Developing a Lightweight Bone Abnormality Classification Model

Charles L Green
Georgia Institute of Technology
CharlesLGreen14@gmail.com

Sai Anurag Pichika
Georgia Institute of Technology
anuragsai97@gmail.com

Saarang Prabhuram
Georgia Institute of Technology
saarang2010@gmail.com

Karthik Subramanian
Georgia Institute of Technology
karkir0003@gmail.com

Abstract

Musculoskeletal radiograph interpretation remains a bottleneck in radiology, particularly in low-resource settings. This study investigates the use of lightweight convolutional neural networks (CNNs) for classifying musculoskeletal radiographs on the MURA dataset. We benchmarked established architectures such as DenseNet, VGG, and ResNet, evaluating their accuracy, Cohen's Kappa, and the number of parameters. To address data imbalance and scalability, we applied data augmentation to enrich and expand the training set and undersampling. Our custom CNNs, including an ensemble of body-part-specific models, demonstrated strong performance, achieving a Cohen's Kappa of 0.591. This outperformed baseline architecture relative to model size, enabling deployment in resource-limited settings. These findings advance the development of scalable, efficient AI tools for radiology, bridging gaps in healthcare accessibility.

1. Introduction

The interpretation of musculoskeletal X-rays has always presented a significant challenge. Radiologists must analyze complex anatomical structures in order to identify abnormalities. Mastery in this field takes years of experience and training, and the process of trying to identify abnormalities itself is time-consuming and is very prone to human error, especially in high-volume clinical settings or areas with limited access to expert radiologists. In recent years, however, AI has taken major steps in this field. Many researchers have come up with many deep-learning models to help categorize and analyze abnormalities in the bodies, however majority of these models prioritized accuracy and performance over computational efficiency, which could be a tad impractical for widespread deployment. Our objective is to come up with a lightweight deep-learning framework that can try to match these models in terms of accuracy and performance while prioritizing computational efficiency.

2. Background/Motivation

2.1. Current Practice and its Challenges

As AI continues to advance and demands for medical imaging by radiologists increase, AI is becoming more of an assistant than ever before. While AI adoption is increasing, reliability remains a concern. A 2020 survey by the American College of Radiology revealed that only one-third of radiologists use AI, with over 90% saying that it has inconsistent performance, and under 6% finding it always working [4, 7]. This shows the need to enhance AI's reliability to build trust and make it an essential diagnostic tool. The challenges radiologists face show why this must occur. Radiologists face immense workloads, spending only 35% of their time on clinical image analysis, leaving about 3 minutes per report on average 1 [5]. Limited analysis time increases errors due to fatigue, contributing to 78% of malpractice cases [1]. For example, analyzing a CT scan with 679 images in a 7-hour workday leaves radiologists with merely a second per image (less with the 1000+ image CRT angiography)[1]. This highlights the urgent need for AI as a diagnostic assistant to alleviate workloads and improve accuracy.

Table 1. Time Allocation for Radiology Clinical Activities

Activity	**Mean Volume**	**Per**	**% of Total Hours**
Reporting X-ray	249	Day	11.9
Reporting CT	54	Day	11.4
Reporting US	80	Day	5.5
Reporting MRI	17	Day	4.4
Reporting mammography	30	Week	0.7
Reporting fluoroscopy	40	Week	0.6
Reporting interventional	10	Day	0.3

2.2. Current AI analysis

Competitions like Stanford ML Group's "Bone X-ray Deep Learning Competition" show AI's potential in muscu-

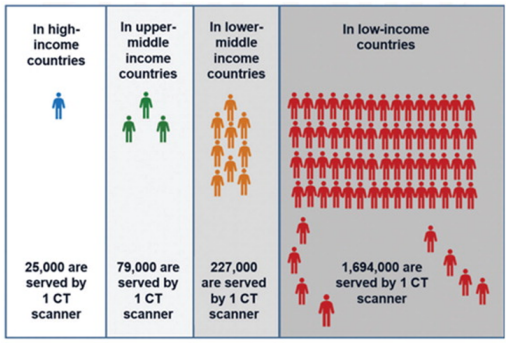

Figure 1. CT accessibility across income levels.

loskeletal anomaly detection [9]. The competition's baseline model, a 169-layer DenseNet Convolutional Neural Network (CNN), had a Cohen's Kappa of .705, which on average was worse than the radiologist, but there are other challenges in addition to this [10].

