MURA Multiclass Classification with Transfer Learning and Explainability Maps

Nikhil Thota
MS in Data Analytics
016791996
nikhil.thota@sjsu.edu

HV Sai Dumpala
MS in Data Analytics
016704805
hemavenkatasai.dumpala@sjsu.edu

Pavan Kumar Kasarla
MS in Data Analytics
016761732
pavankumarreddy.kasarla@sjsu.edu

Irfan Shaik
MS in Data Analytics
016762070
irfan.shaik@sjsu.edu

***Abstract*—This project addresses the problem of multiclass classification of musculoskeletal radiographic abnormalities using the MURA (Musculoskeletal Radiographs) dataset. The existing work on MURA has primarily focused on binary classification, distinguishing between normal and abnormal cases, with the current state-of-the-art achieving a kappa score of 0.843. However, this project aims to extend the problem statement to multiclass classification, predicting the specific body part and abnormality type. The approach taken in this project involves fine-tuning a DenseNet121 architecture using transfer learning techniques and incorporating self-attention mechanisms. The DenseNet121 model, pre-trained on the ImageNet dataset, is fine-tuned on the MURA dataset to classify radiographic images into 14 classes, including normal and abnormal conditions for different body parts (e.g., hand, wrist, elbow, shoulder, and finger). Self-attention modules are integrated into the DenseNet121 architecture to enhance the model's ability to focus on relevant regions and capture long-range dependencies within the images. To enhance the interpretability and transparency of the model's decision-making process, this project incorporates explainability techniques such as attention plots, saliency maps, and Grad-CAM maps. These techniques provide visual explanations by highlighting the regions in the input images that contribute most to the model's predictions. The main results of this project include a confusion matrix and classification report. The confusion matrix reveals that the model struggles with certain classes, such as "Abnormal-HAND," where there is a high number of misclassifications. The classification report shows an overall accuracy of 0.79, with precision, recall, and F1-scores varying across the 14 classes. The highest F1-score of 0.86 is achieved for the "Normal-WRIST" class. The novelty and contribution of this project lie in the extension of the MURA dataset from a binary classification problem to a multiclass classification problem, the utilization of transfer learning with the DenseNet121 architecture augmented with self-attention mechanisms, and the incorporation of explainability techniques for enhanced transparency and interpretability of the model's decisions.**

I. Introduction

In the realm of medical imaging and radiology, the accurate and efficient detection of abnormalities in musculoskeletal radiographs is crucial for early diagnosis, precise treatment planning, and improved patient outcomes. Musculoskeletal fractures, particularly in older adults, present significant health risks. According to the study "Mortality in Older Adults Following a Fragility Fracture," the one-year mortality rate after a hip fracture is alarmingly high—21.5% in women and 32.3% in men aged 65 and older. For non-hip fractures like femur, pelvis, and vertebral fractures, the one-year mortality rates range from 17.9% to 20.2%. Additionally, hip, rib, pelvis, humerus, and vertebral fractures together contribute to 79.3% of all deaths within one year post-fracture in this older adult cohort. The "Burden of Musculoskeletal Diseases" report further highlights the prevalence and severity of these injuries. In 2013, around 900,000 patients with upper and lower limb fractures were hospitalized each year in the United States. For adults aged 65 and older, 66% of lower limb fractures treated in emergency departments were severe enough to require hospital admission. Among those hospitalized, 58% of patients with lower limb fractures were discharged to skilled nursing or intermediate care facilities, indicating the significant impact on healthcare resources and the need for effective treatment planning. These statistics underscore the critical need for advanced diagnostic tools in the early detection and classification of musculoskeletal fractures. Accurate and detailed analysis of radiographic images can significantly aid in early diagnosis, effective treatment planning, and ultimately improve patient outcomes.

The integration of deep learning techniques in medical imaging has shown significant promise in addressing these needs. The MURA (Musculoskeletal Radiographs) dataset, introduced in a Stanford Competition to improve deep learning methodologies for Image Classification, has been instrumental in developing deep learning models for detecting abnormalities in musculoskeletal radiographs. However, existing methods predominantly focus on binary classification, distinguishing between normal and abnormal cases. This binary approach, while informative, does not provide sufficient granularity for clinical use where identifying the specific body part and type of abnormality is essential for targeted diagnosis and treatment. Moreover, the decision-making process of deep learning models often remains opaque, which poses a significant challenge in clinical settings. Medical professionals require a clear understanding of how these models arrive at their conclusions to trust and effectively utilize them. The lack of interpretability in existing models prevents clinicians from validating the AI's reasoning, making it difficult to integrate these tools into clinical workflows confidently.

Despite the utility of the MURA dataset, most research utilizing it has remained confined to binary classification tasks. This limitation hinders the potential for more detailed diagnostic insights that could be derived from these radiographic images.

Specifically, the literature lacks approaches that can effectively handle multiclass classification, which involves identifying the specific body part affected (e.g., hand, wrist, elbow, shoulder, finger) and the type of abnormality present. Additionally, there is a need for models that provide transparency in their decision-making processes to facilitate acceptance and trust among clinicians.

This project addresses the aforementioned gaps by extending the classification task to a multiclass framework. The primary research question is: *Can a deep learning model, fine-tuned on the MURA dataset using transfer learning techniques and augmented with self-attention mechanisms, effectively classify musculoskeletal radiographic images into multiple classes, accurately identifying the body part and type of abnormality?* To achieve these objectives, the project employs a DenseNet121 architecture, pre-trained on the ImageNet dataset, and fine-tunes it on the MURA dataset using transfer learning techniques. DenseNet121 is selected due to its efficiency in feature propagation, parameter efficiency, and reduced vanishing gradient problem. The model is modified to incorporate self-attention modules to enhance its ability to focus on relevant regions and capture long-range dependencies within the images.

- **Pre-training and Fine-tuning**: The DenseNet121 model, pre-trained on ImageNet, is fine-tuned on the MURA dataset. The final fully connected layer is replaced with a new layer tailored to the multiclass classification task. Transfer learning accelerates the training process and improves model performance due to the pre-learned features from ImageNet.
- **Self-Attention Mechanisms**: Self-attention modules are integrated into the DenseNet121 architecture. These modules allow the model to weigh the importance of different parts of the input image, thereby focusing on the most relevant features for classification and improving the model's ability to capture spatial dependencies.

To ensure transparency and interpretability, the project incorporates several explainability techniques:

- **Attention Maps**: These maps visualize the regions of the input image that the self-attention modules focus on. By highlighting relevant regions, attention maps provide insights into which parts of the image are most influential in the model's predictions.
- **Saliency Maps:** Saliency maps identify the pixels or regions that contribute most to the model's output by computing gradients of the output with respect to the input image. This helps in understanding which features or patterns the model relies on.
- **Grad-CAM (Gradient-weighted Class Activation Mapping)**: Grad-CAM generates heatmaps by computing gradients of the target class with respect to the feature maps of convolutional layers. These heatmaps highlight the important regions in the input image, providing a visual explanation of the model's decisions. These maps visualize the regions of the input image that the self-attention modules focus on. By highlighting relevant regions, attention maps provide insights into which parts of the image are most influential in the model's predictions.

The main contributions of this project are threefold. First, it transitions the classification task from binary to multiclass within the MURA dataset, enabling the prediction of specific body parts and types of abnormalities. This extension provides more detailed diagnostic insights that are crucial for targeted diagnosis and treatment. Second, the project enhances model architecture by utilizing transfer learning with DenseNet121, augmented with self-attention mechanisms. This integration improves the model's focus and ability to capture dependencies, thereby enhancing classification accuracy. Third, the project incorporates explainability techniques such as attention maps, saliency maps, and Grad-CAM. These techniques offer visual explanations of the model's decisions, enhancing transparency and fostering trust among medical professionals. Future research can build on this foundation by exploring further enhancements in model architecture, such as integrating additional attention mechanisms or experimenting with different pre-training strategies. Additionally, expanding the dataset with more annotated images and exploring other medical imaging modalities could further improve the robustness and applicability of the models developed in this project. Such advancements would ensure that the models remain relevant and effective in a broader range of clinical scenarios.

II. Literature Survey

The development of convolutional neural networks (CNNs) has significantly advanced the fields of machine learning and computer vision. This journey began with [1] introducing the LeNet architecture for handwritten digit recognition. LeNet, using convolutional layers, pooling layers, and fully connected layers, achieved a 99.05% accuracy on the MNIST dataset, though it was limited to this task. Following this, [2] AlexNet was developed for large-scale image classification on the ImageNet dataset. AlexNet significantly outperformed previous methods with a top-5 error rate of 15.3% and a top-1 error rate of 37.5%, introducing ReLU activation, dropout, and data augmentation, but it required powerful GPUs for training. To better understand CNN representations, [3] developed techniques such as deconvolutional networks and activation maximization, which provided insights into CNN behavior, although these techniques were architecture-specific. Subsequently, [4] introduced GoogLeNet, featuring the Inception module, which improved computational efficiency and achieved a top-5 error rate of 6.67% on ImageNet but required careful hyperparameter tuning. To address the degradation problem in very deep networks, [5] introduced residual connections in ResNets, enabling the training of extremely deep networks and achieving state-of-the-art performance with a top-5 error rate of 3.57% on ImageNet, though these connections required careful design consideration.

[6] introduced DenseNet, a CNN architecture with improved feature propagation and reuse, achieving state-of-the-art performance with a top-5 error rate of 5.77% on ImageNet while

being more parameter-efficient. However, DenseNets required careful regularization to prevent overfitting. In [7], Squeeze-and-Excitation (SE) blocks were introduced to model channel-wise relationships in CNNs, enhancing feature discriminability and improving performance on various tasks, such as achieving a top-5 error rate of 4.47% on ImageNet when integrated into existing architectures, despite adding computational overhead. In [8], EfficientNet was introduced with a compound scaling method to uniformly scale network dimensions, achieving state-of-the-art accuracy with a top-5 error rate of 2.8% on ImageNet and efficiency trade-offs on various datasets, although finding optimal scaling factors could be computationally expensive. The self-attention mechanism in the Transformer architecture, introduced by [9], allowed the model to capture long-range dependencies more effectively, achieving state-of-the-art performance in machine translation with a BLEU score of 41.8 on the WMT 2014 English-to-German translation task and later adapted for image classification by [10]. The Vision Transformer (ViT) achieved competitive performance with a top-1 accuracy of 88.36% on ImageNet, though it required large-scale pretraining.

For visual explanations of CNN predictions, [11] developed Grad-CAM, which provided interpretable visualizations by highlighting important regions in input images and aligning well with human intuition, though it was limited to coarse-grained explanations. [12] introduced integrated gradients for feature attribution, offering a principled approach for interpretable explanations, satisfying various axioms, and demonstrating effectiveness with a top-1 accuracy increase of up to 1.5% on image classification tasks, albeit computationally expensive. The Attention Branch Network (ABN), introduced by [13], aimed to improve interpretability and performance in image classification tasks by learning to attend to relevant regions in input images, achieving competitive performance with top-1 accuracy improvements of 1-2% on various datasets. [14] introduced Score-CAM, an improved visual explanation technique for CNNs based on linear score propagation, providing more precise and interpretable visualizations than previous methods and demonstrating improved interpretability without compromising classification accuracy. To further explore Transformer interpretability, [15] demonstrated various techniques such as integrated gradients, saliency maps, and concept activation vectors, applicable across different tasks and domains, achieving improved interpretability in tasks like sentiment analysis and image classification with consistent performance metrics.

