Segmentation of Pediatric Hand Radiograph Using UNet for Bone Aging

A Thesis

submitted to
Indian Institute of Science Education and Research Pune
in partial fulfillment of the requirements for the
BS-MS Dual Degree Programme

by

Rashmi Sanjayrao Chapke

Indian Institute of Science Education and Research Pune
Dr. Homi Bhabha Road,
Pashan, Pune 411008, INDIA.

April, 2023

Supervisor: Dr. Pranay Goel

Certificate

This is to certify that this dissertation entitled Segmentation of Pediatric Hand Radiograph Using UNet for Bone Aging towards the partial fulfilment of the BS-MS dual degree programme at the Indian Institute of Science Education and Research, Pune represents study/work carried out by Rashmi Sanjayrao Chapke at Indian Institute of Science Education and Research under the supervision of Dr. Pranay Goel, Associate Professor, Department of Biology , during the academic year 2018-2023.

Dr. Pranay Goel

Committee:

Dr. Pranay Goel

Dr. Leelavati Narlikar

This thesis is dedicated to my Parents

Declaration

I hereby declare that the matter embodied in the report entitled Segmentation of Pediatric Hand Radiograph Using UNet for Bone Aging are the results of the work carried out by me at the Department of Data Science, Indian Institute of Science Education and Research, Pune, under the supervision of Dr. Pranay Goel and the same has not been submitted elsewhere for any other degree.

Rashmi Sanjayrao Chapke

Acknowledgments

I want to convey my profound appreciation to everyone who has provided me with their support and motivation throughout this thesis. I would like to express my sincere gratitude to Dr. Pranay Goel, my supervisor, for his unwavering dedication, guidance, encouragement, and insightful feedback throughout this thesis. He has been a great mentor and a source of inspiration for me. I extend my gratitude to Dr. Leelavati Narlikar, my expert member, whose constructive feedback and valuable suggestions helped me enhance the quality of my work.

The collaboration of Dr. Anuradha Khadilkar, Dr. Chirantap Oza, Dr. Shruti Mondkar, and Dr. Vaman from Hirabai Cowasji Jehangir Medical Research Institute (HCJMRI), Pune has been vital to the successful completion of this thesis. Their clinical expertise and guidance in image labeling have been invaluable. I express my deepest gratitude for their significant collaboration and support. Additionally, the support and resources provided by PARAM Brahma Facility under the National Supercomputing Mission, Government of India at the Indian Institute of Science Education and Research; Pune are gratefully acknowledged. Furthermore, I would like to acknowledge the assistance provided by ChatGPT, an AI language model developed by OpenAI, for its writing assistance, which was instrumental in refining and articulating my ideas.

I extend my heartfelt gratitude to Somashree Chakraborty, Sandra, Arjun, Prajjwal, Suyog, and Mukta, my lab members, for their support, guidance, and encouragement during the completion of this thesis. I would also like to thank my friends, Devarsh Patel, Prajwal Jadhav, and Kartik Kakade, who have supported me throughout my academic journey. Their unwavering encouragement and assistance in problem-solving have been invaluable to me. I thank my family for their unconditional love and support throughout my academic journey. My parents have always encouraged me to pursue my dreams and aspirations.

Abstract

The skeletal growth of children is often assessed by calculating bone age. Often developmental age differs from chronological age; hence bone aging is one of the essential steps in the clinical procedure of estimating the biological maturity of children. Assessment of bone age can be done in a traditional, manual way or using automated methods. Manual methods like Greulich-Pyle (GP) and Tanner-Whitehouse (TW) are time-consuming and involve intra- and inter-rater variability. Automated software are probably more reliable, and several are trained on a caucasian dataset given by the Radiological Society of North America (RSNA), consisting of 12,611 X-rays of boys and girls between the ages of 0 and 18 years. The GP method compares the patient's full-hand radiograph to reference images in the GP atlas.

However, in different ethnic groups, the maturation of different bone segments varies compared to Caucasian children having the same chronological age. To address these difficulties, our collaborators have recently developed a method called the "Segmental GP rating" (Oza et al. 2023 submitted) which explicitly accounts for inter-segmental variability in the bones of the hand. We aim to develop an AI-based model that reproduces this novel method.

Our bone age model was developed using the RSNA dataset. A UNet architecture was employed to segment four regions of interest (ROIs) from the X-rays. The image crops generated from segmented ROIs were subsequently used to train a DenseNet regression model for predicting bone age for the full-hand and three individual ROIs (short bones, carpals, and wrist).

The final age of full-hand radiograph was taken as weighted sum of the predicted ages for the three ROIs. The weight values for each ROI were estimated using multivariate linear regression on a validation set. The obtained weight values for short bones, wrists, and carpals in boys were 0.57, 0.26, and 0.18, respectively. For girls, the weight values for the same ROIs were 0.69, 0.16, and 0.17, respectively. By combining the model's age predictions for the three segments using these weights, we obtained a Mean Absolute Distance (MAD) of 6.7 months for boys and 7.4 months for girls on the validation set.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Bone age assessment (BAA) is vital tool in pediatrics that helps clinicians evaluate the skeletal development of a child. It involves estimating the physiological growth of an individual from bone age, which can be determined through careful evaluation of hand radiographs by experts, usually from the non-dominant hand. Skeletal abnormalities can be detected from the discrepancy between bone age (BA) and chronological age (CA), which is the age calculated from the birth date. A difference between the two can indicate delayed or accelerated ossification in bones caused by illness.

One of the primary reasons why clinicians need to calculate bone age is to make informed decisions about a child's healthcare needs. For example, knowing a child's bone age can help clinicians determine the appropriate treatment for conditions such as growth hormone deficiency or delayed puberty.[1] It can also help identify skeletal abnormalities or growth disorders that may require further evaluation or monitoring. Additionally, bone age assessment can track an individual's growth and development over time, which is important for children with chronic illnesses or conditions that affect their growth.

1.1 Motivation

While bone age assessment is crucial for identifying potential health concerns, it is important to note that it is a subjective process and depends on the expertise of the the person

evaluating the radiographs. Additionally, epiphyseal plate closes between ages of 14 to 18,[2] which means that bone age can only be calculated from hand radiographs in children up to the age of 18 years.

The two most widely used methods for BAA are based on hand radiographs [3]. First, Greulich-Pyle(GP), which contains standard reference hand radiograph images from upper-middle class Caucasian children in Ohio, United States. Bone age is measured by comparing the patient's X-ray with standard reference X-rays in the GP atlas [4]. Second, Tanner-Whitehouse(TW) is based on the scoring method, and each bone is assigned a score based on the maturation and sex of the patient [5]. Both these methods are time-consuming and prone to intra or inter-rater variability. The study emphasizes the GP standards for estimating bone age.

Automating BAA can reduce the estimation time, inter and intra-rater variability and relieve doctors from the cumbersome BAA procedure [6]. The automated methods include automatic detection of the region of interest(ROI), followed by BAA through an algorithm. Most of these methods are based on hand and wrist radiographs.

The radiograph provides valuable information about the shape and configuration of bones, which plays a important role in predicting bone maturity. The shape and configuration of bones change as an individual grows, which can be indicative of their overall stage of physical development. The radiograph captures this characteristic information and can help clinicians and researchers make more accurate predictions about bone maturity.

1.2 Related Work

The bone age assessment process has undergone several attempts at automation since the mid-19th century. Beginning in the late 1980s, researchers developed semiautomated and fully automated systems that used image-processing techniques to reduce observer variability and improve accuracy.

- The first automated system, HANDX, was introduced in 1989 and uses image processing techniques to segment bone features in X-ray images of the hand and wrist and reduce observer variability. The system carries three steps: preprocessing, segmenta-

tion, and estimation. It has yet to be evaluated on larger and different data and may not be accurate when the full-hand image is used. Also, it requires human intervention [7].

- Pietka and colleagues, in 1991, developed a method for bone age estimation based on analysis of PROI region. PROI consists of epiphysis and phalanges. The system first scans a horizontal line to detect the lower and upper boundaries of the PROI, and a gradient image is used to segment the bones. The boundary between the proximal, middle, and third distal phalanges is measured. The system was evaluated on 50 computer radiographs and showed a mean difference of 0.02 mm with a measurement error of 0.08 mm compared to a radiologist's measurements. Though it has a low error rate, it is evaluated on a very small scale [8].

- In 1994, Tanner and Gibbons introduced a semi-automated system called CASAS that was tested and assessed using X-ray images from children in stable and normal pathologic positions. The system was more accurate and repeatable than the manual TW method for children in normal situations. Still, it did not work for assessing clinical problems due to bone deformation and required human interventions [9].

- Gross built a system in 1995 that utilized a neural network to estimate BA based on measurements from hand and wrist radiographs. However, the model did not use the morphological features of other methods and had no major advantage over manual methods [10].

- In 2012, Mansourvar et al. created a computerized method for assessing bone age that relied on histogram-based compression techniques. This automated system employed content-based image retrieval (CBIR) methods to analyze images and had a database of 1,100 hand X-ray radiographs, categorized by ethnicity and gender. The system's assessment demonstrated an error rate of 0.170625 years, suggesting it is a reliable tool for BAA. However, the system may not be dependable for low-quality images or X-rays that exhibit abnormal bone structures [11].

- BoneXpert is the first AI-based BAA software introduced in 2008 and most widely used in Europe. It leverages image-processing techniques to automatically reconstruct the borders of 15 bones. It then computes "intrinsic" bone ages for each of the 13 bones (phalanges, metacarpals, and radius ulna) based on their shape, texture, and intensity and subsequently transforms them into GP and TW bone age using statistical

models. It covers boys and girls till the GP age range of 17 and 15, respectively, while its new version extended the age limit to 19 years for boys and 18 years for girls [12]. It rejects the bad quality image and images with abnormal bone morphology [13]. It is developed on approximately 1500 images and validated on GP atlas and TW-rated images, yielding a standard deviation(SD) of 0.42 years (**5.04 months**) and 0.8 (**9.6 months**) years respectively [14].

- Kim et al. conducted an internal validation of BoneXpert in 2017 which gave RMSE(Root Mean Squared Error) of 7.2 months with reference bone age [15].

- External Validation study shows that MAE and RMSE(Root Mean Squared Error) values are greater for CA and reviewer bone age than for CA and software estimated bone age[16].

- RSNA Bone Age Challenge winner 16 Bit Inc achieved MAD-Mean Absolue Distance of **5.99 months** on the validation set using single Inception V3 architecture[13].

- BoneXpert has been utilized for bone age assessment in healthy children of diverse ethnicities. The accuracy of BoneXpert was found to be 0.56 years in a Chinese population [17], and a Dutch study reported a deviation of 0.71 years for both genders combined [18]. Another study on automated BAA in American children for four ethnicities estimated an accuracy of 0.52 years [19]. The Tubingen study from Germany reported an RMSE of 0.72 years for BoneXpert [20].

- Oza et al.[21] conducted a study to determine the accuracy of BoneXpert in determining bone age in healthy children of Asian Indian ethnicity, as compared to traditional manual methods such as GP and TW. The researchers found that the RMSE values of BoneXpert were 0.36, 0.41, and 0.39 years for TW3, TW2, and GP methods, respectively, compared to true bone age. The study also indicated that the RMSE values were significantly lower in girls than boys. The authors discovered that the best agreement between BoneXpert and the standard manual rating was achieved by using a 50 percent weighting on carpals (GP50), due to the fact that carpal bones were comparatively delayed relative to the GP (tubular) bone age, especially in boys. Additionally, the study observed that BoneXpert tended to overestimate bone age in girls and underestimate it in boys in the pubertal age group, indicating varying advancements of bone age concerning chronological age in both genders.

1.3 Our Contribution

The previous literature review has highlighted the variability in the maturation of bone segments among various ethnic groups, which differs from Caucasian children of the same chronological age. Consequently, developing more advanced and comprehensive software for assessing bone age is necessary. Therefore, our study aims to establish a more generalized approach to assessing bone age in diverse ethnic populations, which will provide a more accurate bone age estimation and improve the clinical management of children. By developing a more comprehensive method for assessing bone age, our study intends to fill the gap in the current BAA methods, primarily designed on Caucasian children's data. To address these difficulties, our collaborators have recently developed a method called the "Segmental GP rating"(Oza et al. 2023 submitted, Standardization of weightage assigned to different segments of the hand X-ray for assessment of bone age by the Greulich Pyle method) which explicitly accounts for inter-segmental variability in the bones of the hand. Our study involved estimating weights for each bone segment using multivariate linear regression. The weights reported by Oza et al. differ from those estimated in our study. Subsequently, we calculated the final age by taking the weighted average of each segment.

