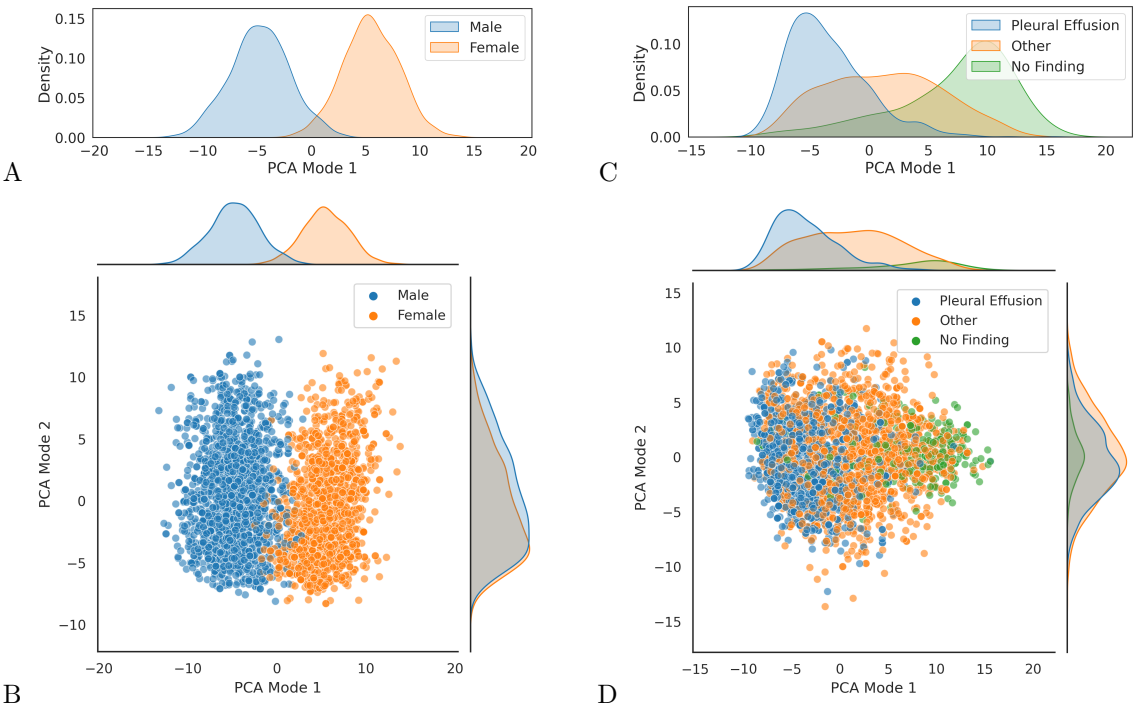


Figure 3.1: **PCA scatter plots and corresponding marginal distributions**. **A.** Normalized representation of the marginal distributions in B for PCA mode 1. We can see a clear shift between Male and Female subgroups, indicating that the model uses these features to make decisions. **B.** Scatter plot for PCA modes 1 + 2 regarding biological sex. This is part of a multitask model (ImageNet backbone + CheXpert fine-tuning) trained for disease detection, sex classification and race classification. As expected, clear clustering for Male and Female in PCA mode 1 can be observed, which is further elucidated in the marginal distributions. **C.** Normalized representation of the marginal distributions in D for PCA mode 1. We can see a clear shift between Pleural Effusion, Other and No Finding, suggesting a disease detection model. **D.** Scatter plot for PCA modes 1 + 2 regarding a single-task model (ImageNet backbone + CheXpert fine-tuning) trained for disease detection. Clustering in PCA mode 1 can be observed.

However, PCA does have limitations. Firstly, PCA may not pick up on all patterns found in the data. For instance, it has difficulty recognizing non-linear patterns, as well as patterns that are highly correlated, but non-orthogonal [40]. Furthermore, since PCA is a dimensionality reduction technique, information and patterns might be lost that can only be represented in higher dimensions.

3.2.2 t-SNE

T-SNE, like PCA, aims to transform high-dimensional feature vectors into low dimensional representations. Therefore, this method can be used to investigate clustering patterns and correlations within patient subgroups. However, rather than capturing the largest variations in data, t-SNE captures similarities between samples, meaning that the orientation of t-SNE projections can be somewhat arbitrary [10]. This makes it more difficult to draw conclusions about the relationship between subgroups, which is why we primarily use PCA for model inspection.

We map PCA modes for particular subgroups (e.g. disease, racial identity, biological sex) onto 2-D scatter plots as illustrated in Fig. 3.1 **B**+**D**. Subsequently, this is used to generate normalized density distributions for these subgroups (Fig. 3.1 **A**+**C**). To test whether differences in distributions are significant, a two-sample Kolmogorov-Smirnov test (Section 3.3.4) can be employed. If distributions are significantly shifted in subgroups not associated with the main prediction task, this may suggest that the model is biased and uses these features spuriously to make decisions.

3.3 Performance Assessment

In addition to feature inspection frameworks, performance disparities can be used as an indicator for the existence of biases in subgroups. That is to say, even if biases may exist in the penultimate feature vectors, what ultimately matters is whether they affect downstream performance [10]. For example, it could be the case that the classifier adjusts how it interprets certain features and as a result, mitigates bias present in the backbone.

The metrics used for assessing the individual performance of models are as follows: (1) the area under the receiver operating characteristic curve (AUROC or AUC), (2) true positive rate (TPR) and (3) false positive rate (FPR). Furthermore, to summarize the classification performance for particular subgroups, Youden's J Statistic can be implemented, which is calculated using FPR and TPR.

3.3.1 AUC

AUC is determined by calculating the area under the ROC curve, which plots the relationship between TPR and FPR for multiple decision thresholds (x: FPR, y: TPR). Therefore, AUC is a measurement of performance that is independent of a discrete decision making threshold and encompasses the total classification ability of the model [7]. The ROC curve can be adapted to display the TPR and FPR for subgroups, thus generating an AUC that can be used to compare performance between subgroups. An AUC of 0.5 means that predictions are random, and an AUC of 1.0 means that the model is able to classify inputs perfectly.

3.3.2 TPR and FPR

In contrast, TPR and FPR are determined using a fixed decision threshold which represents a single point on the ROC curve. The decision threshold is optimized such that the selected threshold has an FPR value closest to the target FPR. If this is a global threshold, it can be applied to calculate the explicit FPR and TPR values for all selected subgroups. To implement a group threshold, the threshold must be selected using the list of FPR values for that subgroup, rather than the FPR values for the whole population.

The TPR for a certain a threshold is defined as $Sensitivity(recall)$ and the FPR is defined as $1 - Specificity$ [7]. Sensitivity represents the probability that a result is positive and the true label is positive. Specificity describes the probability that a result is negative and the true label is negative. Therefore, FPR $(1 - Specificity)$ is the probability that a result is positive but the true label is negative.

3.3.3 Youden's J Statistic

Youden's J Statistic provides a classification performance metric that evaluates the relationship between a model's sensitivity and specificity [7]. It can be calculated as follows:

$$J = Sensitivity + Specificity - 1 \tag{3.1}$$

hence

$$J = TPR - FPR \tag{3.2}$$

This can be used as a target to optimize the decision making threshold and calculate resulting TPR/FPR. However, we primarily use Youden's J Statistic to quantify the classification performance of certain subgroups.

3.3.4 Kolmogorov-Smirnov test

The Kolmogorov-Smirnov test or K-S test is a statistical test used to determine whether two distributions differ significantly i.e., whether the data in these distributions comes from the same distribution (null hypothesis) [41]. The K-S statistic is calculated by finding the maximum absolute difference between the two distributions. This is then compared to a significance operation (dependent on a parameter α) to determine whether the null hypothesis should be rejected. In summary:

$$D^* = \max_x(|F_1(x) - F_2(x)|) \tag{3.3}$$

where D^* is the K-S statistic and, F_1 and F_2 are the distribution functions being compared [41]. This can be implemented using the ks_2samp function from the sklearn library.

3.4 Testing for Spurious Correlations

We analyze models using the above model inspection frameworks to determine whether spurious correlations were formed during training.

Firstly, we explore how these metrics are affected when freezing the neural network backbone i.e., having a fixed feature extractor during training. This is accomplished by employing a modified SPLIT using the core classifiers from the XRV library. These are re-trained on CheXpert for disease detection and compared to XRV models developed using an unfrozen backbone. Functioning as a baseline, ImageNet models are also re-trained with frozen and unfrozen backbones for disease detection. Therefore, if backbone $= BB$ and classification head $= CH$, we develop the following:

Backbone	Frozen	Updated Parameters	Prediction Task
XRV Core Classifier	Y	CH	Disease
XRV Core Classifier	N	$BB + CH$	Disease
ImageNet DenseNet	Y	CH	Disease
ImageNet DenseNet	N	$BB + CH$	Disease

Table 3.2: **Single task model.**

Secondly, we investigate the relationship between subgroups in a multitask model (disease detection, race classification, sex classification). This involves updating parameters in a shared backbone to generate features applicable to all tasks. By comparing these features to a single task model, this might provide insight into how different tasks are correlated [7]:

Backbone	Frozen	Updated Parameters	Prediction Task
XRV Core Classifier	N	$BB + CH_1 + CH_2 + CH_3$	Disease, Race, Sex
ImageNet DenseNet	N	$BB + CH_1 + CH_2 + CH_3$	Disease, Race, Sex

Table 3.3: **Multitask model.**

During multitask training, parameters in the backbone can be frozen. This can be targeted, such that backbone parameters are updated for certain tasks and unchanged for others. We utilize multitask models developed with unfrozen backbone parameters for disease classification but frozen parameters when training for race and sex, as a baseline comparison for DANNs.

3.5 Label Corruption

Label noise in medical datasets, which often stems from systematic mis- or underdiagnosis, poses a challenge for fair model development. In cases where subgroups are imbalanced, methods such as resampling can be implemented for bias mitigation and dataset rebalancing. However, label noise or annotation bias cannot be easily corrected for using such techniques as it refers to an underlying issue with the dataset [7, 42]. In order to simulate how DL models would react to this type of noise, we introduce artificial label noise into certain subgroups of the dataset.

Two types of noise are used: systematic noise and random noise. Let X represent an image, where Y is the true class (in this case the disease label) for that image and $\tilde{Y}$ is the observed label (noisy disease label) [42]. S and R represent the racial identity and biological sex subgroups respectively. Lastly, let us consider E to be the noise generator and variable that indicates the occurrence of label noise. Statistical dependencies are represented by arrows connecting these variables [42].

For systematic noise, noise (E) is dependent on R (or S) and the disease label Y. Specifically, noise is generated for a particular race subgroup (or sex) if a certain disease label is present (no_finding = 0). To imitate underdiagnosis, label values are then flipped to eliminate the presence of positive pathology labels (no_finding = 1).

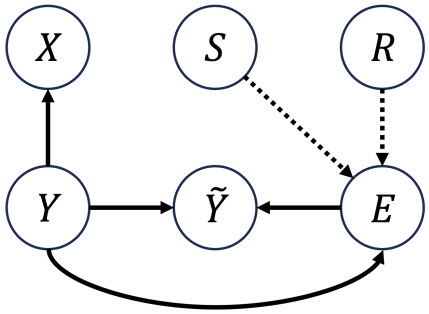


Figure 3.2: **Relationships between variables for systematic label noise in subgroups.**

Random noise, on the other hand, involves assigning labels to a certain subgroup by randomly choosing labels out of the list of available pathologies. Therefore, while E is dependent on R (or S) it does not depend on Y [42]. In both cases, the noisy label $\tilde{Y}$ depends on the true label Y and the noise generator E.

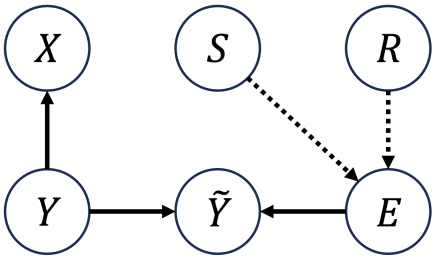


Figure 3.3: **Relationships between variables for random label noise in subgroups.**

3.6 Bias Mitigation

DANNs are used as a bias mitigation technique and method for correcting label noise. They employ a secondary task model to determine whether features in a shared backbone are predictive of the secondary task, whilst being trained for the primary task [25]. Despite its similarities to SPLIT, DANNs differ by using the secondary model to generate domain-invariant features i.e., remove features predictive of the second task during training [25, 7]. Typically, these features are generated using a GRL, which acts as an identity layer during forward propagation and as a

gradient inverter during back propagation [25]. Instead of a GRL, we use two types of operations to update parameters in the backbone:

1. **Gradient Negation**: Negating gradients acts similarly to a GRL and negates the returned loss from the domain classifier. This is achieved by multiplying the domain classifier loss by a hyperparameter α, which determines the strength of the adversary (i.e., the impact the adversary has on the system). Overall, this can be described as $-\alpha * L_d$, where L_d is the loss of the domain classifier.

2. **Loss Confusion**: Instead of explicitly targeting gradients, this method utilizes an alternative loss function for calculating the domain classifier loss. This involves forcing predictions from the domain classifier to be close to a uniform distribution by equalizing softmax outputs [43]. The overall function can be described as:

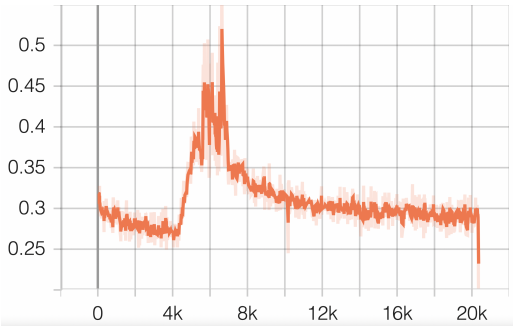
$$L_{conf} = -\frac{1}{S_u}\sum_{s=1}^{S_u}\sum_{k=1}^{N}\frac{1}{N}log(p_{s,k}) \tag{3.4}$$

Because debiasing requires signals from classifiers that work reasonably well in the first place, there is a fading in period where the domain classifier is solely trained to predict the second task (albeit on a frozen backbone). Only after the fading-in-steps are reached, does the model get utilized adversarially and the debiasing becomes active.

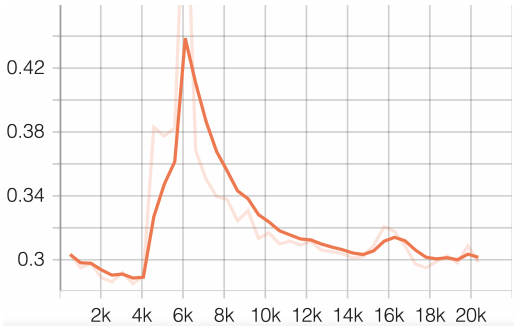
3.7 Loss Monitoring

TensorBoard is a visualization toolkit developed by TensorFlow, that can be used to track the values of certain metrics during training. By logging values at the end of every epoch or after every training step, it is possible to visually summarize the temporal progression of metrics throughout experimentation.

During multitask training and adversarial training the loss of individual classification heads can be monitored. This is especially useful in DANN development to observe how loss values in each head are affected after including the adversary. Since adversarial training is introduced after a selected interval (after a certain amount of fading-in-steps have occured), TensorBoard can be used to track whether this interval is implemented properly.



(a) Train_loss_disease with adversary introduction after 5000 steps (x: number of steps taken, y: training loss)



(b) Val_loss_disease with adversary introduction after 5000 steps (x: number of steps taken, y: validation loss)

Figure 3.4: **Training and validation loss for disease/label predictor in a DANN (BB = ImageNet DenseNet-121).** Gradients are negated based on race classification loss from the domain classifier. To ensure the race classification head functions as an adequate adversary, it is trained on the shared backbone with frozen weights for a certain amount of steps. We can see a spike in both losses once the fading-in-steps are reached and the adversary is introduced.

For instance, in initial experiments, differences between Pytorch-Lighting 2.0 and earlier versions, caused the adversary to be introduced too early. Namely, each optimizer step counted towards the global_step rather than every loop through the list of optimizers counting as one global_step. Since four optimizers were used, the fading-in-steps were short by a factor of four. We were able to observe this in TensorBoard and correspondingly correct the hyperparameters.

3.8 Summary

In this chapter, the performance analysis methodologies and model inspection frameworks utilized throughout this thesis are introduced. This involves inspecting models at (1) a feature level and (2) an output level. At a feature level, dimensionality reduction techniques, such as PCA and t-SNE, can be adopted to observe the relationships and patterns between subgroups. Significant differences between subgroups can be determined using a K-S test. At an output level, differences between subgroups can be quantified by observing performance metrics such as AUC, TPR/FPR, and Youden's J statistic. These techniques are used to investigate different approaches for training models, including single task, multitask and adversarial training. Models of interest are foundation models from the XRV library, trained on various chest X-ray datasets. Furthermore, we describe the mechanisms for introducing label noise into the dataset (systematically and randomly) and methods for mitigating bias (DANN) which arises due to this noise.

Chapter 4

Experimental Setup

4.1 Datasets

As mentioned in Section 3.1, XRV models are pre-trained on various chest X-ray datasets for disease detection and function as feature extractors in transfer learning tasks. Datasets used for pre-training include: RSNA [34], NIH [35], PadChest [36], CheXpert [37] and MIMIC-CXR [38]. Additionally, DenseNet-121 models trained on ImageNet-1K are utilized as a baseline.

4.1.1 CheXpert

In this project, we employ a sampled CheXpert dataset for classification head re-training on XRV and ImageNet backbones. Therefore, to prevent overfitting, the CheXpert XRV model (`densenet121-res224-chex`) is omitted from experimentation. Since CheXpert consists of chest radiographical data, it is primarily used as a training set for disease detection tasks.

Concernedly, Gichoya et al. have shown that protected characteristics such as race and sex can be classified using chest X-rays [13]. Therefore, to investigate this behavior, we also utilize CheXpert for developing racial identity and biological sex prediction tasks. This is performed in multitask and adversarial training.

Demographics

The official CheXpert dataset is a chest X-ray collection of 224,316 images from 65,240 patients [37]. Each image is annotated with 14 observation labels, consisting of two labels marking "No Finding" and "Support Devices", and the remaining 12 describing pathologies found in patients.

Pathology	Positive (%)	Uncertain (%)	Negative (%)
No Finding	16627 (8.86)	0 (0.0)	171014 (91.14)
Enlarged Cardiom.	9020 (4.81)	10148 (5.41)	168473 (89.78)
Cardiomegaly	23002 (12.26)	6597 (3.52)	158042 (84.23)
Lung Lesion	6856 (3.65)	1071 (0.57)	179714 (95.78)
Lung Opacity	92669 (49.39)	4341 (2.31)	90631 (48.3)
Edema	48905 (26.06)	11571 (6.17)	127165 (67.77)
Consolidation	12730 (6.78)	23976 (12.78)	150935 (80.44)
Pneumonia	4576 (2.44)	15658 (8.34)	167407 (89.22)
Atelectasis	29333 (15.63)	29377 (15.66)	128931 (68.71)
Pneumothorax	17313 (9.23)	2663 (1.42)	167665 (89.35)
Pleural Effusion	75696 (40.34)	9419 (5.02)	102526 (54.64)
Pleural Other	2441 (1.3)	1771 (0.94)	183429 (97.76)
Fracture	7270 (3.87)	484 (0.26)	179887 (95.87)
Support Devices	105831 (56.4)	898 (0.48)	80912 (43.12)

Table 4.1: **Counts of pathologies present in the CheXpert dataset.**[37]

These include: Enlarged Cardiomegaly, Cardiomegaly, Lung Lesion, Lung Opacity, Edema,

Consolidation, Pneumonia, Atelectasis, Pneumothorax, Pleural Effusion, Pleural Other, and Fracture. Patients can exhibit multiple pathologies.

CheXpert, like many medical datasets, is severely imbalanced. This is especially prevalent when examining the racial identity distribution of the dataset: White patients make up 78% of the population, whereas other subgroups only make up 15% (Asian) and 7% (Black) of the population [10]. Furthermore, CheXpert displays gender imbalance, with Male and Female subgroups making up 59% and 41% of the population respectively.

Lastly, class imbalance can be observed as well, where some pathologies such as Pleural Effusion (40.34%) and Lung Opacity (49.39%) are present in almost half the samples, whereas Pneumonia (2.44%) or Pleural Other (1.3%) have low representation in the dataset [44]. This is elucidated in Table 4.1, which describes the number of positive, negative and uncertain samples found in the dataset. It is important to note that a patient can display multiple pathologies at once, therefore the columns do not add up to 100%. In the .csv files of the dataset, positive labels are encoded as 1.0, negative as 0.0 and uncertain as -1.0. For simplicity, we ignore labels marked as uncertain.

4.2 Resampling

4.2.1 Training Set

Since the canonical CheXpert dataset does not contain subgroup metadata (racial identity, biological sex, and age), sampling was performed on the original training sets for proper analysis. This allows training to be performed using protected characteristic subgroups. Furthermore, to simplify the quantification of disease classification performance and feature inspection, pathology labels are divided into three subgroups: Pleural Effusion, No Finding and Other. Following the sampling splits from Gichoya et al. [13, 7], the dataset was split into training (76,205 images), validation (12,673 images) and test (38,240 images) sets. The exact procedure for implementing the Gichoya split points was performed using notebooks provided by Glocker et al. [7]. Class and subgroup proportions are preserved in the study population, as illustrated below (Table 4.2).

Dataset	Attribute	All	White (%)	Asian (%)	Black (%)
All data	Scans	127,118	99,027 (78%)	18,830 (15%)	9261 (7%)
	Female	52,436 (41%)	39,735 (40%)	8132 (43%)	4569 (49%)
	No finding	10,916 (9%)	8236 (8%)	1716 (9%)	964 (10%)
	PE	51,574 (41%)	40,545 (41%)	7953 (42%)	3076 (33%)
Training set	Scans	76,205	59,238 (78%)	11,371 (15%)	5596 (7%)
	Female	31,432 (41%)	23,715 (40%)	4976 (44%)	2741 (49%)
	No finding	6514 (9%)	4910 (8%)	1046 (9%)	558 (10%)
	PE	31,015 (41%)	24,405 (41%)	4754 (42%)	1856 (33%)
Validation set	Scans	12,673	9945 (79%)	1809 (14%)	919 (7%)
	Female	5030 (40%)	3933 (40%)	667 (37%)	430 (47%)
	No finding	1086 (9%)	817 (8%)	175 (10%)	94 (10%)
	PE	5049 (40%)	3988 (40%)	738 (41%)	323 (35%)
Test set	Scans	38,240	29,844 (78%)	5650 (15%)	2746 (7%)
	Female	15,974 (42%)	12,087 (41%)	2489 (44%)	1348 (49%)
	No finding	3316 (9%)	2509 (8%)	495 (9%)	312 (11%)
	PE	15,510 (41%)	12,152 (41%)	2461 (44%)	897 (33%)

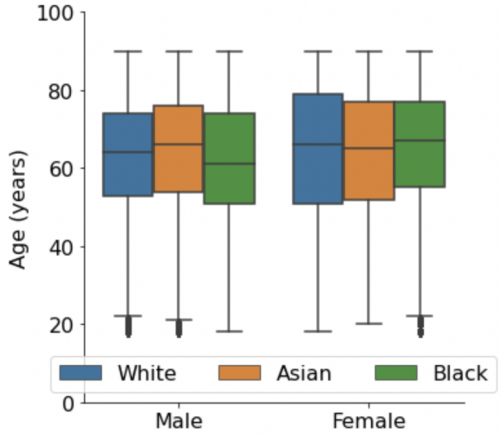
Table 4.2: **Patient demographics in sampled CheXpert set (Gichoya splits).**[7, 13]

4.2.2 Test Set

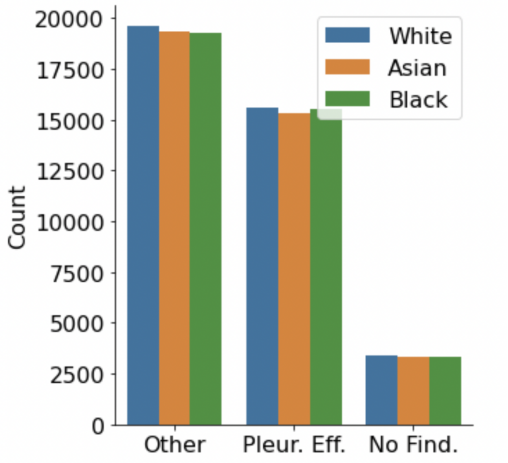
Furthermore, test set re-sampling was conducted to allow for unbiased interpretations of subgroup performance. Since the original test set and training set exhibit similar population disparities,

biases originating during training time might be confounded with subgroup differences from the test set [7]. Therefore, testing must be balanced in order to observe the true behavior of our models.