In the domain of medical imaging, [16] developed an ensemble of deep learning models for abnormality detection in musculoskeletal radiographs using the MURA dataset, achieving a high AUC-ROC of 0.892 for abnormality detection but limited to binary classification. [17] provided an overview of AI and neural network applications in radiology, discussing challenges in data collection, curation, and annotation, and emphasizing the importance of diverse and balanced training data to avoid biases, without presenting specific performance metrics. [18] surveyed various image classification methods and techniques, providing a comprehensive overview of feature extraction, feature selection, and classifier design, along with techniques like preprocessing, data augmentation, and ensemble methods, but did not present specific results. In [19], a fine-tuned MobileNetV2 model for multiclass classification of lung diseases from X-ray images was developed, achieving an overall accuracy of 92.5% and F1-scores ranging from 0.82 to 0.97 for different classes, though the dataset used was unspecified. Finally, [20] introduced a customized MobileNetV2 model for the same task, achieving even higher overall accuracy of 97.8% and F1-scores ranging from 0.95 to 0.99 for different classes, but again without specifying the dataset used for evaluation.

This project utilizes the concepts of Transfer Learning to fine-tune a DenseNet121 architecture for the MURA dataset for a Multiclass Classification Problem statement. The approaches utilized in this project are direct learnings from the literature survey conducted. It also utilizes explainability maps techniques including Attention plots, Saliency Maps, and Grad-CAM Maps to make the decision-making of the models transparent. The novelty of this project is to convert the MURA binary classification problem statement into a Multiclass classification statement and to use transfer learning with a DenseNet121 architecture along with the explainability Maps.

III. Methodology

This project utilizes the MURA dataset to perform multi-class classification by predicting the body part along with the abnormality. The MURA dataset is transformed to be compatible with the multi-class classification problem. After this, the data is prepared for training by dividing the entire dataset into training, validation, and test sets. Transfer learning methodologies are utilized with the DenseNet121 model using ImageNet weights. The training is conducted comprehensively, by unfreezing the last few layers one by one and proceeding accordingly. An important aspect of this project is adding explainability during inference, so the decision-making process of the classifier is understood, thereby improving confidence in the results.

Below figure shows the methodology that has been followed in this project;

Crisp-dm strategy is utilized for its advantages and flexibilities with Deep learning problem statements. Each phase of the Crisp-dm with respect to this project are detailed in the next subsections.

A. Data Collection

The MURA dataset, a comprehensive collection of musculoskeletal radiographs, was meticulously assembled from the Picture Archive and Communication System (PACS) of Stanford Hospital. The institutional review board approved the study, ensuring the collected images were de-identified and HIPAA-compliant. This dataset consists of 14,863 studies from 12,173 patients, resulting in a total of 40,561 multi-view radiographic images. These images span seven standard upper extremity study types: elbow, finger, forearm, hand, humerus,

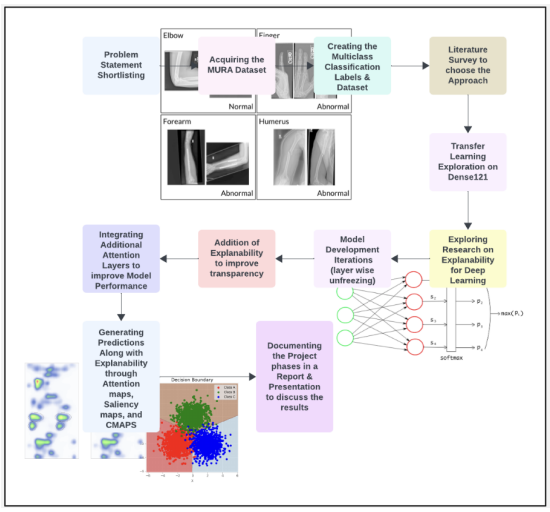


Fig. 1. Project Methodology

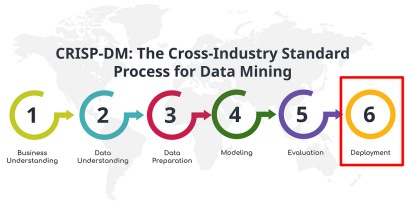


Fig. 2. Project Methodology

shoulder, and wrist.

Each radiographic study was manually labeled as either normal or abnormal by board-certified radiologists from Stanford Hospital. The labeling occurred at the time of clinical radiographic interpretation, performed on DICOM images displayed on high-resolution PACS medical grade monitors. These monitors had at least 3 megapixels, a maximum luminance of 400 cd/m², a minimum luminance of 1 cd/m², a pixel size of 0.2, and a native resolution of 1500 x 2000 pixels. The clinical images in the dataset vary in resolution and aspect ratios, reflecting real-world variability.

The dataset was divided into training, validation, and test. The training set comprises 11,184 patients, 13,457 studies, and 36,808 images. The validation set includes 783 patients, 1,199 studies, and 3,197 images, while the test set contains 206 patients, 207 studies, and 556 images. Crucially, there is no overlap of patients between these sets, ensuring a robust evaluation framework.

To evaluate model performance and obtain an estimate of radiologist performance, additional labels were collected from six board-certified radiologists for the test set. This test set consists of 207 musculoskeletal studies. Each radiologist independently reviewed and labeled each study as either normal or abnormal in a clinical reading room environment using the PACS system. The radiologists had an average of 8.83 years of experience, ranging from 2 to 25 years. Importantly, they did not have access to any clinical information during the labeling process. The labels were entered into a standardized data entry program, ensuring consistency and accuracy.

The MURA dataset has significant applications introduced by [21] in the field of medical imaging and deep learning. It serves as a benchmark for developing and evaluating models designed to detect abnormalities in musculoskeletal radiographs, below are some of the key applications;

- Researchers can develop and train convolutional neural network (CNN) models for abnormality detection using the MURA dataset, leveraging its large size and diverse range of study types to create robust models that generalize well to new data.
- The MURA dataset is ideal for transfer learning techniques, allowing pretrained models like those from ImageNet to be fine-tuned for improved abnormality detection in musculoskeletal radiographs by leveraging existing knowledge from large-scale image recognition tasks.
- It supports the development of methods for visualizing and understanding model predictions, using techniques like Class Activation Mapping (CAM) to highlight critical regions in radiographs and enhance the explainability and interpretability of AI models in clinical settings.

Some of the clinical applications of MURA can be;

- **Worklist Prioritization**: Abnormality detection models trained on the MURA dataset can prioritize worklists by moving studies detected as abnormal ahead in the image interpretation workflow. This allows for quicker diagnoses and treatment for the most critical cases.
- **Preliminary Reading Assignment**: Models can automatically assign a preliminary reading of ”normal” to examinations identified as normal, streamlining the workflow for radiologists. This can lead to faster results for patients and more efficient use of radiologists’ time.
- **Combating Radiologist Fatigue**: Automated abnormality localization can help reduce radiologist fatigue by highlighting the portions of images recognized as abnormal. This can lead to more efficient interpretation, reduced errors, and standardized quality

B. Data Understanding

Below are some of the samples from the MURA dataset for different body parts, along with their respective labels.

From Fig. 3, it can be seen that there are 7 classes (body parts), each categorized as either Abnormal or Normal. This binary classification problem statement is converted into a multiclass classification problem with 14 classes where each class represents bodypart, along with its condition of Abnormality.

The Fig 4 is a typical X-ray of a normal HUMERUS, showing the bone structure without any visible abnormalities. The pixel intensity distribution, shown in the histogram, indicates how frequently each pixel intensity occurs within the image. Most of the pixel values are concentrated in the lower range, reflecting the darker regions in the X-ray. This is consistent

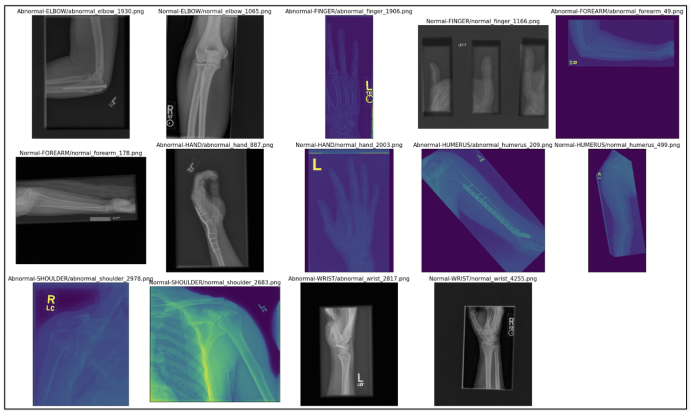


Fig. 3. Radiology Images from MURA dataset for different body parts

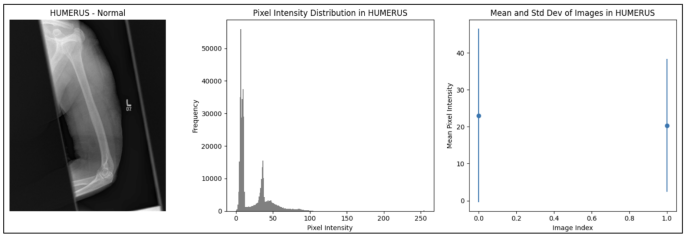


Fig. 4. Pixel Analysis of Normal-HUMERUS

with what is expected in normal bone images, where dense bones appear lighter and the surrounding tissues, which are less dense, appear darker. The peaks in the histogram suggest that certain ranges of pixel intensities are more common, likely corresponding to the different tissues and structures within the elbow.

The scatter plot presents the mean pixel intensity and its standard deviation for the normal elbow images. This two points on the plot, each representing an image, with the error bars indicating the standard deviation. The first image has a lower mean pixel intensity, while the second image shows a higher mean. The larger standard deviation for the first image suggests greater variability in pixel intensities, which could be due to differences in imaging conditions or the inherent variability in the anatomical structure.

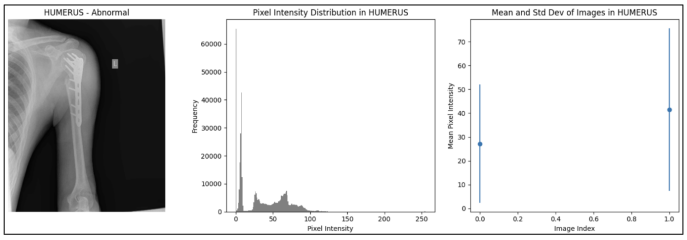


Fig. 5. Pixel Analysis of Abnormal-HUMERUS

Fig 5 focuses on an abnormal HUMERUS radiograph. The image depicts an X-ray of an elbow with visible abnormalities. This could include fractures, dislocations, or other pathological conditions. The pixel intensity distribution in the histogram on the shows significant peaks at lower intensities, indicating more dark regions in the image. This might represent areas with less dense tissue or more open space, which could be associated with the abnormalities present.

The scatter plot shows the mean pixel intensity and its standard deviation for the abnormal HUMERUS images. Similar to the normal images, this plot indicates significant variability in pixel intensities. The first image has a higher mean pixel intensity with a larger standard deviation, while the second image shows a lower mean with less variability. This suggests that abnormal images might have more diverse pixel intensity distributions, possibly due to different types and severities of abnormalities.

The visualizations highlight the importance of considering pixel intensity distributions and their variability when preparing the dataset for training machine learning models. The significant peaks and variability in pixel intensities, especially in abnormal images, suggest that standardizing the image data could be beneficial. Techniques such as mean subtraction and division by the standard deviation can help stabilize the training process by ensuring that the images have consistent statistical properties. This normalization can improve the performance of models by reducing the impact of differences in imaging conditions and enhancing the model's ability to generalize across different types of images.

C. Data Pre-processing

Data Preprocessing can be considered as one of the most important phases of this project. The objective of this phase is to convert the Raw MURA dataset into a Multiclass dataset, by creating 14 distinct classes which include bodypart and presence of abnormality.
With the help of the metadata files for train, valid and test split from the original dataset, a python script was created to use these metadata files to aggregate the directory structure and granularity to images instead of studies. The python script also clubbed the respective class images into their new classes. Below is the metadata snapshot before and after creating the new dataset.