Chapter 2

Preliminaries

2.1 Neural Networks

Neural networks (NNs) are complex computational systems used for various applications, including image recognition, speech recognition, machine learning, and many more. NNs are inspired by the way that the human brain processes information. NNs are also known as stimulated neural networks(SNN) or artificial neural networks (ANN). They consist of several layers of neurons which are inter-connected with each other. Each neuron in the network has an associated weight and bias. Neuron processes data and output information to other neurons.

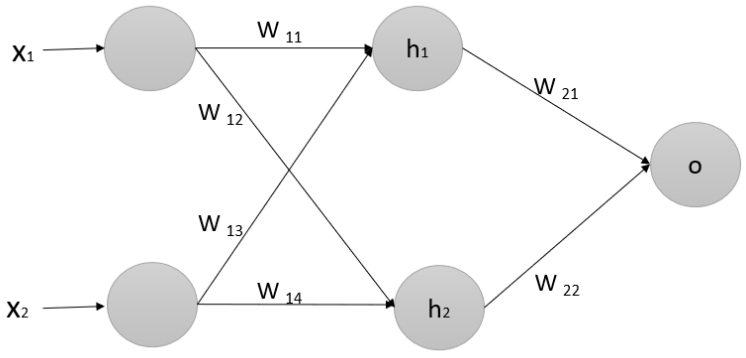


Figure 2.1: Neural Network

$$
\begin{aligned}
h_1 &= g(w_{11}x_1 + w_{13}x_2 + b) \\
h_2 &= g(w_{12}x_1 + w_{14}x_2 + b)
\end{aligned}
\tag{2.1}
$$

Where g is an activation function.

$$
O = w_{21}h_1 + w_{22}h_2 + b \tag{2.2}
$$

2.2 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are a type of deep neural network that are generally used in computer vision tasks, such as video and image video recognition, segmentation, and object detection. They learn patterns and features from the image and video data, which makes them effective for image and video processing.

CNNs process images as a tensor. Convolution measures how much two functions overlap when one passes over the other. It use convolutional layers to extract local features from the input images [22]. These layers apply a set of learnable filters, also known as kernels, to the image or video. In figure 1.2 you can see that kernel slide over the image, performing the multiplication operation between kernel weights and image values to extract the features from the image. The output of these multiplication operations is feature maps that represent different aspects of the image. The kernel weights are then updated after each iteration. The network can learn increasingly complex features and patterns by stacking multiple convolutional layers.

2.3 Activation Function

The activation function is applied to the output of each neuron. It introduces non-linearity in the network and allows the neural network to learn complex patterns and make more accurate predictions. Without an activation function, the neural network would be just a linear regression model. The activation function determines the neuron's output based on its input, which can be either a single value or a vector of values. The most common activation

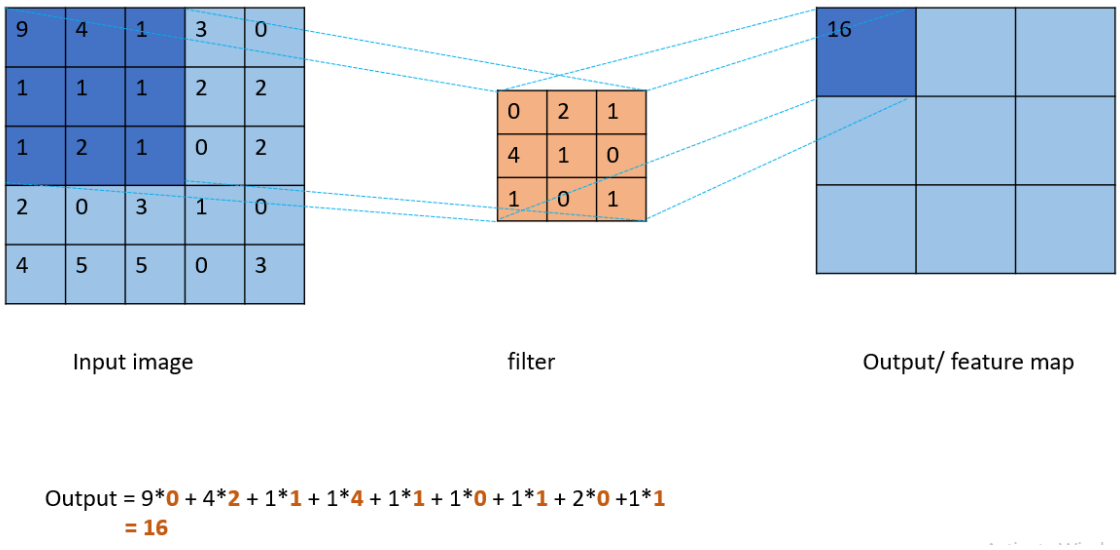


Figure 2.2: Convolution Neural Network sliding window diagram

functions used in CNNs are sigmoid and ReLU (rectified linear unit) functions. The choice of activation function depends on the type of problem solved and the structure of the neural network.

2.3.1 Sigmoid Function

It maps input values between 0 and 1. It is defined as follows:

$$f(x) = 1/(1 + e^{-x}) \tag{2.3}$$

Where x is the input to the function. The sigmoid function is used in the output layer of binary classification models, where the goal is to predict a binary outcome.

Properties of the sigmoid function

- Smoothness: The sigmoid function is differentiable, making it suitable for gradient-based optimization.
- Range limitation: The output of the sigmoid function is always limited to the range between 0 and 1, but it can also lead to the vanishing gradient problem.

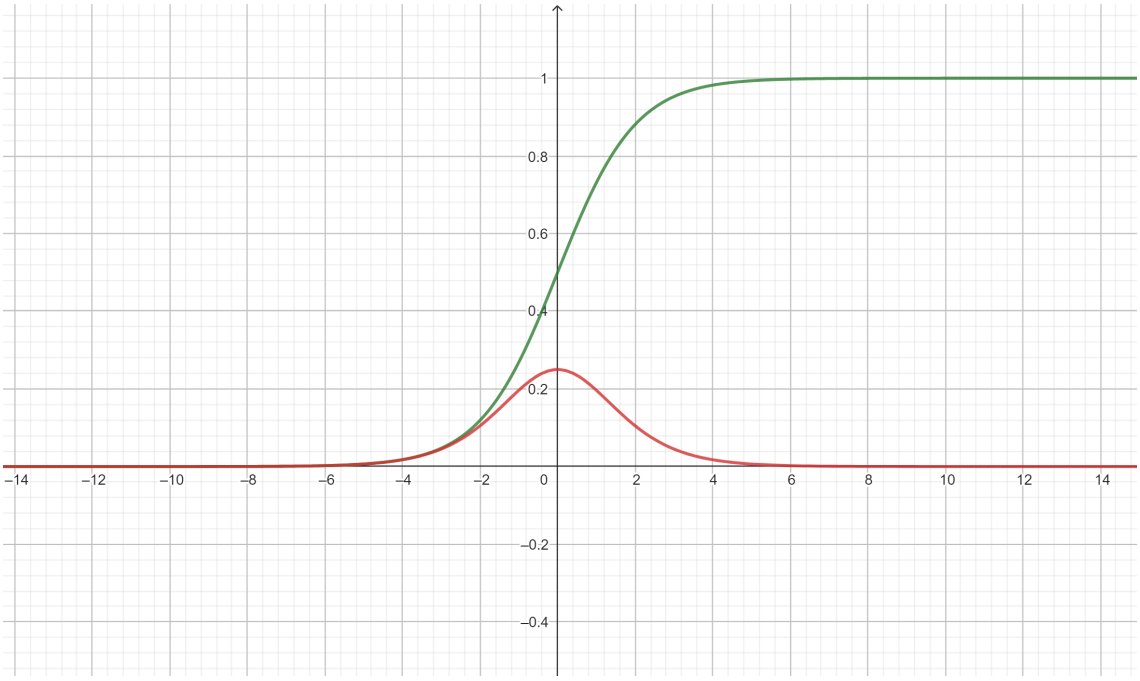


Figure 2.3: Sigmoid Function

- The vanishing gradient problem arises when we train large neural networks. The gradient of the loss function with respect to weights in the previous layer becomes very small, which leads to slow or stalled learning of the network. The sigmoid activation function has a very small derivative between 0 and 0.25. When the network is trained, the derivative becomes exponentially small as they are backpropagated through layers, leading to a vanishing gradient problem.

- Computationally expensive: The sigmoid function involves exponential calculations, making it computationally expensive compared to other activation functions such as the ReLU.

2.3.2 ReLu

The ReLU activation function is a non-linear function that maps input to a value between 0 and infinity [23]. It is defined as follows:

$$f(x) = max(0, x) \tag{2.4}$$

where x is the input to the function. The ReLU function is often used in hidden layers of deep neural networks because of its computational efficiency and ability to avoid the vanishing gradient problem that can occur with sigmoid and tanh activation functions.

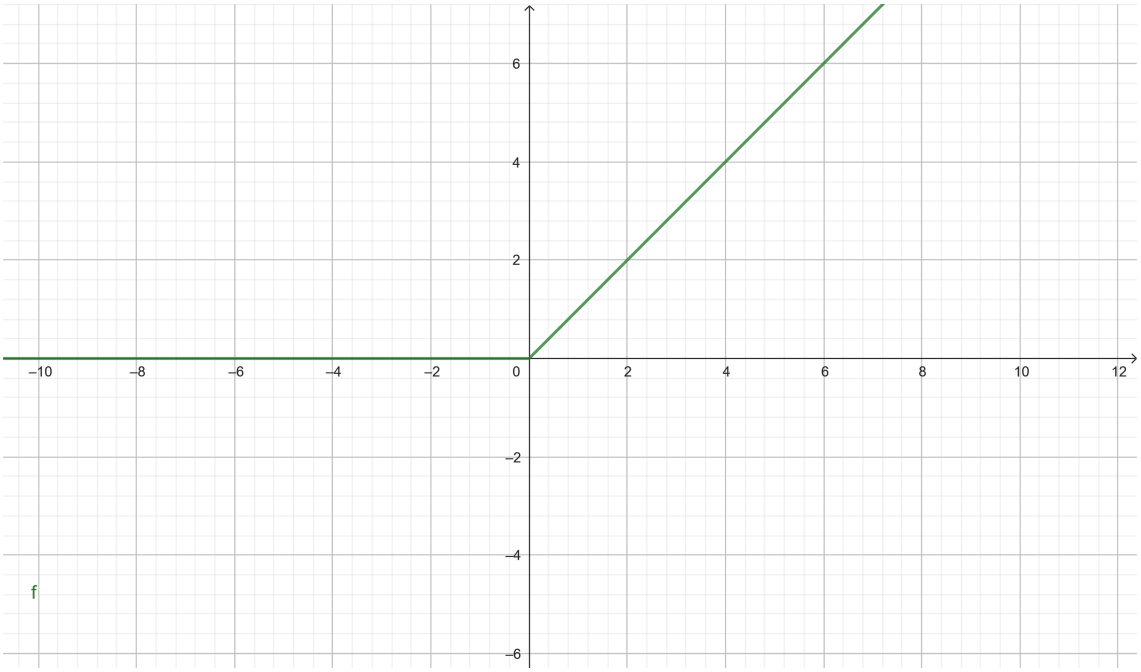


Figure 2.4: ReLU function

Properties of the ReLU function

- Non-linearity: The ReLU function introduces non-linearity to the output of each neuron, which is necessary for deep neural networks to learn complex patterns and make accurate predictions.

- Sparsity: The ReLU function can produce sparse outputs, meaning only a few neurons are activated for a given input. This can reduce the computational complexity of the neural network.

- Avoids vanishing gradient problem: The ReLU function helps to avoid the vanishing gradient problem that can occur with sigmoid and tanh activation functions. This is because the derivative of the ReLU function is either 1 or 0, which means that the gradients will not become too small during backpropagation.

2.4 Batch Normalization

Batch Normalization (BN) is a widely employed method in (CNNs) that enhances their performance and training. It addresses the internal covariate shift problem in CNNs [24], which refers to the variations in the distribution of intermediate feature representations that occur as a result of changes in the input data or parameters during the training process of a neural network. This phenomenon can lead to slower convergence rates and poor generalization performance of the neural network.

BN normalizes the inputs to each convolutional layer across a mini-batch of samples. The normalization is performed independently for each feature map of the convolutional layer. Throughout the training, the mean and standard deviation of each feature map's activations is calculated for each mini-batch. These statistics are then used to normalize the activations by subtracting the mean and dividing by the standard deviation. Finally, the transformed activations are passed to the next layer in the network.

By normalizing the inputs to each layer, Batch Normalization helps to stabilize the training process and reduce the dependence of the network on the initialization of the parameters. It has been shown to improve the accuracy and speed of training of CNNs, reduce overfitting, and enhance the performance of the network. As a result, it has become a standard technique in the development of state-of-the-art CNNs.