2.3. Challenges in Current AI Practice

Two key challenges are slowing down AI adoption in radiology: gaining trust and managing resource cost.

First, building trust among radiologist requires models to achieve near-perfect agreement. Cohen's Kappa, which measures inter-rater reliability, is a critical metric for evaluating trustworthiness, and the gold standard in medical studies. According to McHugh's stricter medical standard, achieving a Kappa score above 0.9 may be essential to gain trust in health-related studies 3. While the Stanford competition's DenseNet model achieved a Kappa of 0.7, indicating substantial agreement, higher standards are needed to ensure adoption[6, 10].

Secondly, the cost of deploying large-scale models limited AI accessibility in low-resource settings.For instance, the 14-million-parameters, nearly 60 MB DenseNet requires advance GPUs and significant computational infrastructure [3]. In low- and middle-income countries, such as Tanzania, where only 60 radiologist serve 58 million people, radiology resources are severely constrained[8]. A single CT scanner in low-income regions serves 1.6 million people compared to 25,000 in high-income countries 1. Resource-poor outreach centers, often staffed by volunteers, highlight the pressing need for lightweight, efficient AI solutions 2. The trustworthiness of AI is one of the critical bottlenecks of AI's adoption in high-income countries, the question in lower-income countries is how much benefit can happen where there are low resources and a low amount of radiologists [8]. This is what this project focuses on.

2.4. Impact and Vision

The success of this research could transform diagnostic workflows in underserved regions by providing lightweight AI models that deliver accurate, efficient analyses without requiring advanced computational resources. In low-resource settings, these models can serve as vital aids, reducing the burden on radiologist and mitigating errors caused by fatigue. By automating abnormality detection, radiologist can allocate more time to complex cases, ultimately improving patient outcomes.

Figure 2. Photograph of medical imaging clinic in resource-poor rural outreach center in the Dominican Republic

Value Range	Cohen's Interpretation	McHugh's Interpretation
Below 0.20	None to slight agreement	No agreement
.21–.39	Fair agreement	Minimal agreement
.40–.59	Moderate agreement	Weak agreement
.60–.79	Substantial agreement	Moderate agreement
.80–.90	Almost perfect agreement	Strong agreement
Above .90	Almost perfect agreement	Almost perfect agreement

Figure 3. Cohen's vs McHugh's Cohen Kappa Standards

Our vision is to address a critical global health challenge by creating scalable, reliable AI-driven diagnostic tools. With their lightweight, efficient design, these models can democratize access to quality radiology, bridging the gap between low- and high-resource settings.

3. MURA Dataset

3.1. Motivation

The MURA dataset [10] focuses on musculoskeletal radiographs, a critical area in radiology where diagnostic errors and workload challenges are prevalent. Its large-scale, labeled X-rays allow for the development of lightweight AI models for assisting radiologist.

3.2. Collection Process and Composition

Curated by the Stanford Machine Learning Group [9], MURA is made up of 40,561 studies of seven upper extremity body parts (wrist, shoulder, hand, finger, elbow, forearm, and humerus), labeled as "normal" or "abnormal" based on radiologist consensus. Table 2 summarizes the dataset's composition before and after augmentation, split into training and validation subsets.

3.3. Preprocessing

To prepare the dataset for training, the following preprocessing pipeline was applied:

- Resizing to 224×224 pixels for uniform input.
- Normalization using ImageNet mean and standard deviation.

Table 2. MURA Dataset Composition with Augmentation (Train/Validation)

Body Part	Original (Train)	Aug. (Train)	Original (Valid)	Aug. (Valid)
Wrist	9752	39008	659	2636
Shoulder	8379	33516	563	2252
Hand	5543	22172	460	1840
Finger	5106	20424	461	1844
Elbow	4931	19724	465	1860
Forearm	1825	7300	301	1204
Humerus	1272	5088	288	1152

- Augmentation with flips, rotations, affine transformations, color jittering, and perspective distortions.
- Undersampling to balance "normal" and "abnormal" cases during training.