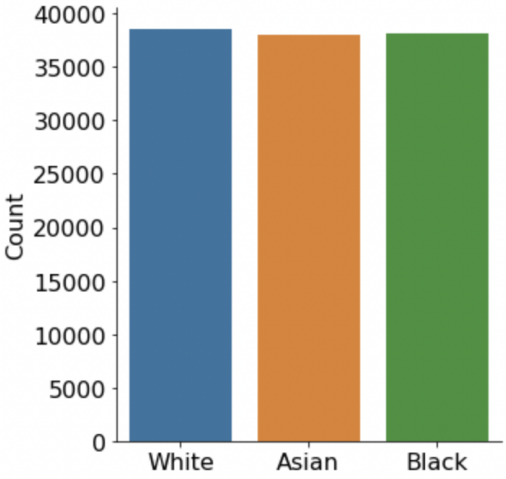
This is implemented by creating race balanced test sets whilst correcting for pathology prevalence and age. However, it should be noted that inequalities between Male and Female subgroups are not corrected for. As mentioned previously, model performance metrics and feature inspection frameworks for disease detection focus on two mutually exclusive tasks to make conclusions; (1) prediction of a certain pathology (Pleural Effusion) and (2) prediction of the absence of pathology (No Finding). Additionally, if the predicted label does not fall into either of these categories, it is referred to as "Other". In the re-sampled test set, these disease outcomes are evenly distributed across race subgroups (4.1 b). In order to accurately compare results with previous experiments conducted by Glocker et al., the implemented re-sampled test set remains consistent with the test set they utilized.



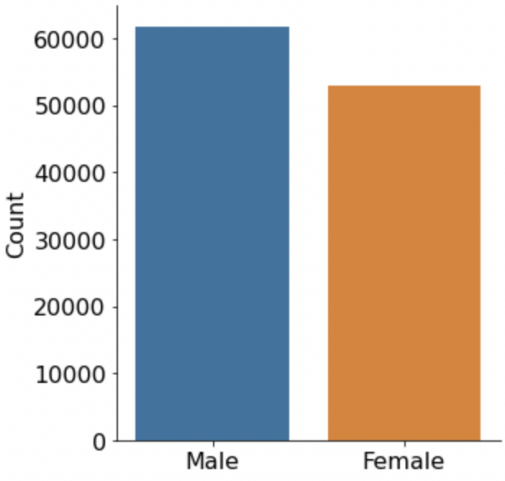
(a) Balanced for age within (race, sex)-subgroups



(b) Within each race subgroup, equal prevalence of disease labels



(c) Racial subgroups are evenly distributed: White: 38,605 Asian: 38,009 Black: 38,106



(d) Sex subgroup imbalance: Male: 61,739 Female: 52,981

Figure 4.1: **Subgroup counts for the re-sampled test set.**

4.3 Preprocessing

Image augmentation is conducted solely on training set data, thus excluding the testing and validation sets. Augmentation is performed by utilizing image transformation methods (also called transforms) available in the `torchvision.transforms` library. These functions are implemented

in our custom `CheXpertDataset` module where transformations are applied to each image sample. Operations can be chained together in a composition object which allows multiple transforms to be conducted in every augmentation sequence. We use the following transforms that geometrically alter the image:

1. `RandomHorizontalFlip`: flip the image randomly with a given probability. We select a probability of 0.5.

2. `RandomAffine`: returns an affine transformed image. Specifically, we utilize a range of degrees between (-15, 15) and a scaling factor interval of (0.9, 1.1). This is wrapped in a RandomApply function which has a 50% probability of applying this transform.

Finally, the dataset images are resized to dimensions the models expect as input (224×224). Throughout this process, the path to the preprocessed image is stored in the sampled CheXpert .csv file.

Chapter 5

Implementation

5.1 PyTorch Lightning

The bulk of our model implementation is built upon the PyTorch Lightning framework. PyTorch Lightning is a PyTorch-based DL tool that offers efficient and scalable methods for model development and experimentation. Acting as a wrapper for PyTorch modules, this framework improves organization and flexibility. In particular, we utilize PyTorch Lighting for the following:

1. **LightningModule**: a wrapped PyTorch neural network module providing code organization and configurability.
2. **Trainer**: works with the LightningModule for controlling and automating training, validation and testing processes. Also provides checkpointing and logging features.
3. **LightningDataModule**: used for loading and implementing datasets efficiently.
4. **Checkpointing**: method for model selection based on logged metrics.
5. **Logging**: allows for visualization of metrics during training with TensorBoard.

5.2 Modules

5.2.1 LightningModule

The neural network modules employed in this project are organized in the PyTorch Lightning (pl) `LightningModule`. In conjunction with a pl `Trainer`, this allows training, validation and testing loops to be easily implemented and personalized. During initialization of the module, the pre-trained convolutional base (XRV or ImageNet) and the corresponding hyperparameters for training are loaded as class attributes. Subsequently, a re-trainable classification head is connected to the loaded backbone. Parameters for all network components are assigned in the `initialize_parameters` function.

The pl `Trainer` moderates training in the `training_step` method, which processes each batch of images via forward passes through the model and returns the appropriate loss. If optimizers are configured to update manually, we integrate optimization steps into this `training_step`. Otherwise, optimization is automated by the pl `Trainer`.

Because sex and race classification are multi-class problems, loss is calculated using cross entropy loss. On the other hand, disease classification is treated as a multi-label problem since patients can exhibit multiple pathologies at once. Therefore, raw outputs are first passed through a sigmoid function and then used to calculate binary cross entropy loss [7, 10]. In DANNs, the confusion loss function is calculated by taking the negative average of the log softmax of the corresponding domain classifier (race or disease) logits [43].

5.2.2 Classification Layers

For multitask and adversarial learning, we designed a multitask head class that can be attached to the pre-trained feature extractor. This is connected to the backbone by a single identity layer,

meaning that all classification layers share the same backbone. The head consists of three separate linear layers, functioning as classification heads for disease, sex and race. For a single task model, a single linear layer is implemented as the classifier, which is directly connected to the backbone.

5.3 Hyperparameters

The *initial* hyperparameters utilized for single task, multitask and DANN development are depicted in the table below (5.1). To improve customizability, hyperparameters can be passed in through the command line and modify training conditions. Their values are stored in a JSON file for future implementation.

Since adversarial training requires more iterations to achieve sufficient performance, single and multitask models are trained for 20 epochs, whereas DANNs run for 40 epochs. Furthermore, fading-in-steps and fading-in-range values can vary ([20000, 800] versus [5000, 200]), since global steps are counted differently depending on the pl version, as mentioned in 3.7.

Hyperparameter	Value
Epochs	20 (for single/multitask) or 40 (for DANN)
Batch size	150
Image size	(224, 224)
Learning rate (all components)	0.001
α	0.0001
No. of disease classes	14
No. of racial classes	3
No. of sex classes	2
Fading in steps	20000 (if PyTorch Lighting version $>$ 2.0, else 5000)
Fading in range	800 (if PyTorch Lighting version $>$ 2.0, else 200)

Table 5.1: **Initial hyperparameters implemented during training.** DANNs utilize twice as many epochs compared to single and multitask models

The behavior of models when adjusting learning rate, α, and fading-in-steps is inspected. In Section 5.9, we describe how the individual learning rates for network components are tuned using the Optuna. Furthermore, we investigate how the magnitude of α and fading-in-steps affects adversarial training performance.

5.4 Training

The overall training procedure is regulated by the pl `Trainer` class, which also provides logging functions, checkpointing, validation steps, and GPU configuration. The implemented parameters are as follows:

Parameter	Value
callbacks	Loaded checkpoint from ModelCheckPoint
log_every_n_steps	5
max_epochs	number of epochs (20/40)
accelerator	"gpu"
devices	number of GPUs (1)
logger	TensorBoardLogger

Table 5.2: **Implemented pl Trainer parameters.**

5.4.1 Single task classification

In a single task setting, we inspect foundation models trained using frozen and unfrozen backbones for potential biases (described in Section 3.4). This involves connecting a new classification head

to a pre-trained XRV feature extractor (DenseNet-121 archtecture), which is then re-trained for disease detection on the sampled CheXpert dataset.

Since CheXpert is labeled with 14 different conditions and XRV models are trained to output 18 classes, the `op_threshs` parameter is cleared such that there are no conflicting tensors. Last-layer re-training is performed using all core XRV models, excluding the CheXpert model, therefore: `-all, -pc, -rsna, -mimic_nb, -mimic_ch,` and `-nih`. Furthermore, ImageNet models, i.e., DenseNet-121 models pre-trained on ImageNet-1K, are implemented as a baseline comparison. However, it is important to note that these models still can exhibit biases.

For XRV model development, images are normalized to [-1024, 1024]. Conversely, when developing ImageNet models, images are kept in the [0, 255] range. Since ImageNet models expect an rgb input, images are passed through a `pseudo_rgb` function generating input with 3 channels.

5.4.2 Multitask learning

Multitask learning is performed by training a multitask head with an XRV foundation model or ImageNet model as a convolutional base. In a traditional multitask model, we train for three tasks (disease, race, sex) whilst generating shared features predictive of these tasks. Therefore, backbone features must be updated during the training of each classification head as well (Table 5.3). Subsequently, by inspecting learned representations, this allows us to observe potential relationships between these tasks.

Task	Updated Parameters
Disease	Disease classification head + backbone
Race	Race classification head + backbone
Sex	Sex classification head + backbone

Table 5.3: **Updatable parameters multitask, unfrozen.**

Multitask models with partially frozen backbones can be developed as a baseline measure for DANNs to inspect the performance of race/sex classification heads. This involves freezing backbone parameters when training for race and sex classification, but updating backbone parameters when training for disease (Table 5.4). Needless to say, the pre-trained model type (XRV model weights/ImageNet), remains consistent with the backbone implemented in the adversarial network.

Task	Updated Parameters
Disease	Disease classification head + backbone
Race	Race classification head
Sex	Sex classification head

Table 5.4: **Updatable parameters multitask, partially frozen.** Baseline for DANN

5.4.3 DANN

For adversarial training, we utilize a modified DANN with three classification heads (multitask head) built upon a shared backbone. Two of the heads maintain the same functionality as in Section 2.4.1 of label predictor and domain classifier, whereas the third added head allows us to monitor a tertiary supplementary task. The label predictor is implemented to predict disease and forms a complete feed-forward network with the backbone. Therefore, during training, backbone parameters and classification layer parameters are updated.

The domain classifier is first trained for a selected task (race or sex classification) with a fixed feature extractor. In order to ensure that the adversary is able to classify the secondary task adequately, fading-in-steps are implemented which disable adversarial training until the fading-in interval has been reached. After the fading-in interval, the secondary task head still undergoes training with a frozen backbone, but its returned loss is also used in a separate task to update the backbone adversarially. This is achieved by either negating the gradients returned by the domain classifier loss function (gradient negation) or computing a new loss function based on the outputs of the domain classifier (confusion loss). The tertiary task is contingent on the domain classifier's

selected task; if the adversary is trained to predict race, then the third head is trained to predict sex (and vice-versa). In this project we primarily utilize a race classifier as the adversary. Similarly to the second task, training is performed using a fixed feature extractor. Therefore, the DANNs we implement perform four optimization steps [25, 43]:

1. *Optimization of the disease classifier (label predictor) and feature extractor for the main task*
2. *Optimization of the race classifier*
3. *Optimization of the sex classifier*
4. *Optimization of the backbone, based on gradient negation/confusion loss to confuse the domain classifier*

and the updated parameters can be summarized in Table 5.5.

Task	Updated Parameters
Disease (label predictor)	Disease classification head + backbone
Race/Sex (domain classifier)	Corresponding classification head
Sex/Race (not adversary)	Corresponding classification head
Gradient negation/confusion loss	Backbone

Table 5.5: **Updatable parameters DANN.**

We inspect differences between implementing confusion loss and gradient negation, as well as differences in features and performance for disease detection.

Secondly, label noise is introduced into the dataset and biased models, with ImageNet or `densenet121-res224-pc` (XRV PadChest) backbones, are purposefully generated to identify if DANNs can recover the initial performance of the model. Multitask models with partially frozen parameters are used as a baseline (Table 5.4). This procedure is described in further detail in Section 5.5.

Adversary Strength

We implement a fading-in interval of 5000 steps, which, when taking multiple optimizers into consideration results in a value of 20000 global steps. Furthermore, a fading-in-range of 800 steps is used to gradually introduce the adversary once the fading-in-steps have been reached. This employs a logistic function omega (ω) bounded by 0 and α, where α is the hyperparameter determining the adversarial strength. This can be summarized as:

$$\omega = \frac{\alpha}{1 + e^{-}(\frac{global\_step - fading\_in\_steps}{fading\_in\_range})} \tag{5.1}$$

Therefore, before the the adversary is introduced, $\omega = 0$ and after $\omega = \alpha$.

Gradient Negation

When training adversarially using negated gradients, a negative ω value is multiplied with the loss returned by the domain classifier. This essentially reverses the gradients by a factor of α. Here, the key to gradient reversal is using the *negative* ω value, thus multiplying the loss by $-1 * \alpha$.

Figure 5.1 illustrates the network structure when implementing gradient negation and a race classifier as the adversary. As depicted, the label predictor's returned loss for disease detection (L_y) propagates back through the whole network, updating parameters in the feature extractor as well. L_s represents the loss for sex classification and does not update the backbone. For the domain classifier, its loss (L_r) propagates back through the whole network, but is reversed by $-\alpha$ in the feature extractor.

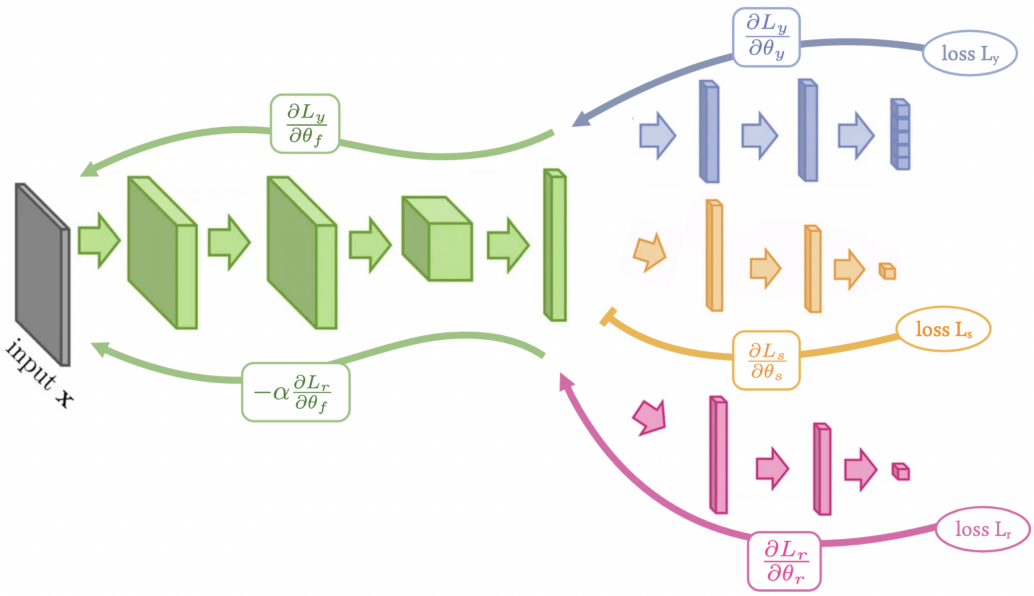


Figure 5.1: **Network structure for DANN, negated gradients.** Label predictor: disease, Domain classifier: race, Tertiary task: sex

Confusion Loss

On the other hand, confusion loss utilizes a secondary loss function which is applied to the outputs of the domain classifier. This loss function is described in Section 3.6 and is implemented in our network as follows:

$$L_{con\_r} = -average(log\_softmax(Y_r)) \tag{5.2}$$

where Y_r is the raw output for a given task. The generated loss value is then multiplied with the value for ω (thus α). This is illustrated in Figure 5.2, which has the same general structure as the gradient negation network, excluding how adversarial loss is implemented. We can see that both L_s and L_r do not affect parameters in the feature extractor.

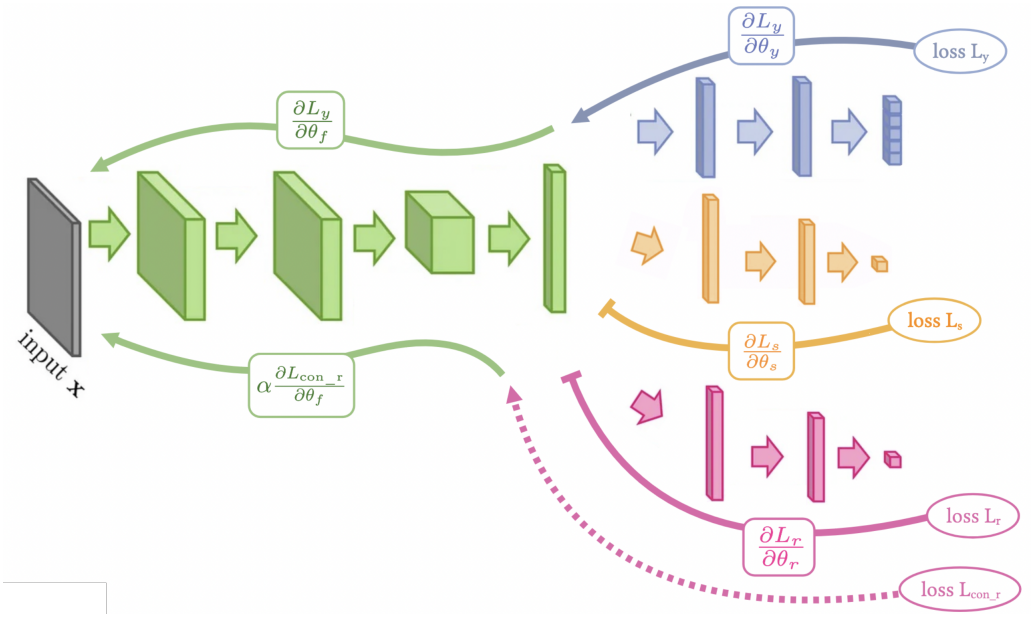


Figure 5.2: **Network structure for DANN, loss confusion.** Label predictor: disease, Domain classifier: race, Tertiary task: sex

However, L_{con_r} which is calculated from the race classification head outputs, is multiplied

with α to update the backbone parameters.

5.5 Label Noise

Noise is introduced image-by-image into the dataset. Using a binomial distribution with 2 trials, the probability of applying noise is set to 0.5. Therefore, if the returned value is 1, noise is implemented based on the given noise type: systematic or random label noise. This is applied to selected subgroups, namely, White, Black, Male and Female groups. The following models and training procedures are implemented:

Training operation	Model backbone	Bias mitigation	Noise type	(%)	Subgroup
Multitask	XRV PadChest	-	-	-	-
Multitask	XRV PadChest	-	Systematic	20	WT
Multitask	XRV PadChest	-	Systematic	20	BK
Multitask	XRV PadChest	-	Systematic	20	M
Multitask	XRV PadChest	-	Systematic	20	F
Multitask	XRV PadChest	-	Random	50	WT
Multitask	XRV PadChest	-	Random	50	BK
Multitask	XRV PadChest	-	Random	50	M
Multitask	XRV PadChest	-	Random	50	F
DANN	XRV PadChest	Race Negation	Systematic	20	WT
DANN	XRV PadChest	Race Negation	Systematic	20	BK
DANN	XRV PadChest	Race Confusion	Systematic	20	WT
DANN	XRV PadChest	Race Confusion	Systematic	20	BK
DANN	XRV PadChest	Race Negation	Random	50	WT
DANN	XRV PadChest	Race Negation	Random	50	BK
DANN	XRV PadChest	Race Confusion	Random	50	WT
DANN	XRV PadChest	Race Confusion	Random	50	BK
DANN	XRV PadChest	Sex Negation	Systematic	20	M
DANN	XRV PadChest	Sex Negation	Systematic	20	F
DANN	XRV PadChest	Sex Confusion	Systematic	20	M
DANN	XRV PadChest	Sex Confusion	Systematic	20	F
DANN	XRV PadChest	Sex Negation	Random	50	M
DANN	XRV PadChest	Sex Negation	Random	50	F
DANN	XRV PadChest	Sex Confusion	Random	50	M
DANN	XRV PadChest	Sex Confusion	Random	50	F

Table 5.6: **Implemented models for label noise investigation.** ImageNet backbones perform these operations only for racial information

5.5.1 Systematic

Systematic label noise is dependent on both race/sex and disease labels (Section 3.5). In other words, noise is applied to a certain race/sex subgroup based on the value of a selected disease label. In order to simulate underdiagnosis, the value of the first disease label in the dataset is changed. This label corresponds to the "No Finding" state, meaning if its value is 0, the sample must exhibit some arbitrary pathology. Therefore, by flipping this label to 1 and setting the remaining labels to 0, this describes a case where no pathologies are present, thus underdiagnosis. For instance:

$$[0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1]$$

$$\Downarrow$$

$$[1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]$$

5.5.2 Random

Random label noise, on the other hand, is added to a certain subgroup, without relying on a specific disease label. This involves simply choosing a random label value for each pathology in the sample. For instance:

$$[0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1]$$
$$\Downarrow$$
$$[0, 1, 0, 0, 0, 1, 1, 1, 0, 0, 1, 0, 1, 1]$$

5.6 Model selection

5.6.1 Logging

The pl `Trainer` utilizes a TensorBoard logger, which saves information logged during model development and visualizes the progression of these metrics in TensorBoard. Logging frequency, which is selected to track training loss every 5 steps, can be adjusted in the Trainer's parameters. In an adversarial model, disease, race, sex and confusion loss is logged, along with the value for ω. Validation loss is tracked during each `validation_step` at the end of every epoch. In DANNs, a modified validation loss for disease is measured, which is stored after the adversary is introduced and used as a metric for model selection. The specific values logged for each model type is illustrated in Table 5.7.

Model type	Logged values
Single task	train_loss, val_loss, test_loss, image samples
Multitask	train_loss_disease/sex/race, val_loss_disease/sex/race, test_loss_disease/sex/race
DANN	train_loss_disease/sex/race/confusion, val_loss_disease/sex/race/confusion, test_loss_disease/sex/race/confusion, val_loss_disease_mod, omega

Table 5.7: **Logged values for implemented model types.**

5.6.2 Checkpointing

The pl `Trainer` employs checkpointing methods for model selection. This is implemented using a `ModelCheckpoint` class which saves checkpoints that the trainer can then load as callbacks. For single task models and multitask training, `ModelCheckpoint` is sufficient. Here, models with the lowest disease validation loss are selected. However, for DANNs, we design a `CustomModelCheckpoint`, which only saves models after the fading-in-steps have been reached. This is to ensure that models are not selected before the adversary has been introduced into the system (since oftentimes the validation loss before adversarial training is lower than after). The `CustomModelCheckpoint` skips saving a checkpoint if the global_step is less than the fading-in-steps and monitors the modified validation loss for disease.

5.7 Optimization

All tasks implement the Adam optimization algorithm. In single and multitask models, an optimizer is utilized for each classification head, hence a single task requires one optimizer, whereas a multitask model (disease, sex, race) requires three. In contrast, the DANNs we implement utilize four optimizers, where three are responsible for updating parameters in disease, sex and race tasks and the remaining optimizer updates backbone parameters based on negated gradients/loss confusion. For each optimizer, parameters are updated by (1) clearing existing gradients

(`opt.zero_grad()`), (2) performing a backward pass using the computed loss (`loss.backward()`), and (3) taking an `opt.step()`. Before PyTorch Lightning 2.0, multiple optimizers could be regulated automatically by the pl `Trainer` and these steps were executed under the hood. However, in later versions of PyTorch Lightning, the implementation of multiple optimizers has to be regulated manually. We accomplish this by adjusting the automatic optimization processes performed under the hood. Namely, for each `training_step`, we make a list of available optimizers and then loop through each optimizer, calculating loss (`manual_backward()`) and updating parameters accordingly.

5.7.1 Optimizer configuration

Optimizers are instantiated in the `configure_optimizers` method. This includes assigning the type of optimization algorithm, and the learning rate schedule (if needed). Furthermore, in each optimizer, we pass in its updatable parameters and the initial learning rate. The parameters we update, given a specific task can be seen in Tables 5.4.2, 5.4 and 5.5. The initial learning rate used across all tasks is 0.001.

5.8 Freezing

When implementing fixed feature extractor methods, the neural network backbone is frozen. For single task models, this involves disabling updates in the backbone during training by setting `requires_grad` to false for these parameters. Thus, only the parameters in the classification head are updated.

In DANNs and some multitask models, where the backbone needs to be frozen for some tasks and unfrozen for others, we initialize parameters for each part of the network and assign them to optimizers accordingly. For instance, in a DANN with three classification tasks, (1) for predicting disease, (2) for predicting sex (with a frozen backbone), and (3) for the adversary predicting race – parameters for (1) would include disease head and backbone parameters, whereas (2) and (3) would only include the classification head parameters.