Training Data Sample:

	path	abnormality	body_part
0	MURA-v1.1/train/XR_SHOULDER/patient00001/study...	1	SHOULDER
1	MURA-v1.1/train/XR_SHOULDER/patient00002/study...	1	SHOULDER
2	MURA-v1.1/train/XR_SHOULDER/patient00003/study...	1	SHOULDER
3	MURA-v1.1/train/XR_SHOULDER/patient00004/study...	1	SHOULDER
4	MURA-v1.1/train/XR_SHOULDER/patient00005/study...	1	SHOULDER

Fig. 6. Metadata of Training dataset before preprocessing

Below figure 7 displays the metadata of the same training sample after the preprocessing, it can be observed that the label has been created with the Python script.

D. Data Preparation

The newly created dataset is then split train, valid and test sets in the ratio of 80-10-10, using the stratify sampling method. This will make sure that the splits are representative of the population. Below are some of the visualizations to understand the class wise distribution in the different splits.

	path	Body_part	abnormality	bodypart_abnormality
0	multiclass_train/Abnormal-ELBOW/abnormal_elbow...	Abnormal	ELBOW	abnormal_elbow
1	multiclass_train/Abnormal-ELBOW/abnormal_elbow...	Abnormal	ELBOW	abnormal_elbow
2	multiclass_train/Abnormal-ELBOW/abnormal_elbow...	Abnormal	ELBOW	abnormal_elbow
3	multiclass_train/Abnormal-ELBOW/abnormal_elbow...	Abnormal	ELBOW	abnormal_elbow
4	multiclass_train/Abnormal-ELBOW/abnormal_elbow...	Abnormal	ELBOW	abnormal_elbow
...	...	...	...	...
31276	multiclass_train/Normal-WRIST/normal_wrist_993...	Normal	WRIST	normal_wrist
31277	multiclass_train/Normal-WRIST/normal_wrist_994...	Normal	WRIST	normal_wrist
31278	multiclass_train/Normal-WRIST/normal_wrist_995...	Normal	WRIST	normal_wrist
31279	multiclass_train/Normal-WRIST/normal_wrist_996...	Normal	WRIST	normal_wrist
31280	multiclass_train/Normal-WRIST/normal_wrist_999...	Normal	WRIST	normal_wrist

31281 rows × 4 columns

Fig. 7. Metadata of Training dataset after preprocessing

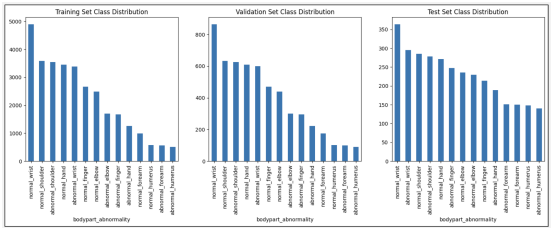


Fig. 8. Data Distribution for Train, Valid and Test Splits

It can be observed that the class imbalances are present, between the different classes in each dataset, but the overall distribution across train, valid and test splits remain consistent.

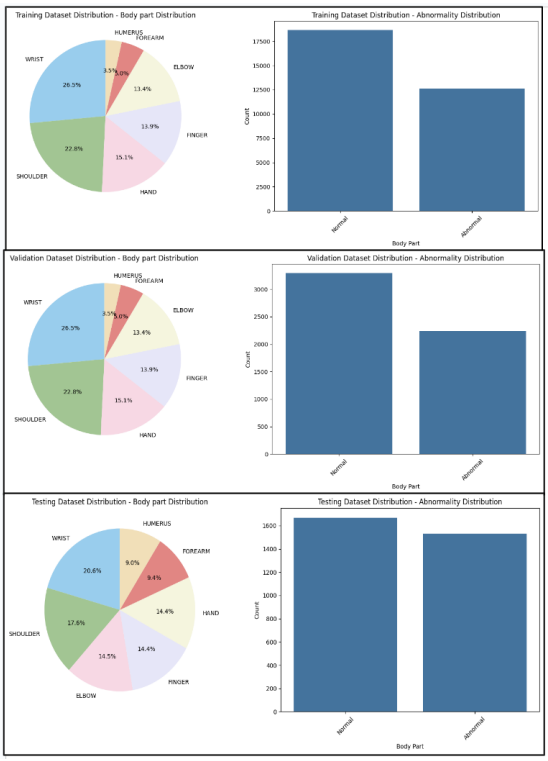


Fig. 9. Abnormality and BodyPart Distribution for Train, Valid and Test Splits

It can be seen that the overall distribution of the abnormality, and the bodypart is remained consistent after the preprocessing. To ensure that there are no data leaks while splitting the dataset, it was made sure that no modifications were done to the newly created MURA dataset, the standardizing, and the PCA is only done on the training data. The standardizing for validation and test images is done individually, and from the results it can be seen that there is no data leak between the splits as well.

E. Data Transformation

Data Augmentation has been done as part of the training process, utilizing the generator capabilities of the keras libraries, below are the augmentations that have been used for training,

- Rescale (Normalization): rescale=1./255 - Normalizes pixel values to the range [0, 1].
- Rotation: rotation_range=40 - Applies random rotations within ±40 degrees.
- Width Shift (Horizontal Shift): width_shift_range=0.2 - Randomly shifts images horizontally by up to 20% of the width.
- Height Shift (Vertical Shift): height_shift_range=0.2 - Randomly shifts images vertically by up to 20% of the height.
- Shear Transformation: shear_range=0.2 - Applies shear transformations with a shear intensity of 0.2.
- Zoom: zoom_range=0.2 - Randomly zooms in and out by up to 20%.
- Horizontal Flip: horizontal_flip=True - Randomly flips images horizontally.
- Fill Mode: fill_mode='nearest' - Fills in new pixels created by transformations using the nearest pixel values.

Above augmentations are done on the training set, to let the model learn diversity of patterns to aid with generalization.
As part of the Data Transformation, this project explored PCA and iPCA (Incremental PCA) with the training data, to reduce the computational load, and help with training times.

- **PCA** is a dimensionality reduction technique used to transform high-dimensional data into a lower-dimensional space while retaining most of the variance in the data. It achieves this by identifying the principal components, which are the directions (eigenvectors) in the feature space that correspond to the largest variances (eigenvalues).
- **Incremental PCA** is a variant of PCA that allows for the processing of large datasets in a memory-efficient manner. Instead of computing the principal components on the entire dataset at once, iPCA processes the data in mini-batches, making it suitable for datasets that do not fit into memory.

iPCA has an advantage over PCA in handling large datasets by processing data in chunks, which is essential when the entire dataset cannot be loaded into memory. iPCA is particularly useful for streaming or incrementally updated datasets, continuously updating principal components as new data arrives, this can be helpful incase if this project extends to building an application for early prediction of abnormalities. iPCA is

more memory-efficient than PCA due to its ability to perform computations incrementally, making it suitable for real-time applications and large-scale data processing.
Below is the detailed results from the exploration of PCA and iPCA with the MURA dataset;

	10 Images Per Class		1000 Images Per Class	
	PCA	iPCA	PCA	iPCA
Images/Class	10	10	1000	1000
Total Images	1400	1400	14000	14000
# Components for 95% variance	8	11	43	19
Runtime	6 secs	3.6 secs	36 secs	9 secs

Fig. 10. Results from the PCA vs iPCA analysis

From the results it can be inferred that iPCA is a great choice for the dimension reduction for the images, but the draw back is that PCA and its variants don't capture the non-linear dependencies in the data, due to this reason PCA reduced dimensions were not utilized for training purposes, this exploration served its purpose by proving that the runtime is exponentially lower in the case of iPCA. If we are dealing with a real time application which can predict abnormalities, this can be a great way to reduce the inference and training time as well

Below are the elbow curves which were generated to understand the number of components needed to capture the percentage of variance.

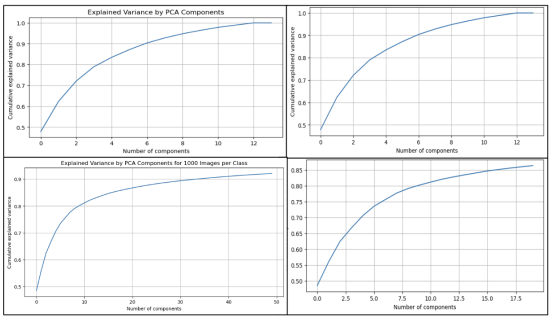


Fig. 11. Elbow curves from the PCA vs iPCA analysis

F. Proposed Model

The Dense Convolutional Network (DenseNet) is a deep learning architecture introduced by [22] designed to ensure maximum information flow between layers by connecting each layer to every other layer in a feed-forward manner. This dense connectivity pattern helps alleviate the vanishing-gradient problem, strengthens feature propagation, encourages feature reuse, and substantially reduces the number of parameters.

The DenseNet architecture is pretty complex and involves various components and layers which are detailed below;

- **Convolutional Layers**: DenseNet employs convolutional layers to extract features from the input images. Each convolutional layer applies a set of filters to the input, producing feature maps that highlight various aspects of the data
- **Batch Normalization (BN)**: Before each convolutional operation, Batch Normalization is applied to standardize the inputs to the layer. This helps in stabilizing the learning process and accelerating convergence by reducing internal covariate shift.
- **Rectified Linear Units (ReLU)**: Following Batch Normalization, ReLU activation functions are used to introduce non-linearity into the network. ReLU is defined as ReLU(x)=max(0,x), which helps in preventing the vanishing gradient problem and accelerates the training process.
- **Pooling Layers**: Pooling layers, specifically average pooling, are used to down-sample the feature maps, reducing their spatial dimensions while retaining the most important information. This helps in reducing the computational complexity and controlling overfitting.
- **Dense Layers**: Dense layers (fully connected layers) are used at the end of the network to combine the features extracted by the convolutional layers and make the final classification. The output of the final dense layer is passed through a softmax activation function to produce probabilities for each class.

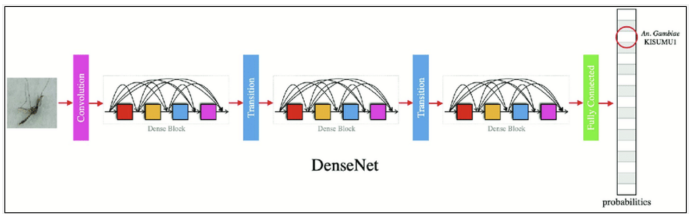


Fig. 12. DenseNet Model Architecture

The important components of the DenseNet are explained below in great detail,

- **Dense Blocks**: Dense blocks are the fundamental building units of the DenseNet architecture, consisting of multiple convolutional layers. Unlike traditional convolutional networks, each layer in a dense block receives inputs from all preceding layers and passes its own feature maps to all subsequent layers within the block. This dense connectivity ensures efficient feature reuse and gradient flow throughout the network. Within a dense block, each layer sequentially performs the following operations: batch normalization to standardize the inputs, ReLU activation to introduce non-linearity, and a 3x3 convolution to extract features. This unique arrangement allows the network to utilize the "collective knowledge" of all previous layers, enhancing learning efficiency and feature propagation.
- **Transition Layers**: Transition layers are placed between dense blocks to down-sample the feature maps and manage the number of feature maps. Each transition layer includes batch normalization, a 1x1 convolution, and a

2x2 average pooling operation. The batch normalization standardizes the input to the transition layer, the 1x1 convolution reduces the number of feature maps, acting as a bottleneck to decrease computational cost, and the 2x2 average pooling reduces the spatial dimensions of the feature maps. These layers ensure the network remains compact and computationally efficient while preserving important information.