The augmentation process quadrupled the training set size, resulting in a total of 147,232 training samples and 3,197 validation samples.

The training set shows varying negative-to-positive ratios across body parts, with approximately 1.46:1 for the elbow, 1.59:1 for the finger, 1.76:1 for the forearm, 2.74:1 for the hand, 1.12:1 for the humerus, 1.01:1 for the shoulder, and 1.45:1 for the wrist. The validation set (no augmentation) maintains a similar balance, with ratios of 1.02:1 for the elbow, 0.87:1 for the finger, 0.99:1 for the forearm, 1.43:1 for the hand, 1.06:1 for the humerus, 1.03:1 for the shoulder, and 1.23:1 for the wrist.

4. Approach

4.1. Experiment Process

4.1.1 Models Explored

- Experimented with several well-known architectures, including ResNet, VGG, and DenseNet.
- Selected lightweight versions of these architectures as they were also featured in the MURA leaderboard [9], aiming for comparable performance.
- Designed and trained a lightweight Custom CNN tailored for the MURA dataset, optimized for resource-constrained environments to balance efficiency and performance.
- Enhanced the Custom CNN by incorporating attention mechanisms and developing a specialized ensemble approach with body-part-specific models to improve accuracy and robustness.

4.1.2 Evaluation Process

In order to evaluate how well the models perform relative to human performance, we use the Cohen's Kappa κ statistic. This statistic is used for measuring how frequently "raters" agree on the labels for each example in a dataset [6]. We also calculated the following evaluation metrics in order to understand model performance and size: Parameter Count, Accuracy, Precision, Recall, and Specificity.

To interpret the model's behavior, we leveraged Grad-CAM to visualize the last convolutional layer of each architecture in order to see what higher level features were learned [12].

4.2. Training Details

The training pipeline for all models followed a consistent setup:

- **Training Configurations:** Learning rate (`1e-3`), weight decay (`1e-4` and `1e-5` for VGG models), batch size (`32`) epochs (`10`) , and binary cross-entropy loss with dynamic class weights.
- **Optimizer:** Adam optimizer with learning rate scheduler (ReduceLROnPlateau) for adaptive adjustments.
- **Architectures:** Explored both Custom and Pre-trained architectures including DenseNet-169, DenseNet-121, ResNet-18, ResNet-34, VGG-11, and VGG-13.
- **Pre-trained Configuration:** Pre-trained models were seeded with weights from the ImageNet dataset and were fine-tuned. Weights in were frozen, except in the final linear layer to preserve the learning.[11]

4.3. Challenges Encountered

- **Computing Resources Used:** Google Colab, PACE-ICE (HPC resource provided by GaTech), 2 local GPUs for extended training sessions
- **Duration:** Limited time per GPU session (eg: 8 hour session with 2 GPU cluster on PACE-ICE)

5. Experiments and Results

5.1. DenseNet Models

5.1.1 DenseNet169 and DenseNet121 Architectures

DenseNet169 and DenseNet121, introduced by Huang et al. [3], are CNN architectures featuring dense connectivity, where each layer connects to all subsequent layers via feature concatenation. Their architectures consist of multiple dense blocks, each containing convolutional layers followed by Batch Normalization and ReLU activations, connected through transition layers that use convolutions and average pooling. While both share the same foundational design, DenseNet121 is lighter and computationally more efficient whereas DenseNet169 provides a deeper structure capable of capturing finer details in the data.

5.1.2 Custom DenseNet Model Architecture

The proposed model is a lightweight DenseNet variant with three DenseBlocks (growth rate: 32) and Dropout (p=0.2) for regularization. Transition layers include 1×1 convolutions and average pooling. Key modifications include:

- **Compact Architecture**: Three smaller DenseBlocks and two Transition Layers reduce model size.
- **Growth Rate**: The model maintains a consistent growth rate. This reduces number of layers per block.