5.9 Learning rate tuning

We implement the Optuna hyperparameter tuning framework which utilizes a modified bayesian search algorithm (Optuna Search) to optimize hyperparameters. Learning rates for each network component are chosen within a range of (1e-4, 1e-1) and suggested to the Optuna search algorithm log uniformly. Optuna utilizes a `PyTorchLightningPruningCallback` which can be implemented in the trainer for early stopping. We create an Optuna study, which is able to track and visualize metrics across different learning rate trials, which is carried out for 30 trials with 10 epochs each.

5.10 Analysis

Performance analysis is employed using outputs from the `test_multi` method. This function (in a multitask setting) returns logits, predictions and targets for all classification tasks. For disease predictions, logits are passed through a sigmoid function and for race/sex predictions, logits are passed through a softmax. These are then organized and converted into a .csv file which can be used to assess performance.

We measure AUC, TPR, FPR and Youden's J Statistic to quantify performance. In order to compare and contrast subgroups, disease detection performance is determined for each subgroup of interest (White, Asian, Black, Male and Female). This involves addressing the two mutually exclusive classification labels "Pleural Effusion" and "No Finding". A target FPR of 0.20 is chosen to optimize the global threshold and determine corresponding subgroup TPR/FPRs. Youden's J statistic for each subgroup is calculated based on these TPR/FPR values. In multitask and DANN models, besides disease, we inspect the performance of racial identity subgroups in race and sex classification tasks.

Model inspection frameworks i.e., dimensionality reduction is performed using embeddings from the penultimate layer. For a single task model, these are acquired by replacing the classification head with an identity layer and then running the test set through the model. For multitask/DANN,

this involves generating a test set output using only the model backbone instead of the whole model. The generated embeddings are used to calculate the first four PCA modes and t-SNE projections.

Utilizing a random sample of 3000 patients, with 1000 patients from each racial subgroup, patient characteristics are overlayed on top of these projections and mapped into a 2-D scatter plot [10, 7]. The points in the plot can be converted into a marginal distribution which to clearly visualize the relationship between subgroups. If two distributions are statistically different (as determined by K-S test), this suggests that these features might be used to make decisions during prediction time. Indeed, in disease detection models, distribution disparities can be observed between "Pleural Effusion", "No Finding" and "Other" subgroups. Therefore, if differences are found in subgroups not related to the primary task within the same modes of variation, this may indicate that spurious information is encoded. For each PCA, t-SNE, and logit plot, we visualize disease (Pleural Effusion, No Finding, Other), race (White, Asian, Black), sex (Male, Female) and age subgroups.

Chapter 6

Evaluation

The following chapter is organized into three sections. In each section, we present the raw results of our experiments, followed by a discussion and evaluation of these results. The sections are organized as follows:

1. Performance disparities in XRV foundation models
2. Feature inspection
3. Adversarial training and label corruption

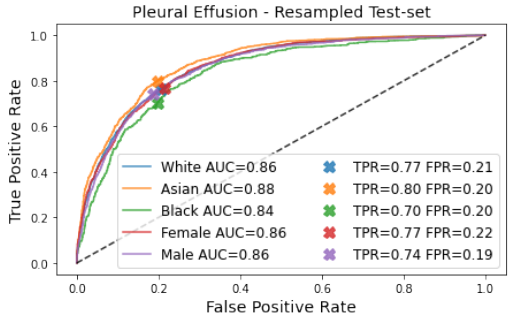
6.1 Performance disparities in foundation models

We measure performance disparities in XRV foundation models and ImageNet models as explained in 5.4.1. This involves implementing transfer learning techniques to train pre-trained models for disease detection. In particular, the fixed feature extractor method (frozen backbone) and fine-tuning method (unfrozen backbone) are utilized. Since disease detection is being monitored, we focus on the classification performance for detecting the labels "Pleural Effusion" and "No Finding". This is quantified using the performance metrics in 3.3. To calculate subgroup TPR/FPR, a global threshold is set, such that the population FPR is closest to the target FPR of 0.20. Lastly, testing is performed on a re-sampled test set as described in 4.2.2.

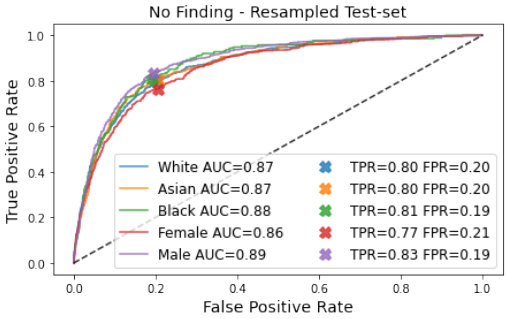
6.1.1 Results

Overall, we observe high classification accuracy for ImageNet and XRV models trained on an unfrozen backbone, with an AUC of 0.86 for "Pleural Effusion" (PE) and 0.87 for "No Finding" (NF) across models. While performance metrics do not vary greatly *between* unfrozen models, disparate performance *within* subgroups of each model can be observed. These shifts are most apparent when inspecting subgroup TPR/FPR. A detailed breakdown of performance metrics across model subgroups can be found in Appendix A (as well as Fig. 6.2 + 6.1). A performance disparity trend can be observed among models predicting for PE. Namely, Black patients tend to display lower TPRs (0.69-0.70) compared to the average rate. Let us consider the baseline ImageNet model for instance – here, Black patients exhibit a 2% decrease in AUC and 8% decrease in TPR compared to the population average. Conversely, Asian patients display a 2% and 5% increase for AUC and TPR respectively. While the remaining subgroups (White, Male, Female) largely maintain consistent levels of performance, some models demonstrate increased TPR for Female patients as well. This is summarized in Fig. 6.4 with the corresponding ROC curves shown in Fig. 6.1.

When regarding NF, an almost inverse behavior can be observed, with Black patients exhibiting higher AUC (0.88) and TPR (0.81). Additionally, gender disparities can be identified: Male patients now show increased performance (AUC 0.89, TPR 0.83) in contrast to Female patients (AUC 0.86, TPR 0.77). However, it is important to note that subgroup distribution shifts for NF are not as prominent as those present for predicting PE. The corresponding Youden's J Statistic for subgroups demonstrates similar disparities as previously described, despite corresponding changes that occur between FPR and TPR. These values are summarized in Fig. 6.4 and Fig. 6.5.

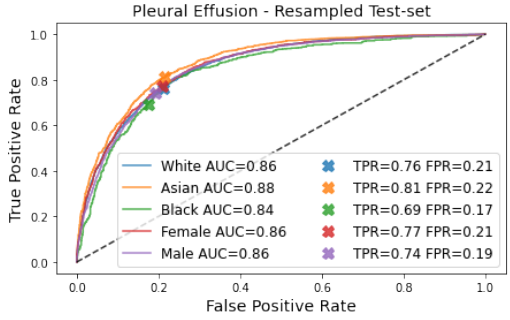


(a) ImageNet Pleural Effusion, Unfrozen

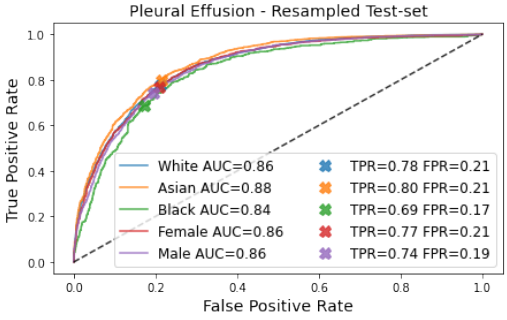


(b) ImageNet No Finding, Unfrozen

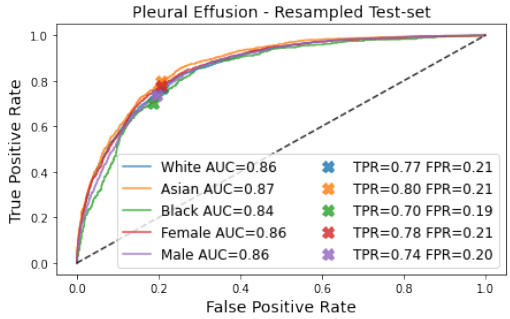
Figure 6.1: **Disease detection performance on ImageNet model with unfrozen backbone.** ROC curves for the detection of "Pleural Effusion" and "No Finding" with an ImageNet DenseNet-121 trained on CheXpert and evaluated on the re-sampled test set.



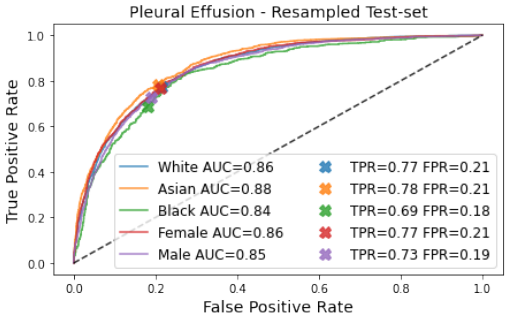
(a) RSNA Pleural Effusion, Unfrozen



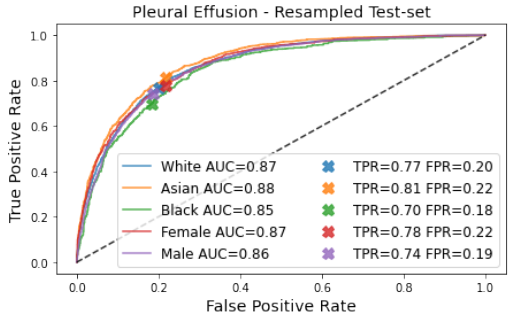
(b) MIMIC_CH Pleural Effusion, Unfrozen



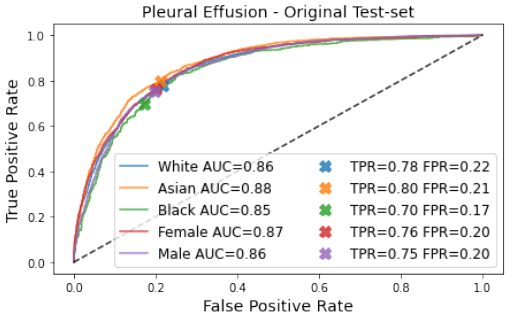
(c) MIMIC_NB Pleural Effusion, Unfrozen



(d) NIH Pleural Effusion, Unfrozen



(e) PadChest Pleural Effusion, Unfrozen

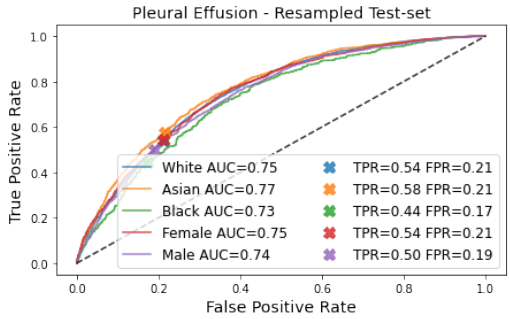


(f) All Pleural Effusion, Unfrozen

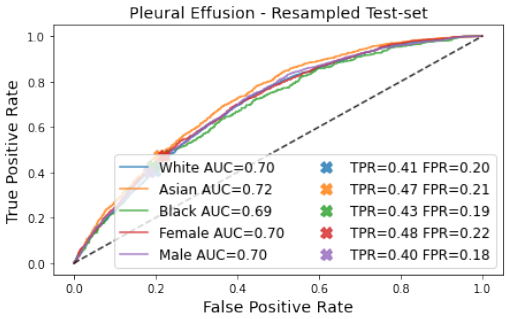
Figure 6.2: **Disease Detection performance on XRV foundation models with unfrozen backbones.** ROC curves for the detection of "Pleural Effusion" with an XRV core classifier re-trained on CheXpert and evaluated on the re-sampled test set. Similar TPR/FPR shifts compared to ImageNet can be observed.

Inspection of the foundation models shows mostly analogous behavior to ImageNet, with similar contrasts between Black and Asian subgroups, as well as Male and Female subgroups. However,

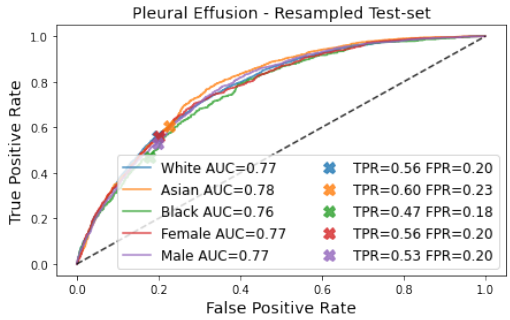
some FPR differences can be observed as well, where XRV models show a decreased FPR for Black patients and increased FPR in Asian patients (for PE).



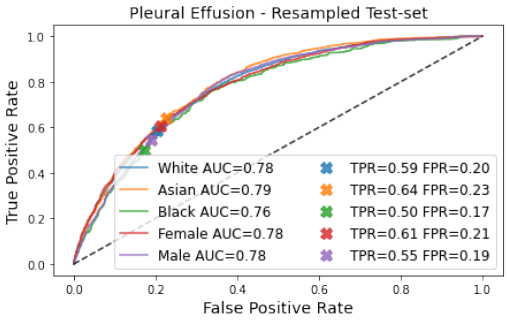
(a) ImageNet Pleural Effusion, Frozen



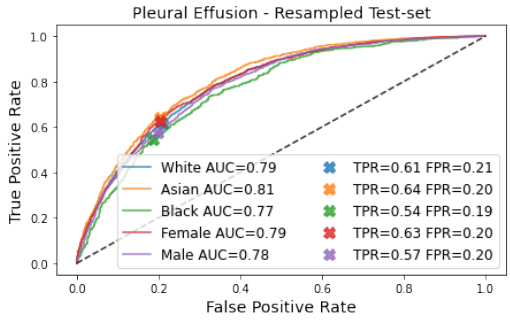
(b) RSNA Pleural Effusion, Frozen



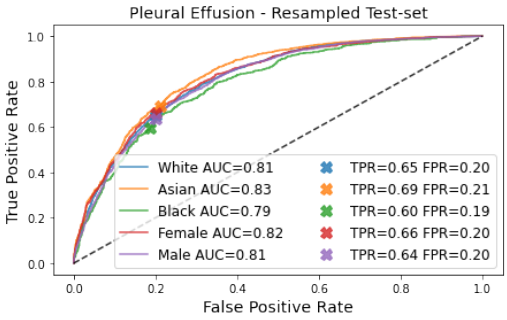
(c) MIMIC_CH Pleural Effusion, Frozen



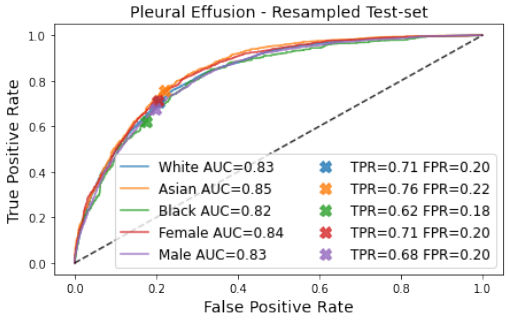
(d) MIMIC_NB Pleural Effusion, Frozen



(e) NIH Pleural Effusion, Frozen



(f) PadChest Pleural Effusion, Frozen



(g) All Pleural Effusion, Frozen

Figure 6.3: **Disease detection performance on XRV and ImageNet models with frozen backbones.** ROC curves for the detection of "Pleural Effusion" with an ImageNet backbone (a) and XRV core classifiers (b-g) re-trained on CheXpert and evaluated on the re-sampled test set. Flatter ROC curves are depicted compared to unfrozen backbones, however similar TPR/FPR shifts can be observed.

Conversely, the performance in models trained with frozen backbones differs on both a model and subgroup level. The `densenet121-res224-rsna` model displays the lowest performance for the

labels PE (AUC 0.70, Youden's 0.24) and NF (AUC 0.79, Youden's 0.44). In contrast, the model with the highest performance is the `densenet121-res224-all` model with an AUC of 0.83 and 0.87 for PE and NF respectively (Youden's: 0.5, 0.60). Other core XRV models report performance levels within this range. Visually, we observe flatter ROC curves for all frozen models, which corresponds to a subsequent decrease in AUC. While frozen and unfrozen models diverge in terms of how model performance differs, the relationship between subgroups and observed subgroup disparities remain similar.

This is summarized in Fig. 6.4 where we see a clear underperformance for Black subgroups and increased performance for Asian subgroups. The frozen RSNA model is an outlier, exhibiting a considerably decreased Youden's J Statistic for White patients (0.21) as well. Despite NF subgroup differences not being as clear-cut, we do observe overall increases in Black and Male subgroup performance, and decreases in Female performance, similar to what is observed with frozen backbones.

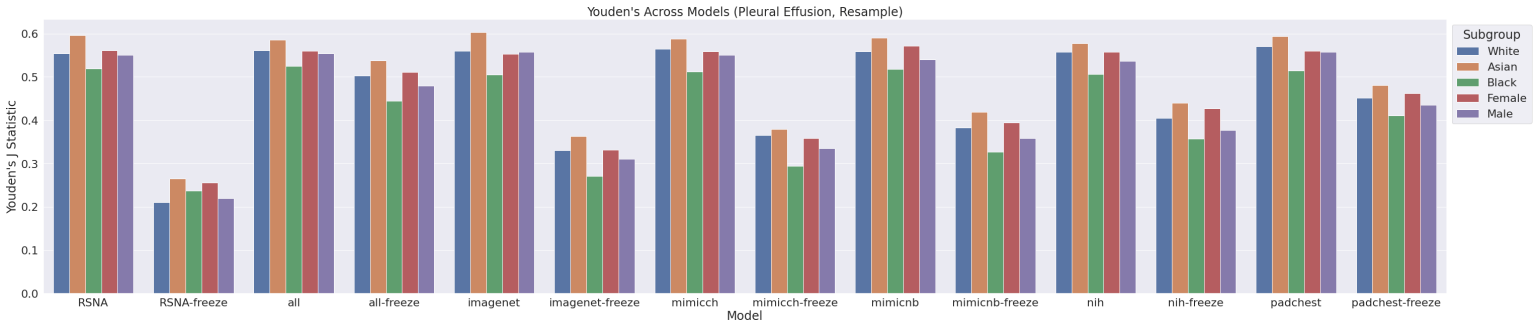


Figure 6.4: **Summary of Youden's J Statistic across models, Pleural Effusion, Resampled test set.** Models trained with a frozen backbone are labeled with "-freeze". Overall, they display decreased performance compared to unfrozen models. Disparities between subgroups are similar across all models, with differences between Black and Asian subgroups being the most prominent.

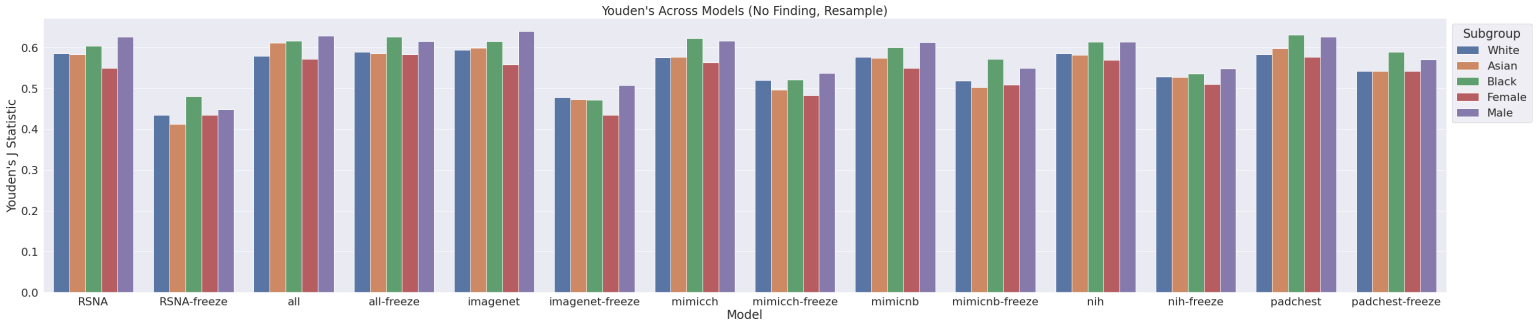


Figure 6.5: **Summary of Youden's J Statistic across models, Pleural Effusion, Resampled test set.** Models trained with a frozen backbone are labeled with "-freeze". Overall, they display decreased performance compared to unfrozen models. Disparities between subgroups are similar across all models, with Black and Male subgroups exhibiting the highest performance.

6.1.2 Discussion

The objective of this set of experiments was to build upon the CXR foundation model study conducted by Glocker et al. [10]. They implemented a model that was not publicly available, meaning a fixed feature extractor method had to be employed for classification head re-training. Therefore, parameters in the backbone could not be updated, and any biases encoded during pre-training could not be overridden, potentially causing them to affect downstream predictions. By inspecting the behavior of foundation models (XRV) with frozen *and* unfrozen backbones, we might be able to discern how biases introduced during pre-training affect downstream predictions in

unfrozen conditions. However, this does not necessarily provide conclusions on whether protected identity information and disease detection are spuriously correlated.

Clear performance disparities between frozen and unfrozen models can be observed, where frozen models consistently display a decreased Youden's J Statistic, elucidated in Fig. 6.6. This behavior is expected, since generated features remain unchanged from pre-training, hindering adaptation to the new domain. Therefore, even though the XRV foundation models are pre-trained for disease, they still exhibit discrepancies, since (1) the implemented datasets differ, (2) the expected output classes differ and (3) the backbone cannot be updated to correct for these differences.

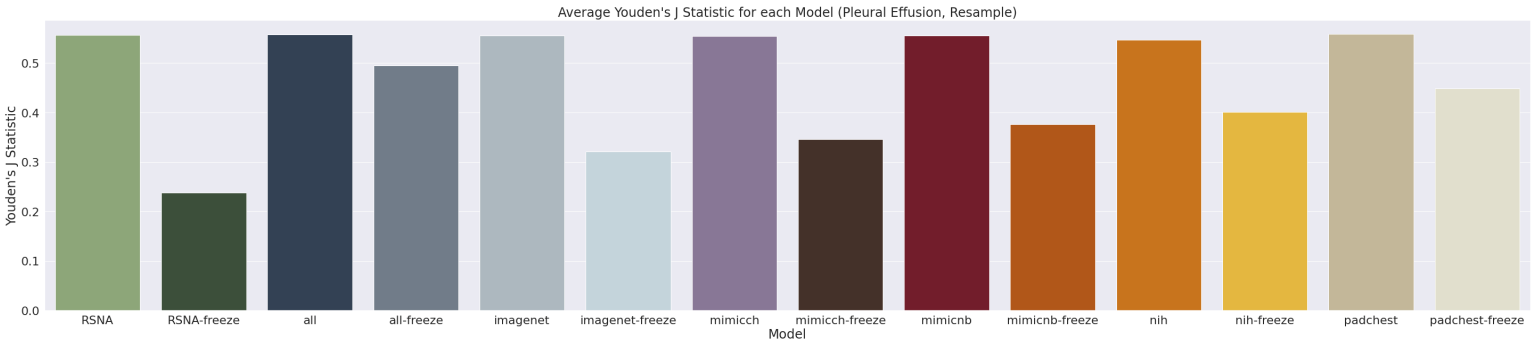


Figure 6.6: **Average Youden's J Statistic across frozen and unfrozen models for Pleural Effusion.**

However, it is also important to note that XRV models *were* trained on a domain related dataset (chest X-rays) and *do* have a number of overlapping labels with CheXpert. These include Atelectasis, Consolidation, Pneumothorax, Edema, Effusion, Pneumonia, Cardiomegaly, Lung Lesion, Fracture, Lung Opacity, and Enlarged Cardio. Therefore, relevant features for disease classification (albeit not all) can be utilized and we observe certain models (XRV All and XRV PadChest) that achieve comparable performance to unfrozen models. Interestingly, increases in performance seem to correspond with the number of output classes a model was pre-trained to predict. A potential explanation for this behavior is that, in learning to detect more pathologies, more applicable features are generated – for disease detection in general, as well as for specific overlapping pathologies. For instance, the worst-performing model, RSNA, is only trained to predict 2 pathologies (Lung Opacity, Pneumonia), whereas the best-performing model is trained to output 18 classes (`densenet121-res224-all`). This behavior is illustrated in Table 6.1.

Model	Number of output classes	Youden's J Statistic (Frozen, PE)
RSNA	2	0.24
ImageNet	-	0.32
Mimic_ch	11	0.35
Mimic_nb	11	0.38
NIH	14	0.40
PadChest	15	0.45
All	18	0.50

Table 6.1: **Pre-trained number of output classes and corresponding Youden's J Statistic.**

The elevated performance for frozen XRV All could be attributed to fact that it was pre-trained on a variety of datasets. Therefore, the model might have learned a wide range of features that could be employed for disease detection. However, one could also argue that since CheXpert is one of the datasets included during pre-training, this model is overfitting, since our implemented test set is taken from the original training set. Whether these probable causes are the reason for this behavior is ambiguous, therefore let us consider the XRV PadChest to be the highest performing model.