- **Growth Rate**: The growth rate, denoted by k, is a critical parameter in DenseNet that determines the number of new feature maps each layer within a dense block produces. For instance, if the growth rate is set to 32, each layer will add 32 new feature maps. This growth rate controls the capacity of the network, ensuring that each layer contributes a small, manageable amount of new information to the overall network. This allows the network to be both deep and efficient without unnecessary redundancy.
- **Compression Factor**: The compression factor, denoted by , is applied in transition layers to reduce the number of feature maps. For example, if =0.5, the number of feature maps is halved in the transition layer, making the network more compact and efficient. This factor helps in controlling the growth of the network's size, ensuring that it remains scalable and less prone to overfitting.
- **Bottleneck Layers**: To further improve computational efficiency, DenseNet incorporates bottleneck layers. These layers consist of a 1x1 convolution followed by a 3x3 convolution. The 1x1 convolution reduces the number of input feature maps, acting as a bottleneck that decreases computational load before the 3x3 convolution is applied. This design choice minimizes the number of parameters and computational cost while maintaining the network's expressive power.
- **Global Average Pooling**: Instead of using fully connected layers directly after the convolutional layers, DenseNet uses global average pooling. This operation reduces each feature map to a single value by taking the average across its spatial dimensions. This significantly reduces the number of parameters and helps prevent overfitting by ensuring that the final feature representation is compact and robust.
- **SoftMax Classification Layer**: The final layer in the DenseNet architecture is a fully connected layer with a softmax activation function. This layer takes the output of the global average pooling layer and produces the probabilities for each class, enabling multiclass classification. The softmax function ensures that the output probabilities sum up to one, providing a clear and interpretable classification result.
- **Loss function**: In the DenseNet-121, the commonly used loss function is **categorical cross-entropy**. This loss function is well-suited for multiclass classification problems, which aligns with the task of predicting body parts and abnormalities in the MURA dataset.

Below Fig 13, architecture from [22] helps in understanding the layers and components structure;

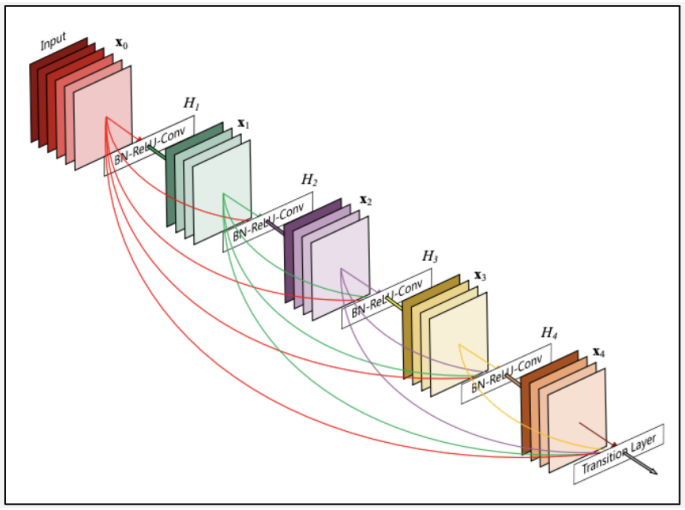


Fig. 13. A 5-layer dense block with a growth rate of k=4. Each layer takes all preceding feature-maps as input

The model utilized for this project is Dense121 which is one of the variants proposed in the [22]. The dense121 architecture details below;

- **7x7 Convolution with stride 2**: This layer applies a large convolutional filter to the input image, reducing its spatial dimensions and capturing initial features.
- **3x3 Max Pooling with stride 2**: This pooling layer further reduces the spatial dimensions of the feature maps, helping in down-sampling and retaining important features.
- **Dense Block 1**: Comprises 6 convolutional layers, each with a growth rate of 32. Within this block, each layer performs batch normalization (BN), a rectified linear unit (ReLU) activation, and a 3x3 convolution (Conv). Each layer connects directly to all subsequent layers via feature-map concatenation, ensuring efficient feature reuse and enhanced gradient flow
- **Transition Layer 1**: 1x1 Convolution acts as a bottleneck layer, reducing the number of feature maps. 2x2 Average Pooling with stride 2 reduces the spatial dimensions of the feature maps, managing the network capacity and dimensionality.
- **Dense Block 2**: Contains 12 convolutional layers with a growth rate of 32, following the same BN, ReLU, and 3x3 Conv structure. The dense connectivity pattern continues, with each layer accessing all preceding layers' feature maps.
- **Transition Layer 2**: Similar to Transition Layer 1, with 1x1 Convolution and 2x2 Average Pooling.
- **Dense Block 3**: Includes 24 convolutional layers with a growth rate of 32. This block maintains the dense connections, ensuring extensive feature reuse and robust learning.

- **Transition Layer 3**: Comprises 1x1 Convolution and 2x2 Average Pooling.
- **Dense Block 4**: Consists of 16 convolutional layers with a growth rate of 32, continuing the dense connectivity pattern.
- **Global Average Pooling**: This layer reduces each feature map to a single value by averaging the spatial dimensions, significantly reducing the number of parameters and preventing overfitting.
- **Fully Connected Layer**: Softmax activation outputs the probabilities for each class, enabling multiclass classification.

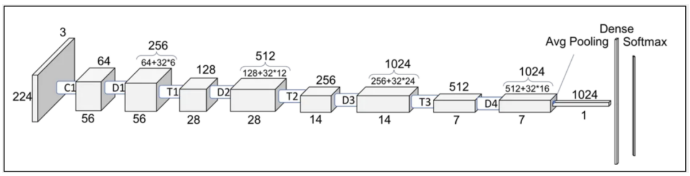


Fig. 14. Dense121 architecture

DenseNet-121's architecture, with its dense connectivity, efficient feature reuse, and robust design, makes it highly suitable for the multiclass classification tasks required for analyzing the MURA dataset. Its ability to handle high-dimensional data, mitigate overfitting, and provide accurate predictions ensures its effectiveness in predicting body parts and abnormalities in medical images.

As this project utilizes Transfer learning Practices to train the Dense121 with ImageNet weights, it removes the last layer of the architecture - which is a classifier trained for ImageNet data. Additional Global Average Pooling layers, Dropouts, Dense layers are added based on the results, as the training progresses, and a softmax layer is also added as a classifier, and then trained on the radiology images from MURA dataset.

Below is the architecture diagram of the model to initialize the training, this architecture has all the layers frozen, and only dense layers with ReLU, and Softmax for classification.

conv5_block16_1_bn (BatchNormalization)	(None, 7, 7, 128)	512	['conv5_block16_1_conv[0][0]']
conv5_block16_1_relu (Activation)	(None, 7, 7, 128)	0	['conv5_block16_1_bn[0][0]']
conv5_block16_2_conv (Conv2D)	(None, 7, 7, 32)	36864	['conv5_block16_1_relu[0][0]']
conv5_block16_concat (Concatenate)	(None, 7, 7, 1024)	0	['conv5_block15_concat[0][0]', 'conv5_block16_2_conv[0][0]']
bn (BatchNormalization)	(None, 7, 7, 1024)	4096	['conv5_block16_concat[0][0]']
relu (Activation)	(None, 7, 7, 1024)	0	['bn[0][0]']
global_average_pooling2d (GlobalAveragePooling2D)	(None, 1024)	0	['relu[0][0]']
dense (Dense)	(None, 1024)	1049600	['global_average_pooling2d[0][0]']
dense_1 (Dense)	(None, 14)	14350	['dense[0][0]']

Total params: 8101454 (30.90 MB)
Trainable params: 1063950 (4.06 MB)
Non-trainable params: 7037504 (26.85 MB)

Fig. 15. Modified Dense121 architecture with all layers frozen - first iteration, ready for Multiclass classification

The DenseNet-121 architecture has been enhanced with several modifications to improve its performance for multiclass classification on the MURA dataset:

- The last 5 convolutional layers of the DenseNet-121 are unfrozen, allowing them to be trainable and fine-tuned.
- Squeeze-and-Excite blocks are incorporated to enhance the network's sensitivity to important features by recalibrating channel-wise feature responses adaptively.
- Three attention heads are added to focus on different parts of the feature maps. These attention heads are concatenated to enhance the network's ability to capture relevant features.
- Increased dropout rates are used for regularization, with 0.5 before the Global Average Pooling (GAP) layer and 0.7 after GAP to prevent overfitting.
- Two fully connected layers are added with heavy dropout rates (0.7 and 0.6) and L2 regularization to improve generalization and reduce overfitting.
- The ReduceLROnPlateau scheduler is utilized to adjust the learning rate dynamically based on validation performance, ensuring efficient training.
- EarlyStopping is also utilized for efficient training process.

multiply_1 (Multiply)	(None, 7, 7, 1024)	0	['multiply[0][0]', 'attention_0[0][0]']
multiply_2 (Multiply)	(None, 7, 7, 1024)	0	['multiply[0][0]', 'attention_1[0][0]']
multiply_3 (Multiply)	(None, 7, 7, 1024)	0	['multiply[0][0]', 'attention_2[0][0]']
concatenate (Concatenate)	(None, 7, 7, 3072)	0	['multiply_1[0][0]', 'multiply_2[0][0]', 'multiply_3[0][0]']
dropout (Dropout)	(None, 7, 7, 3072)	0	['concatenate[0][0]']
global_average_pooling2d_1 (GlobalAveragePooling2D)	(None, 3072)	0	['dropout[0][0]']
dropout_1 (Dropout)	(None, 3072)	0	['global_average_pooling2d_1[0][0]']
dense_2 (Dense)	(None, 512)	1573376	['dropout_1[0][0]']
batch_normalization (BatchNormalization)	(None, 512)	2048	['dense_2[0][0]']
dropout_2 (Dropout)	(None, 512)	0	['batch_normalization[0][0]']
dense_3 (Dense)	(None, 512)	262656	['dropout_2[0][0]']
dense_4 (Dense)	(None, 512)	1572864	['dropout_1[0][0]']
batch_normalization_1 (BatchNormalization)	(None, 512)	2048	['dense_3[0][0]']
batch_normalization_2 (BatchNormalization)	(None, 512)	2048	['dense_4[0][0]']
add (Add)	(None, 512)	0	['batch_normalization_1[0][0]', 'batch_normalization_2[0][0]']
activation (Activation)	(None, 512)	0	['add[0][0]']
dropout_3 (Dropout)	(None, 512)	0	['activation[0][0]']
dense_5 (Dense)	(None, 256)	131328	['dropout_3[0][0]']
dropout_4 (Dropout)	(None, 256)	0	['dense_5[0][0]']
dense_6 (Dense)	(None, 14)	3598	['dropout_4[0][0]']

Total params: 10722705 (40.90 MB)
Trainable params: 10635985 (40.57 MB)
Non-trainable params: 86720 (338.75 KB)

Fig. 16. Modified Model Architecture with Best results out of the experiments conducted

IV. Experimental Setup

This section will detail the experiments, configurations, training, and validation progress, and other details. It also

details about the hardware and software requirements that have been used for Model training and inference purposes.

A. Hardware Requirements

This project was completely executed from end-to-end in the ADS lab which was booked for different sessions, and executed accordingly. Below is the quick summary of the hardware details, including cost to conduct this project.