2.5 Loss Function

The loss function sometimes called a cost function, assesses how well the model predicts the desired outcome from a given input. The loss function takes ground truth and predicted output as the input to measure how well the model did the prediction. While training, our goal is to minimize the loss function. Different types of loss function can be used depending on the problem being solved. Choosing an appropriate loss function is important.

2.5.1 Binary Cross Entropy Loss

Binary cross-entropy loss is a commonly used loss function for pixel level classification [25] and in machine learning, it measures the difference between predicted probabilities and true labels for binary classification problems [26]. It is defined as the negative the logarithm of the likelihood function, which assumes that each observation in the dataset is independent and follows a Bernoulli distribution.

$$L(y, \hat{y}) = -[y \cdot log(\hat{y}) + (1 - y) \cdot log(1 - \hat{y})] \tag{2.5}$$

For a batch of training examples, the BCE Loss is the average of the loss for each example.

2.5.2 Mean Squared Error Loss (MSE)

The MSE loss is given by:

$$L(y, \hat{y}) = (y - \hat{y})^2 \tag{2.6}$$

Commonly used in regression problem. For a batch of training examples, the MSE loss is the average of the losses for each example. It is a quadratic loss function, meaning that it penalizes the model more heavily for making large errors than small errors. It is sensitive to outliers. Other types of loss such as mean absolute error (MAE) loss may be more appropriate in certain cases.

2.5.3 Mean Absolute Error (MAE)

MAE loss is used in a regression problem. It measures the difference between the predicted output of the model and the true output in terms of their absolute difference. MAE is calculated as follows:

$$L(y, \hat{y}) = |y - \hat{y}| \tag{2.7}$$

For a batch of training examples, the MAE loss is the average of the losses for each example. It is a linear loss function, meaning that it penalizes the model equally for making small and

large errors. MAE is less sensitive to outliers.

2.6 Peformance Evaluation Matrices

2.6.1 Mean Absolute Distance (MAD)

MAD calculates absolute difference between true label and predicted label.

$$MAD = |y - \hat{y}| \tag{2.8}$$

2.6.2 Dice Score

In medical image segmentation, the Dice score is a common metric used to evaluate the similarity between the predicted and ground truth segmentations. It measures the overlap between the predicted and true segments of an image. The dice score measures the performance of a model, ranging from 0 to 1. One corresponds to a pixel-perfect match between the model's output and annotated ground truth.

$$DiceScore = \frac{2\times \mid X \cap Y \mid}{\mid X \mid + \mid Y \mid} \tag{2.9}$$

i.e., simply two times the area of overlap between output and ground truth divided by the sum of the total number of pixels in both images.

2.7 Optimizer

Optimizer modifies the model's parameters to reduce the loss function during training. It determines the neural network's weights and biases for best fitting the training set of data. To do this, the parameters are iteratively updated using the gradient of the loss function. It updated the weights using the following formula:

$$W_{new} = W_{old} - \eta \times gradient \tag{2.10}$$

where η is the learning rate.

The optimizer uses the direction of the steepest gradient descent to update the parameters to reduce the loss. There are different types of optimizers:

2.7.1 Batch Gradient Descent

Batch gradient descent computes the gradient over the complete training set at once in batch gradient descent [27]. This can be computationally expensive for large datasets, but it has the advantage of guaranteeing convergence to a global minimum if the learning rate is chosen appropriately.

One disadvantage of batch gradient descent is that it can get stuck in local minima or saddle points. Other optimization algorithms like stochastic gradient descent and mini-batch gradient descent can be used to overcome this.

2.7.2 Stochastic Gradient Descent (SGD)

SGD updates parameters for each training example. Thus, over conventional gradient descent, SGD has faster convergence [27]. As we only take into account one example at a time, the cost will vary over the training examples and won't definitely go down. Yet, over time, you'll see that costs are fluctuating downward.

2.7.3 Mini-Batch Gradient Descent

It divides training data into mini-batch using random sampling, computes the gradient of the loss function with respect to model weights using data only in minibatch, then updates the parameters based on the gradient. This process is carried out repeatedly until convergence [27].

2.7.4 Adagrad- Adaptive Gradient Algorithm

Gradient descent uses the same learning rate for each input variable. It is challenging to set the learning rate. If it is too small, learning will be very slow and will take a long time to train, or if it is too large, the loss function will oscillate and may not reach the global minimum [27]. In addition, the high-dimensional nature of neural network optimization leads to varying sensitivity in different dimensions. To tackle this problem, we could choose different learning for different network dimensions. Adagrad automatically adapts the learning rate based on the calculated gradient over the search.

2.7.5 Adam optimizer - Adaptive moment estimate

It is a combination of Adagrad and RMSProp (Root Mean Square Propagation) [27]. RMSProp maintains per-parameter learning rates that are adjusted in accordance with the recent magnitudes of weight gradients' average.

Adam calculates the second moment of the gradients instead of the first moment, as in RMSProp. It specifically calculates squared error and exponential moving average of the gradient.

2.8 Backpropogation

Backpropagation is the backward propagation of error. It calculates the gradient of the loss function with respect to the weight of each neuron in a layer. These gradients are calculated from the last layer to the first layer in a backward direction [28].

Chapter 3

Methodology

3.1 Resources and Tools

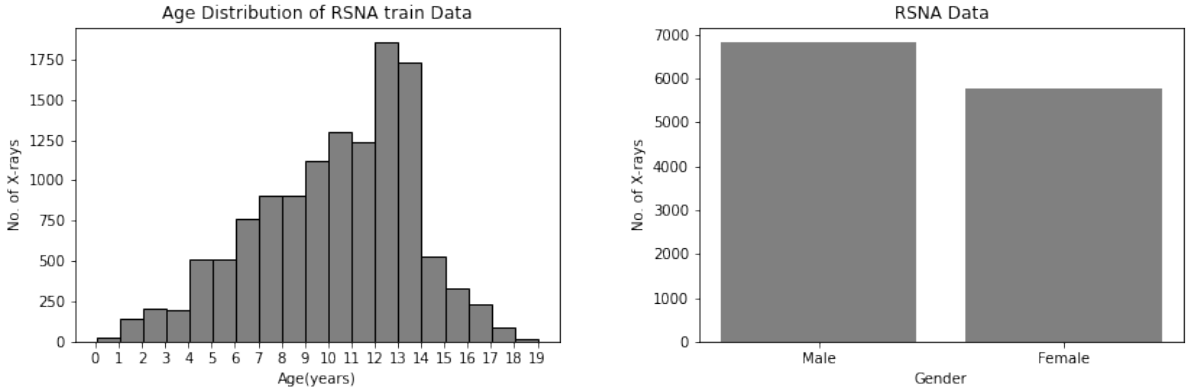
The project was developed using Python version 3.8.1 and employed the PyTorch library (1.13.1+cu117) for creating deep neural network models. Image processing tasks were heavily reliant on the OpenCV library, while CVAT was used for data labeling. Visualization was done using Matplotlib and torchshow. Jupyter notebooks were utilized for prototyping and data exploration on an HPC Linux machine named Param-Brahma. The project relied heavily on the numpy and pandas Python modules for scientific computing functions and data structures.

3.2 Dataset details

The study uses a public database provided by the Radiological Society of North America (RSNA) for the Pediatric Bone Age Challenge 2017 [13]. It consists of a train and validation data.You can find train and validation data here. The training set consists of 12,611 hand radiograph images and their corresponding age in months from male and female subjects ranging from 0 to 19 years of age. Figure 3.1 (a) and 3.3 (a) shows that both training and validation datasets have skewed normal distribution. Number of training images belonging

to male and female subjects:

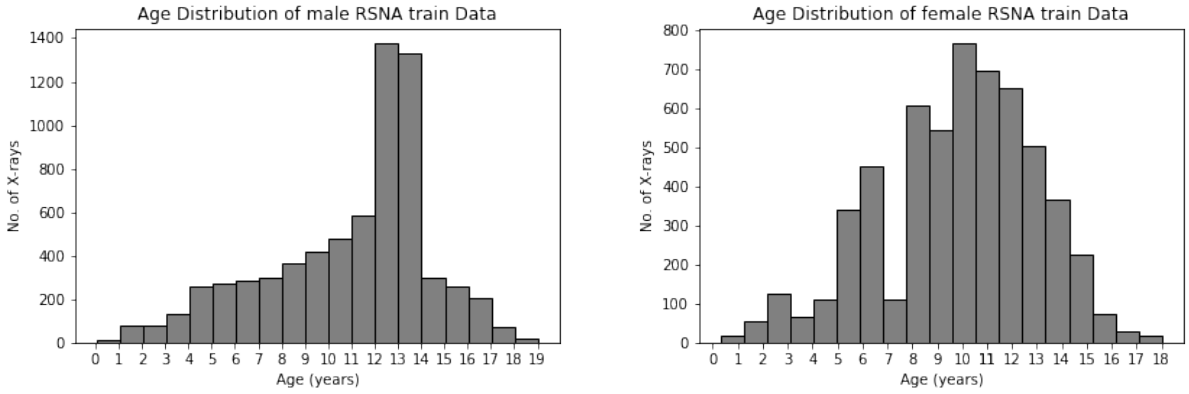
$$Male : 6833$$
$$Female : 5778$$



(a) Training data distribution across 0-19 years of age.

(b) Training data distribution across gender.

Figure 3.1: The figure represents the distribution of 12,611 training images by age and gender.



(a) Distribution of male hand radiographs across 0-19 years of age.

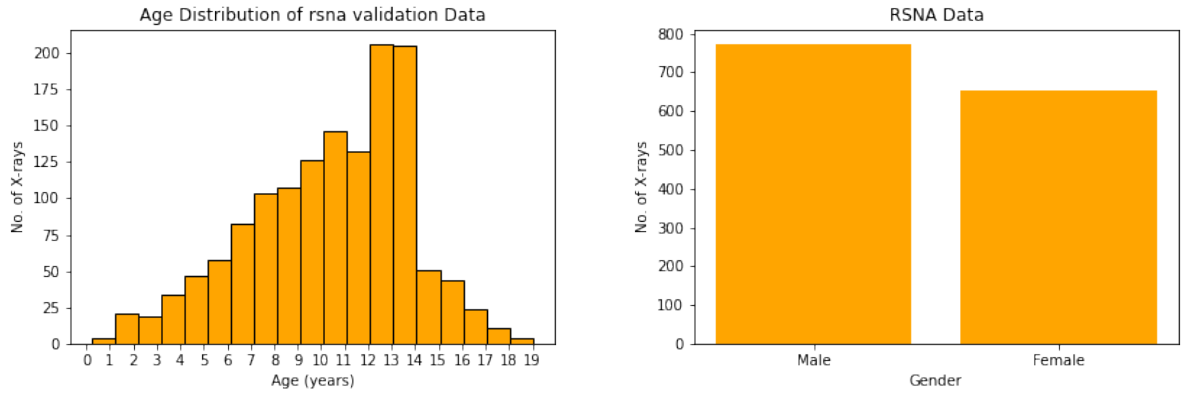
(b) Distribution of female hand radiographs across 0-19 years of age.

Figure 3.2: Distribution of training images belonging to males and females across 0-19 years of age

The validation set consists of 1,425 hand radiograph images.

$$Male : 773$$

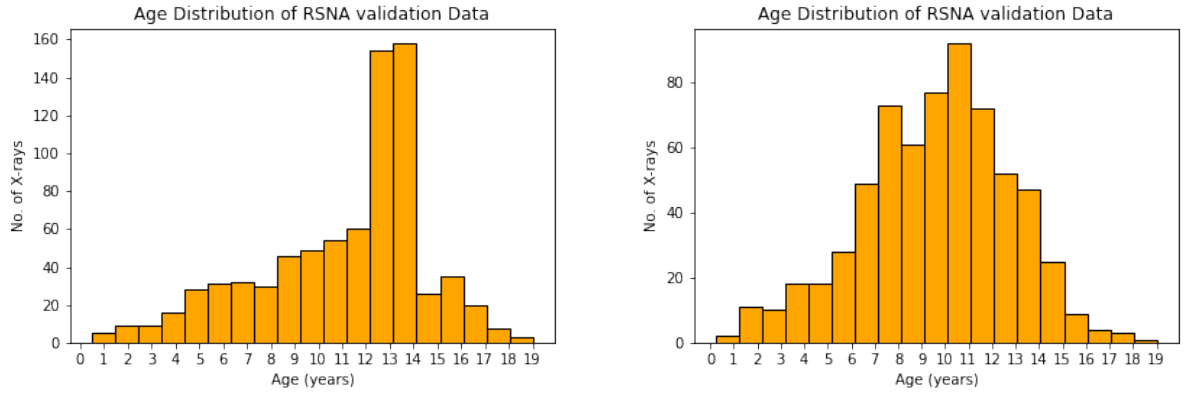
$$Female : 652$$



(a) Validation data distribution across 0-19 years of age.