It is important to note that these performance disparities between models mostly apply to models trained with a frozen backbone. If an unfrozen backbone is implemented, these variations are corrected for, as illustrated in Fig. 6.7. However, this figure further elucidates that, while the

performance between unfrozen models stays consistent, certain subgroups in each model perform differently. If we look at the percentage difference from the average Youden's for each subgroup, we notice an interesting trend between frozen and unfrozen backbones. Particularly, while frozen backbones have more pronounced disparities compared to unfrozen backbones, all disparities point in a similar direction.

For example, if a frozen model displays a decreased performance for Male subgroups, the corresponding unfrozen model also displays a decreased Male performance (though not as large of a decrease). Analyzing the subgroups with the highest differences for predicting the PE label, Asian patients average a 10% difference for frozen backbones and 6% difference for unfrozen; a 40% decrease between the two. In contrast, the Black subgroup averages an -11% difference for frozen backbones and -9% difference for unfrozen; an 18% decrease. Overall, decreases in variance whilst maintaining subgroup performance relationships, can be observed across all models and subgroups when comparing frozen to unfrozen models. Therefore, this suggests that biases in the backbone might not be necessarily overridden during fine-tuning and could potentially affect future predictions.

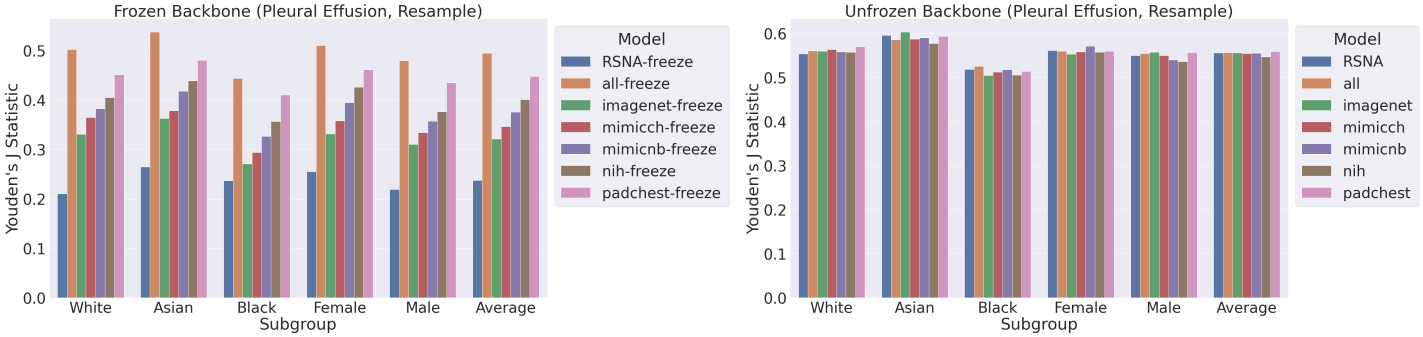


Figure 6.7: **Youden's J Statistic for frozen and unfrozen backbones across all subgroups.**

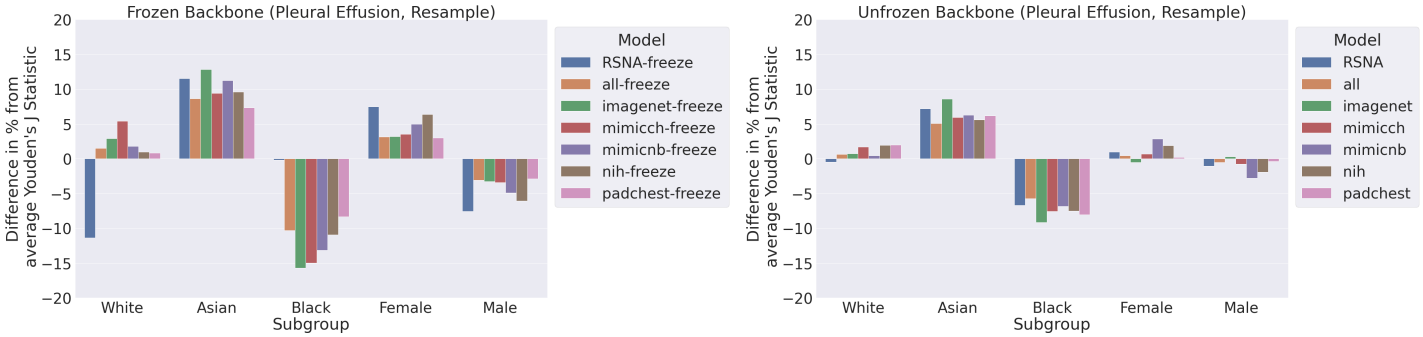


Figure 6.8: **Percentage difference from the average Youden's J Statistic across all subgroups.**

The TPR/FPR disparities we observe indicate that particular subgroups generalize poorly based on the chosen population threshold and suggest that certain predictions could be biased. Thus, despite implementing an unfrozen backbone during training, the performance disparities which existed in a frozen model are still present. However, whether models use this information to make decisions when classifying disease needs to be further inspected. Therefore, we investigate the relationship between disease classification and other subgroups by implementing feature exploration frameworks described in 3.2.

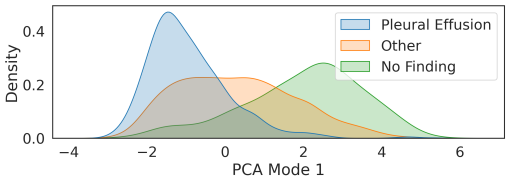
6.2 Model inspection

Feature representations are inspected by implementing dimensionality reduction techniques to summarize information present in the penultimate layer of a model. By overlaying patient subgroup metadata over the extracted modes, this allows us to capture relationships and potential biases between subgroups. The primary method we employ is Principal Component Analysis or PCA.

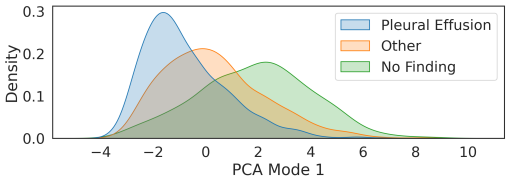
This is performed as described in Sections 5.10 and 3.2. Additionally, marginal distributions are normalized for improved visualization.

6.2.1 Results

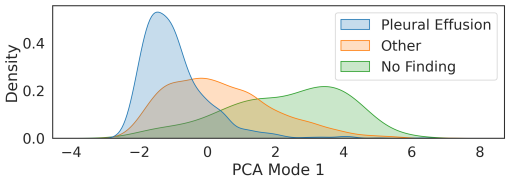
In all XRV models, we observe clear separations in marginal distributions for disease labels in the first mode of PCA. This indicates that features were generated to discriminate between the selected disease classes; namely, "Pleural Effusion", "Other" and "No Finding" [7]. Furthermore, models with frozen backbones show a separation of disease labels along the largest mode of variation, signifying that pre-training was conducted on similar (chest X-ray) datasets. However, visually, we can see less distinct separations between PE and NF for frozen compared to unfrozen models. This could explain the decreased performance and flatter ROC curves present when classifying disease.



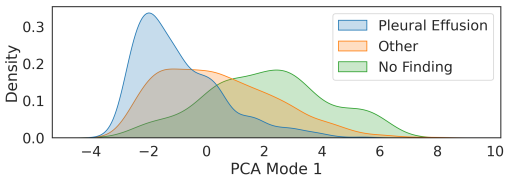
(a) PCA 1 Mimic_ch, disease labels, unfrozen



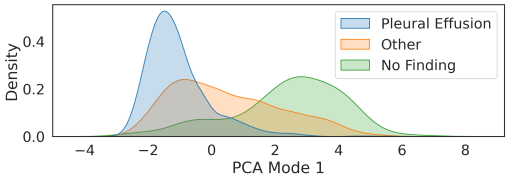
(b) PCA 1 Mimic_ch, disease labels, frozen



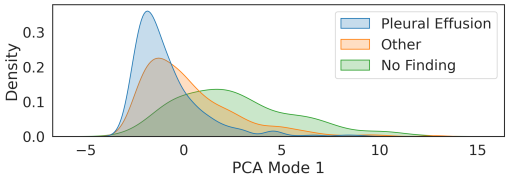
(c) PCA 1 Mimic_nb, disease labels, unfrozen



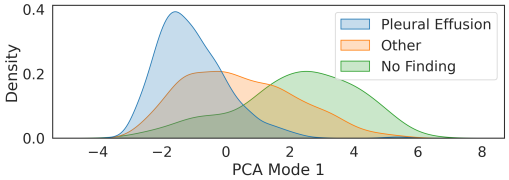
(d) PCA 1 Mimic_nb, disease labels, frozen



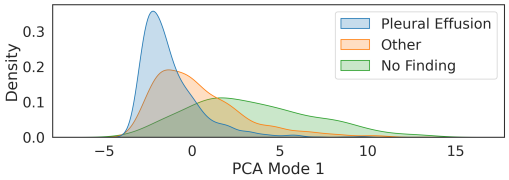
(e) PCA 1 NIH, disease labels, unfrozen



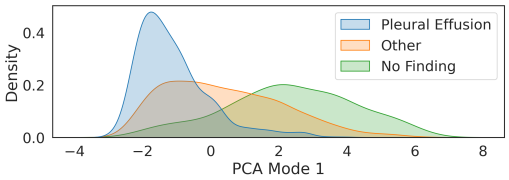
(f) PCA 1 NIH, disease labels, frozen



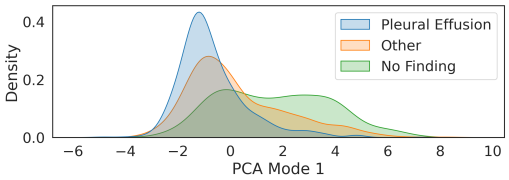
(g) PCA 1 PadChest, disease labels, unfrozen



(h) PCA 1 PadChest, disease labels, frozen



(i) PCA 1 RSNA, disease labels, unfrozen



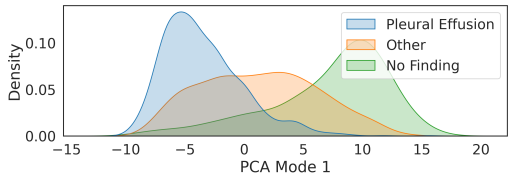
(j) PCA 1 RSNA, disease labels, frozen

Figure 6.9: **First PCA mode of variation for disease detection in XRV models.** Since backbone parameters are not updated when the model is frozen, the extracted embeddings are indicative of the dataset it was pre-trained on. On the other hand, fine-tuning changes the feature representations in the backbone.

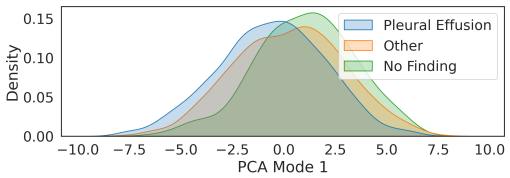
In contrast, while the unfrozen ImageNet model displays similar disease label separations to XRV, the frozen model contains more ambiguous features. We observe minimal separation for discriminating disease, as well as small shifts within protected characteristic subgroups (6.10 b, d,

f). Training on an unfrozen backbone seems to mitigate these shifts, which is elucidated in the evenly distributed subgroups for racial identity and biological sex (6.10 c, e).

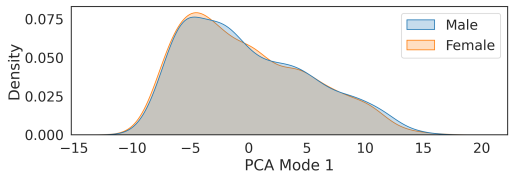
By investigating different subgroup distributions across the four PCA modes, potential correlations can be observed. In order for these to occur, distribution shifts within subgroups must be significant *and* must be within the same mode of variation as the primary disease detection task. Therefore, we primarily scrutinize variations within the first PCA mode to test for spurious correlations; however, we analyze the remaining modes comprehensively, since shifts could still indicate the presence of bias (and correspondingly affect performance). Furthermore, logit space can be scrutinized to detect potential biases between certain subgroups and output predictions.



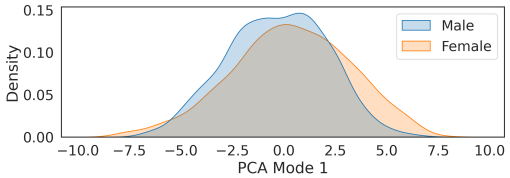
(a) PCA 1 ImageNet, disease labels, unfrozen



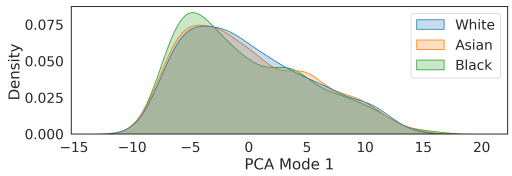
(b) PCA 1 ImageNet, disease labels, frozen



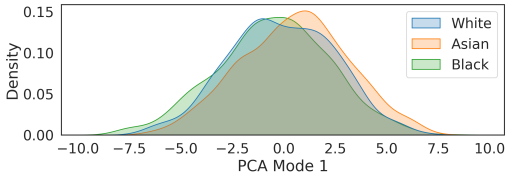
(c) PCA 1 ImageNet, sex labels, unfrozen



(d) PCA 1 ImageNet, sex labels, frozen



(e) PCA 1 ImageNet, race labels, unfrozen



(f) PCA 1 ImageNet, race labels, frozen

Figure 6.10: **ImageNet PCA mode 1 for disease, sex, and race labels, frozen and unfrozen models.** We can see that fine-tuning the backbone allows clearer separations for disease to be formed, whilst evening out any separations present in sex and racial subgroups from pretraining. The shape of the features in the frozen ImageNet model, differs from the representations in the frozen XRV models.

For unfrozen backbones, shifts in marginal distributions unrelated to the primary task are less evident and less prevalent compared to frozen backbones. In ImageNet models for example, we observe significant shifts between NF and PE labels (which is to be expected since the model is trained for disease). However, upon further inspection, unfrozen models contain no additional significant differences between subgroup distributions in PCA 1. In contrast, frozen models display significant shifts between AS/WT, AS/BK and M/F groups, suggesting some correlation between these tasks and disease detection. Furthermore, observing higher PCA modes, an increased number of significant shifts for all subgroups can be identified. In particular, the M/F, AS/BK and AS/WT distributions display differences in PCA modes 1, 2 + 4. These relationships can be further visualized in 2-D PCA scatter plots.

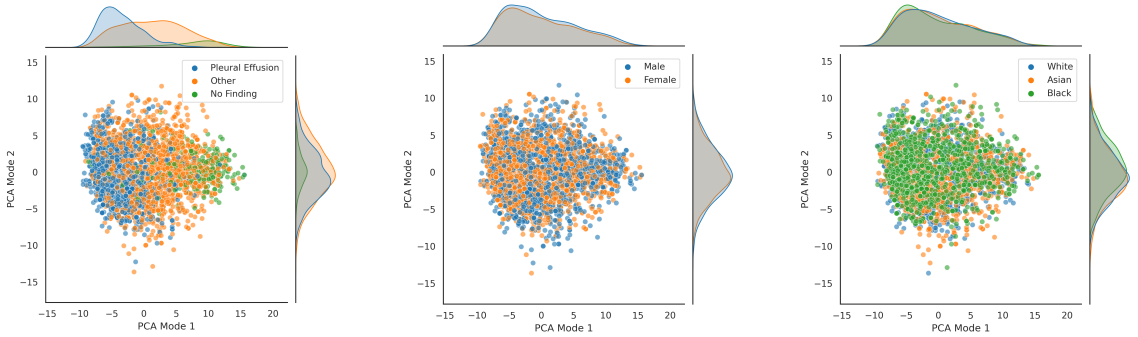
Frozen XRV models display significant differences in the first PCA mode for AS/BK subgroups, suggesting a weak correlation between racial information and disease (Fig. 6.13). In contrast, unfrozen XRV models (Fig. 6.12) do not display any shifts within racial identity or biological sex. Therefore, we can speculate that the spurious correlations present in frozen models are 'unlearned' when updating parameters in the backbone.

However, it is important to note that similar shifts are observed in higher PCA modes. In particular, racial information seems to be encoded in PCA 2 for unfrozen models, where differences between AS/BK subgroups can be observed. Therefore, while the model might not rely on racial information to form predictions for disease, these shifts can still contribute to performance disparities, as we see between Black and Asian subgroups in 6.1. Concernedly, this shows that

models might encode underlying racial information during training, despite being trained solely for detecting disease.

Model	Mode	NF/PE	WT/AS	AS/BK	BK/WT	M/F
ImageNet	PCA 1	<0.0001*	1.00	0.68	0.19	1.00
	PCA 2	<0.0001*	1.00	<0.0001*	<0.0001*	1.00
	PCA 3	<0.0001*	0.016*	0.0006*	1.00	0.098
	PCA 4	0.0052	0.32	0.033*	1.00	0.001*
ImageNet (Frozen)	PCA 1	<0.0001*	<0.0001*	<0.0001*	0.07	<0.0001*
	PCA 2	<0.0001*	<0.0001	0.0002*	1.00	<0.0001*
	PCA 3	<0.0001*	1.00	1.00	0.43	0.076
	PCA 4	<0.0001*	<0.0001*	<0.0001*	0.009*	0.053

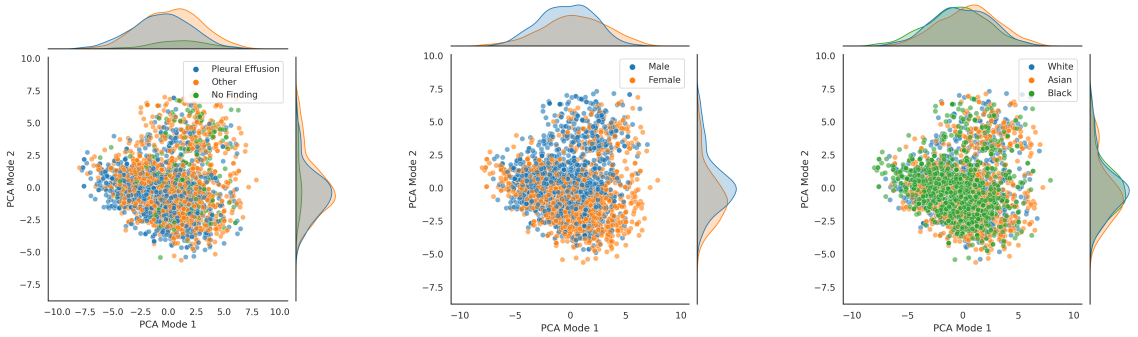
Table 6.2: **Kolmogorov-Smirnov Test.** Comparison of marginal distributions between subgroups in disease, race and sex for all 4 PCA modes. Statistically significant differences are marked with * next to their p-value, rejecting the null hypothesis. NF: No Finding, PE: Pleural Effusion, WT: White, AS: Asian, BK: Black, M: Male, F: Female. P-values are significant if $p<0.05$ or $p<0.001$



(a) PCA 1 + 2 ImageNet, disease labels, unfrozen

(b) PCA 1 + 2 ImageNet, sex labels, unfrozen

(c) PCA 1 + 2 ImageNet, race labels, unfrozen



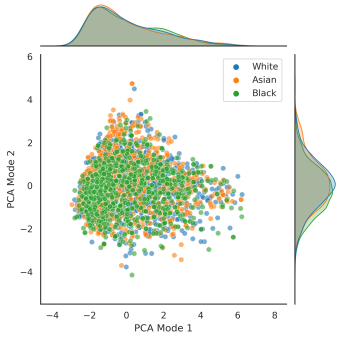
(d) PCA 1 + 2 ImageNet, disease labels, frozen

(e) PCA 1 + 2 ImageNet, sex labels, frozen

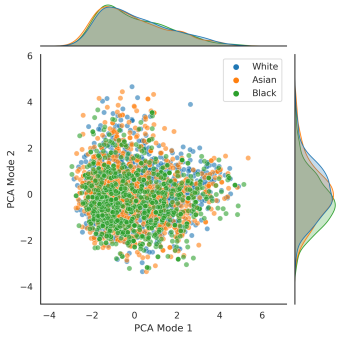
(f) PCA 1 + 2 ImageNet, race labels, frozen

Figure 6.11: **PCA Mode 1 + 2 Scatter plot ImageNet frozen + unfrozen.** The density of points in the scatter plot are summarized in corresponding marginal distributions. In (a), we can see slight clustering between disease labels which is shown to be statistically significant (K-S test), indicating the ability to discriminate between these labels for classification tasks. PCA plots for frozen models (d-f) also visualize differences between subgroups computed in the K-S test. Minor separations between M/F and WT/AS, AS/BK can be seen.

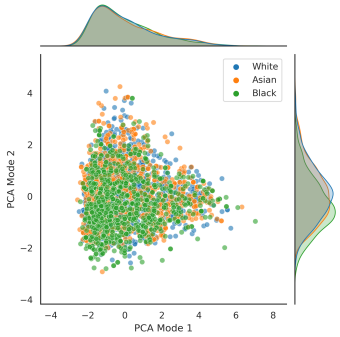
Similarly to ImageNet models, frozen XRV backbones exhibit an increased amount of distribution shifts across PCA modes. Outliers are the XRV PadChest models, where the frozen model contains the same number of shifts and does not exhibit differences in protected attributes for the largest mode of variation. Observing the computed p-values, both PadChest models have 11/20 comparisons rejecting the null hypothesis.



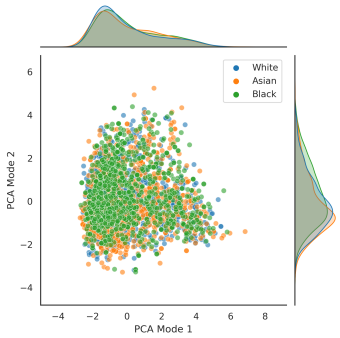
(a) PCA 1 + 2 RSNA, race labels, unfrozen



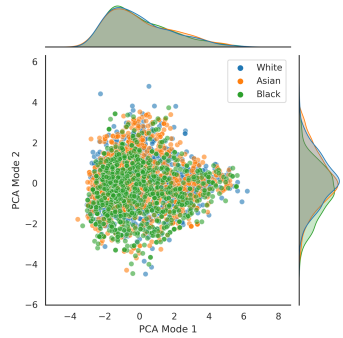
(b) PCA 1 + 2 Mimic_ch, race labels, unfrozen



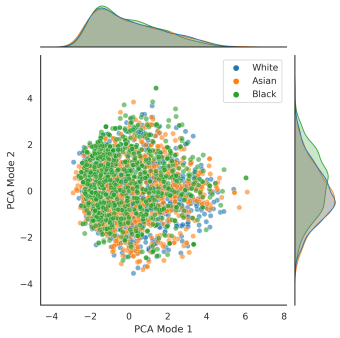
(c) PCA 1 + 2 Mimic_nb, race labels, unfrozen



(d) PCA 1 + 2 NIH, race labels, unfrozen

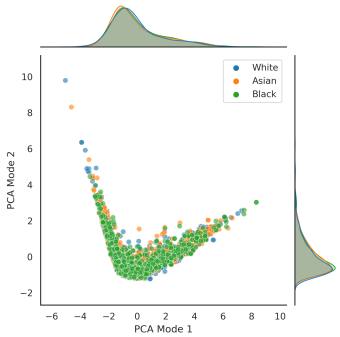


(e) PCA 1 + 2 PadChest, race labels, unfrozen

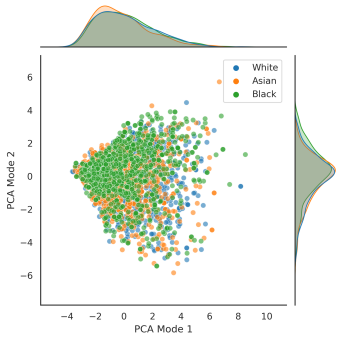


(f) PCA 1 + 2 All, race labels, unfrozen

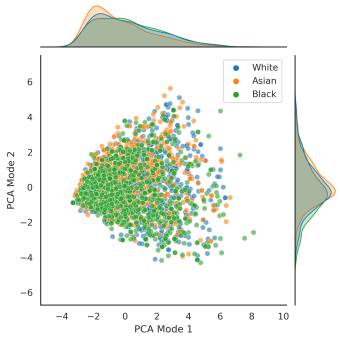
Figure 6.12: **PCA Mode 1 + 2 Scatter plot XRV unfrozen, racial information.**



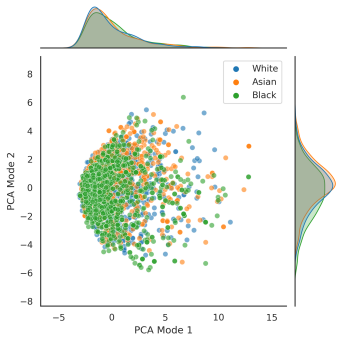
(a) PCA 1 + 2 RSNA, race labels, frozen



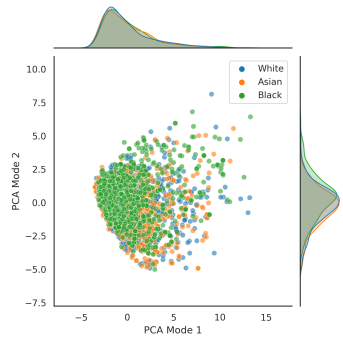
(b) PCA 1 + 2 Mimic_ch, race labels, frozen



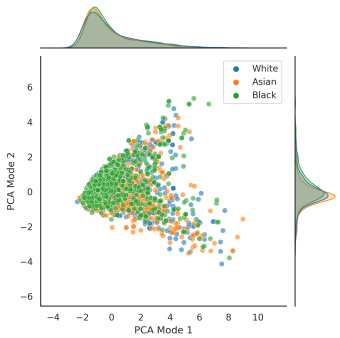
(c) PCA 1 + 2 Mimic_nb, race labels, frozen



(d) PCA 1 + 2 NIH, race labels, frozen



(e) PCA 1 + 2 PadChest, race labels, frozen



(f) PCA 1 + 2 All, race labels, frozen

Figure 6.13: **PCA Mode 1 + 2 Scatter plot XRV frozen, racial information.**

6.2.2 Discussion

Since XRV models were trained for disease detection on domain relevant data, the frozen and unfrozen backbones display similar distribution shifts for discriminating disease labels (Fig. 6.9). These shifts are observed in the largest mode of variation: PCA 1. When adopting an unfrozen backbone, the overall shape of subgroup distributions changes such that PE and NF have less of an overlap and can distinguish between these labels more effectively. This explains the corresponding reduction in disease detection performance for frozen models. These differences in feature representations can be observed across all models.