Table 3. Performance Metrics for All Models

Model	Params	Accuracy	Precision	Recall	Specificity	Cohen's Kappa
DenseNet-169	12M	0.7745	0.8003	0.7046	0.8386	0.5459
DenseNet-121	7M	0.7919	0.8074	0.7425	0.8374	0.5818
DenseNet-169 (*PT)	12M	0.8305	0.8569	0.7752	0.8812	0.6589
DenseNet-121 (*PT)	7M	0.8255	0.8394	0.7856	0.8620	0.6493
Custom DenseNet	540k	0.7272	0.8225	0.5483	0.8914	0.4458
VGG11 (*PT)	128.7M	0.6728	0.6847	0.5863	0.7522	0.3405
VGG13 (*PT)	128.9M	0.6850	0.7107	0.5765	0.7846	0.3639
Custom VGG11	128.8M	0.6925	0.8169	0.4608	0.9052	0.3727
Custom VGG13	128.9M	0.7222	0.7734	0.5935	0.8404	0.4381
ResNet-18	11.2M	0.7417	0.7711	0.6542	0.8218	0.4790
ResNet-34	21.3M	0.7134	0.7304	0.6359	0.7846	0.4228
ResNet-18 (*PT)	11.2M	0.6703	0.6758	0.5890	0.7367	0.3362
ResNet-34 (*PT)	21.3M	0.6578	0.6251	0.7117	0.6082	0.3182
Custom CNN 1	1.3M	0.7647	0.8220	0.6490	0.8710	0.5250
Custom CNN With Attention	55,644	0.7341	0.7120	0.7464	0.7229	0.4683
Custom CNN Ensemble	1.3M	0.7961	0.8310	0.7203	0.8656	0.5891

**PT = Pre Trained on ImageNet*

- **Dropout**: Dropout is applied after each convolutional layer to prevent overfitting.
- **Classifier**: An Adaptive Avg. Pooling layer and a single fully connected layer output binary classification.

5.1.3 Results

Table 3 contains performance metrics of both pre-trained DenseNet models and custom DenseNet models. Figure 4 is the grad cam visualizations of the best performing DenseNet model.

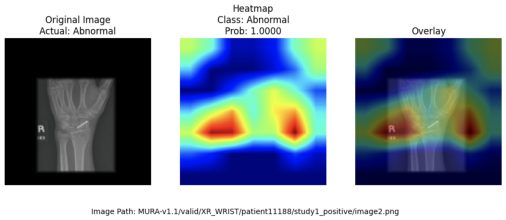

Figure 4. Grad-CAM visualizations for DenseNet169

5.1.4 Analysis

- Pre-trained DenseNet models outperform non-pretrained counterparts, highlighting the impact of transfer learning in medical imaging tasks.
- Pre-trained DenseNet169 achieved the highest accuracy and Cohen's Kappa score, with DenseNet121 as a close second despite its smaller size.
- The custom compact DenseNet model, designed for resource-constrained settings, showed competitive specificity and precision but low recall.
- The reduced capacity of the custom model may limit its ability to capture fine-grained fracture details.
- DenseNet169 demonstrated high Kappa scores but showed inefficiencies in localization based on grad-cam visualizations.

5.2. VGG Models

VGG, or Visual Geometry Group, is a CNN developed at the University of Oxford to tackle image classification problems by increasing model depth while maintaining simple, uniform convolution layers.

5.2.1 General Model Architecture

Simonyan et al.[13] present various architectures, including VGG11, VGG13, and VGG16, differing only by network depth. All these models feature blocks of convolution and max-pooling layers, ending with fully connected layers for classification. We primarily focus on VGG11 and VGG13 for computational efficiency. Below is a more in-depth look into the general architecture:

- All convolution blocks in the VGG networks contain 1,2, or 3 convolution layers that use 3x3 filters, Each convolutional layer is followed by a ReLU layer.
- A Max-pooling layer follows every convolution block
- The last layers of the model are fully connected layers for classification

5.2.2 Custom VGG11 and VGG13 Architecture

The Custom VGG11 and VGG13 architectures, inspired by the original models [13], include Batch-Norm layers after each convolutional layer and one Average-Pool layer before the fully-connected layers. These additions hope to improve generalization and reduce over-fitting of the models. The training configuration follows the same pattern as Section 4.2, with gradient flow enabled for weight updates.