(b) Validation data distribution across gender.

Figure 3.3: The figure represents the distribution of 1425 validation images by age and gender.



(a) Distribution of male hand radiographs across 0-19 years of age.

(b) Distribution of female hand radiographs across 0-19 years of age.

Figure 3.4: The figure represents the distribution of validation images belonging to males and females across 0-19 years of age

3.3 Thesis Structure

Thesis is divided into two main parts. First, is automating data labeling i.e, segmentation of radiographs and second, bone age prediction. The details of this are discussed in chapter 4 and chapter 5 respectively.

Chapter 4

Segmentation of Hand Radiographs

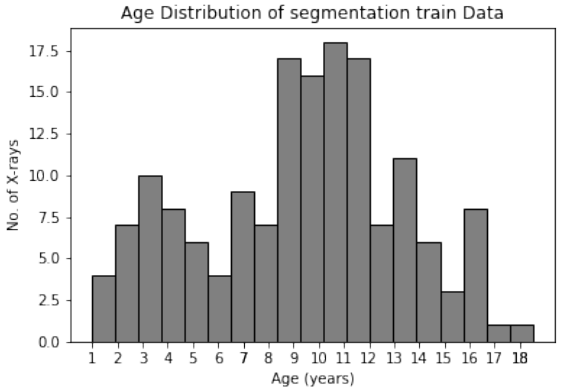
According to G-P based BAA methods, multiple ROIs are required to assess bone maturity. These ROIs in hand radiographs serve as the foundation for Deep Learning (DL)-based BAA. Annotation of these ROIs is an essential step in data preprocessing. ROIs must be detected before the prediction of bone age in automated BAA systems. This work utilizes commonly used UNET architecture for ROI extraction. We have considered four ROI or segments. These four segments are phalanges, metacarpals, carpals, and radius ulna.

We first extracted the desired bone segments from the radiograph for predicting bone age. Manual segmentation of radiographs in large datasets is not possible. So, we have automated the segmentation process so that if a new hand radiograph is given to the model, it will give you the desired bone segment.

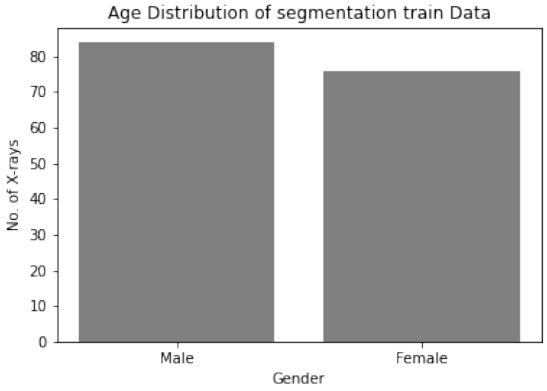
4.1 Data Preparation

A set of 200 radiographs were randomly chosen from RSNA training set to train CNN models for image segmentation. The original images in the dataset were not all of the same sizes. In order to address this, black borders were added to each image. The size of the borders was determined by calculating the difference between the height and width of each image. If the height of the image was greater than its width, then an equal border, which was half the difference between the height and width, was added to both sides of the width. Alternatively,

if the width was greater than the height, then a border of equal size was added to the top and bottom of the image. After this images were resized to 2000 by 2000. This ensured that all images in the dataset had the same dimensions and could be used for training without causing issues due to variations in size. These radiograph data were then split into two sets: a training set of 160 images and a validation set of 40 images. In order to ensure that the training set was representative of the overall population, we included images of both genders across all age ranges. The training and validation data distribution across different ages can be seen in Figures 4.1 and 4.2.

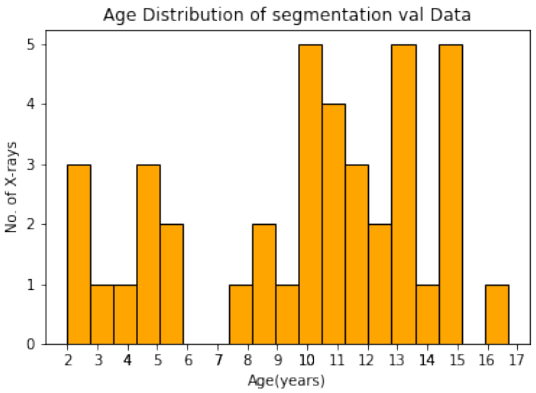


(a) Training data distribution across 0-19 years of age.

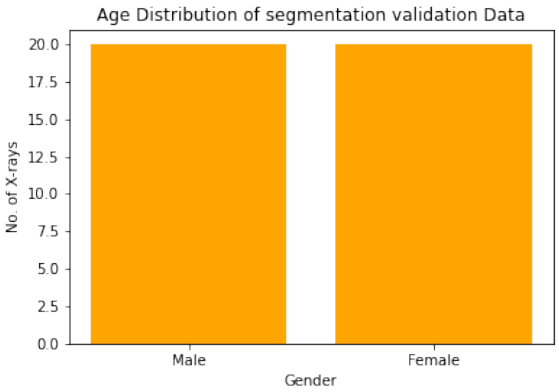


(b) Training data distribution across gender.

Figure 4.1: The figure represents the distribution of 160 randomly chosen training images by age and gender.



(a) Validation data distribution across 0-19 years of age.



(b) Validation data distribution across gender. Both the gender category contains same number of images.

Figure 4.2: The figure represents the distribution of 40 validation images by age and gender.

4.2 Image Annotation

Image annotation refers to adding additional information or labels to an image. This information could include objects, regions, or specific features within the image. Image annotation is commonly used in computer vision and machine learning applications to train algorithms to identify and recognize objects within images. The annotations could be in the form of bounding boxes, polygonal shapes, or semantic labels. The process of image annotation is usually performed by humans, who manually add the relevant information to the image, although there are also some automated methods for image annotation. In order to train a model that can accurately segment desired bone segments as output, we labeled the four bone segments individually on an X-ray. The labeling was done using CVAT, which is a free online tool designed for annotating images and videos for computer vision purposes. The labeling process was performed on both training and validation sets. Following the annotation, we constructed four sets of labels, each consisting of 200 labeled X-rays. These labeled datasets can now be used to train and validate the model to ensure precise bone segmentation. Figure 4.3 shows four labels or ROI of an X-ray.

4.3 Data Augmentation

Data augmentation is a widely used technique that involves the creation of new data points from the existing dataset. It achieves this by applying a range of methods to generate additional samples, which ultimately enhance the performance of the model and improve the generalization of learned features to new and unseen data. These small modifications can create new data points similar to the original data but with slight variations that allow for a more robust training process and better generalization to new data. By increasing the size and diversity of the dataset, data augmentation can help to prevent the overfitting of the model. We have applied the following transformation to the training set.

4.3.1 Contrast Adjustment

The adjust contrast transform function in PyTorch is used to adjust the contrast of an image. It takes a PIL-Python Imaging Library image as input and returns the adjusted image. The

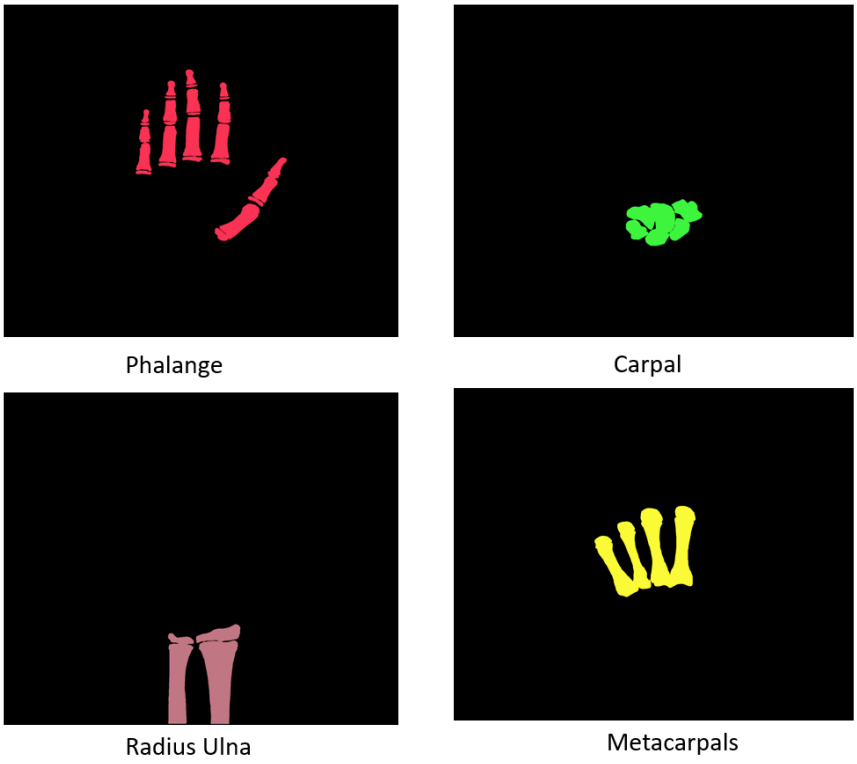


Figure 4.3: Hand Radiograph labels. These labels were painted on the original image using CVAT software. The labels were drawn for 160 training images and 40 validation sets.

function works by multiplying the pixel values of the image by a **contrast factor alpha**.

$$g(x) = \alpha f(x) + \beta \tag{4.1}$$

The contrast factor is a scalar value that is greater than 1.0 to increase the contrast of the image or less than 1.0 to decrease the contrast of the image. If the contrast factor is 1.0, the function returns the original image with no changes.

The alpha values-0.1, 0.2, 0.4 for the image processing operations were chosen by considering different types of image quality in the training dataset. The aim was to ensure that the image features such as edges, textures, and contours were not destroyed while applying the processing operations. The use of appropriate alpha values ensures that the image processing operations do not result in a significant loss of image details, which may affect the performance of the model.

4.3.2 Gaussian Blur

The GaussianBlur function in PyTorch performs a convolution operation on an input image with a 2D Gaussian kernel. The Gaussian kernel is a 2D array of numbers that represent the values of a Gaussian function, which is a bell-shaped curve that describes the distribution of values around a central point. The operation of the Gaussian blur function can be broken down into the following steps:

- The Gaussian kernel is generated based on the standard deviation and size specified by the user. The size and standard deviation determine the extent of blurring that will be applied to the image.
- The kernel is convolved with the input image by sliding it over the image and computing the weighted sum of pixel values within the kernel window at each position.
- The resulting convolved image is returned as the output of the function.

We chose kernels of size 1, 3, 5, 7, and 9, and sigma value was chosen randomly between 0.1 and 2, which is the default range set in the GaussianBlur function of PyTorch.

4.3.3 Rotation

The rotation transforms function in PyTorch applies a rotation to an image by a specified angle. Specifically, it rotates the image counter-clockwise around its center. When you apply the rotation, transform function to an image in PyTorch, the pixels of the image are rotated by the specified angle. This means that each pixel in the original image is moved to a new location in the rotated image. The result of the rotation transform depends on the angle of rotation, as well as the size and shape of the image. If the image is square, the center of the image will be the midpoint of the image's width and height. If the image is rectangular, the center of rotation will be located at the midpoint of the shorter dimension. In either case, the size and shape of the image will affect how the image appears after it has been rotated. The angle of rotation used for augmentation are 30, 60, and 90 degrees. Random rotations during image augmentation can help to simulate real-world scenarios where the orientation of the object in the image may not be consistent. This can help to make the model more robust to variations in the orientation of objects in the image and improve its overall accuracy.

4.3.4 Horizontal Flip

Horizontal flip transform flips the image along its vertical axis. Pytorch horizontal flip transform function takes a probability value as input, which determines the probability that the image will be flipped horizontally. We have applied horizontal transform with a probability of 0.5.

4.3.5 Vertical Flip

Vertical flip transform flips the image along its horizontal axis. Pytorch vertical flip transform function takes a probability value as input, which determines the probability that the image will be flipped vertically. We have applied vertical transform with a probability of 0.5.

The image augmentations mentioned above were first manually applied to the images before training, with the parameters' values selected through multiple trials and error attempts. In total, 13 augmentations were applied to the training images. As a result of the

augmentation process, the size of the training dataset increased from 160 to 2240 images. Figure 4.4 shows the augmented images.