In contrast, ImageNet models, i.e., frozen ImageNet models do not display large separations between disease classes. This can be attributed to pre-training on ImageNet-1K, since this consists of natural images and not domain-specific data. Therefore, features for distinguishing chest X-ray pathologies were not learned. However, minor shifts between disease, race and sex subgroups are still notable, most likely a result of learning a wide range of general features.

Frozen models display significant shifts in marginal distributions for race and sex subgroups in PCA 1, meaning that these features might be discriminated for when predicting disease. Therefore, spurious correlations between unrelated subgroups exist, which can diminish performance in the primary task and propagate existing disparities. The significance values between all subgroups in PCA 1 are illustrated in Table 6.3. In line with our expectations, each model has a significant shift for NF/PE. In addition to this however, a majority (66%) of frozen models display significant shifts between AS/BK and M/F, validating the subgroup disparities we observe in 6.1.

Model	NF/PE	WT/AS	AS/BK	BK/WT	M/F
ImageNet	<0.0001*	1.00	0.68	0.19	1.00
ImageNet (Frozen)	<0.0001*	<0.0001*	<0.0001*	0.07	<0.0001*
Mimic_ch	<0.0001*	0.68	1.00	1.00	1.00
Mimic_ch (Frozen)	<0.0001*	0.12	0.007*	1.00	0.016*
Mimic_nb	<0.0001*	1.00	1.00	1.00	1.00
Mimic_nb (Frozen)	<0.0001*	0.0002*	<0.0001*	0.58	0.014*
NIH	<0.0001*	0.20	0.71	0.16	1.00
NIH (Frozen)	<0.0001*	1.00	0.037*	0.009*	0.058
PadChest	<0.0001*	1.00	1.00	1.00	1.00
PadChest (Frozen)	<0.0001*	1.00	0.39	0.64	1.00
RSNA	<0.0001*	1.00	0.60	1.00	1.00
RSNA (Frozen)	<0.0001*	0.52	0.48	1.00	<0.0001*

Table 6.3: **p-values across models for all subgroup comparisons in PCA 1.**

The increased frequency of distribution shifts observed in frozen models, can be attributed to the fact that generated features in the backbone cannot be updated. Since potential undesirable correlations present in the training data cannot be overridden, models are more likely to form shortcuts for making predictions [10].

On the other hand, updatable backbone parameters allow spurious connections between disease detection and protected characteristics to be corrected for. Thus, we observe biases which exist initially in frozen models, that are mitigated during fine-tuning. This can be seen in Table 6.3, which show spurious connections in AS/BK and M/F subgroups for frozen models, yet no significant distribution shifts for unfrozen models. However, as noted previously, shifts present in frozen models can be observed in higher PCA modes for unfrozen models (PCA 2, AS/BK). While these shifts do not play a direct role in determining disease predictions, predictive disparities could arise as shown in Fig. 6.4.

Referring back to the CXR foundation model investigated by Glocker et al.: this confirms their speculation that the model likely exhibited biased correlations between subgroups due to its implementation as a frozen backbone. We show that adopting fine-tuning methods can correct these existing correlations, but may contain discrepancies in higher PCA modes that can affect the overall disease detection performance in certain subgroups. In summary, while *frozen* backbones exhibit spurious correlations, *unfrozen* foundation models do not show compelling evidence that protected characteristics are explicitly used for disease detection tasks. In order to draw further conclusions, additional foundation models with differing architectures (i.e., ResNet, ViT) and pre-training methods (i.e., REMEDIS [8]) should be investigated.

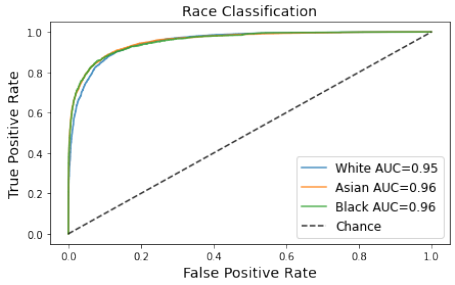
6.3 Adversarial training and label corruption

In order to adopt a more systematic approach for bias analysis, noise can be injected into specific subgroups in the dataset to generate an intentionally biased model. This simulates annotation bias which can present itself in medical datasets as a result of systematic mis- or underdiagnosis catalyzed by healthcare disparities. By generating a model known to be biased, we can more effectively test subgroup analysis frameworks and bias mitigation techniques. Furthermore, since bias is being introduced artificially and therefore the source of bias is known, it can be more finely controlled and targeted. Bias mitigation is performed using domain adversarial networks as explained in 5.4.3 and 5.5.

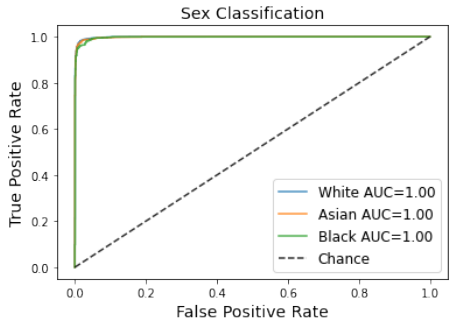
6.3.1 Results

Multitask

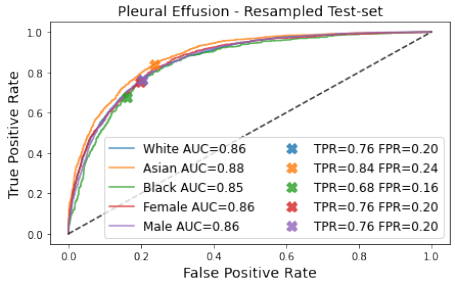
A multitask model with partially frozen features is utilized as the baseline comparison for DANNs. This involves training a multi-classification head for disease, sex, and race, whilst keeping the backbone frozen for sex and race tasks. The backbones we implement for multitask and adversarial training are ImageNet and XRV PadChest. Firstly, let us inspect the *initial* behavior of a multitask model where backbone updates are provided during all three classification tasks.



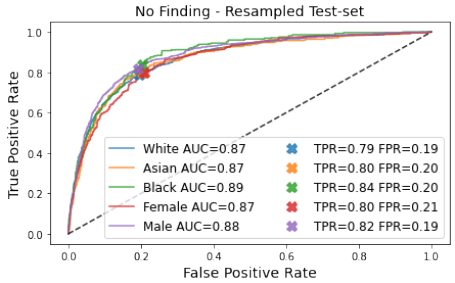
(a) Race classification multitask, ImageNet, unfrozen BB



(b) Sex classification multitask, ImageNet, unfrozen BB



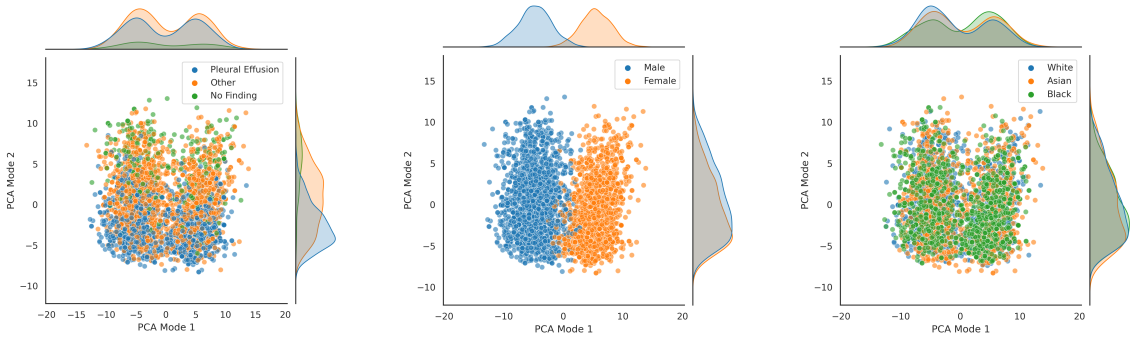
(c) Disease classification multitask, ImageNet, PE unfrozen BB



(d) Disease classification multitask, ImageNet, NF unfrozen BB

Figure 6.14: **Disease, race, and sex classification performance for a multitask model, ImageNet unfrozen BB.**

In this model, unreasonably high performance between racial subgroups for sex and race classification can be observed, reporting an AUC of 0.96 for race classification and 1.00 for sex classification. Disease detection performance is similar to a single task model's, exhibiting comparable disparities between subgroups, such as AS/BK for PE (Fig. 6.14).



(a) PCA 1 + 2 ImageNet multitask, disease labels, unfrozen

(b) PCA 1 + 2 ImageNet multitask, sex labels, unfrozen

(c) PCA 1 + 2 ImageNet multitask, race labels, unfrozen

Figure 6.15: **Multitask feature inspection, ImageNet unfrozen BB**

Unsupervised feature exploration shows clear clustering between biological sex in the first PCA mode, as well as significant shifts in racial distributions. However, even though these shifts are significantly discernible in PCA 1, race seems to be primarily encoded in modes 3 + 4 (Fig. 6.16). Disease detection seems to have migrated to the second PCA mode, as elucidated in Fig. 6.17 (a).

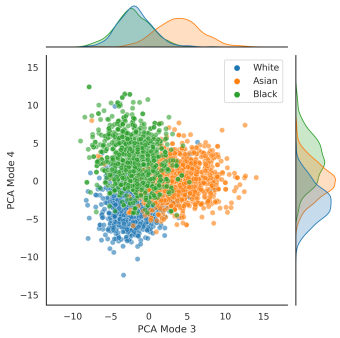
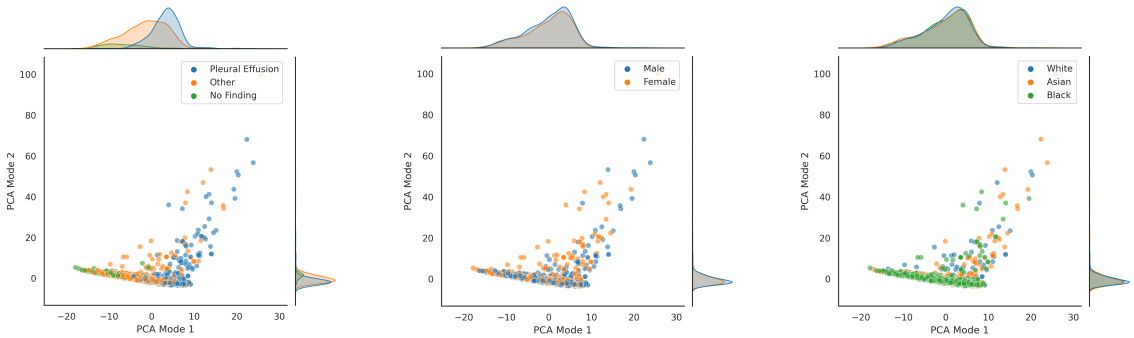


Figure 6.16: **PCA 3 + 4 ImageNet multitask, race labels, unfrozen.**

Compare this to an ImageNet multitask model developed with frozen sex and race classification heads (DANN baseline, ImageNet BB):



(a) PCA 1 + 2 ImageNet multitask, disease labels, unfrozen

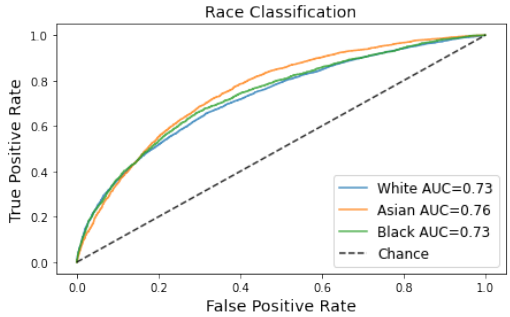
(b) PCA 1 + 2 ImageNet multitask, sex labels, frozen

(c) PCA 1 + 2 ImageNet multitask, race labels, frozen

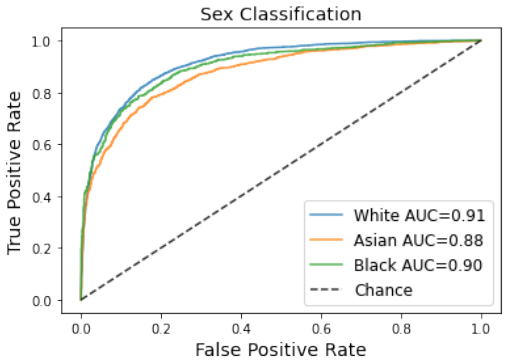
Figure 6.17: **Multitask feature inspection, ImageNet frozen BB**

Feature inspection shows significant shifts for disease in PCA 1, but no shifts for sex or race in

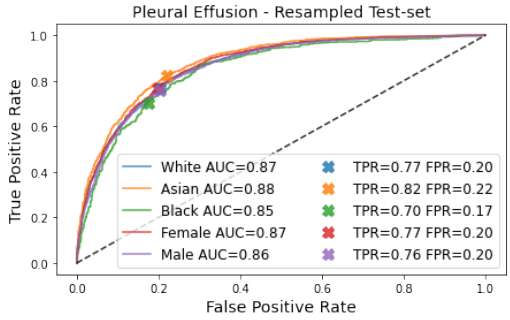
modes 1 + 2. It is worth noting however, that the dispersion of plot points is irregular and highly conglomerated along PCA 2 for all investigated labels.



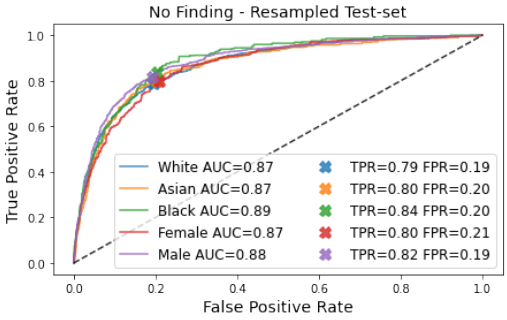
(a) Race classification multitask, ImageNet, frozen BB



(b) Sex classification multitask, ImageNet, frozen BB



(c) Disease classification multitask, ImageNet, PE unfrozen BB

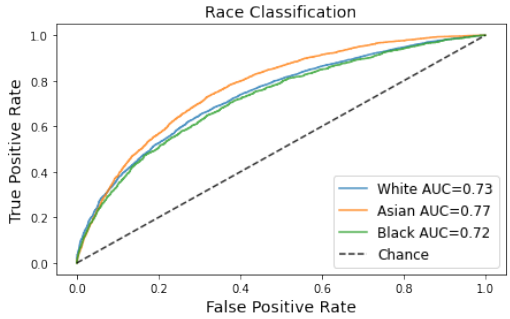


(d) Disease classification multitask, ImageNet, NF unfrozen BB

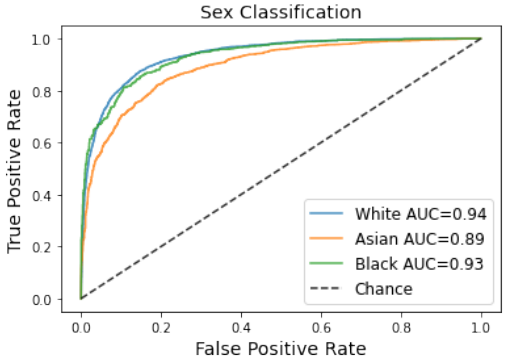
Figure 6.18: **Disease, race, and sex classification performance for a multitask model, ImageNet frozen BB.**

The frozen multitask model maintains a relatively consistent level of disease detection performance compared to the original multitask model, while the performance in classifying race and sex differs. However, even though backbone features have not been adapted specifically for these tasks, race and sex classification still report a relatively high AUC of 0.74 and 0.90 respectively. Comparable gaps in performance between AS/BK can be observed for PE, while other subgroups mostly show equal values of TPR/FPR.

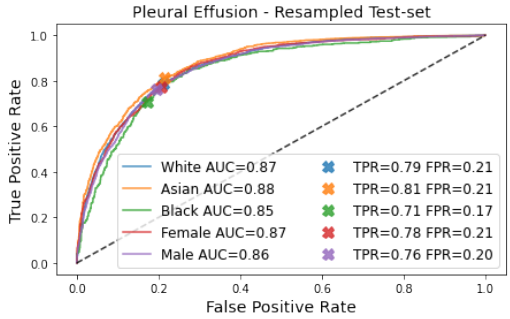
Subsequently, the XRV PadChest multitask model developed with partially frozen parameters (DANN baseline, PadChest BB), displays similar ROC curves as the ImageNet multitask model. This is characterized by a slight decrease in performance when it comes to classifying sex and race (Fig. 6.14), whilst maintaining consistent classification performance for disease.



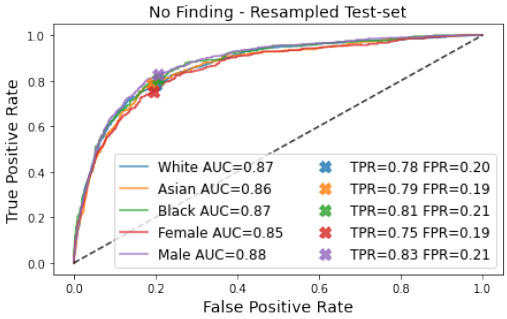
(a) Race classification multitask, PadChest, frozen BB



(b) Sex classification multitask, PadChest, frozen BB



(a) Disease classification multitask, PadChest, PE unfrozen BB



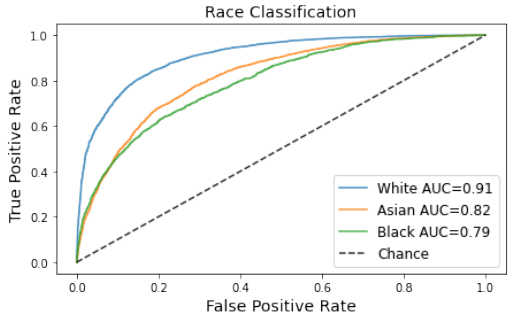
(b) Disease classification multitask, PadChest, NF unfrozen BB

Figure 6.20: **Disease, race, and sex classification performance for a multitask model, PadChest frozen BB.**

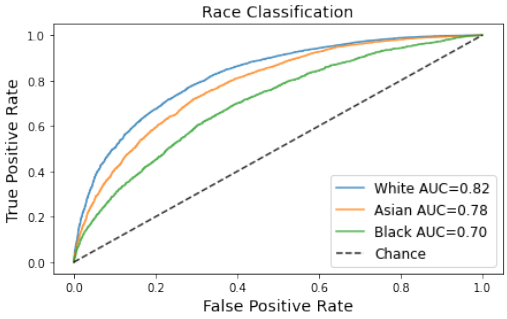
DANN

DANNs implement two methods for training adversarially: gradient negation and loss confusion. We investigate how both these methods affect disease, race, and sex classification performance after noise has been introduced into the dataset. Namely, we generate a biased model and ascertain whether adversarial training can mitigate performance disparities that arise as a result of this noise being added. Noise is introduced systematically or randomly into targeted subgroups (5.5) to simulate annotation bias.

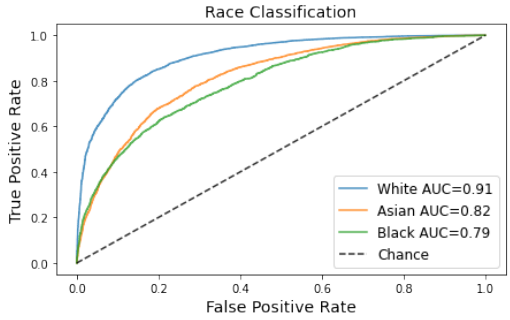
For comparison we utilize (1) a multitask model trained with a partially frozen backbone *without* label noise, to display the unperturbed disease performance (6.18, 6.20), and (2) a multitask model trained with a partially frozen backbone *with* label noise, to display disease performance without adversarial training. The performance for race and sex is monitored as well, as different adversarial methods can affect these tasks accordingly, as illustrated in Fig. 6.21.



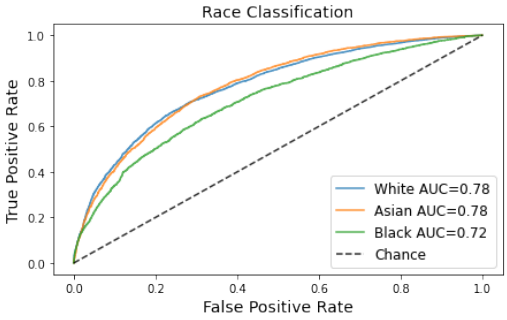
(a) ImageNet, random noise 50%, WT



(b) ImageNet, systematic noise 20%, WT



(c) PadChest, random noise 50%, WT



(d) PadChest, systematic noise 20%, WT

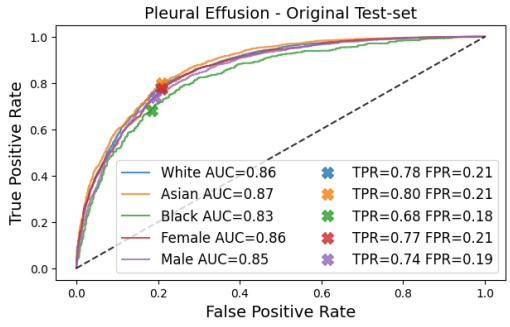
Figure 6.21: **Addition of systematic/random noise into White subgroups.** For random noise, White subgroups exhibit an AUC increase of 20% compared to the baseline. For systematic noise, ImageNet models display a 12% increase and PadChest models display a 7% increase.

Let us add systematic or random noise to a selected subgroup. If noise is added to White patients, we observe an increase in classification performance for this subgroup. For systematic noise, the performance in Asian and Black patients remains largely unaffected compared to the baseline. However, slight increases can be observed for random noise.

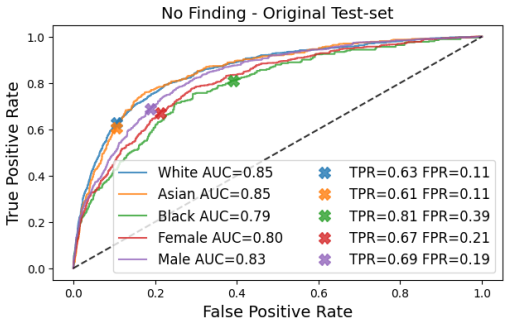
Random noise is applied to subgroups with a 50% probability, whereas systematic noise is applied with a 20% probability. In order to differentiate how adding noise to particular classes affects model behavior, we inject noise in Black/White subgroups (or Male/Female subgroups). Furthermore, for each subgroup, and each type of noise, we compare the affect of utilizing negated gradients or confusion loss for bias mitigation.

Due to the particular imbalance of the CheXpert dataset, labels are skewed towards White/Male patients, therefore Black/Female patients are considered minority subgroups. Regarding disease detection, we observe TPR/FPR shifts that arise when implementing noise. In particular, adding random label noise to majority subgroups, seems to cause a reduction in performance for the minority subgroups, yet an increase/stagnant performance for the majority subgroup (when predicting PE). This is especially noticeable in Fig. 6.23 (c) where noise is added to the White subgroup, giving rise to a TPR/FPR of 0.58/0.14 for Black patients but 0.79/0.26 for White patients.

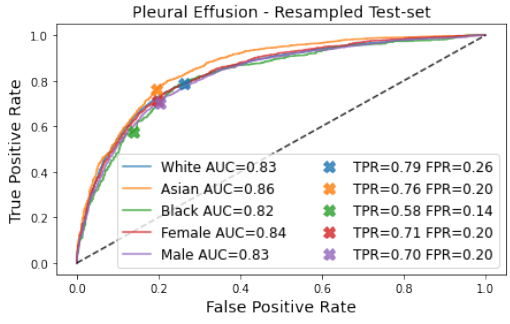
The effect of noise is most apparent when regarding the ROC curves for "No Finding". The subgroup experiencing noise displays a substantial increase in TPR, tied with a subsequent increase in FPR. Furthermore, opposing subgroups exhibit reduced TPR/FPR. This behavior can also be observed between M/F patients when introducing noise into one of these subgroups.