Resource	Specification	Est. Cost (USD)	Justification
Server	32-core CPU, 128GB RAM, 4TB SSD	3,000	High-performance servers, ensuring fast response times, fast training times and data processing.
Workstation	16-core 4.50 GHz CPU, 128GB RAM, 1TB SSD	5,000	For development and testing of the model, ensuring efficient model training and application development.
GPU details	Proc.: Rtx 4090, RAM: 128 GB, vRAM: 24 GB, CPU: ryzen9, Storage: 5TB SSD	Variable	GPUs accelerate model training with parallel processing power
CPU Instances	16-core, 64GB RAM (Cloud Instances)	1500 (Annual)	Cloud-based CPU instances for additional computing power, scalability, and flexibility in resource allocation for peak loads.
Cloud Storage	Data Storage	Variable	Data storage for backup
SSD	100GB SSD Storage	200	Additional storage for backups and redundancy, ensuring data integrity and availability.

Fig. 17. Hardware Requirements for this Project Execution

B. Software Requirements

Below is the list of software requirements that were needed to execute this project. This project utilizes Keras, and Tensorflow as primary python libraries utilized for Modeling phase.

Library	Version	Submodules	Description
pandas	1.5.3	read_csv(), DataFrame(), to_numpy()	Data manipulation and analysis library.
os	-	listdir(), makedirs(), path.join()	Provides functions for interacting with the operating system.
tqdm	4.64.0	tqdm()	Fast, extensible progress bar for loops and iterables.
sklearn	1.2.2	metrics.classification_report(), metrics.confusion_matrix(), metrics.roc_curve(), metrics.auc(), preprocessing.label_binarize()	Machine learning library for Python, offering tools for metrics, model selection, and preprocessing.
matplotlib	3.6.3	pyplot.figure(), pyplot.plot(), pyplot.imshow()	Comprehensive library for creating static, animated, and interactive visualizations in Python.
seaborn	0.11.2	heatmap(), distplot(), pairplot()	Statistical data visualization library based on matplotlib.
numpy	1.24.0	array(), zeros(), ones()	Fundamental package for numerical computing in Python.
tensorflow	2.11.0	keras.applications.DenseNet121(), keras.models.Sequential(), keras.layers.Conv2D(), keras.layers.MaxPooling2D(), keras.layers.Dense(), keras.optimizers.Adam()	End-to-end open-source platform for machine learning.
scipy	1.9.3	ndimage.zoom()	Library for scientific and technical computing, with a submodule for multi-dimensional image processing.
tf_keras_vis	0.13.0	activation_maximization.ActivationMaximization(), saliency.Saliency(), utils.model_modifiers.ExtractIntermediateLayer(), utils.scores.CategoricalScore()	Library for the visualization of TensorFlow and Keras models, with submodules for different visualization techniques.
itertools	-	cycle()	Implements a number of iterator building blocks.

Fig. 18. Software Requirements for this Project Execution

C. Experiments

For model building and training, there were quite a few iterations that were experimented with. The model was properly developed using the Transfer learning methodologies, where each iteration had a change of architecture, majorly focusing on adding relevant dropouts, regularizers, learning rates, and optimizers.

1) Experiment - 1: Initially, the model experimented with had all layers frozen, removed the last layer (classifier of Dense121), and added a Global Average Pooling, along with Dense layers with ReLU and Softmax for multiclass classification.

Below Fig 19 are the results of training. Epoch wise training loss, accuracy and validation loss & accuracy are plotted and shown below.

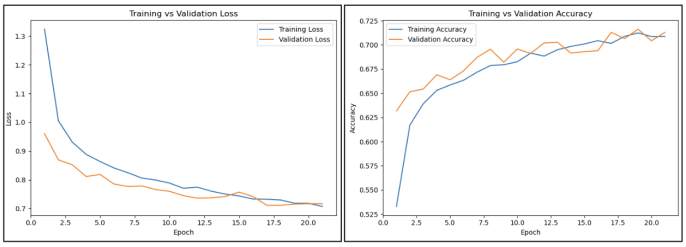


Fig. 19. Training and Validation Curves for the first Iteration

From the above Fig 19 curves, it can be inferred that the Models are not yet stable. The high loss in training, and validation hints the case of underfitting and overfitting, meaning the generalization won't be stable.

2) Experiment - 2: Learning from the previous Experiment, in this iteration last 5 convolutional layers have been unfreezed, and a Learning rate scheduler is added, along with L2 decay factor, with a GAP layer, and Dense(ReLU), Dense with a dropout (0.6), and a Dense (Softmax) is added, and trained for few epochs to understand the training progress.

Below Fig 20 is the updated model architecture;

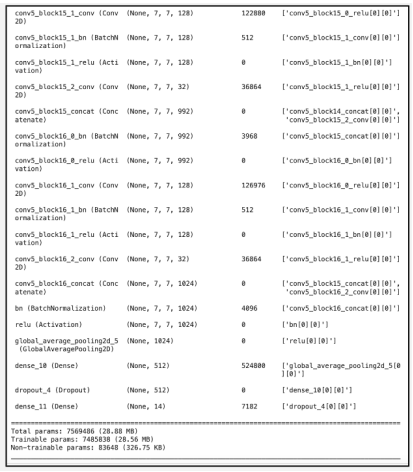


Fig. 20. Updated Model Architecture Utilized in the Experiment - 2

Below Fig 21 are the training and validation curves to understand the characteristics of the trained model.

The ReduceLROnPlateau is a dynamic learning rate scheduler used to adapt the learning rate during training based

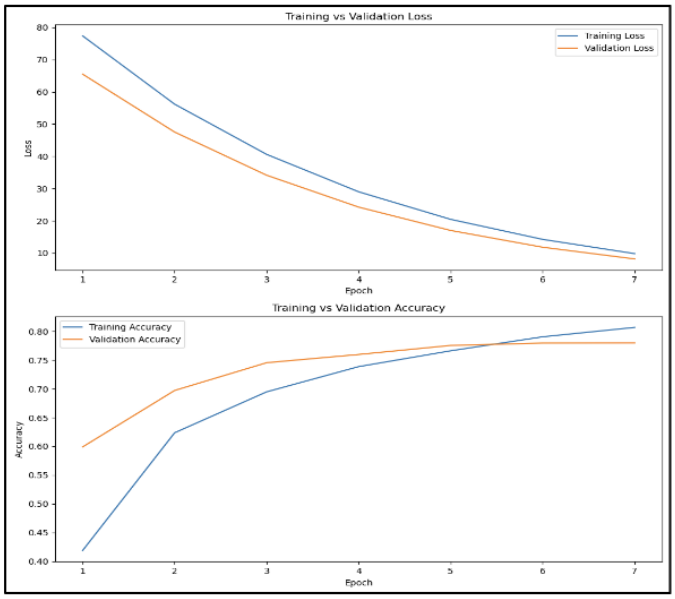


Fig. 21. Training and Validation Curves to understand the Training process

on the performance of the model on the validation set. It helps to ensure that the model converges efficiently without overshooting or getting stuck in suboptimal minima.

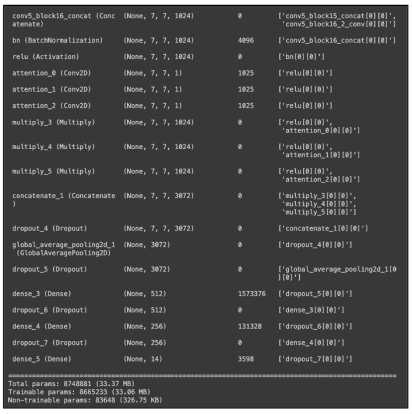
The scheduler monitors a specified metric which is validation loss in this case, typically the validation loss or validation accuracy. When the monitored metric stops improving for a defined number of epochs (patience), the learning rate is reduced by a specified factor. Patience parameter defines the number of epochs to wait before reducing the learning rate. If the metric does not improve within this patience period, the learning rate is decreased. Patience is set to 5 in this case. Factor parameter specifies the multiplication factor by which the learning rate will be reduced. For example, in this experiment a factor of 0.2 means the learning rate will be reduced to 20% of its current value. Minimum Learning Rate parameter sets a lower bound on the learning rate, ensuring it does not reduce to an impractically low value, the set value for this experiment is 1e-7.

By reducing the learning rate when the improvement plateaus, the model can converge more efficiently, avoiding overshooting the minimum of the loss function. Lower learning rates late in training help to fine-tune the model and avoid overfitting, as the model parameters adjust more delicately. The dynamic adjustment of the learning rate based on validation performance makes the training process adaptive and robust to different phases of learning.

From the training and validation curves, it can be observed that the model is able to converge, and the curves look good, meaning they are learning efficiently. The dropout, and learning rate scheduler have worked.

3) Experiment - 3: The results of experiment-2 were satisfactory, so the next iteration developed on it, by adding attention mechanism to the model. The last 5 convolutional layers are unfreezed for fine-tuning and integrates 3 attention heads to focus on different image regions, improving feature extraction. Attention heads use Conv2D layers with sigmoid activations, and their outputs are concatenated. The model employs significant dropout regularization, with rates of 0.5 before and 0.7 after the Global Average Pooling (GAP) layer, and rates of 0.7 and 0.6 in the fully connected layers to prevent overfitting. Additionally, it includes two dense layers with L2 regularization and uses the ReduceLROnPlateau scheduler to dynamically adjust the learning rate, ensuring stable and efficient training.

Below Fig 22 is the enhanced architecture that has been created, and trained accordingly.



conv5_block16_concat (Concatenate)	(None, 7, 7, 1024)	0	['conv5_block15_concat[0][0]', 'conv5_block16_2_conv[0][0]']
bn (BatchNormalization)	(None, 7, 7, 1024)	4096	['conv5_block16_concat[0][0]']
relu (Activation)	(None, 7, 7, 1024)	0	['bn[0][0]']
attention_0 (Conv2D)	(None, 7, 7, 1)	1025	['relu[0][0]']
attention_1 (Conv2D)	(None, 7, 7, 1)	1025	['relu[0][0]']
attention_2 (Conv2D)	(None, 7, 7, 1)	1025	['relu[0][0]']
multiply_3 (Multiply)	(None, 7, 7, 1024)	0	['relu[0][0]', 'attention_0[0][0]']
multiply_4 (Multiply)	(None, 7, 7, 1024)	0	['relu[0][0]', 'attention_1[0][0]']
multiply_5 (Multiply)	(None, 7, 7, 1024)	0	['relu[0][0]', 'attention_2[0][0]']
concatenate_1 (Concatenate)	(None, 7, 7, 3072)	0	['multiply_3[0][0]', 'multiply_4[0][0]', 'multiply_5[0][0]']
dropout_4 (Dropout)	(None, 7, 7, 3072)	0	['concatenate_1[0][0]']
global_average_pooling2d_1 (GlobalAveragePooling2D)	(None, 3072)	0	['dropout_4[0][0]']
dropout_5 (Dropout)	(None, 3072)	0	['global_average_pooling2d_1[0][0]']
dense_3 (Dense)	(None, 512)	1573376	['dropout_5[0][0]']
dropout_6 (Dropout)	(None, 512)	0	['dense_3[0][0]']
dense_4 (Dense)	(None, 256)	131328	['dropout_6[0][0]']
dropout_7 (Dropout)	(None, 256)	0	['dense_4[0][0]']
dense_5 (Dense)	(None, 14)	3598	['dropout_7[0][0]']

Total params: 8748881 (33.37 MB)
Trainable params: 8665233 (33.06 MB)
Non-trainable params: 83648 (326.75 KB)

Fig. 22. Enhanced Model Architecture with Attention heads for Experiment-3

Below Fig 23 is the training, validation curves to understand and throw light on the training process.

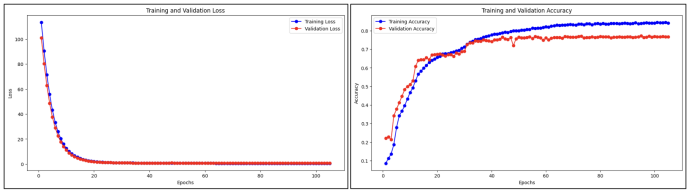


Fig. 23. Results of training for Experiment-3

The training was conducted for 150 epochs with learning rate scheduler, from the curves, the model was able to converge, but the overlap of training and validation loss after converging hints that the model is underfitting, which can be dangerous for generalization on unseen data.