4.4 Model Architecture

We have used state-of-art UNET architecture for automating the image segmentation process. The UNET architecture is a widely used deep learning model particularly useful for image segmentation tasks. It consists of two main parts - an encoder and a decoder network which are linked by a bottleneck layer. The encoder network comprises multiple convolutional and pooling layers that downsample the input image. This downsampling is useful in extracting low-level features of the image. On the other hand, the decoder network contains up-convolutional and concatenation layers that upsample the image to its original size. This process helps the model capture high-level features of the image.

One of the key advantages of the UNET architecture is the use of skip connections that connect corresponding layers between the encoder and decoder networks. These skip connections help the model preserve spatial information during the upsampling process, improving its accuracy in segmenting objects in images.

The number of convolutional layers in the UNET architecture is determined by the complexity of the input image, the level of detail required in the output segmentation, and the computational resources available for training and inference. It is important to strike a balance between the number of convolutional layers and the model's overall performance in terms of accuracy, speed, and generalization ability.

The UNet used for training has few modifications from the original Unet. First, the convolution layer in both the encoder and decoder have a padding of one. The padding ensures that the output after applying the convolution operation will have the same size as the input. Second, every convolution is followed by the Batch Normalization Layer. We have trained four separate UNet models for the segmentation of phalanges, carpals, metacarpals, and radius ulna.

(a) Original image

(b) contrast adjusted

(c) Gaussian Blurred

(d) Rotated by 30 degree

(e) Horizotal Flip

(f) Vertical Flip

Figure 4.4: Figure represents the different augmentations applied to training images. These augmentations were applied to each image and augmented images were saved in a directory. These augmented images were then used for the model training.

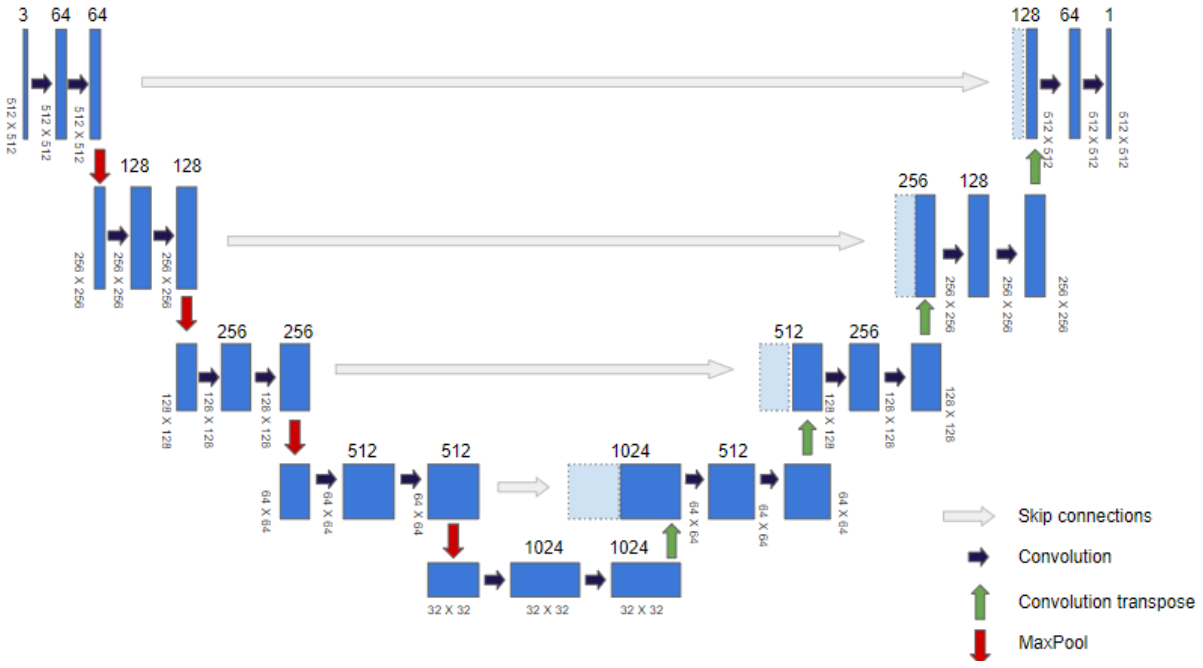


Figure 4.5: The figure depicts the architecture of the UNet model, where a blue block represents an output feature map of a convolution layer. The model includes several different types of layers, with the blue arrow indicating a (3×3) convolution layer, the green arrow indicating a (2×2) convolution transpose layer, and the red arrow indicating a (2×2) Maxpool layer which reduces the spatial resolution of feature maps. Additionally, the figure includes gray arrows representing skip connections that concatenate feature maps from the encoder with corresponding feature maps from the decoder.

4.5 Hyperparameters

In machine learning, hyperparameters are parameters that are set prior to training a model, and are not learned during the training process. In the context of CNNs, hyperparameters include the number of layers in the network, the size of the filters used in each layer, the learning rate used during training, the batch size, and the number of epochs. Hyperparameters can significantly impact the performance of the CNN, so it is important to choose the right values for each one to achieve the best results. This process of choosing the right hyperparameters is done through trial and error, by testing different values and observing the impact on the model's performance. Following multiple trials and adjustments, we settled on an input image size of 512 $\times$ 512, variable batch size for different segments, and trained our model for 300 epochs. For weight updation, we utilized the ADAM optimizer with a learning rate of 1e-5, while Binary Cross Entropy Loss was employed as the loss function during training.

4.6 Results and Discussion

To segment four hand bone segments, we trained four separate models using different configurations of UNet architecture. We experimented with reducing the convolution layers, using batch normalization, padding the image for segmentation, and evaluated the model's performance using dice score and loss on validation data. We selected the best-performing model based on the combination of reduced validation loss and high dice score and used it to segment all four segments. The chosen model and hyperparameters were optimized to achieve the best results. Table 4.1 displays the performance of the various trained UNet models, with the best-performing models highlighted. The "SHORT UNET" architecture refers to a version of UNet with three layers in the encoder and decoder. Each layer in the encoder has two convolution layers followed by one downsampling layer, while each layer in the decoder has two convolution layers followed by one upsampling layer. The "Original UNet" architecture contains four layers, while the "SHORT UNET" architecture only has three layers.

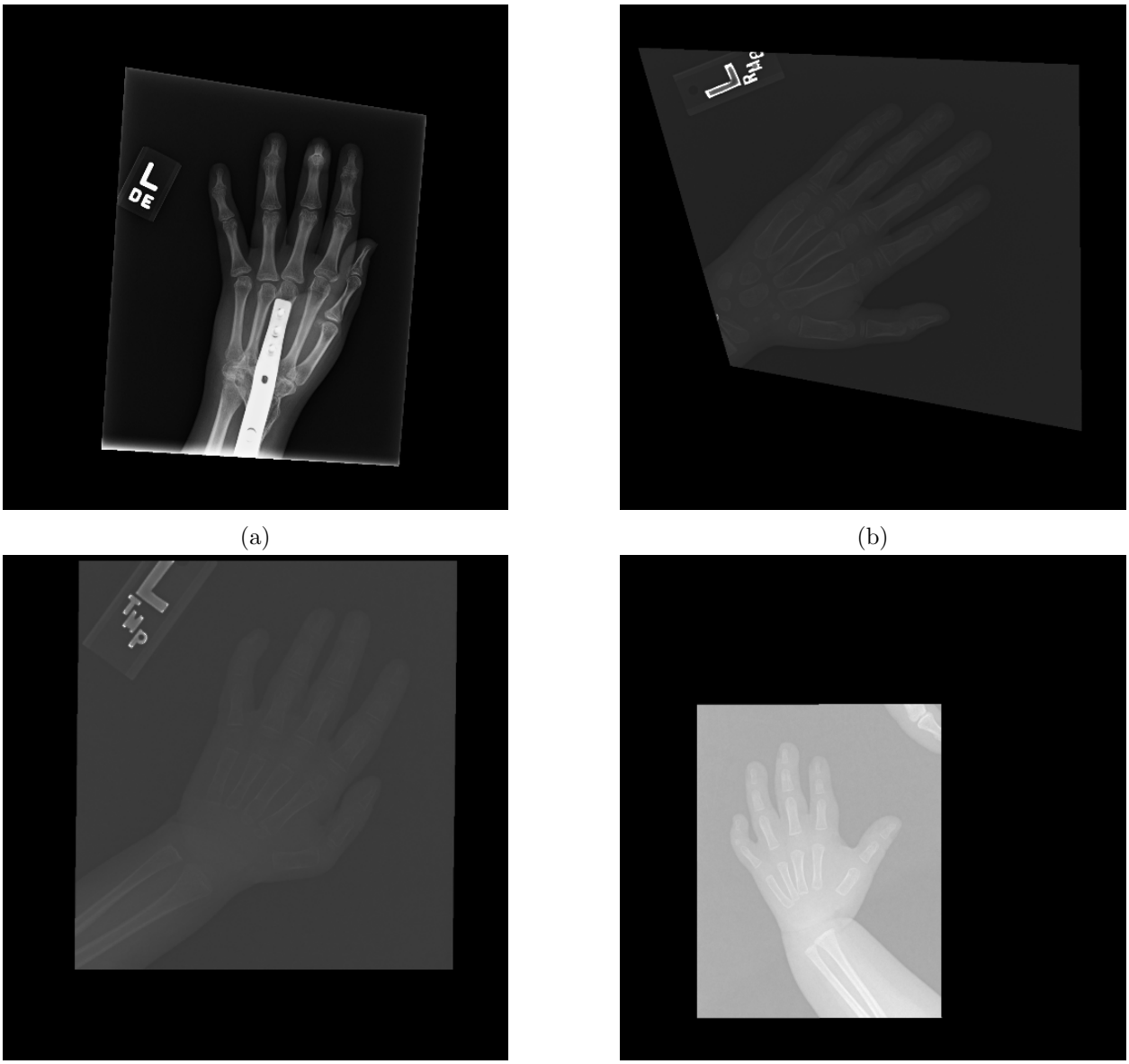
Figure 4.7 shows the training and validation dice score and per epoch loss plots for models trained for the segmentation of all four segments.

Architecture	Epochs	Training Dice Score	Validation Dice Score	Training Loss	Validation Loss
UNET, BN, p=1	300	0.99	0.87	0.035	0.069
SHORT UNET, BN	120	0.97	0.86	0.023	0.056
UNET, p=1, no BN	120	0.94	0.86	0.006	0.021
UNET, p=0, no BN	120	0.86	0.80	0.015	0.016

Table 4.1: Training and validation loss and dice score results on different UNet architecture. UNET, with batch normalization and padding of one, gave the highest dice score on validation images. BN-Batch Normalization, p-padding

When we examine the per-epoch loss for all four hand bone segments, we can observed that the training and validation loss values remain constant after 100 epochs, except for the metacarpal bone in Figure 4.7-b, which saturates after 180 epochs. However, we still trained the models for a greater number of epochs because the visualization of predictions on different validation images showed improvement with increasing training epochs. Therefore, we continued training the model for 300 epochs. We got validation dice scores of 0.90, 0.91, 0.93, and 0.94 for phalanges, carpals, metacarpals, and radius ulna, respectively.

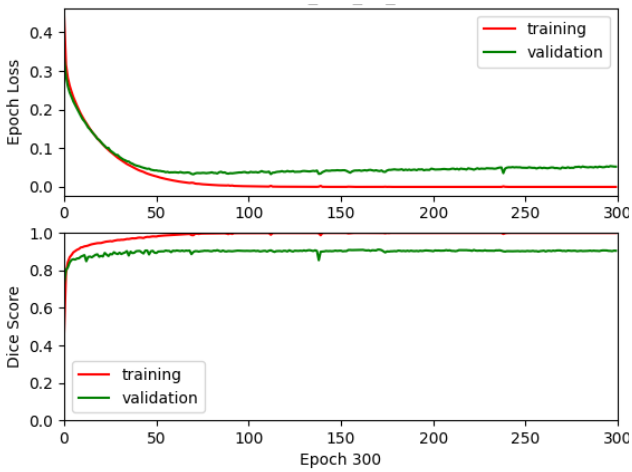
Using the trained segmentation models, we labeled all the training and validation images for four segments. The models successfully segmented almost all 12611 training and 1425 validation images, but there were some instances of low image quality, like low contrast, and a white patch in between the image, that caused difficulties in accurate segmentation. Examples of such images can be seen in Figure 4.6. Despite these challenges, the majority of the images were successfully segmented using the trained models. We overlaid predicted masks onto the corresponding input images to visualize the accuracy of the predicted masks generated by segmentation models. An example of this visualization can be seen in Figure 4.8. The predicted masks are highlighted in pink for easy identification. The results of the segmentation model on training and validation are further utilized for bone aging.