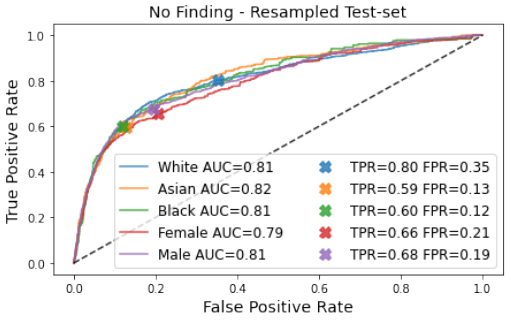
(a) PadChest, PE, random label noise 50%, BK



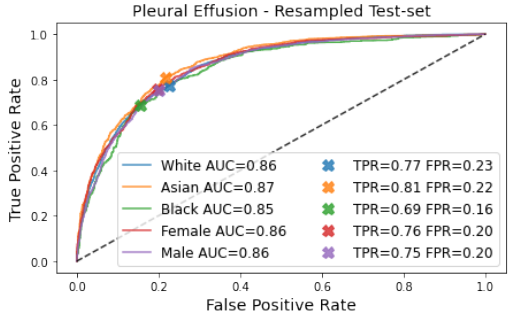
(b) PadChest, NF, random label noise 50%, BK



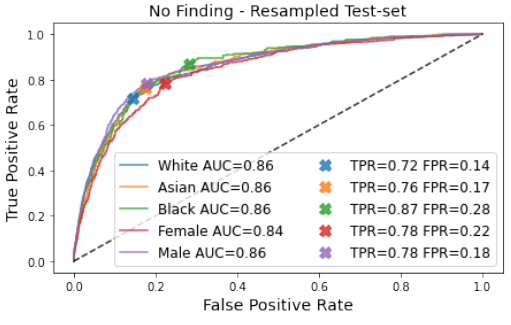
(c) PadChest, PE, random label noise 50%, WT



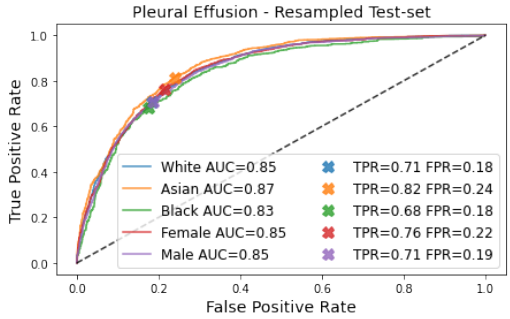
(d) PadChest, NF, random label noise 50%, WT



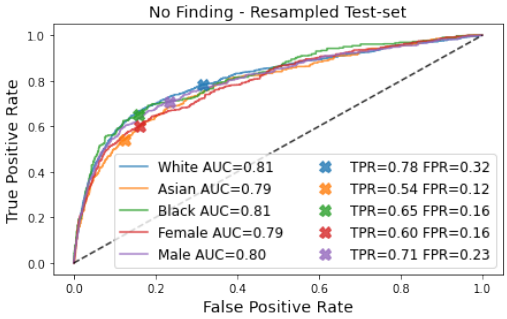
(e) PadChest, PE, systematic label noise 20%, BK



(f) PadChest, NF, systematic label noise 20%, BK



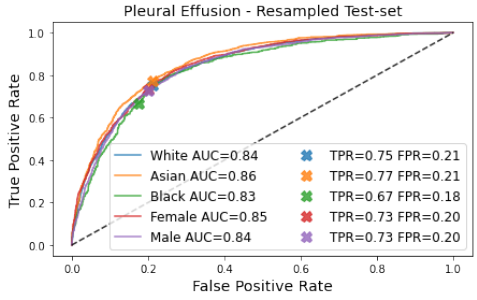
(a) PadChest, PE, systematic label noise 20%, WT



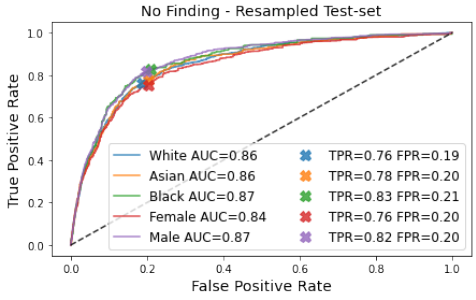
(b) PadChest, NF, systematic label noise 20%, WT

Figure 6.23: **Disease detection performance, label noise, PadChest, WT/BK.** In both models predicting NF where noise is introduced into BK subgroups (b + f), an increased TPR/FPR can be observed for BK. Similarly in models where noise is introduced into WT subgroups (d + h), high TPR/FPR can be observed. Remaining subgroups display an overall lower TPR/FPR. For systematic noise, general reduction in BK TPR/FPR is present.

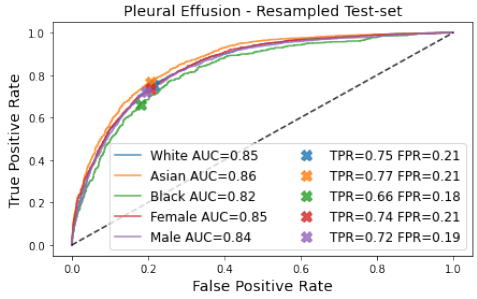
Looking at the feature space of these models, we can distinguish significant shifts in marginal distributions. Furthermore, some models display patterns in logit space that can be seen observed for certain subgroups along the output for disease labels, suggesting a connection between these factors. Thus, predictions might be biased, and could possibly be determined by spurious correlations between unrelated tasks. After implementing adversarial training, an overall correction in TPR/FPR back to baseline levels can be seen. For instance, examining PadChest with 50% random label noise added to Black subgroups, we now observe equitable performance between subgroups for NF and a more balanced AUC distribution for both PE/NF. Inspecting the generated feature representations, shifts in marginal distributions for race and sex as well as clusters for logit outputs are no longer present.



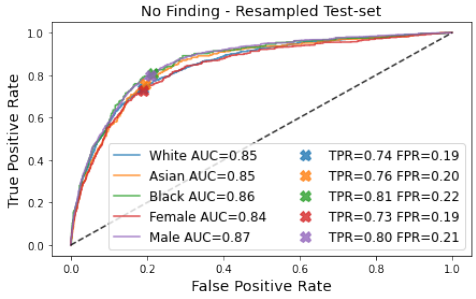
(a) PadChest, confusion loss, PE, random label noise 50%, BK



(b) PadChest, confusion loss, NF, random label noise 50%, BK



(c) PadChest, gradient negation, PE, random label noise 50%, BK



(d) PadChest, gradient negation, NF, random label noise 50%, BK

Figure 6.24: **Disease detection performance, with random label noise 50% and adversarial training, PadChest, BK.**

6.3.2 Discussion

The behavior of confusion loss versus gradient negation differs in terms of how parallel classification tasks are affected. Since gradient negation is implemented using the loss returned from the domain classifier and multiplying it by a negative hyperparameter, this should target selected features in the backbone more precisely. Therefore, we can expect a decrease in classification performance for the adversarial task, while the primary and tertiary tasks remain largely unaffected. On the other hand, confusion loss acts as a secondary loss function that aims to equalize all softmax outputs of the domain classifier, generating domain-invariant features in the process.

Let us inspect the previously mentioned PadChest model that implements domain adversarial training and is affected by 20% systematic noise added to Black patients. Racial information is targeted adversarially, meaning gradient negation inverts the loss returned by the race classifier and confusion loss utilizes the outputs from the race classifier to calculate the secondary loss.

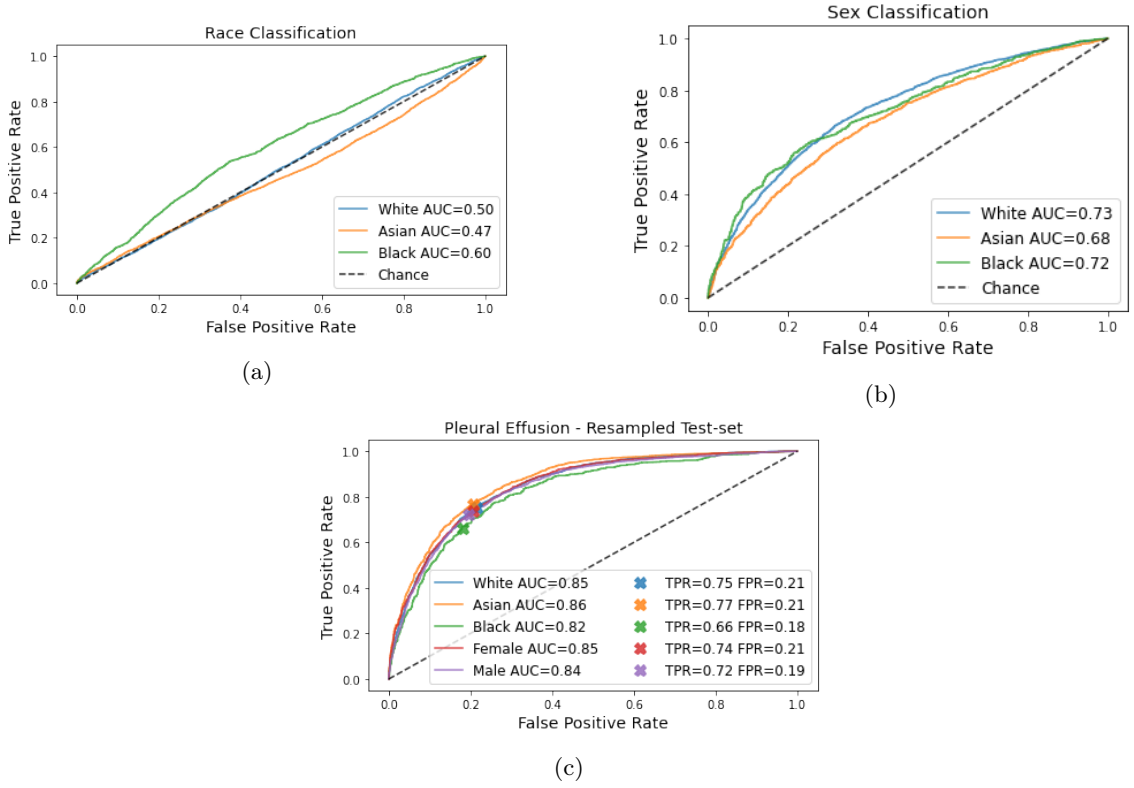


Figure 6.25: **Overall performance for systematic noise 20% with gradient negation, PadChest, BK.**

Since we negate gradients related to race loss, the AUC for race classification is observed to be close to 0.50, indicating random predictions. In contrast, sex classification maintains performance substantially above 0.50, though slightly decreased in comparison to the baseline (Fig. 6.20). A possible cause for this decrease in performance is that backbone parameters are not updated during training of the sex classifier. Therefore, when these are altered for disease detection, the tertiary task is not able to adapt as well. Lastly, disease detection performance is comparable to the baseline, displaying only minor downward shifts in AUC/TPR/FPR due to adversarial training. This behavior can be observed across all models implementing negated gradients.

For confusion loss, we observe AUC values signifying random classification (approximately 0.50) for both sex and race classification. This decrease in sex classification performance could be attributed to backbone parameters not being updated during training of the sex classifier. However, this does not necessarily explain why sex classification is worse for confusion loss compared to negated gradients. Possibly, the confusion loss function aggressively generates domain-invariant features such that backbone parameters are so altered that sex classification suffers substantially. On the other hand, one might conclude that the confusion loss function inadvertently generates features invariant to both tasks. Therefore, it would be interesting to investigate whether a model

implementing confusion loss based on race outputs can negate bias if label noise is added to biological sex subgroups.

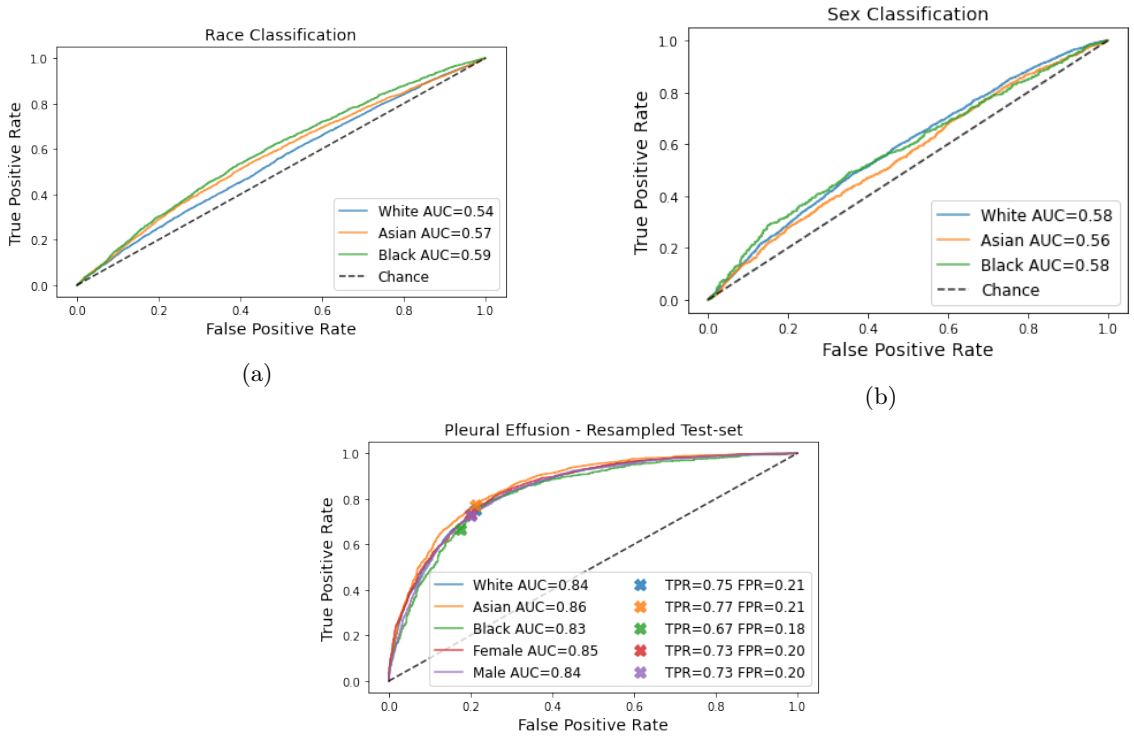


Figure 6.26: **Overall performance for systematic noise 20% with confusion loss, PadChest, BK.**

Adversarial strength

In the above described models, an α of 0.0001 is chosen to determine the adversary strength. One might hypothesize that changing α may not have a significant effect on sex and race classification performance since this is included in a separate optimization task (with negated gradients/confusion loss).

Conversely, this might determine how impactful updates to the feature extractor are and affect how much the classification heads need to adapt. Therefore, in prediction heads that are trained with a frozen backbone, having a weak adversary may not cause such a strong reduction in performance. We inspect different values for α, starting with a value of 0.1 and decreasing this exponentially by a power of 2. Therefore, the inspected values are 0.1, 0.001, 0.0001, 0.00001, and 0.000001.

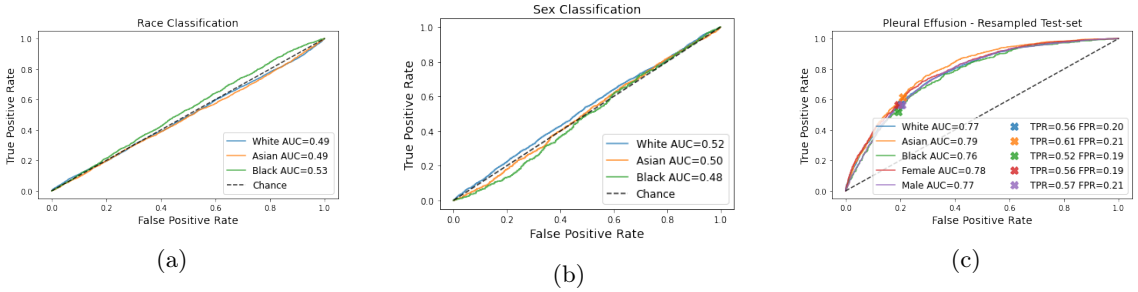
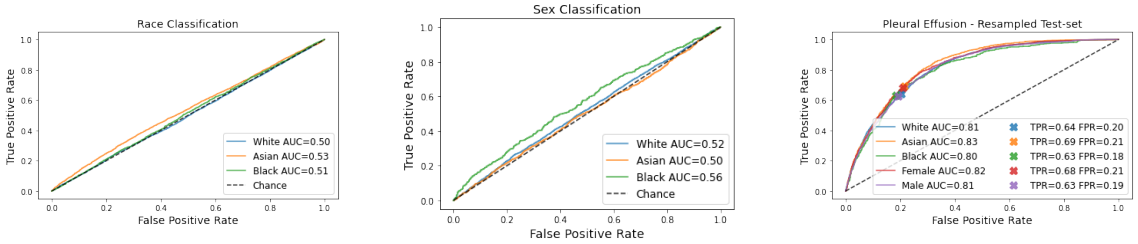
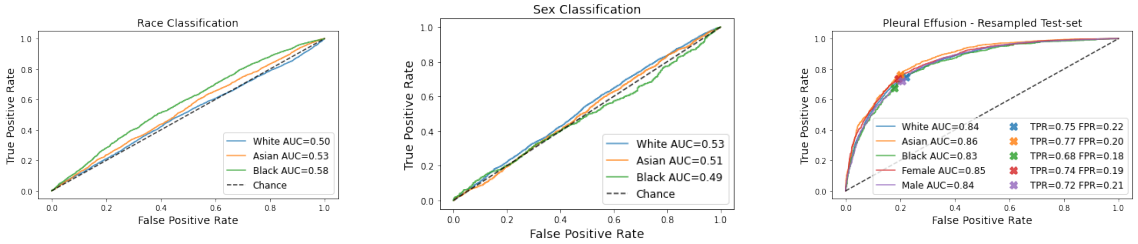


Figure 6.27: α = **0.1, ImageNet, race confusion.**



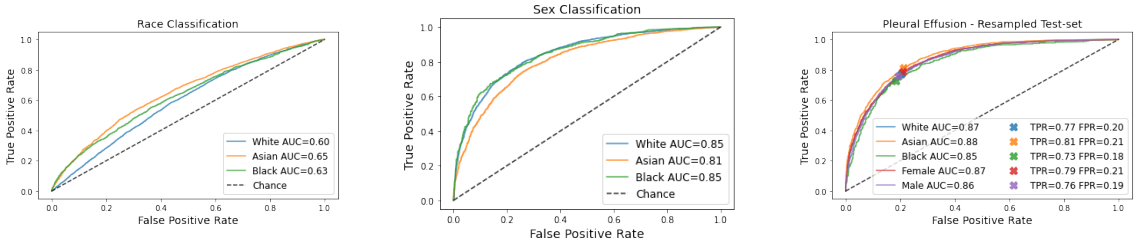
(a) (b) (c)

Figure 6.28: α = **0.001, ImageNet, race confusion.**



(a) (b) (c)

Figure 6.29: α = **0.00001, ImageNet, race confusion.**



(a) (b) (c)

Figure 6.30: α = **0.000001, ImageNet, race confusion.**

Visually, a general trend that can be determined, is that the AUC for disease detection increases as the value for α decreases. In the case of larger α values, this could be interpreted as implementing an adversary that is too powerful, resulting in an excessive performance trade-off from the label predictor. This can be seen in Fig. A.15, which displays an AUC of 0.77 for the entire population, which is less than the observed performances for all subsequent α's (which increases gradually).

α	AUC
0.1	0.77
0.01	0.812
0.001	0.814
0.0001	0.832
0.00001	0.844
0.000001	0.866

Table 6.4: **AUC progression with decreasing** α

Race and sex classification see a corresponding increase in AUC with a smaller $\alpha = 0.000001$, though this could simply be attributed to α being so small that it becomes negligible. Thus, ω would range from 0 to $\sim$0, essentially stopping updates from being provided to the backbone and hindering adversarial training. Subsequently, the model starts displaying performance similar to the baseline in all classification tasks (Fig. A.19).

This behavior was also investigated using negated gradients and yielded similar observations. The corresponding ROC curves can be found in Appendix A.

Performance

At first glance, after implementation of the adversarial network, model performance seems to decrease without correcting subgroup disparities and biases. Let us examine Youden's J Statistic calculated in the presence of systematic label noise for instance (Fig. 6.31 + 6.32).

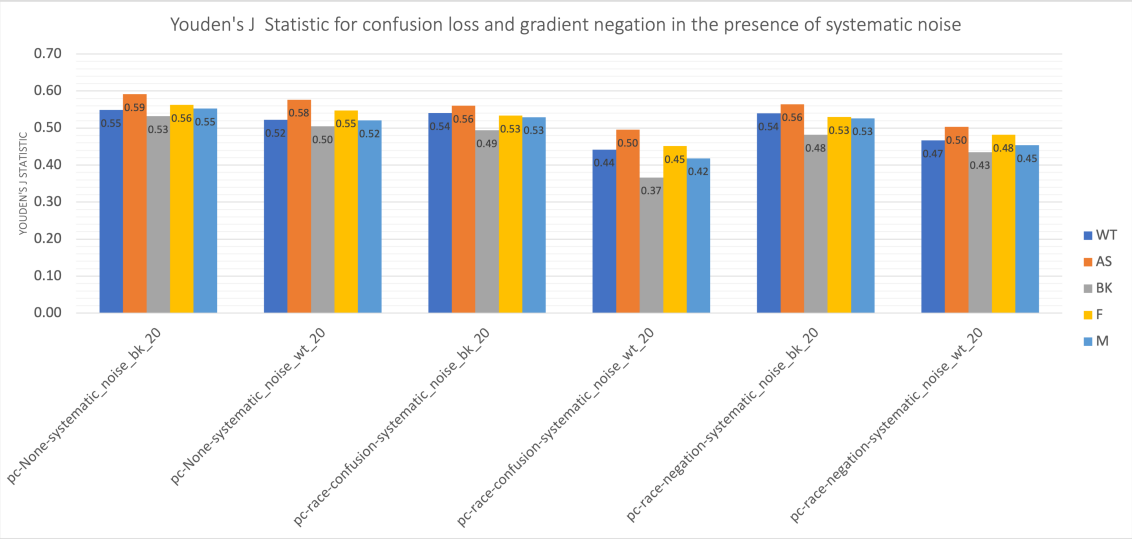


Figure 6.31: **Injected systematic label noise into race subgroups 20%, PadChest, Pleural Effusion.**

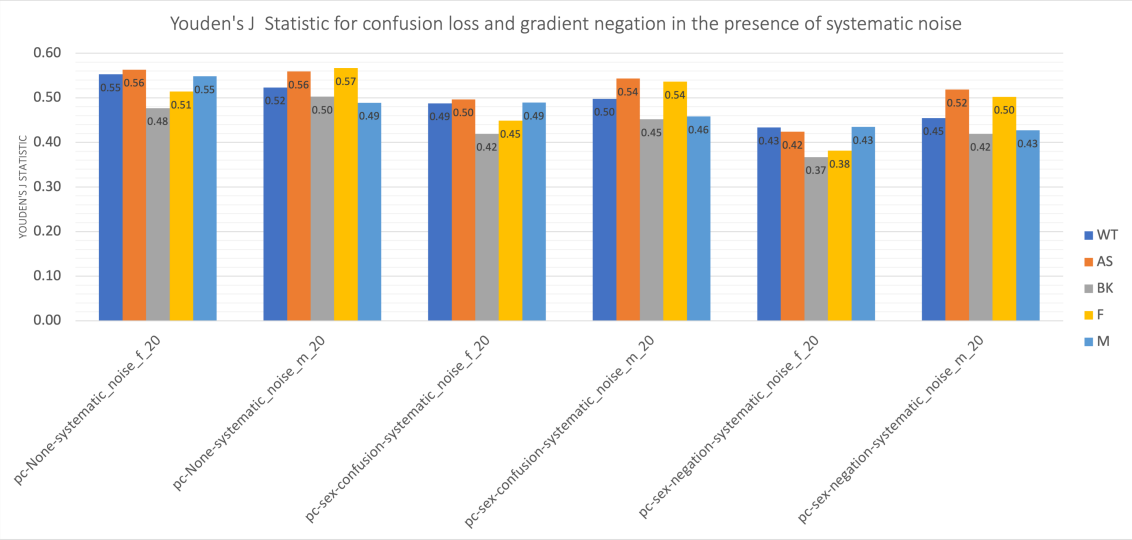
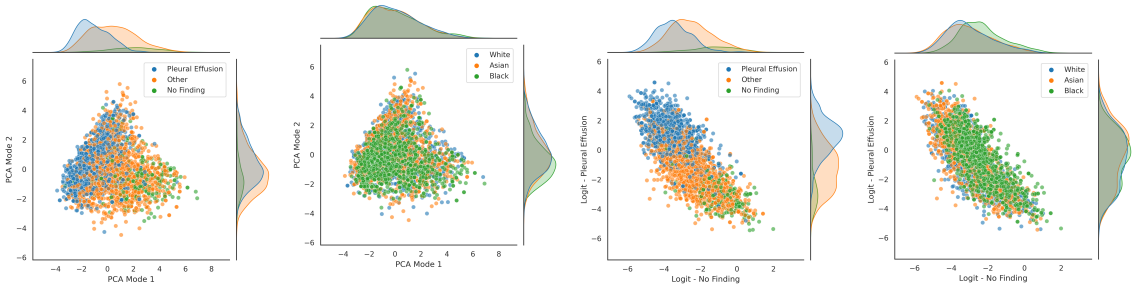


Figure 6.32: **Injected systematic label noise into sex subgroups 20%, PadChest, Pleural Effusion.**

Disparities from models without adversarial training can still be seen in models that employ confusion loss/negated gradients. In some cases, these disparities are even exaggerated, as seen in "pc-sex-confusion-systematic_noise_m_20" and "pc-race-confusion-systematic_noise_wt_20".