5.2.3 Results

Table 3 provides some of the metrics and results of both the custom VGG11 and VGG13 models as well as their pre-trained versions. Figure 5 is a Grad-CAM visualization of the custom model.

5.2.4 Analysis

- VGG11 achieved a Cohen's Kappa of 0.37, indicating fair agreement according to Cohen's Interpretation (Figure 3), with higher specificity, making it better at predicting negative cases.

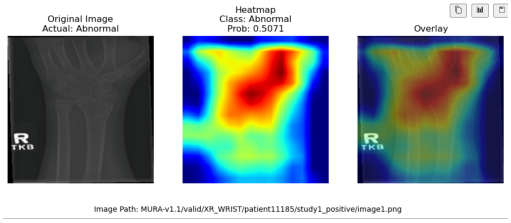

Figure 5. Grad-CAM visualizations for custom VGG13

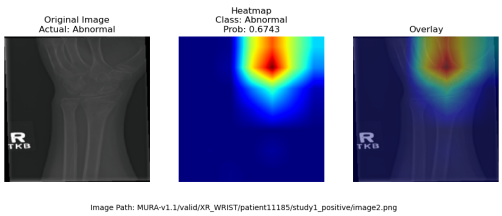

Figure 6. Grad-CAM visualizations for custom ResNet18

- VGG13 achieved a Cohen's Kappa of 0.43, indicating moderate agreement according to Cohen's Interpretation (Figure 3), with 3% higher accuracy and better overall generalization and learning.
- Both models, while smaller than VGG16 or VGG19, still have a high number of parameters, resulting in "per-epoch" training times of approximately 30 mins.
- Increasing network depth enhances performance but comes at the expense of a larger model and longer training periods.
- To achieve both efficiency and sufficient performance, further strategies had to be explored

5.3. ResNet Models

5.3.1 General Model Architecture

ResNet is a type of CNN architecture structured similar to VGG, but contains residual connections. These residual connections allow data to flow through a convolutional layer or skip that block, allowing the model to learn if a given convolutional layer is important or not to determine the output [2]. The key architectural components for ResNet 18 and 34 are:

- **Initial Convolution Layer**: Convolution, Batch Norm, ReLU, Max Pool
- **Convolutional Block ("building block")**: Convolution, Batch Norm, ReLU, Residual Connection.
- **Identity Shortcut**: Adds input to the block's output
- **Projection Shortcut**: Used when increasing channels between blocks.

5.3.2 Custom Resnet18 and Resnet34 Architecture

We explore training a ResNet-18 and Resnet-34 model from scratch. The training configuration is as mentioned in Section 4.2.

5.3.3 Results

Table 3 provides some of the metrics and results of both the custom ResNet-18 and ResNet-34 models as well as their pre-trained versions. Figure 6 is a sample Grad-CAM visualization of the custom model.

5.3.4 Analysis

- Pretrained ResNet Cohen's kappa score(s) of 0.3362 and 0.3182 reflect fair agreement
- Pretrained ResNet-18 has a higher specificity than Pretrained ResNet-34, meaning that this model is better at predicting negative cases amongst the negative labeled examples. Pretrained ResNet-34 performed better on recall, meaning that this model is better at predicting positive cases amongst the positive labeled examples.
- Improved Cohen's Kappa for custom ResNet-18 and ResNet-34 are improved compared to pretrained, representing moderate agreement
- The difference may stem from pretrained weights coming from ImageNet, which focus on everyday objects, limiting generalizability to medical imaging datasets.

5.4. Custom CNN Models

5.4.1 Custom CNN Architectures

We designed three variations of lightweight CNNs to classify musculoskeletal radiographs effectively while addressing computational efficiency. Each architecture shares a common foundation but introduces unique modifications to balance performance and resource constraints.

1. Baseline Architecture (`CustomCNN1`): The baseline model consists of three stages:

- **Stem Block:** Three convolutional layers with 3×3 kernels, batch normalization, ReLU activation, and a max-pooling layer for initial feature extraction.
- **Middle Block:** Three additional convolutional layers with batch normalization, ReLU activation, and max-pooling for down-sampling.
- **Final Block:** Two convolutional layers, followed by global average pooling, a dropout layer (`rate=0.3`), and a fully-connected classification layer.