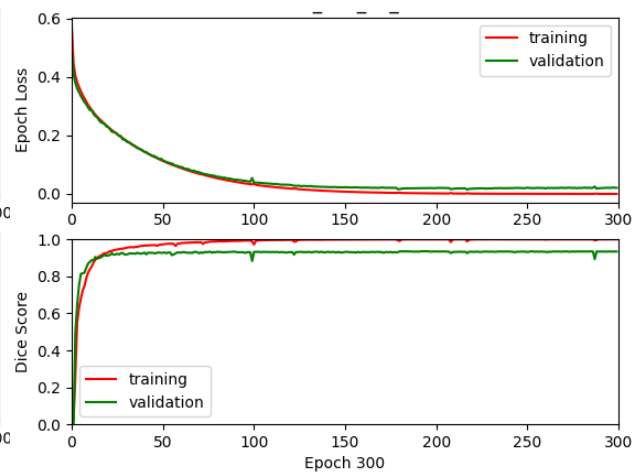
(a) (b)

(c) (d)

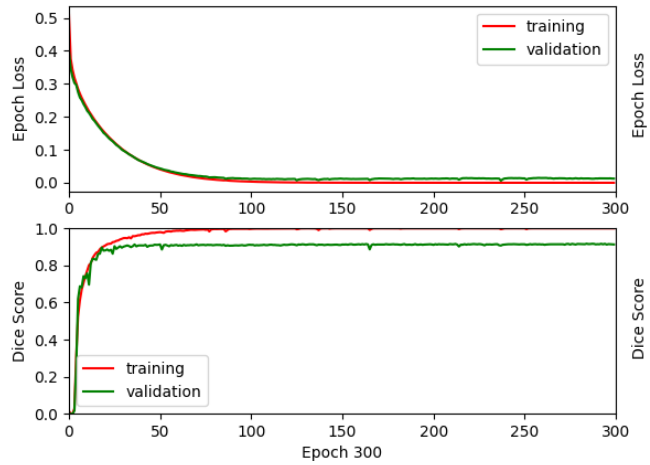
Figure 4.6: Examples of the low-quality images from the RSNA training dataset. Image (a) contains a white patch of some element captured during imaging. Image (b), (c), and (d) has very low contrast between bones and the background. These types of images cause difficulty in accurate segmentation.



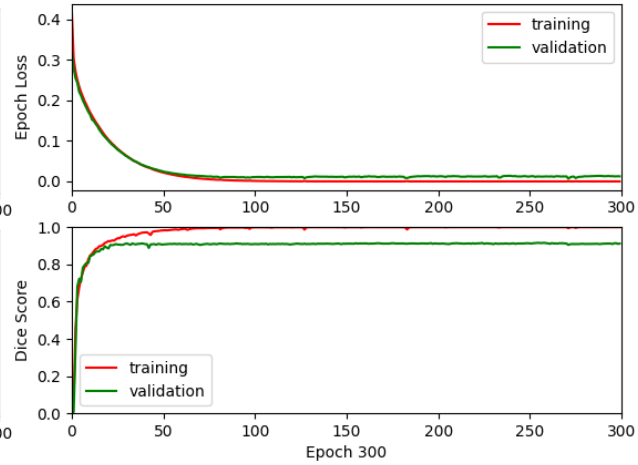
(a) Training and validation per epoch loss and dice score of a model trained for phalange segment.



(b) Training and validation per epoch loss and dice score of a model trained for the metacarpal segment.



(c) Training and validation per epoch loss and dice score of a model trained for the carpal segment.



(d) Training and validation per epoch loss and dice score of a model trained for radius ulna segment.

Figure 4.7: Epoch loss and dice score plot of training and validation of the best-performing model. We have achieved a validation dice score of more than 0.90 for all four segments.

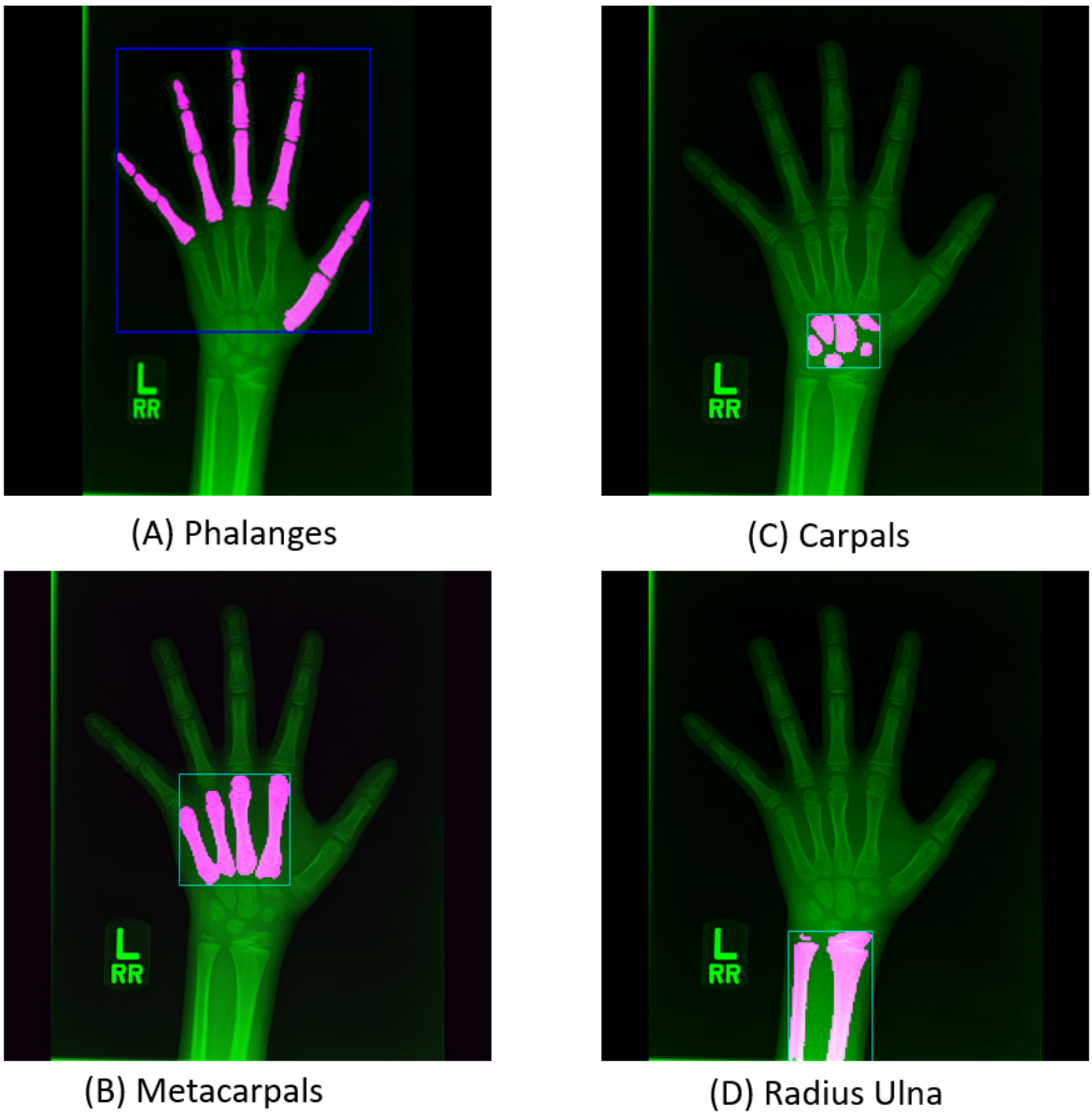


Figure 4.8: Predicted output masks (in pink) of four segmentation models overlayed on the input image to visualize the accuracy of predicted masks.

Chapter 5

Bone Aging

In the GP method of bone aging, X-rays of the left hand and wrist are considered for assessment, and the radiographs are compared to a standard atlas of bone development. The atlas contains a series of X-ray images of the left hand and wrist bones of children of known ages, with each image corresponding to a specific age.

The radiologist or physician compares the X-ray of the child's hand and wrist bones to the reference x-ray in the atlas and selects the image that best matches the appearance of the child's bones. Based on this comparison, the radiologist or physician estimates the child's bone age.

The Greulich-Pyle method is widely used to estimate bone age because it is noninvasive and inexpensive. However, it should be noted that the method is not foolproof and can be affected by factors such as ethnicity, nutrition, and hormonal disorders. In different ethnic populations, there may be differences in the maturation of hand bones. For instance, some hand bones may mature earlier in one population, while the same bones may mature later in another population. As a result, the features of the hand bones on X-ray can differ between populations.

To address this issue, we calculate the bone age of each hand bone segment independently and then calculate the final age of full-hand by two methods, first by the weighted sum of the three segments and second by the average prediction of the three segments. This process is carried out separately for males and females. This approach takes into account the differences

in bone maturation between different ethnic populations, which can lead to a more accurate estimation of bone age. By segmenting the hand bones, we can also identify any specific bones that may be maturing faster or slower than others, which can provide additional insight into a child's growth and development.

5.1 Data Preparation

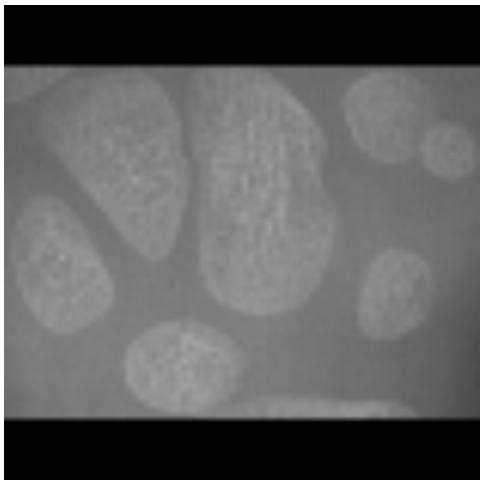
Segmenting hand bone images for age detection is a critical process that involves cropping the required bone segments from a full-hand radiograph. While this process can be done manually, it is challenging and time-consuming, particularly when dealing with a large dataset of images.

To obtain three bone segments (short bones, carpals, and wrist) from a full-hand radiograph, we followed the steps outlined below:

- First, masks for each segment were obtained using the trained segmentation models.
- Second, calculated the coordinates of each bone segment that needed to be cropped from the full-hand radiograph using the OpenCV **findContour()**function by giving the corresponding segmented mask as input.
- Finally, we cropped the full-hand radiograph into the three bone segments (short bones, carpals, and wrist) using the coordinates obtained from the previous step.
- To obtain the short bones segment, we subtracted the carpal segment from the phalange segment.

This automated approach of cropping full-hand images enabled us to save time and improve accuracy in the age detection process. It reduced the need for manual intervention and minimized the potential for errors associated with manual cropping. Figure 5.1 shows the crops of an image obtained using the automated process.

(a) Original image



(b) carpals

(c) short bones: phalanges and metacarpals

(d) Wrist/ Radius Ulna

Figure 5.1: figure (a) depicts a full-hand radiograph, from which we have extracted three cropped segments represented by (b), (c), and (d). We obtained these segments by first utilizing segmented masks of the original image to calculate the bounding box coordinates and then using these coordinates to crop out the desired segments.

We used these crops to train regression models for age detection. Crops of the image are given the same age as the original image. We trained four regression models separately for boys and girls to detect the age of full-hand, short bones, carpals, and wrist.

5.2 Model Architecture

Bone age is a continuous variable that varies across a range of values. As such, linear regression is an appropriate statistical technique for predicting bone age from hand radiograph images. In our study, we trained a linear regression model to predict bone age from hand radiograph images and their crops.

In order to learn the relationship between skeletal features derived from hand radiographs and bone age, we utilized PyTorch's DenseNet161 architecture with pre-trained weights (IMAGENET1K V1). We followed the recommended image preprocessing techniques for DenseNet161 [29]. This deep learning model is specifically designed to process and analyze images and is capable of learning complex representations of image features.