4) Experiment - 4: From the Exp-3, it was concluded that attention heads are indeed being a good addition to the model architecture. Firstly, the last five convolutional layers are unfrozen to allow fine-tuning, which helps in adapting the pre-trained model more effectively to the specific task.
The model integrates squeeze-and-excite (SE) blocks, which perform feature recalibration by adaptively adjusting the weights of different feature channels. This mechanism involves a ”squeeze” operation where global spatial information is condensed into a compact form using global average pooling. The

subsequent ”excite” operation applies a gating mechanism with fully connected layers and a ReLU activation followed by a sigmoid activation. This gating mechanism learns to emphasize informative features and suppress less useful ones dynamically, thus improving the model's ability to focus on relevant parts of the input data. Additionally, the model employs three attention heads that focus on different regions of the image, enhancing the feature extraction process. These attention heads use Conv2D layers with sigmoid activations, and their outputs are concatenated to form a comprehensive feature map. The architecture features significant dropout regularization, with a rate of 0.5 before GAP and 0.7 after, to prevent overfitting.
The fully connected layers include heavy dropout rates of 0.7 and 0.6 and utilize L2 regularization to further prevent overfitting and improve model generalization. The model also incorporates a learning rate scheduler, ReduceLROnPlateau, which dynamically adjusts the learning rate based on validation loss, ensuring stable and efficient training. An early stopping mechanism with patience set to 5 is used to halt training when the validation loss stops improving, preventing overfitting and saving computational resources.
By leveraging the SE blocks and attention mechanisms, the model is able to focus on the most informative features, enhancing its ability to distinguish between different classes effectively. This combination of techniques is crucial for achieving high performance in the multiclass classification task on the MURA dataset.

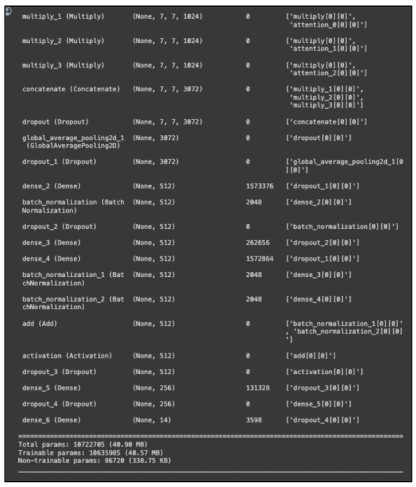


Fig. 24. Enhanced Model Architecture with Attention heads and Squeeze and Excite Blocks for Experiment-4

Below Fig 25, are the training and validation curves, implemented with Learning rate scheduler, and Early stopping to make sure the best performance of the model is saved, and the training process is logged.

From the Fig 25 curves, it can be concluded that the results of exp-4 are improved over exp-3, as it can be observed that the validation and training loss, are not exactly overlapping, meaning the model will be able to generalize better than before.

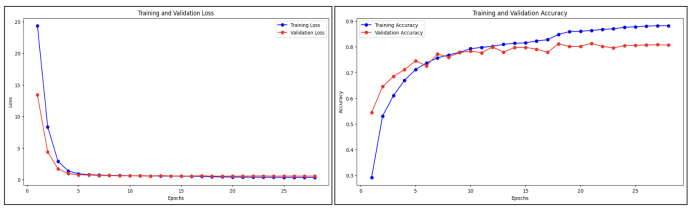


Fig. 25. Results of training for Experiment-3

Below Fig 26 is the learning rate curve for this experiment;

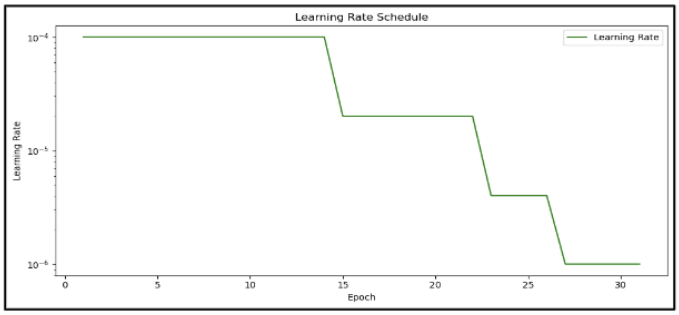


Fig. 26. Learning Rate Scheduler Behavior for this experiment

The efforts from the experimentation process resulted in the below hyperparameters which prove to be a good starting point for the MURA Multiclass classification.

Parameter	Value
Base Model	DenseNet121
Optimizer	Adam
Initial Learning Rate	0.00001
Loss Function	Categorical Crossentropy
Learning Rate Scheduler	Yes
Batch Size	64
Epochs	150
Target Size	224 x 224
Data Augmentation	Yes (rotation, width/height shift, shear, zoom, horizontal flip)
Rescale	1/255
Callbacks	ModelCheckpoint, ReduceLROnPlateau, EarlyStopping
ModelCheckpoint Mode	min
EarlyStopping Patience	10
EarlyStopping Mode	min

Fig. 27. Hyperparameters achieved after Experimentation, based on Exp-4

The chosen hyperparameters yielded the best results during the training and evaluation phases. The DenseNet121 model was fine-tuned using the Adam optimizer with an initial learning rate of 0.00001. The categorical crossentropy loss function was employed due to the multiclass nature of the classification task. A batch size of 64 and 150 epochs were set for training. Images were resized to 224x224, and data augmentation techniques such as rotation, width/height shift, shear, zoom, and horizontal flip were applied to enhance model generalization. The pixel values were rescaled to a range of 0-1 by dividing by 255. Callbacks included ModelCheckpoint

to save the best model, ReduceLROnPlateau to adjust the learning rate based on validation loss, and EarlyStopping to halt training if the validation loss did not improve for 10 epochs. The primary focus of experimentation was on the placement of different layers like Global Average Pooling (GAP), Dropout, and Attention to yield better performance.

Below Fig 27 is the table to summarize the experimentation results,

		Training		Validation	
# Exp	**Model Details**	**Loss**	**Accuracy**	**Loss**	**Accuracy**
Exp 1	Dense121- No layers Unfrozen	0.72	0.71	0.72	0.70
Exp 2	Dense 121 - Last Layer Unfrozen	0.68	0.72	0.69	0.72
Exp 3	Dense 121 - Last 4 layers Unfrozen, Strong L1, L2 and learning rate scheduler	0.25	0.91	0.64	0.80
Exp 4	Dense121 - Last 5 layers, L1, L2, learning rate scheduler	0.41	0.82	0.55	0.81
Exp 5	Dense121 - Last 5 layers, Attention heads = 3, strong L1, L2, learning rate scheduler with additional dropout	0.61	0.84	0.79	0.77
Exp 6	Dense121 - SE feature extraction, Last 5 layers, Attention heads = 3, strong L1, L2, learning rate scheduler with additional dropout	**0.37**	**0.88**	**0.59**	**0.81**

Fig. 28. Above is the summary of the experimentation process, ONLY THE HIGHLIGHTED ROWS WERE EXPLAINED IN THIS REPORT

The experimentation process highlighted four key configurations of DenseNet-121. In Exp 2, unfreezing the last layer slightly improved performance with training and validation accuracies of 0.72 each. Exp 4 introduced L1, L2 regularization, and a learning rate scheduler with the last five layers unfrozen, achieving training and validation accuracies of 0.82 and 0.81, respectively. Exp 5 added three attention heads and additional dropout, resulting in accuracies of 0.84 for training and 0.77 for validation. Exp 6, incorporating squeeze-and-excite feature extraction and the same attention mechanism, achieved the best results with train accuracy of 0.88 and val. accuracy of 0.81.

V. Inference and Results

The resultant model of the Experiment-4, was saved as it produced the best training process. For the inference, as its a classification problem statement, the standard set of Performance metrics derived from the Confusion Matrix are utilized.

In this context of multiclass classification of body part abnormalities, metrics such as precision, recall, F1-score, and accuracy are crucial for evaluating the model's performance comprehensively. Each of these metrics provides insights into different aspects of the model's classification ability:

- **Precision**: Indicates the accuracy of the positive predictions made by the model. It answers the question: Of all the samples predicted as a certain class, how many were correctly predicted?
- **Recall**: Measures the model's ability to correctly identify all relevant instances of a specific class. It answers the question: Of all the actual samples of a certain class, how many were correctly predicted?
- **F1-score**: The harmonic mean of precision and recall. It is particularly useful when the class distribution is imbalanced, as it balances the trade-off between precision and recall.
- **Accuracy**: The overall correctness of the model, calculated as the number of correct predictions divided by the total number of predictions.
- **Weighted Average**: This considers the support (the number of true instances) of each class when calculating the average, providing a balanced performance measure that accounts for class imbalance.

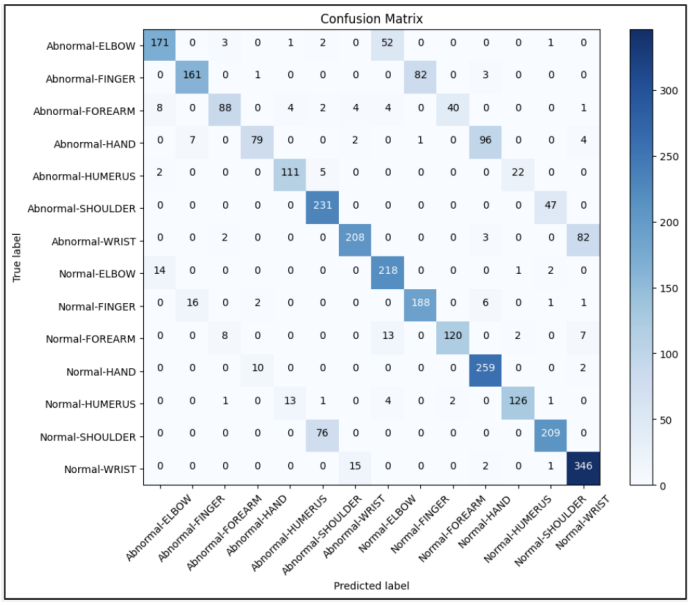


Fig. 29. Confusion Matrix Derived derived from the test set

The weighted average F1-score of 0.78 and accuracy of 0.79 indicate a relatively balanced performance across all classes, despite the inherent class imbalances. The macro average metrics (which treat all classes equally) are slightly lower than the weighted averages, suggesting the model performs better on more frequent classes.

Abnormal-ELBOW shows the Precision of 0.88 and recall of 0.74, resulting in an F1-score of 0.80. This indicates high precision but moderate recall, suggesting the model is good at identifying true positives but misses some actual abnormal elbow cases. Predictions on Abnormal-HAND results in the Precision of 0.86 and recall of 0.42, with an F1-score of 0.56. This low recall highlights a significant number of false negatives, indicating the model struggles to identify all abnormal hand cases. Analyzing the results on Normal-ELBOW, it can be observed that it produces High recall of 0.93 but moderate precision of 0.75, with an F1-score of 0.83. This suggests the model is very effective at identifying normal elbows but makes some false positive errors.

The classes Abnormal WRIST, Abnormal FINGER, and Ab-

normal HAND exhibit significant misclassification rates, as evidenced by the confusion matrix. The model frequently confuses these abnormalities with other categories, resulting in lower recall and F1-scores for these specific classes. This indicates a need for improved differentiation between these abnormal categories to enhance the model's diagnostic accuracy.