2. Attention-Enhanced Architecture (`CustomCNN WithAttention`): Building based on `CustomCNN1`, this variant incorporates:

- **Depthwise Separable Convolutions:** Reducing computational complexity and parameter count.
- **Squeeze-and-Excitation (SE) Modules:** Dynamically reweighting feature maps to focus on clinically relevant regions.

3. Ensemble Strategy: This approach uses specialized `CustomCNN1` models trained independently for each body part:

- **Knowledge Transfer:** Weights from body-part-specific models initialize the main model, enhancing

generalization.

- **Performance Enhancement:** This ensemble method significantly improves accuracy and robustness, despite not being directly deployable.

5.4.2 Results

Table 3 compares DenseNet-169 with Custom CNN models, highlighting trade-offs in performance and computational efficiency. Key observations:

- CustomCNN1 achieves competitive accuracy (0.7647) and Cohen's Kappa (0.525) with minimal parameters, making it suitable for resource-constrained deployments.
- CustomCNNWithAttention prioritizes interpretability with a compact design (55,644 parameters) but lower performance (Kappa = 0.4683).
- The ensemble approach, though not deployed, significantly improves the main model's generalization through weight transfer, achieving a Cohen's Kappa of 0.5891.

5.4.3 Analysis

- DenseNet-169 achieves the highest Cohen's Kappa (0.6590), indicating substantial agreement, but its high parameter count limits deployability.
- CustomCNN1 balances lightweight design (1.3M parameters) and moderate agreement (Kappa = 0.525), ideal for constrained environments.
- CustomCNNWithAttention achieves moderate agreement (Kappa = 0.4683) in an ultra-compact design, suitable for edge devices prioritizing interpretability.
- The ensemble strategy, enhances the main model's performance (Kappa = 0.5891) through specialized training on specific body-parts.

For medical applications, high Cohen's Kappa values establish trustworthiness. While DenseNet-169 outperforms in agreement, `CustomCNN1` and `CustomCNNWithAttention` achieve substantial and moderate agreement, respectively, within significantly smaller footprints. Figure 7 shows Grad-CAM visualizations for the `CustomCNN1 Ensemble`, emphasizing clinically relevant focus areas.

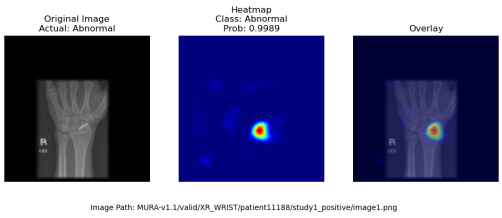

Figure 7. Grad-CAM visualizations for Custom CNN with Ensemble learning

5.4.4 Deployment Feasibility

Table 3 highlights each model's suitability for deployment:

- CustomCNN1: Optimal for low-resource environments due to its balance of accuracy (0.7647) and compact design (1.3M parameters).
- CustomCNNWithAttention: Its ultra-lightweight architecture (55,644 parameters) is ideal for edge devices with storage constraints.
- Custom CNN Ensemble: Enhances main model generalization (through knowledge transfer) with same final size.
- DenseNet-169: Superior performance (Kappa = 0.6589) but unsuitable for edge or low-resource scenarios due to its size.

Optimizations like quantization, pruning, and on-device learning could further enhance deployability.

6. Discussion and Conclusion

This study explored the use of lightweight convolutional neural networks (CNNs) for musculoskeletal radiograph analysis, demonstrating the potential of these models to address critical challenges in radiology. By experimenting with Custom CNNs, DenseNet, VGG, and ResNet architectures, we saw key trade-offs between performance, computational efficiency, and deployment feasibility. Our findings highlight the following:

- Custom CNN models, particularly the ensemble approach, achieved comparable performance to state-of-the-art models like DenseNet-169 while being significantly smaller in size and faster in inference.
- Incorporating attention mechanisms improved model interpretability, enhancing their suitability for clinical adoption.
- Data augmentation and dynamic class weighting effectively addressed dataset imbalance and improved generalization.