However, by inspecting the feature space of generated models, we might be able to ascertain how biases are encoded and whether they are still present after adversarial training. Adversarial training is able to mitigate clustering in logit space for secondary/tertiary tasks that arises due to label noise. Furthermore, we observe corrections of ROC curves back to their original shapes.

However, in some subgroups, statistical differences between distributions can still be observed, though these shifts are less prominent compared to distributions without adversarial training.



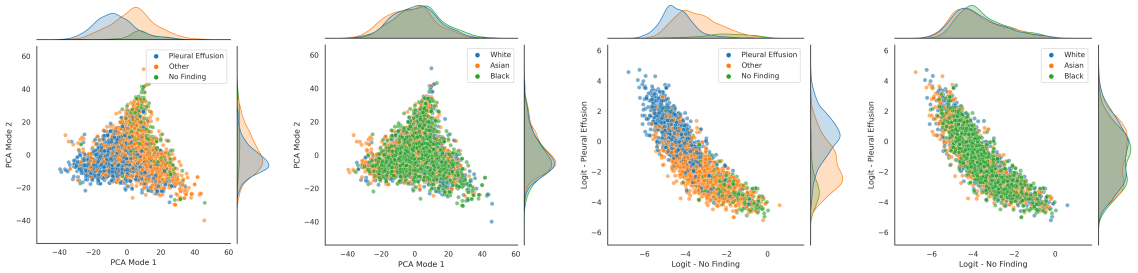
(a) PCA 1 + 2 PadChest, disease labels, systematic noise 20%, BK, no adv

(b) PCA 1 + 2 PadChest, race labels, systematic noise 20%, BK, no adv

(c) Logits PadChest, disease, systematic noise 20%, BK, no adv

(d) Logits PadChest, race, systematic noise 20%, BK, no adv

Figure 6.33: **Feature inspection for PadChest, systematic label noise 20%, BK, no adv.**



(a) PCA 1 + 2 PadChest, disease labels, systematic noise 20%, BK, negated gradients

(b) PCA 1 + 2 PadChest, race labels, systematic noise 20%, BK, negated gradients

(c) Logits PadChest, disease, systematic noise 20%, BK, negated gradients

(d) Logits PadChest, race, systematic noise 20%, BK, negated gradients

Figure 6.34: **Feature inspection for PadChest, systematic label noise 20%, BK, negated gradients**

In summary, we observe that adversarial training and DANNs are able to partially mitigate biases that were introduced into the dataset. This can be targeted (gradient negation) or applied more generally (confusion loss). However, it is important to take into account, that in a real-world scenario many sources of bias exist and it is difficult to point out a distinct origin of bias. This could hinder implementations of adversarial networks, since targetable groups cannot be specified properly. Therefore, if confusion loss turns out to have a broader effect on the network, this could be useful for adversarial training without the need for a specific source of bias.

Chapter 7

Conclusions and Future Work

In this project, we have looked at various methods for developing "fair" disease detection algorithms involving the use of foundation models and domain adversarial neural networks. To summarize our contributions:

- We analyzed the subgroup performance and feature representations in pre-trained chest X-ray disease detection models from the XRV library. These models function as feature extractors and were retrained for domain-specific tasks on CheXpert. Fixed feature extractor (frozen) and fine-tuning (unfrozen) transfer learning methods were implemented for this step. Differences in behavior emerging from training on a frozen versus unfrozen backbones were investigated and showed consistent reduction in performance for frozen backbones.

- Through feature exploration, we showed that the models developed with frozen backbones often display spurious connections between disease classification tasks and protected patient characteristics, subsequently forming shortcuts to predict disease outcomes. Following this, we emphasized how training with an unfrozen backbone allows these biases in the backbone to be overridden, and found no strong evidence suggesting that protected characteristics are correlated with disease detection in unfrozen XRV foundation models.

- We implemented adversarial neural networks for bias mitigation, which were able to nearly restore irregular ROC curves due to label noise back to their original state. However, this came with a performance trade-off and DANNs consistently showed reduced Youden's J Statistic compared to the baseline. Confusion loss showed to affect both race and sex classification tasks, suggesting that it might the backbone broadly or that it updates features in such a way that sex classification is randomized. On the other hand, negated gradients target the domain classifier more precisely.

This project shows the difficulties of quantifying bias and determining what information is used to make decisions. The foundation models we investigated did not show correlations between unrelated subgroups, however biases from pre-training might be encoded in higher PCA modes that affect downstream performance. DANNs are able to correct shifts in logit space and balance representations in feature space as well. In some cases however, adversarial training was not as effective.

Finally, we suggest further areas for investigation and improvement.

1. **Foundation models** In order to determine more significant conclusions about how foundation models encode bias, further architectures and pre-training methods need to be inspected. For instance, ViT or REMEDIS models. Using models that are not necessarily pre-trained for disease detection, would allow us to more easily observe differences in features for comparison between frozen and unfrozen backbones.

2. **DANN and label noise** Bias mitigation is a topic with many branches that can be investigated. Firstly, we can look at other mitigation techniques such as Deep Feature Reweighting (DFR) which corrects for spurious correlations, and might prove helpful in discovering methods for diminishing bias. Furthermore, investigating additional levels of noise in DANNs can be inspected, as well as introducing it into other subgroups such as age or Asian patients.

Adding noise to both sex and race classification heads can be investigated to see how this affects model development. Subsequently, this raises the question of whether adversarially training with two heads is logical and how effective this is. Furthermore, in order to see how confusion loss affects the backbone, we suggest training adversarially for race, while injecting label noise into sex subgroups. If the backbone is broadly affected this may prove to be a helpful solution for bias mitigation, since we often do not know where biases in a dataset stem from.

Lastly, it is important to note that, while generating unbiased models is crucial, it is only the first step in implementing fair algorithms for medical AI. Biases and prejudice can still be introduced after model development, for instance at the point of care. In conclusion, deep learning in medicine is a growing field that offers efficient and effective methods for diagnosis, but also has a significant impact on a patient's health outcomes and quality of care. Therefore, it of paramount importance that fair algorithms and unbiased decision making processes are used in future medical applications.

Appendix A

First Appendix

Model	Metric	All	White	Asian	Black	Female	Male
ImageNet	AUC	0.86	0.86	0.88	0.84	0.86	0.86
	TPR	0.76	0.77	0.80	0.70	0.77	0.74
	FPR	0.20	0.21	0.20	0.20	0.22	0.19
	Youden's	0.56	0.56	0.60	0.50	0.55	0.56
rsna	AUC	0.86	0.86	0.88	0.84	0.86	0.86
	TPR	0.76	0.76	0.81	0.69	0.77	0.74
	FPR	0.20	0.21	0.22	0.17	0.21	0.19
	Youden's	0.56	0.55	0.60	0.52	0.56	0.55
mimic_ch	AUC	0.86	0.86	0.88	0.84	0.86	0.86
	TPR	0.75	0.78	0.80	0.69	0.77	0.74
	FPR	0.20	0.21	0.21	0.17	0.21	0.19
	Youden's	0.56	0.56	0.59	0.51	0.56	0.56
mimic_nb	AUC	0.86	0.86	0.87	0.84	0.86	0.86
	TPR	0.76	0.77	0.80	0.70	0.78	0.74
	FPR	0.20	0.21	0.21	0.19	0.21	0.20
	Youden's	0.56	0.56	0.60	0.52	0.57	0.54
nih	AUC	0.86	0.86	0.88	0.84	0.86	0.85
	TPR	0.75	0.77	0.78	0.69	0.77	0.73
	FPR	0.20	0.21	0.21	0.18	0.21	0.19
	Youden's	0.55	0.56	0.58	0.51	0.56	0.54
pc	AUC	0.86	0.87	0.88	0.85	0.87	0.86
	TPR	0.76	0.77	0.81	0.70	0.78	0.74
	FPR	0.20	0.20	0.22	0.18	0.22	0.19
	Youden's	0.56	0.57	0.59	0.51	0.56	0.56
all	AUC	0.86	0.86	0.88	0.85	0.87	0.86
	TPR	0.76	0.78	0.80	0.70	0.76	0.75
	FPR	0.20	0.22	0.21	0.17	0.20	0.20
	Youden's	0.56	0.56	0.58	0.53	0.56	0.55

Table A.1: XRV and ImageNet model performance metrics, single task, unfrozen BB, PE

Model	Metric	All	White	Asian	Black	Female	Male
ImageNet	AUC	0.87	0.87	0.87	0.88	0.86	0.89
	TPR	0.80	0.80	0.80	0.81	0.77	0.83
	FPR	0.20	0.20	0.20	0.19	0.21	0.19
	Youden's	0.60	0.59	0.60	0.62	0.56	0.64
rsna	AUC	0.87	0.86	0.87	0.88	0.85	0.88
	TPR	0.79	0.79	0.78	0.81	0.76	0.82
	FPR	0.20	0.20	0.20	0.20	0.19	0.21
	Youden's	0.59	0.59	0.58	0.60	0.55	0.63
mimic_ch	AUC	0.87	0.87	0.86	0.88	0.86	0.88
	TPR	0.79	0.77	0.77	0.83	0.75	0.82
	FPR	0.20	0.20	0.20	0.20	0.19	0.21
	Youden's	0.59	0.58	0.58	0.62	0.56	0.62
mimic_nb	AUC	0.87	0.86	0.87	0.88	0.85	0.88
	TPR	0.80	0.78	0.80	0.83	0.77	0.83
	FPR	0.20	0.21	0.20	0.20	0.18	0.21
	Youden's	0.59	0.59	0.58	0.58	0.60	0.56
nih	AUC	0.87	0.87	0.86	0.87	0.85	0.88
	TPR	0.80	0.78	0.81	0.82	0.79	0.83
	FPR	0.20	0.20	0.20	0.20	0.19	0.22
	Youden's	0.60	0.58	0.60	0.63	0.58	0.63
pc	AUC	0.87	0.87	0.86	0.88	0.86	0.88
	TPR	0.80	0.78	0.80	0.83	0.77	0.83
	FPR	0.20	0.20	0.20	0.20	0.19	0.21
	Youden's	0.60	0.58	0.60	0.63	0.58	0.63
all	AUC	0.87	0.87	0.87	0.88	0.86	0.88
	TPR	0.80	0.78	0.81	0.82	0.79	0.83
	FPR	0.20	0.20	0.20	0.20	0.19	0.21
	Youden's	0.60	0.61	0.61	0.58	0.57	0.63

Table A.2: XRV and ImageNet model performance metrics, single task, unfrozen BB, NF

Model	Metric	All	White	Asian	Black	Female	Male
ImageNet	AUC	0.75	0.75	0.77	0.73	0.75	0.74
	TPR	0.52	0.54	0.58	0.44	0.54	0.50
	FPR	0.20	0.21	0.21	0.17	0.21	0.19
	Youden's	0.32	0.33	0.36	0.27	0.33	0.31
rsna	AUC	0.70	0.70	0.72	0.69	0.70	0.70
	TPR	0.44	0.41	0.47	0.43	0.48	0.40
	FPR	0.20	0.20	0.21	0.19	0.22	0.18
	Youden's	0.24	0.21	0.26	0.24	0.26	0.22
mimic_ch	AUC	0.77	0.77	0.78	0.76	0.77	0.77
	TPR	0.55	0.56	0.60	0.47	0.56	0.53
	FPR	0.20	0.20	0.23	0.18	0.20	0.20
	Youden's	0.34	0.37	0.38	0.29	0.36	0.33
mimic_nb	AUC	0.78	0.78	0.79	0.76	0.78	0.78
	TPR	0.58	0.59	0.64	0.50	0.61	0.55
	FPR	0.20	0.20	0.23	0.17	0.21	0.19
	Youden's	0.38	0.38	0.42	0.33	0.39	0.36
nih	AUC	0.79	0.79	0.81	0.77	0.79	0.78
	TPR	0.60	0.61	0.64	0.54	0.63	0.57
	FPR	0.20	0.21	0.20	0.19	0.20	0.20
	Youden's	0.40	0.41	0.44	0.36	0.43	0.38
pc	AUC	0.81	0.81	0.83	0.79	0.82	0.81
	TPR	0.65	0.65	0.69	0.60	0.66	0.64
	FPR	0.20	0.20	0.21	0.19	0.20	0.20
	Youden's	0.45	0.45	0.48	0.41	0.46	0.44
all	AUC	0.83	0.83	0.85	0.82	0.84	0.83
	TPR	0.69	0.71	0.76	0.62	0.71	0.68
	FPR	0.20	0.20	0.22	0.18	0.20	0.20
	Youden's	0.50	0.50	0.54	0.44	0.51	0.48

Table A.3: XRV and ImageNet model performance metrics, single task, frozen BB, PE

Model	Metric	All	White	Asian	Black	Female	Male
ImageNet	AUC	0.81	0.80	0.81	0.82	0.80	0.82
	TPR	0.67	0.68	0.66	0.68	0.62	0.72
	FPR	0.20	0.21	0.19	0.20	0.19	0.21
	Youden's	0.47	0.47	0.47	0.43	0.51	0.44
rsna	AUC	0.79	0.78	0.79	0.81	0.78	0.80
	TPR	0.64	0.64	0.62	0.67	0.64	0.65
	FPR	0.20	0.20	0.21	0.19	0.20	0.20
	Youden's	0.44	0.43	0.41	0.48	0.43	0.45
mimic_ch	AUC	0.83	0.83	0.82	0.85	0.82	0.84
	TPR	0.71	0.73	0.69	0.72	0.65	0.76
	FPR	0.20	0.21	0.19	0.20	0.17	0.22
	Youden's	0.51	0.52	0.50	0.52	0.48	0.54
mimic_nb	AUC	0.84	0.83	0.82	0.87	0.83	0.85
	TPR	0.73	0.73	0.69	0.77	0.69	0.76
	FPR	0.20	0.21	0.19	0.20	0.19	0.21
	Youden's	0.53	0.52	0.50	0.57	0.51	0.55
nih	AUC	0.84	0.83	0.83	0.85	0.83	0.84
	TPR	0.73	0.74	0.72	0.73	0.70	0.76
	FPR	0.20	0.21	0.19	0.19	0.19	0.21
	Youden's	0.53	0.53	0.53	0.54	0.51	0.55
pc	AUC	0.85	0.84	0.85	0.87	0.84	0.86
	TPR	0.76	0.75	0.74	0.79	0.73	0.78
	FPR	0.20	0.20	0.20	0.20	0.19	0.21
	Youden's	0.56	0.54	0.54	0.59	0.54	0.57
all	AUC	0.87	0.86	0.86	0.88	0.86	0.87
	TPR	0.80	0.79	0.78	0.83	0.78	0.82
	FPR	0.20	0.20	0.20	0.20	0.20	0.20
	Youden's	0.60	0.59	0.59	0.63	0.58	0.62

Table A.4: XRV and ImageNet model performance metrics, single task, frozen BB, NF