The weighted average metrics provide a comprehensive overview of the model's performance across all classes. The precision of 0.80 indicates that, on average, 80% of the positive predictions across all classes were correct. The recall of 0.77 suggests that the model correctly identified 77% of the actual positive cases across all classes. The F1-score of 0.78, which balances precision and recall, shows that the model maintains a good trade-off between accurately identifying true positives and minimizing false positives.

In the context of medical imaging tasks, the F1-score is especially important because both false positives and false negatives can have serious implications. A high F1-score means that the model is reliable in its predictions, making it a critical metric for evaluating performance. High precision ensures that the abnormalities detected by the model are indeed present, reducing the chances of false alarms. High recall ensures that most abnormalities are detected, which is crucial for patient care. These metrics collectively ensure that the model's predictions are both accurate and comprehensive, enhancing its utility in clinical settings.

The classification report summarizes the class wise performance metrics, and it can be observed that the overall

```
Found 3197 images belonging to 14 classes.
50/50 [==============================] - 16s 303ms/step
Classification Report
                    precision    recall  f1-score   support

   Abnormal-ELBOW       0.88      0.74      0.80       230
  Abnormal-FINGER       0.88      0.65      0.75       247
 Abnormal-FOREARM       0.86      0.58      0.70       151
    Abnormal-HAND       0.86      0.42      0.56       189
 Abnormal-HUMERUS       0.86      0.79      0.83       140
Abnormal-SHOULDER       0.73      0.83      0.78       278
   Abnormal-WRIST       0.91      0.71      0.79       295
     Normal-ELBOW       0.75      0.93      0.83       235
    Normal-FINGER       0.69      0.88      0.78       214
   Normal-FOREARM       0.74      0.80      0.77       150
      Normal-HAND       0.70      0.96      0.81       271
   Normal-HUMERUS       0.83      0.85      0.84       148
  Normal-SHOULDER       0.80      0.73      0.76       285
     Normal-WRIST       0.78      0.95      0.86       364

         accuracy                           0.79      3197
        macro avg       0.80      0.77      0.78      3197
     weighted avg       0.80      0.79      0.78      3197
```

Fig. 30. Class wise Classification Report on the test dataset

The model achieved a weighted average F1-score of 0.78 and an overall accuracy of 0.79 across the 14 classes. These metrics indicate a balanced performance in terms of precision and recall for the multiclass classification task. Specifically, the class 'Abnormal HAND' showed modest recall, primarily due to high misclassification rates involving 'Abnormal WRIST', 'Abnormal FINGER', and 'Abnormal HAND'. This suggests that while the model is generally reliable, there are specific areas, particularly among similar abnormalities, where further refinement is needed to improve classification accuracy.

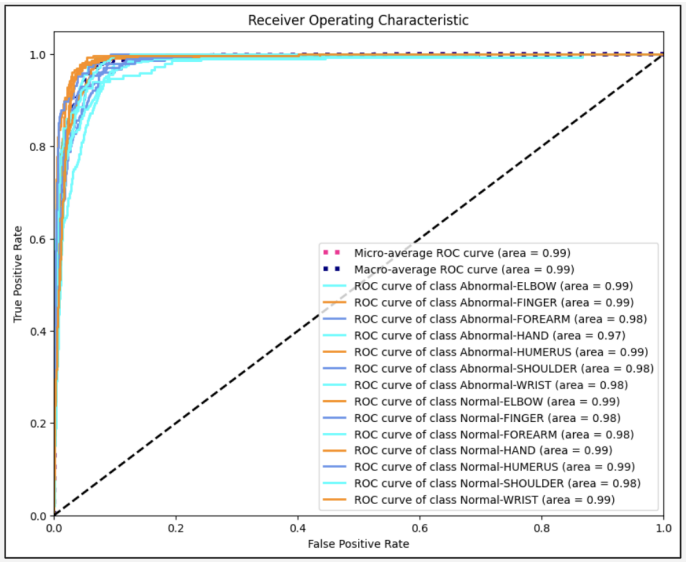


Fig. 31. Class wise AUCROC curves

The ROC AUC analysis demonstrates the model's high overall performance, with both micro-average and macro-average AUC values of 0.99, indicating exceptional performance across all classes. Most classes, including 'Abnormal-ELBOW', 'Abnormal-FINGER', 'Abnormal-HUMERUS', 'Abnormal-WRIST', 'Normal-ELBOW', 'Normal-HUMERUS', and 'Normal-WRIST', exhibit near-perfect AUC values of 0.99, reflecting high true positive rates and low false positive rates. Classes like 'Abnormal-FOREARM', 'Abnormal-HAND', 'Abnormal-SHOULDER', 'Normal-FINGER', 'Normal-FOREARM', and 'Normal-SHOULDER' show slightly lower AUC values of 0.98 and 0.97, suggesting minor misclassification issues. The close clustering of ROC curves near the top-left corner signifies consistent model performance with high sensitivity and specificity. Despite the overall excellent performance, there is room for improvement in the classes with AUC values of 0.97 and 0.98 through enhanced data augmentation, class-specific adjustments, or advanced model tuning.

Figure 32 displays some predictions from the test dataset;

A. *Explainability Maps*

One of the primary objectives of this project is to utilize the Explainability maps to understand the decisions taken by the models. The importance of explainability in this context is described with the below pointers,

- **Clinical Relevance**: In medical imaging, particularly with datasets like MURA (Musculoskeletal Radiographs), the stakes are high. Incorrect predictions can lead to misdiagnosis, affecting patient care. Explainability ensures that the model's decisions can be understood and

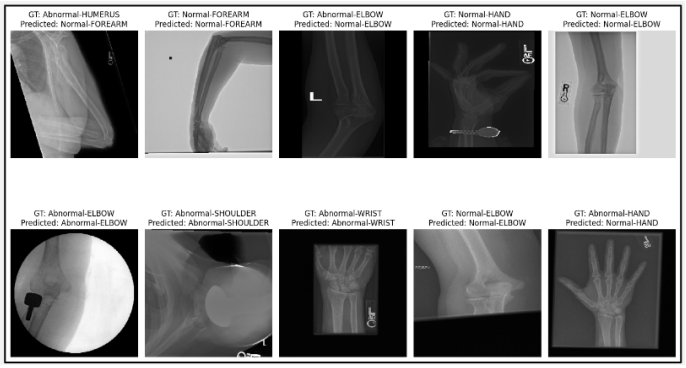


Fig. 32. Some of the Prediction Samples

trusted by radiologists and other healthcare professionals. It bridges the gap between complex AI algorithms and clinical practice, facilitating adoption in real-world settings.

- **Transparency**: Explainability provides transparency in the decision-making process of the model. For example, if a model predicts an abnormal wrist, the healthcare provider needs to understand why this prediction was made. Tools like Class Activation Mapping (CAM) highlight the regions in the radiograph that contributed to the model's decision, allowing practitioners to verify the validity of the model's focus.
- **Model Validation**: Explainability aids in validating the model's performance and identifying potential biases or errors. If the model consistently misclassifies certain types of abnormalities, understanding the decision process can reveal whether the issue lies in the data, the feature extraction process, or the model itself. This insight is crucial for iterative improvements and ensuring the model's robustness.
- **Patient Trust**: Involving explainable AI in clinical diagnostics can enhance patient trust. When healthcare providers can explain AI-driven diagnoses to patients, it improves the transparency and perceived reliability of the diagnosis. Patients are more likely to trust and accept AI-assisted medical advice if they know how conclusions are reached.
- **Regulatory Compliance**: In many regions, medical devices and diagnostic tools must comply with stringent regulatory standards. Explainability is often a requirement for regulatory approval, as it demonstrates that the model's predictions are based on understandable and medically relevant features. This is essential for the MURA dataset's application in real-world clinical settings.
- **Error Analysis**: Explainability helps in detailed error analysis. By understanding which parts of the image led to a particular classification, practitioners can identify whether errors are due to specific visual patterns or artifacts in the radiographs. This targeted analysis enables more effective corrections and refinements in both data preprocessing and model training.
- **Training and Education**: Explainable AI can serve as an educational tool for training radiologists. By showing how models interpret different radiographic features, it helps trainees understand complex patterns and improves their diagnostic skills. This dual benefit of training both AI models and human experts can lead to overall better diagnostic practices.

The methods utilized for Explainability in this project are - Attention Maps, Saliency Maps, Grad-CAM Maps, the below subsections detail the results and illustrate the explainability of the predictions.

B. Attention Maps

Attention maps are visual representations that highlight the regions of an image that a neural network model focuses on while making a prediction. In the context of convolutional neural networks (CNNs), attention maps are generated by applying weights to feature maps, which are derived from the convolutional layers of the model. These weights are learned during the training process and reflect the importance of different spatial regions in the image for the classification task. These attention maps are generated by utilizing the 3 attention heads which were included in Model architecture.

In the context of the MURA dataset, which involves classifying abnormalities in different body parts, attention maps are particularly useful. They help in understanding which regions of the musculoskeletal radiographs the model is focusing on, ensuring that the model is attending to relevant anatomical structures. This is crucial for both validating the model's accuracy and for gaining insights into potential areas of improvement. By providing a visual explanation of the model's decisions, attention maps enhance the trust and reliability of AI-assisted diagnoses in medical imaging.

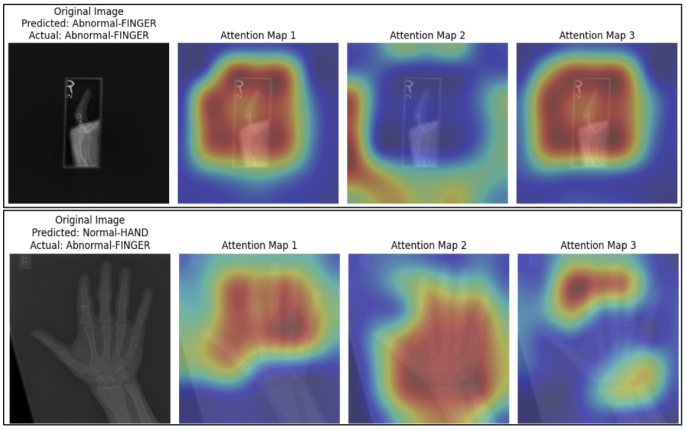


Fig. 33. Illustration of Attention Maps for a correct and wrong prediction

These visual representations illustrate which parts of the input image were most influential in the model's decision-making process. Typically, these maps are created by overlaying a heat-map onto the original image. The areas highlighted in brighter colors (such as red or yellow) indicate where the

model focused more attention, meaning these regions had a greater impact on the output prediction. By examining the highlighted regions in the attention maps, you can assess whether the model is focusing on anatomically relevant areas when making its predictions. For instance, if the model consistently highlights joint areas or visible abnormalities in the bones when predicting "Abnormal-FINGER," it likely understands the key features that define an abnormal X-ray.

C. Saliency Maps

Saliency maps are a type of visualization used to highlight the regions of an input image that most influence the model's predictions. These maps are created by calculating the gradient of the output class score with respect to the input image pixels. In simpler terms, saliency maps show which pixels in the image, when changed, would most affect the model's confidence in its prediction. The resulting map often highlights important features in the image, with higher values indicating greater influence on the model's decision. Radiologists and other medical professionals can use saliency maps to validate the model's predictions. By ensuring that the highlighted areas correspond to medically relevant features, experts can assess whether the model is making decisions based on sound medical evidence.