These results demonstrate the viability of lightweight AI models for resource-constrained settings, with opportunities to further address limitations. Future work should focus on:

- **Refining and Optimizing Architectures:** Explore advanced lightweight architectures (e.g., MobileNet, EfficientNet) and apply techniques like quantization and pruning to enhance efficiency.
- **Real-World Validation and Deployment:** Collaborate with hospitals for clinical trials to validate model performance and usability.
- **Expanding Applications:** Extend the framework to additional imaging modalities (e.g., MRI, CT scans).

By pursuing these directions, this project can advance lightweight, scalable AI-driven diagnostic tools, helping to bridge gaps in global healthcare and improve patient outcomes in diverse settings.

References

[1] Robert Alexander, Stephen Waite, Michael A. Bruno, Elizabeth A. Krupinski, Leonard Berlin, Stephen Macknik, and Susana Martinez-Conde. Mandating limits on workload, duty, and speed in radiology. *Radiology*, 304(2):274–282, 2022. PMID: 35699581. 1

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. 5

[3] Gao Huang, Zhuang Liu, and Kilian Q. Weinberger. Densely connected convolutional networks. *CoRR*, abs/1608.06993, 2016. 2, 3

[4] Curtis P. Langlotz. Will artificial intelligence replace radiologists? *Radiology: Artificial Intelligence*, 1(3):e190058, 2019. PMID: 33937794. 1

[5] Sharyn LS MacDonald, Ian A Cowan, Richard A Floyd, and Rob Graham. Measuring and managing radiologist workload: A method for quantifying radiologist activities and calculating the full-time equivalents required to operate a service. *Journal of Medical Imaging and Radiation Oncology*, 57(5):551–557, 2013. 1

[6] Mary L McHugh. Interrater reliability: the kappa statistic. *Biochem. Med. (Zagreb)*, 22(3):276–282, 2012. 2, 3

[7] Claudia Mello-Thoms and Carlos A B Mello. Clinical applications of artificial intelligence in radiology. *British Journal of Radiology*, 96(1150):20221031, 04 2023. 1

[8] Daniel J. Mollura, Melissa P. Culp, Erica Pollack, Gillian Battino, John R. Scheel, Victoria L. Mango, Ameena Elahi, Alan Schweitzer, and Farouk Dako. Artificial intelligence in low- and middle-income countries: Innovating global health radiology. *Radiology*, 297(3):513–520, 2020. PMID: 33021895. 2

[9] Pranav Rajpurkar. Bone x-ray deep learning competition. 2, 3

[10] Pranav Rajpurkar, Jeremy Irvin, Aarti Bagul, Daisy Ding, Tony Duan, Hershel Mehta, Brandon Yang, Kaylie Zhu, Dillon Laird, Robyn L. Ball, Curtis Langlotz, Katie Shpanskaya, Matthew P. Lungren, and Andrew Y. Ng. Mura: Large dataset for abnormality detection in musculoskeletal radiographs, 2018. 2

[11] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *CoRR*, abs/1409.0575, 2014. 3

[12] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-CAM: Visual explanations from deep networks via gradient-based localization. Oct. 2016. 3

[13] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. 4

Table 4. Contributions of team members.

Student Name	Contributed Aspects	Details
Karthik Subramanian	Data Engineering, Model Training/Experimentation, Code Reviewer	Setup infrastructure in AWS to store dataset in the cloud. Created utility functions to make it easier to work with MURA dataset and run experiments in the Colab Notebook (eg: fixing bugs with file paths and abstracting the dataset loading logic). Performed 3-4 experiments related to training pretrained and from scratch ResNet 18 and 34 architectures on MURA dataset in hopes to improve the Cohen's Kappa score. Reviewed 13 Pull Requests from the team and provided helpful comments/clarifying questions relating to the project direction.
Sai Anurag Pichika	Custom DenseNet Design, Development, and Experimentation Development of DenseNet121 and DenseNet169 from scratch Experimented with DenseNet pretrained models Models Training, Performance Benchmarking, and Optimization Under-Sampling Techniques for Class Imbalance Mitigation Modular Training Framework Development	Designed and implemented a lightweight custom DenseNet model from scratch, optimized for size and efficiency. Developed DenseNet121 and DenseNet169 models from scratch. Also, experimented with DenseNet121 and DenseNet169 pretrained architectures. To address class imbalance, applied under-sampling techniques, improving model performance through balanced training. Developed a modular training framework to streamline training, hyperparameter tuning, and evaluation across models, enabling systematic performance comparisons.
Saarang Prabhuram	Design and Trained Basic Custom Densenet model, Repository Refactoring, Development, Training, and Experimentation of VGG models, both pre-trained and custom	Was involved in the initial design and training of a basic custom densenet model. Refactored the repository for easy addition, experimentation, and training of the models. Refactored and cleaned the Jupyter Notebook for better readability and maintainability and for usage locally and on Google Colab. Performed experiments with pretrained VGG11 and VGG13 models. Coded, experimented, and trained custom VGG11 and VGG13 models on the MURA dataset to improve Cohen's Kappa score.