By training DenseNet 161 on a dataset of hand radiograph images and corresponding bone age labels, we aimed to create a powerful predictive model that could accurately estimate bone age from new radiograph images. During training, the model learns to adjust its weights and biases in order to minimize the difference between its predicted bone age values and the true bone age labels in the training dataset. We utilized the pre-trained DenseNet-161 model in PyTorch for training a regression model. We modified the output of the final Linear layer to be a single value, as our goal is to predict age, which is a numerical value. Figure 5.2 shows the summary of the DenseNet-161 architecture. Here are the details of the DenseNet-161 architecture:

- Input layer: The network takes a 256 $\times$ 256 $\times$ 3 RGB image as input.
- Convolutional layers: The first layer of the network is a convolutional layer with 96 filters and a 7$\times$7 kernel. This is followed by a batch normalization layer, a ReLU activation layer, and a max pooling layer with a 3$\times$3 kernel and a stride of 2.
- Dense blocks: The network consists of four dense blocks. Each dense block consists of multiple convolutional layers with a 1x1 kernel followed by a 3x3 kernel. Each convolutional layer within a dense block is followed by batch normalization and ReLU activation.
- Transition layers: Between each pair of dense blocks, there is a transition layer that consists of a 1x1 convolutional layer, a batch normalization layer, and a 2x2 average

Layers	**Filters (**Number/size)	**Input size**	**Output size**
Input layer		256 x 256 x 3	256 x 256 x 3
Conv layer (stride 2)	96/ (7 x 7)	256 x 256 x 3	128 x 128 x 96
Max pool (stride 2)	96/ (2 x 2)	128 x 128 x 96	64 x 64 x 96
Dense block 1	6 / (1 x 1) (3 x 3)	64 x 64 x 96	64 x 64 x 384
Transition layer 1	1/ (1 x 1) (2 x 2)	64 x 64 x 384	32 x 32 x 192
Dense block 2	12 / (1 x 1) (3 x 3)	32 x 32 x 192	32 x 32 x 768
Transition layer 2	1/ (1 x 1) (2 x 2)	32 x 32 x 768	16 x 16 x 384
Dense block 3	36 / (1 x 1) (3 x 3)	16 x 16 x 384	16 x 16 x 2112
Transition layer 3	1/ (1 x 1) (2 x 2)	16 x 16 x 2112	8 x 8 x 1056
Dense block 4	24/ (1 x 1) (3 x 3)	8 x 8 x 1056	8 x 8 x 2208
Global Average Pooling	2208/ 8 x 8	8 x 8 x 2208	1 x 1 x 2208
Linear layer		1 x 1 x 2208	1

Figure 5.2: DenseNet-161 Model Summary

pooling layer with a stride of 2. The 1x1 convolutional layer reduces the number of channels in the feature maps.

- Classification layer: The last dense block is followed by a global average pooling layer, which reduces the dimensions of the feature maps to 1x1. This is followed by a fully connected layer with 1 output feature which gives the age of the image.

Overall, DenseNet-161 has 49 million parameters and achieves state-of-the-art performance on several benchmark image classification tasks, such as ImageNet.

5.3 Hyperparameters

We trained the regression models for each segment for boys and girls separately. Therefore, for each gender, we trained four regression models. After conducting multiple trials and making adjustments, we determined that the input image size would be 256 x 256, and a batch size of 32 should be used during training. Each segment was trained for a specific number of epochs. During training, we employed the ADAM optimizer with a learning rate of 1e-5 to update the weights and used MSE as the training loss function, while MAD was used for calculating validation loss.

Models were trained on multiple GPUs. For this, we used PyTorch Distributed Data Parallelism (DDP)[30]. DDP is a technique used in distributed computing to train deep learning models. It works by replicating the model on multiple devices and dividing data into smaller parts, and distributing them across multiple devices or computers, such as GPUs or nodes in a cluster. Each device or computer trains the model on its portion of the data and the resulting gradients are combined to update the model parameters.
The basic workflow of DDP can be summarized in the following steps:

- Data parallelism: The data is divided into smaller batches, which are distributed across multiple devices or computers.

- Model parallelism: The model is replicated on multiple devices.

- Forward pass: Each device or computer performs a forward pass through the model, using its portion of the data.
- Backward pass: The gradients for each model are computed separately on each device.
- Gradient aggregation: The gradients from each device are combined, either by averaging them or using a more complex algorithm like the ring all-reduce algorithm.
- Parameter update: The combined gradients are used to update the model parameters and the process repeats for the next batch of data.

By using DDP, we were able to take advantage of parallel processing to train a CNN model, DenseNet161, on a large dataset of 12,611 images. This approach allowed us to effectively handle the significant computational burden of a model with 49 million parameters, resulting in a notable reduction in training time.

5.4 Results

Segment	weights	
	Boys	Girls
Short bones	0.57	0.69
Wrist	0.26	0.16
Carpal	0.18	0.17

Table 5.1: Table shows the estimated weights for each segment in girls and boys. These weights were estimated by multivariate linear regression of ground truth against individual segment prediction.

5.4.1 Independent Segment Rating

We validated the bone age model on RSNA validation data in the bone age range up to 19 years in boys and 18 years in girls. Figures 5.3 and 5.4 shows the comparison between the ground truth and the predictions made using the full-hand and three-bone segments in boys and girls, respectively. The mean value of the differences is close to zero in all plots, indicating good agreement between the predictions and ground truth, giving the lowest MAD

of 7.2 using the full-hand in boys and MAD of 7.7 using short bones in girls. The MAD of all the segments is reported in Table 5.2.

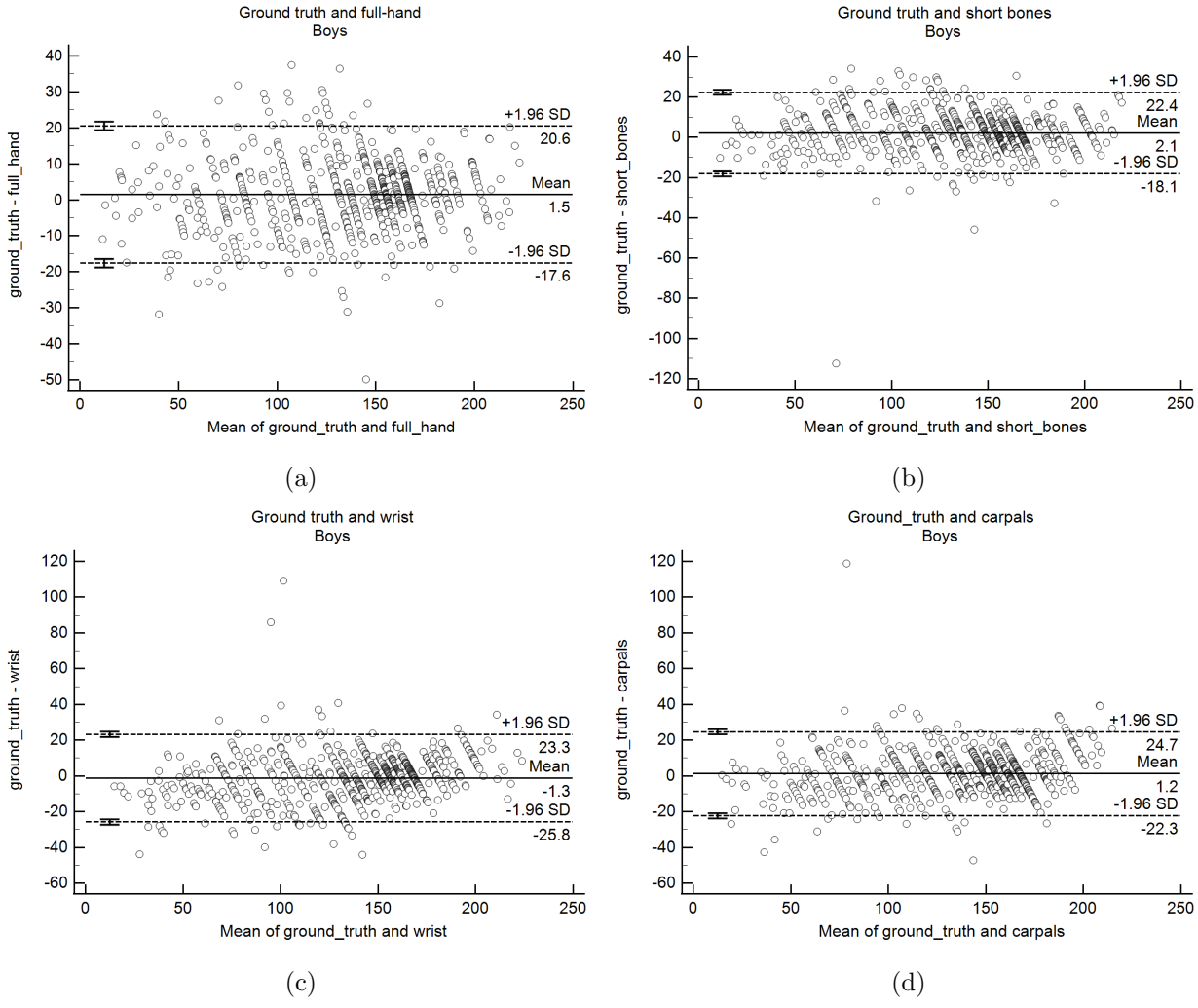


Figure 5.3: A Bland-Altman plot visualizes the agreement between the ground truth and (a)full-hand radiograph, as well as the ground truth and three bone segments: (b) short bones, (c)wrist, and (d) carpals in boys. The x-axis represents the average of the two methods being compared, while the y-axis shows the difference between the two methods.

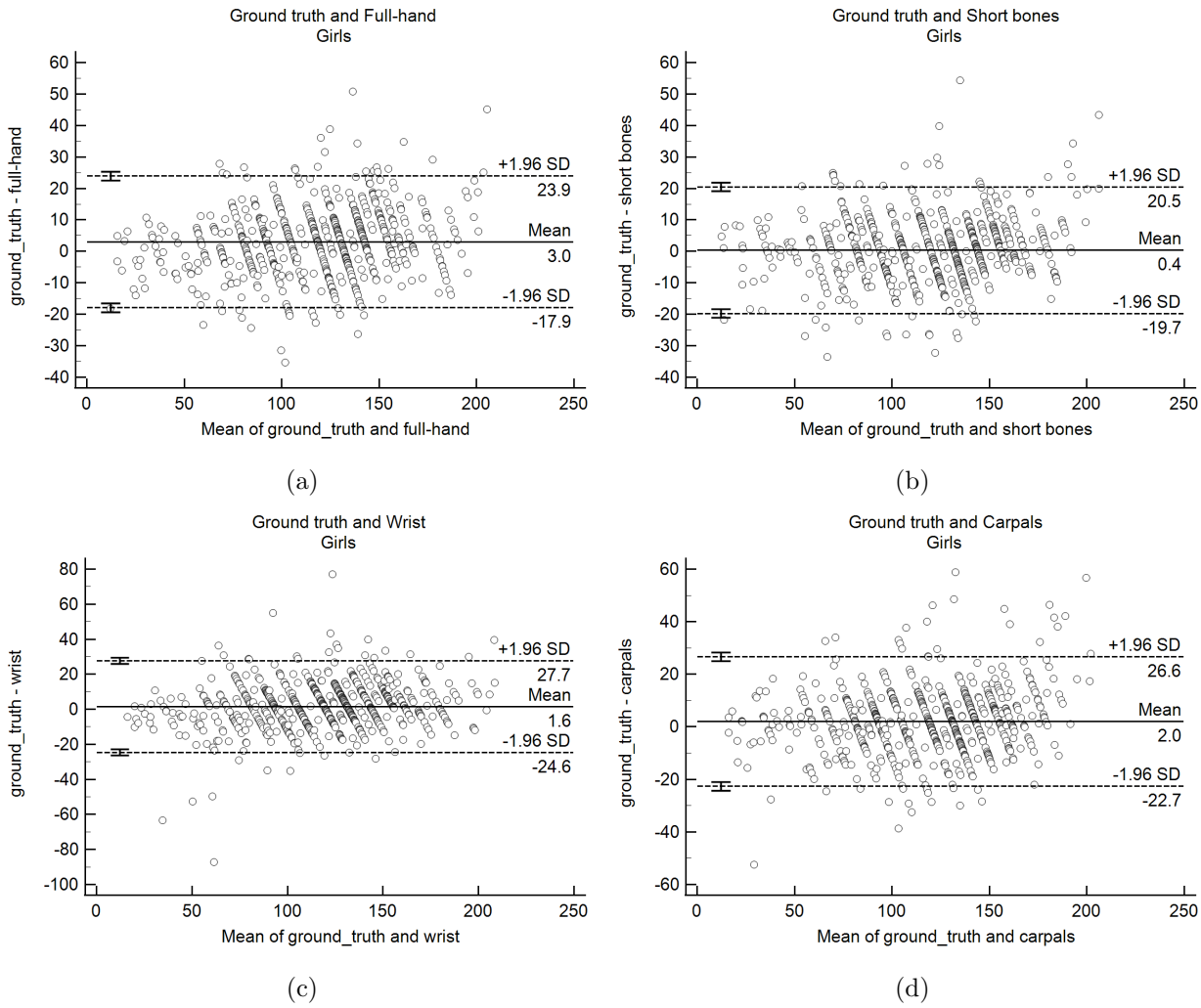
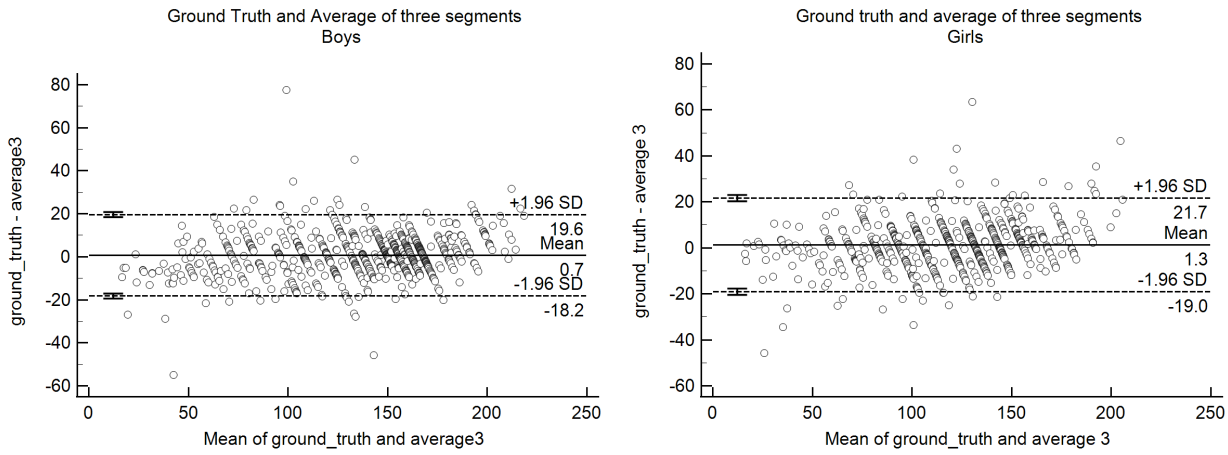


Figure 5.4: A Bland-Altman plot visualizes the agreement between the ground truth and (a)full-hand radiograph, as well as the ground truth and three bone segments: (b) short bones, (c)wrist, and (d) carpals in girls.