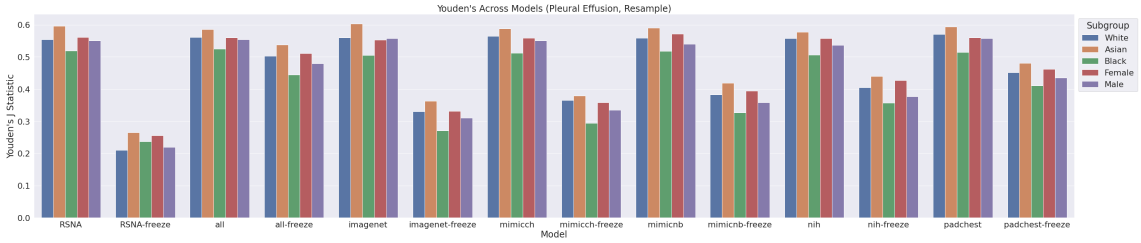


Figure A.1: Youden's across models with subgroup performance, PE

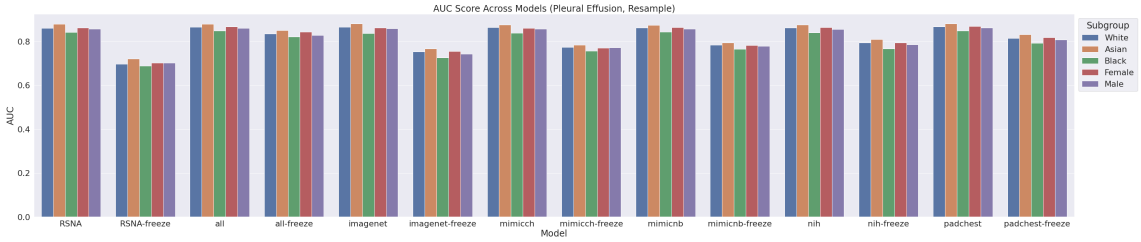


Figure A.2: AUC across models with subgroup performance, PE

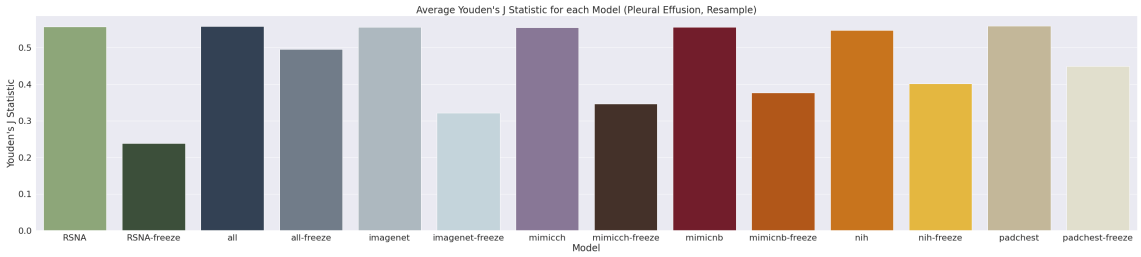


Figure A.3: Average Youden's across models, PE

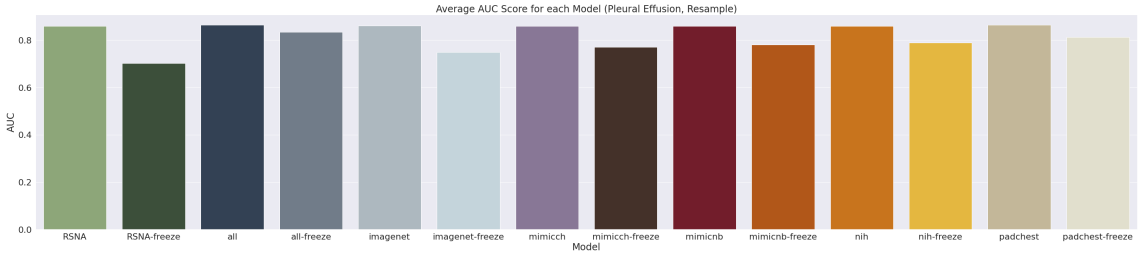


Figure A.4: Average AUC across models, PE

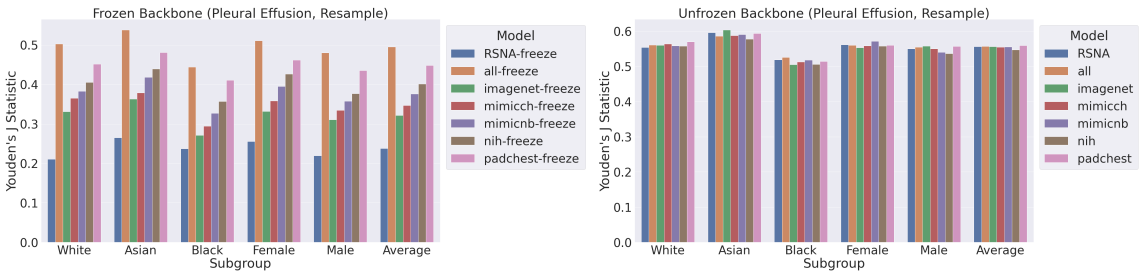


Figure A.5: Youden's separated into frozen and unfrozen models, PE

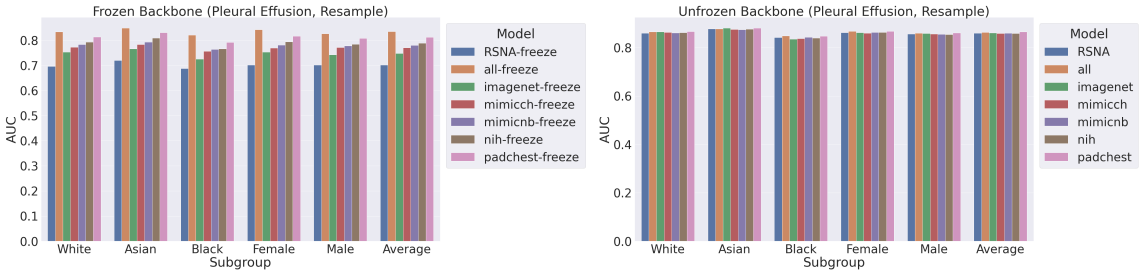


Figure A.6: AUC separated into frozen and unfrozen models, PE

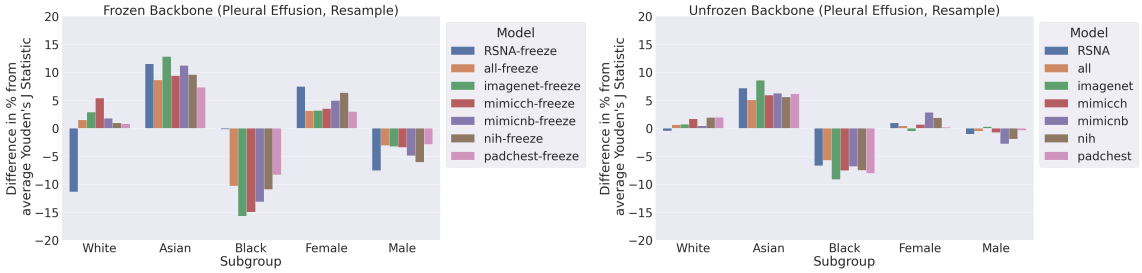


Figure A.7: % difference average from Youden's, PE

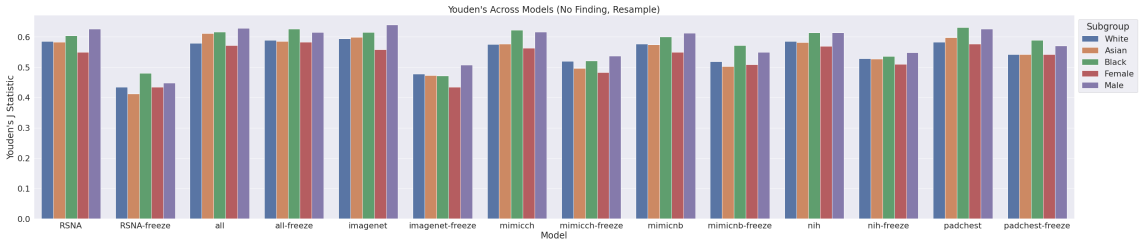


Figure A.8: Youden's across models with subgroup performance, NF

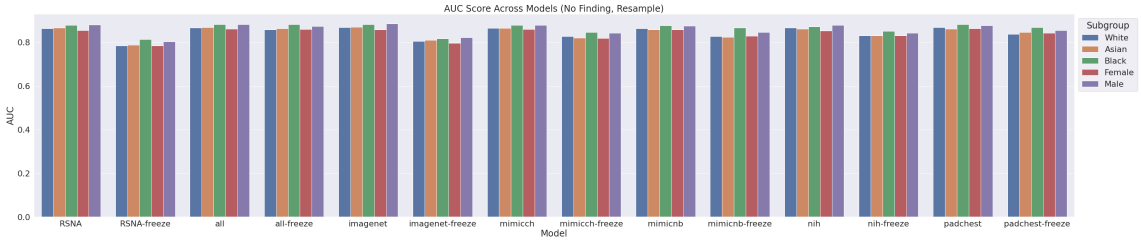


Figure A.9: AUC across models with subgroup performance, NF

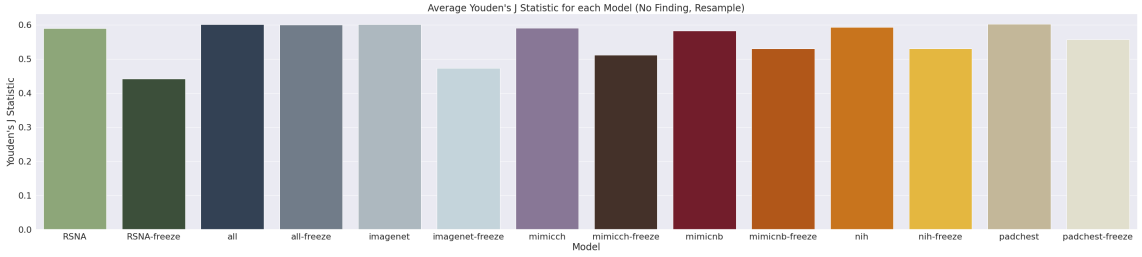


Figure A.10: Average Youden's across models, NF

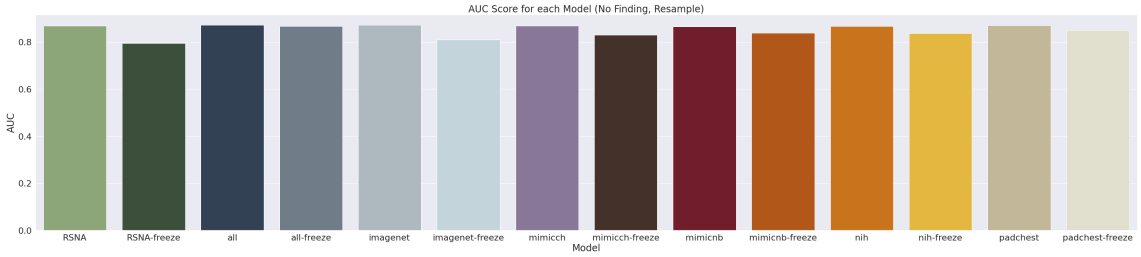


Figure A.11: Average AUC across models, NF

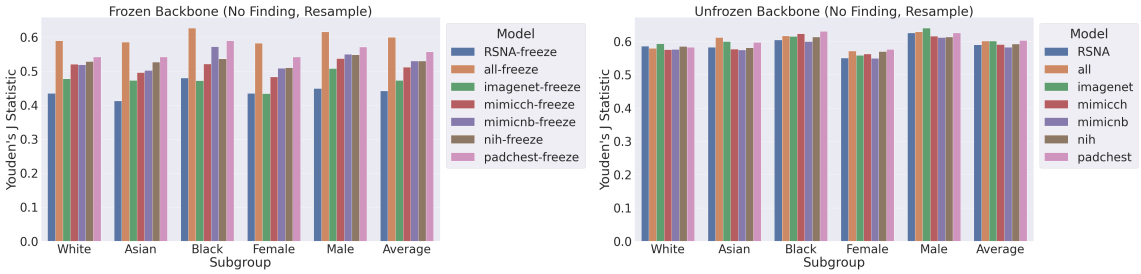


Figure A.12: Youden's separated into frozen and unfrozen models, NF

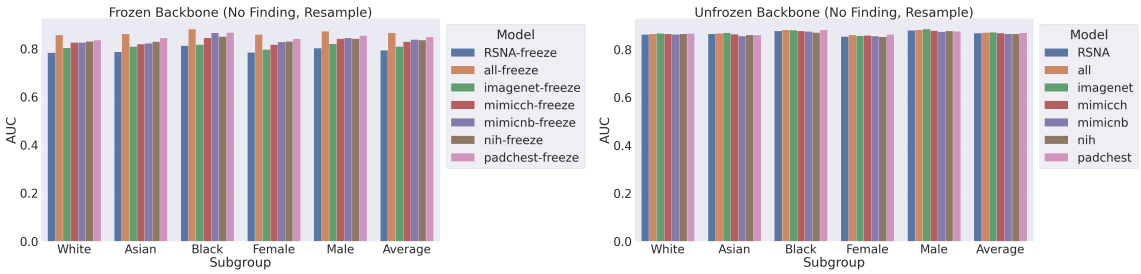


Figure A.13: AUC separated into frozen and unfrozen models, NF

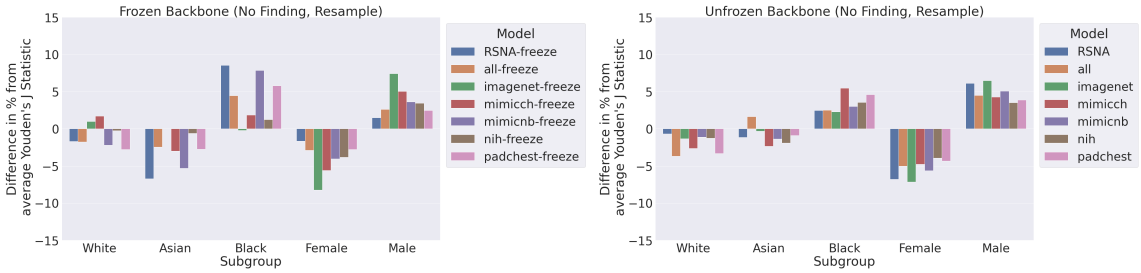
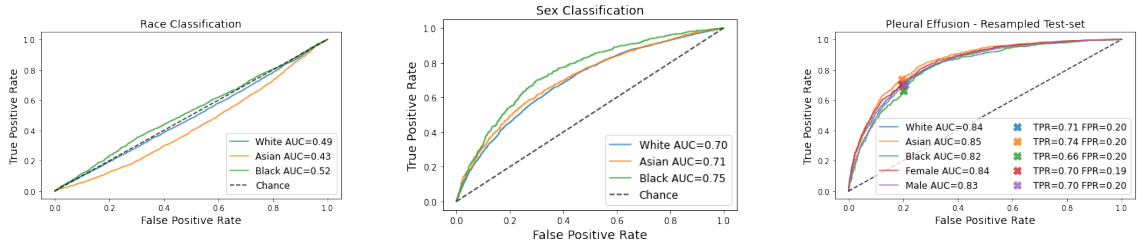
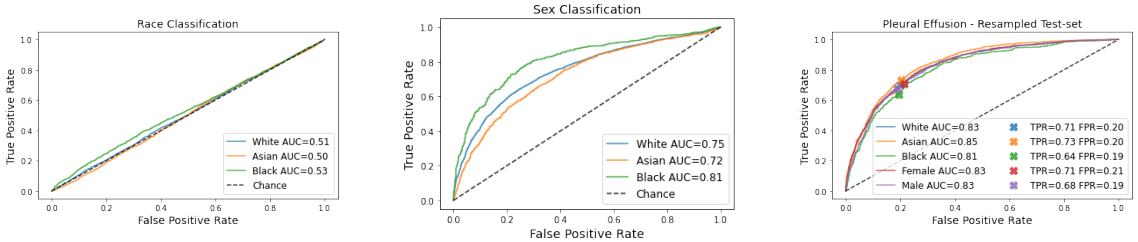


Figure A.14: % difference average from Youden's, NF



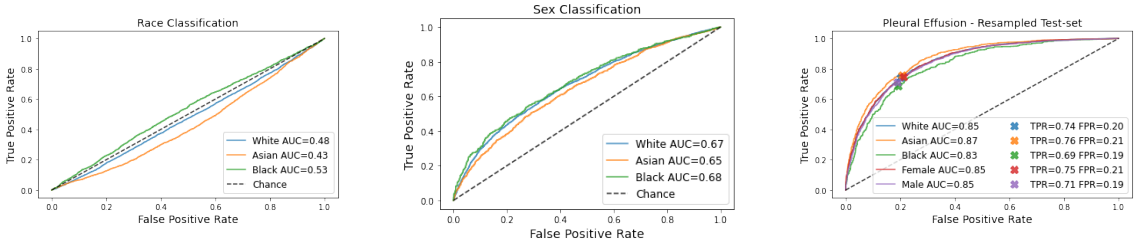
(a) (b) (c)

Figure A.15: $\alpha = \mathbf{0.1}$**, ImageNet, race negation.**



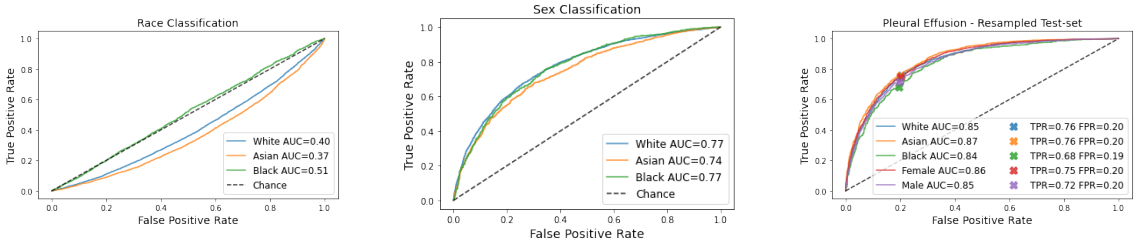
(a) (b) (c)

Figure A.16: α = **0.01, ImageNet, race negation.**



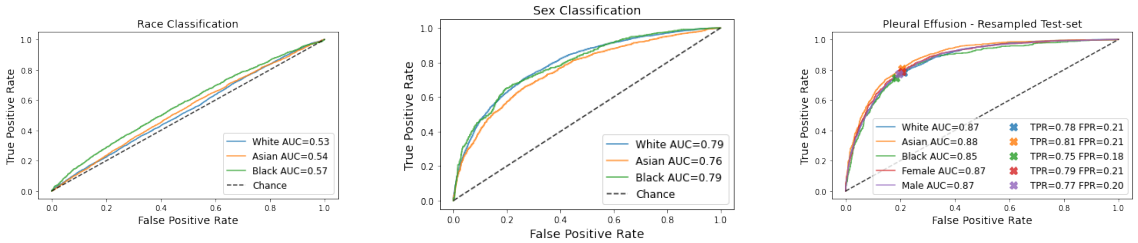
(a) (b) (c)

Figure A.17: α = **0.0001, ImageNet, race negation.**



(a) (b) (c)

Figure A.18: α = **0.00001, ImageNet, race negation.**



(a) (b) (c)

Figure A.19: α = **0.000001, ImageNet, race negation.**

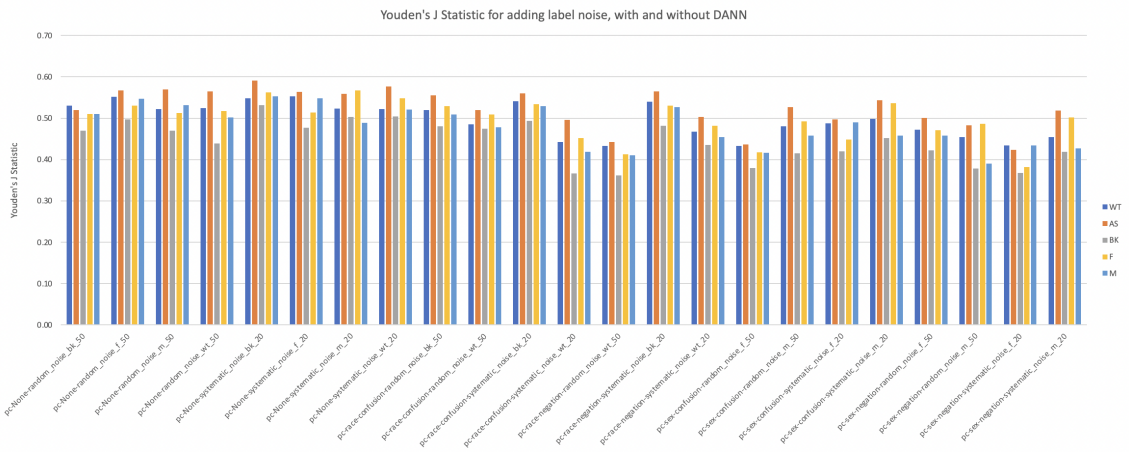


Figure A.20: Performance for all models with label noise + adversarial training

Bibliography

[1] Agostina J. Larrazabal, Nicolás Nieto, Victoria Peterson, Diego H. Milone, and Enzo Ferrante. Gender imbalance in medical imaging datasets produces biased classifiers for computer-aided diagnosis. *Proceedings of the National Academy of Sciences*, 117(23):12592–12594, June 2020. Publisher: Proceedings of the National Academy of Sciences.

[2] S. Kevin Zhou, Hayit Greenspan, Christos Davatzikos, James S. Duncan, Bram Van Ginneken, Anant Madabhushi, Jerry L. Prince, Daniel Rueckert, and Ronald M. Summers. A Review of Deep Learning in Medical Imaging: Imaging Traits, Technology Trends, Case Studies With Progress Highlights, and Future Promises. *Proceedings of the IEEE*, 109(5):820–838, May 2021. Conference Name: Proceedings of the IEEE.

[3] Richard J. Chen, Judy J. Wang, Drew F. K. Williamson, Tiffany Y. Chen, Jana Lipkova, Ming Y. Lu, Sharifa Sahai, and Faisal Mahmood. Algorithmic fairness in artificial intelligence for medicine and healthcare. *Nature Biomedical Engineering*, 7(6):719–742, June 2023. Number: 6 Publisher: Nature Publishing Group.

[4] Nancy E. Adler, M. Maria Glymour, and Jonathan Fielding. Addressing Social Determinants of Health and Health Inequalities. *JAMA*, 316(16):1641–1642, October 2016.

[5] Ričards Marcinkevičs, Ece Ozkan, and Julia E. Vogt. Debiasing Deep Chest X-Ray Classifiers using Intra- and Post-processing Methods, July 2022. arXiv:2208.00781 [cs, stat].

[6] 2022 National Healthcare Quality and Disparities Report Appendix B. Quality Trends and Disparities Tables. 2019.

[7] Ben Glocker, Charles Jones, Mélanie Bernhardt, and Stefan Winzeck. Algorithmic encoding of protected characteristics in chest X-ray disease detection models. *eBioMedicine*, 89:104467, March 2023.

[8] Shekoofeh Azizi, Laura Culp, Jan Freyberg, Basil Mustafa, Sebastien Baur, Simon Kornblith, Ting Chen, Patricia MacWilliams, S. Sara Mahdavi, Ellery Wulczyn, Boris Babenko, Megan Wilson, Aaron Loh, Po-Hsuan Cameron Chen, Yuan Liu, Pinal Bavishi, Scott Mayer McKinney, Jim Winkens, Abhijit Guha Roy, Zach Beaver, Fiona Ryan, Justin Krogue, Mozziyar Etemadi, Umesh Telang, Yun Liu, Lily Peng, Greg S. Corrado, Dale R. Webster, David Fleet, Geoffrey Hinton, Neil Houlsby, Alan Karthikesalingam, Mohammad Norouzi, and Vivek Natarajan. Robust and Efficient Medical Imaging with Self-Supervision, July 2022. arXiv:2205.09723 [cs].

[9] Rikiya Yamashita, Mizuho Nishio, Richard Kinh Gian Do, and Kaori Togashi. Convolutional neural networks: an overview and application in radiology. *Insights into Imaging*, 9(4):611–629, August 2018. Number: 4 Publisher: SpringerOpen.

[10] Ben Glocker, Charles Jones, Mélanie Bernhardt, and Stefan Winzeck. Risk of Bias in Chest X-ray Foundation Models.

[11] Dan Hendrycks, Kimin Lee, and Mantas Mazeika. Using Pre-Training Can Improve Model Robustness and Uncertainty, October 2019. arXiv:1901.09960 [cs, stat].

[12] Andrew B. Sellergren, Christina Chen, Zaid Nabulsi, Yuanzhen Li, Aaron Maschinot, Aaron Sarna, Jenny Huang, Charles Lau, Sreenivasa Raju Kalidindi, Mozziyar Etemadi, Florencia Garcia-Vicente, David Melnick, Yun Liu, Krish Eswaran, Daniel Tse, Neeral Beladia, Dilip

Krishnan, and Shravya Shetty. Simplified Transfer Learning for Chest Radiography Models Using Less Data. *Radiology*, 305(2):454–465, November 2022. Publisher: Radiological Society of North America.

[13] Judy Wawira Gichoya, Imon Banerjee, Ananth Reddy Bhimireddy, John L Burns, Leo Anthony Celi, Li-Ching Chen, Ramon Correa, Natalie Dullerud, Marzyeh Ghassemi, Shih-Cheng Huang, Po-Chih Kuo, Matthew P Lungren, Lyle J Palmer, Brandon J Price, Saptarshi Purkayastha, Ayis T Pyrros, Lauren Oakden-Rayner, Chima Okechukwu, Laleh Seyyed-Kalantari, Hari Trivedi, Ryan Wang, Zachary Zaiman, and Haoran Zhang. AI recognition of patient race in medical imaging: a modelling study. *The Lancet Digital Health*, 4(6):e406–e414, June 2022.

[14] Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A. Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673, November 2020. Number: 11 Publisher: Nature Publishing Group.

[15] Polina Kirichenko, Pavel Izmailov, and Andrew Gordon Wilson. Last Layer Re-Training is Sufficient for Robustness to Spurious Correlations, June 2023. arXiv:2204.02937 [cs, stat].

[16] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, May 2015. Number: 7553 Publisher: Nature Publishing Group.

[17] Paul H. Yi, Jinchi Wei, Tae Kyung Kim, Jiwon Shin, Haris I. Sair, Ferdinand K. Hui, Gregory D. Hager, and Cheng Ting Lin. Radiology "forensics": determination of age and sex from chest radiographs using deep learning. *Emergency Radiology*, 28(5):949–954, October 2021.

[18] Ryan Poplin, Avinash V. Varadarajan, Katy Blumer, Yun Liu, Michael V. McConnell, Greg S. Corrado, Lily Peng, and Dale R. Webster. Prediction of cardiovascular risk factors from retinal fundus photographs via deep learning. *Nature Biomedical Engineering*, 2(3):158–164, March 2018. Number: 3 Publisher: Nature Publishing Group.

[19] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely Connected Convolutional Networks, January 2018. arXiv:1608.06993 [cs].

[20] João Alexandre Lôbo Marques, Francisco Nauber Bernardo Gois, João Paulo do Vale Madeiro, Tengyue Li, and Simon James Fong. Chapter 4 - Artificial neural network-based approaches for computer-aided disease diagnosis and treatment. In Akash Kumar Bhoi, Victor Hugo C. de Albuquerque, Parvathaneni Naga Srinivasu, and Gonçalo Marques, editors, *Cognitive and Soft Computing Techniques for the Analysis of Healthcare Data*, Intelligent Data-Centric Systems, pages 79–99. Academic Press, January 2022.

[21] Karl Weiss, Taghi M. Khoshgoftaar, and DingDing Wang. A survey of transfer learning. *Journal of Big Data*, 3(1):9, May 2016.

[22] Rich Caruana. Multitask Learning. *Machine Learning*, 28(1):41–75, July 1997.

[23] John R. Zech, Marcus A. Badgeley, Manway Liu, Anthony B. Costa, Joseph J. Titano, and Eric Karl Oermann. Variable generalization performance of a deep learning model to detect pneumonia in chest radiographs: A cross-sectional study. *PLoS medicine*, 15(11):e1002683, November 2018.

[24] Harrison Edwards and Amos Storkey. Censoring Representations with an Adversary, March 2016. arXiv:1511.05897 [cs, stat].

[25] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-Adversarial Training of Neural Networks, May 2016. arXiv:1505.07818 [cs, stat].

[26] Jenny Yang, Andrew A. S. Soltan, David W. Eyre, Yang Yang, and David A. Clifton. An adversarial training framework for mitigating algorithmic biases in clinical machine learning. *npj Digital Medicine*, 6(1):1–10, March 2023. Number: 1 Publisher: Nature Publishing Group.

[27] Melissa D. McCradden, Shalmali Joshi, Mjaye Mazwi, and James A. Anderson. Ethical limitations of algorithmic fairness solutions in health care machine learning. *The Lancet Digital Health*, 2(5):e221–e223, May 2020. Publisher: Elsevier.

[28] Social Determinants of Health - Healthy People 2030 | health.gov.

[29] Paula Braveman and Laura Gottlieb. The Social Determinants of Health: It's Time to Consider the Causes of the Causes. *Public Health Reports*, 129(Suppl 2):19–31, 2014.

[30] E. M. Lewiecki, N. C. Wright, and A. J. Singer. Racial disparities, FRAX, and the care of patients with osteoporosis. *Osteoporosis international : a journal established as result of cooperation between the European Foundation for Osteoporosis and the National Osteoporosis Foundation of the USA*, 31(11):2069, November 2020. Publisher: NIH Public Access.

[31] Joseph Paul Cohen, Joseph D. Viviano, Paul Bertin, Paul Morrison, Parsa Torabian, Matteo Guarrera, Matthew P. Lungren, Akshay Chaudhari, Rupert Brooks, Mohammad Hashir, and Hadrien Bertrand. TorchXRayVision: A library of chest X-ray datasets and models, October 2021. arXiv:2111.00595 [cs, eess].

[32] Dina Demner-Fushman, Marc D. Kohli, Marc B. Rosenman, Sonya E. Shooshan, Laritza Rodriguez, Sameer Antani, George R. Thoma, and Clement J. McDonald. Preparing a collection of radiology examinations for distribution and retrieval. *Journal of the American Medical Informatics Association*, 23(2):304–310, March 2016.

[33] Chest Radiograph Interpretation with Deep Learning Models: Assessment with Radiologist-adjudicated Reference Standards and Population-adjusted Evaluation.

[34] George Shih, Carol C. Wu, Safwan S. Halabi, Marc D. Kohli, Luciano M. Prevedello, Tessa S. Cook, Arjun Sharma, Judith K. Amorosa, Veronica Arteaga, Maya Galperin-Aizenberg, Ritu R. Gill, Myrna C.B. Godoy, Stephen Hobbs, Jean Jeudy, Archana Laroia, Palmi N. Shah, Dharshan Vummidi, Kavitha Yaddanapudi, and Anouk Stein. Augmenting the National Institutes of Health Chest Radiograph Dataset with Expert Annotations of Possible Pneumonia. *Radiology: Artificial Intelligence*, 1(1):e180041, January 2019. Publisher: Radiological Society of North America.

[35] Xiaosong Wang, Yifan Peng, Le Lu, Zhiyong Lu, Mohammadhadi Bagheri, and Ronald 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)*, pages 3462–3471, July 2017. arXiv:1705.02315 [cs].

[36] Aurelia Bustos, Antonio Pertusa, Jose-Maria Salinas, and Maria de la Iglesia-Vayá. PadChest: A large chest x-ray image dataset with multi-label annotated reports. *Medical Image Analysis*, 66:101797, December 2020.

[37] Jeremy Irvin, Pranav Rajpurkar, Michael Ko, Yifan Yu, Silviana Ciurea-Ilcus, Chris Chute, Henrik Marklund, Behzad Haghgoo, Robyn Ball, Katie Shpanskaya, Jayne Seekins, David A. Mong, Safwan S. Halabi, Jesse K. Sandberg, Ricky Jones, David B. Larson, Curtis P. Langlotz, Bhavik N. Patel, Matthew P. Lungren, and Andrew Y. Ng. CheXpert: A Large Chest Radiograph Dataset with Uncertainty Labels and Expert Comparison, January 2019. arXiv:1901.07031 [cs, eess].

[38] Alistair E. W. Johnson, Tom J. Pollard, Seth J. Berkowitz, Nathaniel R. Greenbaum, Matthew P. Lungren, Chih-ying Deng, Roger G. Mark, and Steven Horng. MIMIC-CXR, a de-identified publicly available database of chest radiographs with free-text reports. *Scientific Data*, 6:317, December 2019.

[39] Joseph Paul Cohen, Mohammad Hashir, Rupert Brooks, and Hadrien Bertrand. On the limits of cross-domain generalization in automated X-ray prediction, May 2020. arXiv:2002.02497 [cs, eess, q-bio, stat].

[40] Jake Lever, Martin Krzywinski, and Naomi Altman. Principal component analysis. *Nature Methods*, 14(7):641–642, July 2017. Number: 7 Publisher: Nature Publishing Group.

[41] The Kolmogorov-Smirnov Test for Goodness of Fit. ISSN: 0162-1459.

[42] Benoit Frenay and Michel Verleysen. Classification in the Presence of Label Noise: A Survey. *IEEE Transactions on Neural Networks and Learning Systems*, 25(5):845–869, May 2014. Conference Name: IEEE Transactions on Neural Networks and Learning Systems.

[43] Nicola K. Dinsdale, Mark Jenkinson, and Ana I. L. Namburete. Deep learning-based unlearning of dataset bias for MRI harmonisation and confound removal. *NeuroImage*, 228:117689, March 2021.

[44] Christian Garbin, Pranav Rajpurkar, Jeremy Irvin, Matthew P. Lungren, and Oge Marques. Structured dataset documentation: a datasheet for CheXpert, May 2021. arXiv:2105.03020 [cs, eess].