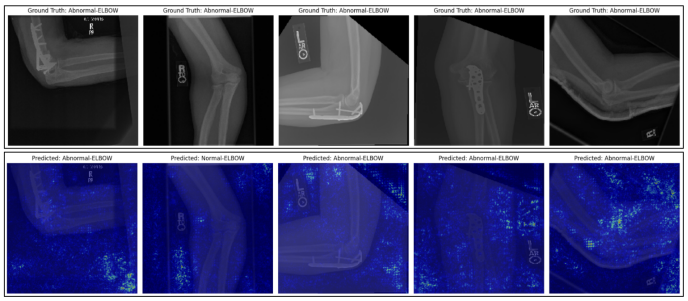


Fig. 34. Illustration of Saliency Maps for a correct and wrong prediction

The illustrated Saliency maps are generated using tf_keras_vis, a visualization library compatible with TensorFlow and Keras models. For our analysis, we focused on a specific convolutional layer (conv5_block16_2_conv), which is located deeper in the network. This layer is likely to capture high-level features that are crucial for making predictions about elbow abnormalities. It can be observed that for wrong predictions, the white dotted area is visible, and is being focused on other parts of the image, rather than focusing on the important segment.

D. Grad-CAM

Grad-CAM (Gradient-weighted Class Activation Mapping) is a technique used to generate visual explanations for decisions made by convolutional neural networks (CNNs). It works by utilizing the gradients of the target class flowing into the final convolutional layer to produce a localization map highlighting the important regions in the image. This helps in understanding which parts of the image the model focuses on when making a particular prediction.

In this context of multiclass classification with the MURA dataset, Grad-CAM is particularly useful because it allows us to visually interpret the model's decisions by highlighting areas in the radiographs that are most indicative of each class. For instance, when predicting whether an image depicts an abnormal wrist, Grad-CAM can show if the model is correctly focusing on regions where abnormalities typically appear. This not only provides insights into the model's decision-making process but also helps in validating that the model is learning relevant medical features, thereby increasing trust in the model's predictions.

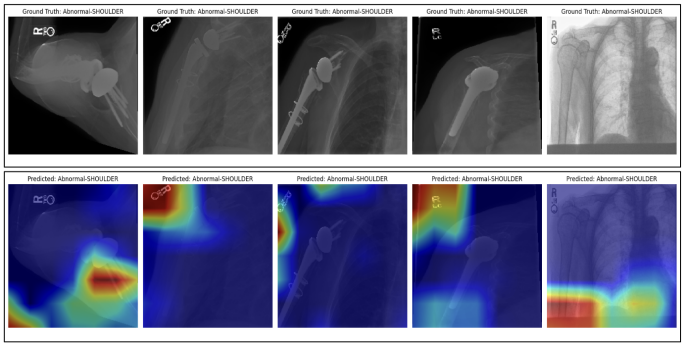


Fig. 35. Illustration of Grad-CAM Maps for a correct and wrong prediction

A specific convolutional layer, conv5_block16_2_conv, is utilized to capture high-level features in the image that are essential for making predictions. The feature maps from this layer are used to generate Grad-CAM heatmaps. Grad-CAM leverages the gradients of a target concept that flow into the final convolutional layer to create a coarse localization map. This map highlights the crucial regions in the image that influence the prediction of the target concept.

E. Cohen-Kappa Statistic

The Cohen Kappa statistic is a robust metric for evaluating the agreement between two raters, adjusting for the possibility of agreement occurring by chance. It ranges from -1 to 1, where a value of 1 indicates perfect agreement, 0 indicates no agreement beyond chance, and negative values indicate disagreement. In the context of multiclass classification with the MURA dataset, Cohen Kappa is particularly relevant as it provides a measure of the model's consistency and reliability in classifying medical images into various categories.

For the original competition involving the MURA dataset, Cohen Kappa was the primary metric used to judge model performance. This choice underscores the importance of not only achieving high accuracy but also ensuring that the model's predictions are consistent with expert human annotations, thus reflecting the practical utility of the model in clinical settings.

The Cohen-Kappa statistic in the comparison table, has all the models which worked on Binary Classification, so for this MultiClass classification problem statement, the achieved

Rank	Model Name	Cohen Kappa
	Best Radiologist Performance Stanford University Rajpurkar & Irvin et al.	0.778
1	base-comb2-xuan-v3(ensemble) jzhang Availink	0.843
2	base-comb2-xuan(ensemble) jtzhang Availink	0.834
3	muti_type (ensemble model) SCU_MILAB	0.833
4	base-comb4(ensemble) jtzhang Availink	0.824
5	base-comb2-jun2(ensemble)	0.814
	dl-team7-dense121	0.612
65	Bhaukali_v1.0 (single model) IIT BHU, Varanasi	0.589
68	Densenet169-lite(single model) Tang	0.56
69	ensemble1 ensemble	0.534
70	DenseNet (single model) Zhou	0.518

Fig. 36. Cohen-Kappa statistic leaderboard for the MURA classification

Cohen-Kappa score proves that the model will be able to predict the results fairly.

VI. Conclusion

This project successfully addressed the challenge of multiclass classification of musculoskeletal radiographic abnormalities using the MURA dataset. By extending the problem from binary to multiclass classification, the project aimed to predict specific body parts and types of abnormalities, thus providing more granular diagnostic insights. The DenseNet121 architecture, pre-trained on the ImageNet dataset, was fine-tuned using transfer learning techniques and augmented with self-attention mechanisms to enhance feature extraction and improve model accuracy.

The incorporation of explainability techniques such as attention maps, saliency maps, and Grad-CAM provided visual explanations of the model's decisions, thereby enhancing transparency and fostering trust among medical professionals. These techniques highlighted relevant regions in the radiographic images that contributed most to the model's predictions, ensuring that the model's focus aligned with anatomical relevance. The use of Cohen Kappa as a primary evaluation metric underscored the model's reliability and consistency in classifying medical images, aligning with expert human annotations.

The experimentation process demonstrated that modifications such as unfreezing the last five convolutional layers, integrating squeeze-and-excite blocks, adding multiple attention heads, and employing dropout regularization and L2 regularization significantly improved model performance. The final model achieved a weighted average F1-score of 0.78 and an overall accuracy of 0.79. The ROC AUC analysis further confirmed the model's high performance, with micro-average and macro-average AUC values of 0.99, indicating exceptional classification capability across all classes. This project not only achieved significant advancements in the classification of musculoskeletal radiographs but also set a foundation for future research to explore further enhancements in model architecture, data augmentation, and integration of additional explainability methods.

Overall, this project's contributions include transitioning the MURA dataset to a multiclass classification framework, enhancing model interpretability with advanced explainability techniques, and achieving robust performance metrics, making it a valuable tool for clinical use in diagnosing musculoskeletal abnormalities. Future research can build upon these findings to improve model accuracy, extend to other medical imaging modalities, and further integrate AI into clinical workflows for better patient outcomes.

VII. Future Scope

The future scope for this project is vast and promising, offering numerous avenues for enhancing model performance and expanding its applicability in medical imaging. Key areas for future work include:

- **Advanced Training Strategies**: Experimenting with the number and arrangement of layers in the DenseNet architecture could yield further improvements in accuracy and efficiency. Fine-tuning additional layers or incorporating novel layer configurations might better capture the intricacies of musculoskeletal radiographs.
- **Ensemble Models**: Creating an ensemble of neural networks or integrating various DenseNet-like architectures can boost the model's performance comprehensively. Ensembles are known to reduce variance and improve generalization, leading to more robust and reliable predictions.
- **Vision Transformers (ViT)**: Implementing Vision Transformers with transfer learning could offer a powerful alternative to traditional CNNs. ViTs are capable of capturing long-range dependencies in images and can potentially provide more nuanced explainability through their attention mechanisms. Exploring ViTs in this context could significantly enhance both performance and interpretability.
- **Edge-Efficient Models**: Developing models that are optimized for edge devices would enable real-time predictions in clinical settings, facilitating immediate diagnostic support. These models need to be lightweight yet accurate, ensuring they can run efficiently on devices with limited computational resources.
- **Explainability Techniques**: Exploring additional eXplainable AI (XAI) techniques will enhance the comprehension of neural network results. Techniques such as Local Interpretable Model-agnostic Explanations (LIME) or SHapley Additive exPlanations (SHAP) could provide deeper insights into model decisions, further bridging the gap between AI and clinical practice.
- **Synthetic Minority Over-sampling Technique (SMOTE)**: Addressing class imbalances in the dataset through techniques like SMOTE can improve model performance. SMOTE generates synthetic samples for underrepresented classes, ensuring the model receives a balanced training set, which is crucial for improving recall and precision in minority classes.
- **Weight Initialization for Radiography Images**: Developing specialized weight initialization techniques tailored for radiographic images could significantly enhance

model performance. By creating weights that are pre-optimized for the unique characteristics of medical images, training times can be reduced, and model accuracy can be improved.

- **Broader Research Opportunities**: The domain of medical image classification is ripe for research. Further studies can investigate various aspects of AI in healthcare, such as integrating multi-modal data (e.g., combining radiographs with patient records) or developing predictive models for other types of medical images.

By pursuing these avenues, the project can continue to advance, providing more accurate, efficient, and interpretable diagnostic tools for medical professionals, ultimately improving patient care and outcomes.

VIII. Important Links

- Github link - click here
- Drive link - click here

References

[1] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE 1998*, 1998.

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems 25 (NIPS 2012)*, 2012.

[3] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," *Computer Vision – ECCV 2014*, 2013.

[4] C. Szegedy, Y. J. Wei Liu, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," 2014.

[5] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *2016 IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[6] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[7] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018.

[8] M. Tan and Q. V. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," *International Conference on Machine Learning*, 2019.

[9] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *31st Conference on Neural Information Processing Systems (NIPS 2017)*, 2017.

[10] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," *ICLR 2021 Oral*, 2021.

[11] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, ""grad-cam: Visual explanations from deep networks via gradient-based localization," *International Conference on Computer Vision (ICCV)*, 2017.

[12] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Axiomatic attribution for deep networks," *Advances in Neural Information Processing Systems 25 (NIPS 2012)*, 2012.

[13] M. D. Zeiler and R. Fergus, "Attention branch network: Exploiting attentions for interpretable deep neural networks," *Computer Vision – ECCV 2014*, 2013.

[14] C. Szegedy, Y. J. Wei Liu, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Score-cam: Score-weighted visual explanations for convolutional neural networks," 2014.

[15] K. He, X. Zhang, S. Ren, and J. Sun, "Transformer interpretability beyond attention visualization," *2016 IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[16] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "A calibrated deep learning ensemble for abnormality detection in musculoskeletal radiographs," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[17] S. FAZEKAS, B. K. BUDAI, R. STOLLMAYER, P. N. KAPOSI, and V. BÉRCZI, "Artificial intelligence and neural networks in radiology," 2022.

[18] D. Lu and Q. Weng, "A survey of image classification methods and techniques for improving classification performance," *International Journal of Remote Sensing*, 2007.

[19] F. M. J. M. Shamrat, S. Azam, A. Karim, R. Islam, Z. Tasnim, P. Ghosh, and F. D. Boer, "Lungnet22: A fine-tuned model for multiclass classification and prediction of lung disease using x-ray images," *Journal of Personalized Medicine)*, 2022.

[20] F. J. M. Shamrat, S. Azam, A. Karim, K. Ahmed, F. M. Bui, and F. D. Boer, "High-precision multiclass classification of lung disease through customized mobilenetv2 from chest x-ray images," *Comput Biol Med 2023*, 2023.

[21] P. Rajpurkar, J. Irvin, A. Bagul, D. Ding, T. Duan, H. Mehta, B. Yang, K. Zhu, D. Laird, R. L. Ball, C. Langlotz, K. Shpanskaya, M. P. Lungren, and A. Y. Ng, "Mura: Large dataset for abnormality detection in musculoskeletal radiographs," *Standford Competition*, 2017.

[22] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," *Computer Vision and Pattern Recognition*, 2018.