5.4.2 Ensemble of Carpals, Wrist, and Short Bones Rating

Figure 5.5 compares the ground truth and the average prediction of individual bone segments in boys and girls. The results show that the mean difference between the two methods is smaller when compared to the independent segment rating for both genders. By combining the segments and taking the average, the MAD is reduced to 6.9 in boys and 7.4 in girls, as compared to the MAD of the individual segments. The MAD values for the individual segments are reported in Table 5.2.

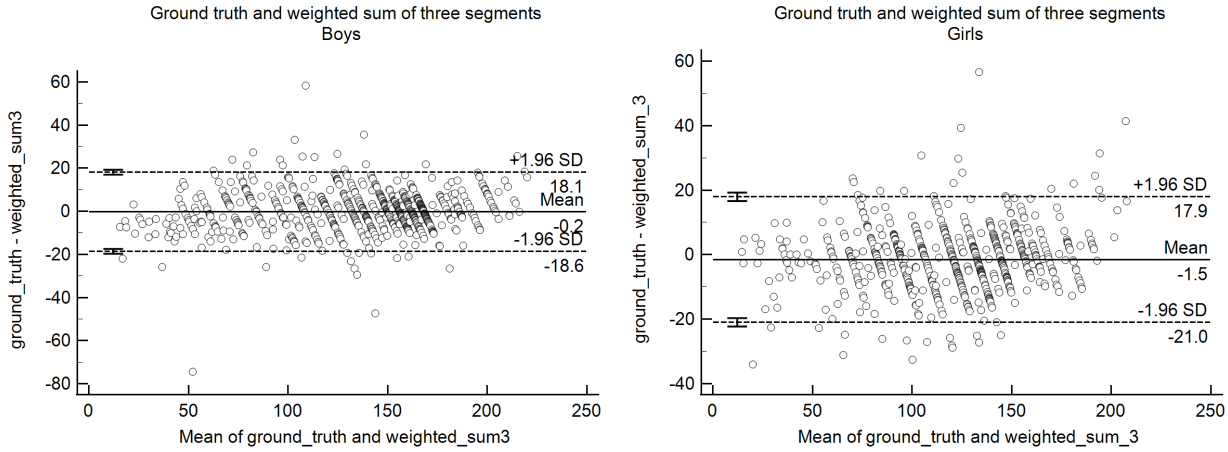


(a) (b)

Figure 5.5: A Bland-Altman plot visualizes the agreement between ground truth and the average predicted age for short bones, carpals, and wrist segments in boys (a) and girls (b).

5.4.3 Weighted Ensemble of Carpals, Wrist, and Short Bones Rating

Figure 5.6 (a), (b) compares the ground truth and the weighted average of individual bone segments in boys and girls, respectively. Weights were calculated using multivariate linear regression of ground truth against the prediction of each segment. Table 5.1 lists the estimated weights for each segment in girls and boys. The weighted average of individual segments has resulted in a further reduction of the MAD to 6.7 in boys and 7.4 in girls, compared to the MAD obtained by simply averaging the predictions of the individual segments.



(a) (b)

Figure 5.6: A Bland-Altman plot visualizes the agreement between ground truth and the weighted average of predicted age for short bones, carpals, and wrist segments in boys (a) and girls(b).

Segment	MAD (months)	
	Boys	Girls
Full-Hand	7.2	8.4
Short Bones	7.4	7.7
Wrist	8.8	9.7
Carpals	8.6	9.3
Mean (short bones + carpals + wrist)	6.9	7.6
Weighted sum (short bones + carpals + wrist)	6.7	7.4

Table 5.2: Mean Absolute Distance in months between ground truth and bone segment age prediction.

5.5 Discussion

After evaluating our bone age model on RSNA validation data, we found that a weighted ensemble approach resulted in a reduced MAD compared to other methods, such as full-hand, individual segments, and an average of individual segments. Table 5.1 lists the weights for each segment in girls and boys.

The weights provide insights into the relative contribution of each segment to bone age prediction. For both genders, short bones are the most important segment, but the wrist plays a more significant role in bone maturation in boys compared to girls. This suggests differences in bone growth patterns between boys and girls. These findings highlight the importance of considering the contributions of different segments when predicting bone age and the need to account for gender differences in bone growth patterns.

5.6 Conclusion

Our results demonstrate that using segmented hand bone images and combining the predictions from different segments can improve the accuracy of bone age predictions in both girls and boys. We also found that weighing the predictions from each segment can provide insights into the relative contribution of each segment to bone age prediction. These findings have implications for clinical practice, where accurate bone age prediction is essential for assessing growth and development in children and adolescents. Our approach can streamline the process of bone age prediction and help clinicians make more informed decisions about patient care.

Bibliography

[1] D. D. Martin, J. M. Wit, Z. Hochberg, L. Sävendahl, R. R. Van Rijn, O. Fricke, N. Cameron, J. Caliebe, T. Hertel, D. Kiepe *et al.*, "The use of bone age in clinical practice–part 1," *Hormone research in paediatrics*, vol. 76, no. 1, pp. 1–9, 2011.

[2] A. M. Mughal, N. Hassan, and A. Ahmed, "Bone age assessment methods: a critical review," *Pakistan journal of medical sciences*, vol. 30, no. 1, p. 211, 2014.

[3] V. Gilsanz and O. Ratib, *Hand bone age: a digital atlas of skeletal maturity*. Springer, 2005, vol. 1.

[4] M. Satoh, "Bone age: assessment methods and clinical applications," *Clinical Pediatric Endocrinology*, vol. 24, no. 4, pp. 143–152, 2015.

[5] V. R. Preedy, *Handbook of growth and growth monitoring in health and disease*. Springer Science & Business Media, 2011, vol. 1.

[6] H. H. Thodberg, "An automated method for determination of bone age," *The Journal of Clinical Endocrinology & Metabolism*, vol. 94, no. 7, pp. 2239–2244, 2009.

[7] D. J. Michael and A. C. Nelson, "Handx: a model-based system for automatic segmentation of bones from digital hand radiographs," *IEEE transactions on medical imaging*, vol. 8, no. 1, pp. 64–69, 1989.

[8] E. Pietka, M. F. McNitt-Gray, M. Kuo, and H. Huang, "Computer-assisted phalangeal analysis in skeletal age assessment," *IEEE transactions on medical imaging*, vol. 10, no. 4, pp. 616–620, 1991.

[9] J. M. Tanner and R. D. Gibbons, "A computerized image analysis system for estimating tanner-whitehouse 2 bone age," *Hormone Research in Paediatrics*, vol. 42, no. 6, pp. 282–287, 1994.

[10] G. W. Gross, J. M. Boone, and D. M. Bishop, "Pediatric skeletal age: determination with neural networks." *Radiology*, vol. 195, no. 3, pp. 689–695, 1995.

[11] M. Mansourvar, R. G. Raj, M. A. Ismail, S. A. Kareem, S. Shanmugam, S. Wahid, R. Mahmud, R. H. Abdullah, F. H. F. Nasaruddin, and N. Idris, "Automated web based system for bone age assessment using histogram technique," *Malaysian Journal of Computer Science*, vol. 25, no. 3, pp. 107–121, 2012.

[12] H. H. Thodberg, R. R. van Rijn, O. G. Jenni, and D. D. Martin, "Automated determination of bone age from hand x-rays at the end of puberty and its applicability for age estimation," *International journal of legal medicine*, vol. 131, pp. 771–780, 2017.

[13] S. S. Halabi, L. M. Prevedello, J. Kalpathy-Cramer, A. B. Mamonov, A. Bilbily, M. Cicero, I. Pan, L. A. Pereira, R. T. Sousa, N. Abdala *et al.*, "The rsna pediatric bone age machine learning challenge," *Radiology*, vol. 290, no. 2, pp. 498–503, 2019.

[14] H. H. Thodberg, S. Kreiborg, A. Juul, and K. D. Pedersen, "The bonexpert method for automated determination of skeletal maturity," *IEEE transactions on medical imaging*, vol. 28, no. 1, pp. 52–66, 2008.

[15] J. R. Kim, W. H. Shim, H. M. Yoon, S. H. Hong, J. S. Lee, Y. A. Cho, and S. Kim, "Computerized bone age estimation using deep learning based program: evaluation of the accuracy and efficiency," *American Journal of Roentgenology*, vol. 209, no. 6, pp. 1374–1380, 2017.

[16] W. W.-i. Lea, S.-J. Hong, H.-K. Nam, W.-Y. Kang, Z.-P. Yang, and E.-J. Noh, "External validation of deep learning-based bone-age software: a preliminary study with real world data," *Scientific Reports*, vol. 12, no. 1, p. 1232, 2022.

[17] J. Zhang, F. Lin, and X. Ding, "Automatic determination of the greulich-pyle bone age as an alternative approach for chinese children with discordant bone age," *Hormone Research in Paediatrics*, vol. 86, no. 2, pp. 83–89, 2016.

[18] R. R. van Rijn, M. H. Lequin, and H. H. Thodberg, "Automatic determination of greulich and pyle bone age in healthy dutch children," *Pediatric radiology*, vol. 39, pp. 591–597, 2009.

[19] H. H. Thodberg and L. Sävendahl, "Validation and reference values of automated bone age determination for four ethnicities," *Academic radiology*, vol. 17, no. 11, pp. 1425–1432, 2010.

[20] H. H. Thodberg, "An automated method for determination of bone age," *The Journal of Clinical Endocrinology & Metabolism*, vol. 94, no. 7, pp. 2239–2244, 2009.

[21] C. Oza, A. V. Khadilkar, S. Mondkar, K. Gondhalekar, A. Ladkat, N. Shah, N. Lohiya, H. K. Prasad, P. Patil, M. Karguppikar *et al.*, "A comparison of bone age assessments using automated and manual methods in children of indian ethnicity," *Pediatric Radiology*, vol. 52, no. 11, pp. 2188–2196, 2022.

[22] K. O'Shea and R. Nash, "An introduction to convolutional neural networks," *arXiv preprint arXiv:1511.08458*, 2015.

[23] A. F. Agarap, "Deep learning using rectified linear units (relu)," *arXiv preprint arXiv:1803.08375*, 2018.

[24] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*. pmlr, 2015, pp. 448–456.

[25] S. Jadon, "A survey of loss functions for semantic segmentation," in *2020 IEEE conference on computational intelligence in bioinformatics and computational biology (CIBCB)*. IEEE, 2020, pp. 1–7.

[26] Y. Ho and S. Wookey, "The real-world-weight cross-entropy loss function: Modeling the costs of mislabeling," *IEEE access*, vol. 8, pp. 4806–4813, 2019.

[27] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv preprint arXiv:1609.04747*, 2016.

[28] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[29] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[30] S. Li, Y. Zhao, R. Varma, O. Salpekar, P. Noordhuis, T. Li, A. Paszke, J. Smith, B. Vaughan, P. Damania *et al.*, "Pytorch distributed: Experiences on accelerating data parallel training," *arXiv preprint arXiv:2006.15704*, 2020.