Table 5. Contributions of team members.

Student Name	Contributed Aspects	Details
Charles L Green	Data Preprocessing	Created 'image_utils.py' to process data, including creating a load entity and data transformations for training and validation sets.
	Metrics Collection	Developed 'metrics.py' to generate confusion matrix, ROC curve, class weights, Cohen's Kappa with confidence interval, and other metrics.
	Visualization Creator	Created 'visualization.py' using Grad-CAM for heatmap visualizations explaining model predictions. Functions include finding last convolutional layer, running Grad-CAM on specific batches/body parts/images.
	Custom CNN (1) Model Development	Developed a custom CNN architecture with stem, middle, and final convolutional blocks, followed by global average pooling, dropout, and a final linear layer for binary classification.
	Custom CNN with Attention Model Development	Based on the custom CNN, created a model using depthwise separable convolutions for reduced computational cost and Squeeze-Excitation blocks for dynamic feature reweighting, followed by global average pooling and a final classification layer.
	Lightweight CNN Model Development	Developed a lightweight CNN model for efficient body part classification using convolutional and pooling layers, followed by a final classification layer.
	Ensemble Based Learning Development	Created a notebook using an ensemble of body part models to improve the main model's performance by setting weights based on the body part models.
	Dynamic Class Weights Computation	Developed a function to dynamically compute class weights.
	Model Training/Experimentation	Trained various models (Custom DenseNet, Custom CNNs, Lightweight CNN, Ensemble) and optimized hyperparameters for both computational efficiency and improved Cohen's Kappa and accuracy.

Table 6. Time Allocation for Radiology Activities

Activity	**Mean Volume**	**Per**	**Number/ Month**	**Time/ Activity (min)**	**Mean Hours/ Day**	**Mean Hours/ Month**	**% of Total Hours**
Reporting X-ray	249	Day	5410	3.42	14.2	303	11.9
Reporting CT	54	Day	1173	15.08	13.6	295	11.4
Reporting US	80	Day	1738	4.90	6.5	142	5.5
Reporting MRI	17	Day	369	18.33	5.2	113	4.4
Reporting mammography	30	Week	130	8.90	1.7	19	0.7
Reporting fluoroscopy	40	Week	174	5.00	0.7	14	0.6
Reporting interventional	10	Day	217	2.33	0.4	8	0.3
Procedure diagnostic	39	Day	847	15.00	9.8	212	8.2
Advanced diagnostic studies	10.4	Day	226	50.00	8.7	188	7.3
Procedure interventional	5.1	Day	110	65.00	5.5	119	4.6
Procedure fluoroscopy	27	Week	117	30.00	2.7	59	2.3
Procedure mammography	26	Week	113	15.00	1.8	28	1.1
Referral-related admin	191	Day	4150	1.00	3.2	69	2.7
Informal case discussions	120	Week	521	30.00	12.0	261	10.1
Conference preparation	49	Week	213	35.00	5.1	124	4.8
Conference run-time	49	Week	213	50.00	8.1	177	6.8
Registrar, fellow supervision	12	Day	261	90.00	18.0	391	15.1
Tutorial preparation	9	Week	39	35.00	1.0	23	0.9
Tutorial run-time	9	Week	39	60.00	1.8	39	1.5