Bone Age Comparison Using Convolutional Neural Network

by

Fariha Nawaz
15301121
Md. Samiul Akib
15101074
Asif Imtiaz
14301086
Sakib Uddin Ahmad
14301106

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science

Department of Computer Science and Engineering
Brac University
April 2019

Declaration

It is hereby declared that

1. The thesis submitted is our own original work while completing degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

Student's Full Name & Signature:

Fariha Nawaz 15301121	Md. Samiul Akib 15101074
Asif Imtiaz 14301086	Sakib Uddin Ahmad 14301106

Approval

The thesis titled "Bone Age Comparison Using Convolutional Neural Network" submitted by

1. Fariha Nawaz (15301121)
2. Md. Samiul Akib (15101074)
3. Asif Imtiaz (14301086)
4. Sakib Uddin Ahmad (14301106)

Of Spring, 2019 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science on April 25, 2019.

Examining Committee:

Supervisor:
(Member)

Dr. Amitabha Chakrabarty
Associate Professor
Department of Computer Science and Engineering
Brac University

Program Coordinator:
(Member)

Dr. Jia Uddin
Associate Professor
Department of Computer Science and Engineering
Brac University

Head of Department:
(Chair)

Dr. Md. Abdul Mottalib
Professor
Department of Computer Science and Engineering
Brac University

Abstract

In the last few years,Machine Learning has taken the world by storm. From predictive web browsing to the email and text classification,from the autonomous car to facial recognition, machine learning is the main core of every intelligent application that we can see now a days. Predicting bone age is another field that has been benefited exceedingly from the exposure of this technology. For this reason, we have proposed convolutional neural network for predicting the age of a child and doing a comparative analysis on with other available techniques. We have choose four models for it and they are: InceptionV3, VGG16, ResNet50 and MobileNet. By pre-processing the image and selecting the various parameters the framework has been trained and tested in "RSNA Pediatric Bone Age Machine Learning Challenge" dataset. Highest accuracy of 91.13% has been achieved for MobileNet with mean absolute error of 8.87, the explained variance score for this method is 0.92 and value loss during the training is 0.0809 whereas the lowest accuracy has been achieved for VGG16 with mean absolute error 32.58,the explained variance score for this method is 0.032 and value loss during the training is 1.0281.

Keywords: Convolutional Neural Network; Pre-processing; Bone-age; InceptionV3; VGG16; ResNet50; MobileNet;Comparative analysis

Acknowledgement

First of all, we would like to thank Almighty Allah to enable us to work on this thesis which has been a great learning experience for us. By the grace of Allah, we could able to put our best efforts and successfully complete it on time. Secondly, we would like to convey our gratitude to our supervisor Dr. Amitabha Chakrabarty for his guidance and handful contribution throughout the whole phase of our thesis work and also to write this report. From the very beginning to the end of the work he has provided us with all kinds of help and inspired us to move forward to our goal. We are also very much thankful to our parents, friends, and well-wishers who have helped us directly or indirectly while conducting our research and continuing our work. We would also like to acknowledge the assistance that we received from a number of resources over the Internet especially from the work of our fellow researches. Finally, we would like to thank BRAC University for giving us the opportunity to conclude the thesis and for giving us the chance to complete our Bachelor degree.

Table of Contents

List of Figures

List of Tables

Nomenclature

The next list describes several symbols & abbreviation that will be later used within the body of the document

BAA Bone Age Assessment

CNN Convolutional Neural Network

$CNTK$ Microsoft Cognitive Toolkit

CSV Comma Separated Values

$DCNN$ Deep Convolutional Neural Network

DRU Distal Radius and Ulna

$EWMM$ Element Wise Multiplication

$FPGA$ Field Programmable Gate Array

$GEMM$ General Purpose Matrix Multiplication

GP Greulich And Pyle Method

$ILSVRC$ ImageNet Large Scale Visual Recognition Competition

LTI Linear and Time-Invariant

MAE Mean Absolute Error

NN Neural Network

$RCNN$ Regional Convolutional Neural Network

$ReLu$ Rectified Linear Units

$ResNet$ Residual Network

ROI Regions Of Interest Pooling

SF Sigmoid Function

SGD Stochastic Gradient Descent

$Tanh$ Hyperbolic Tangent Function

TW Tanner Whitehouse Method

VGG Visual Geometry Group

Chapter 1

Introduction

This chapter gives a general review of the purpose of this research work and an overview of the entire thesis.
In deep learning, a convolutional neural network is a class of deep neural networks, most commonly applied to analyzing visual imagery [30].CNN is mainly an version of deep learning . Deep learning is a sub-field of machine learning concerned with algorithms inspired by the structure and the functionality of the brain called artificial neural network [9] .CNN mainly focuses on developing a computer program that will take the image as input and by processing it to learn the features from it so if a new image is given as input it can identify it as accurately as possible. So like the other machine learning algorithms, CNN also focuses on recognizing the pattern among the image by its own. In previous years many CNN models have been invented. In this work we have tried to use some pre-trained models in bone-age x-ray image dataset to predict the bone age(in months) and tried to do a comparative analysis on the models we have used.

1.1 Motivation

Deep learning architectures like deep neural networks, deep belief networks and repeated neural networks are applied to fields together with computer vision, speech recognition, natural language process, audio recognition, social network filtering, machine translation, bioinformatics, drug design, medical image analysis, material inspection and board game programs, wherever they need created results comparable to and in some cases superior to human specialists [7], [8], [25]. In last 10 years, the implementation of CNN has been increased vastly and with the recent improvement of the processing power, the domain of CNN is rapidly increasing. Day by day the CNN models shows more improved accuracy as well as optimized use of computational resources. In this thesis we are putting a comparative analysis on the models which already exists.

1.2 Aims and Objectives

The main objectives of this thesis:

- Converting the digital images into a dataset for building training and pre-training models.

- Applying various pre-trained Convolutional Neural Network (CNN) models to predict a child's age and making a comparative analysis between various models.
- Analyzing reasons why performance varies between models having tested with the same dataset.

1.3 Thesis Overview

Chapter 1 is the introduction of our thesis work. Our motivations and objectives to do this thesis are also mentioned here as well as a short overview of the methodology that has been followed by us.

Chapter 2 consists of the literature review where we have demonstrated the background study that we have done for this thesis.

Chapter 3 consists of workflow,pre-prossesing and compilation of the pre-trained models for our thesis.

Chapter 4 consists of pretrained models and implementation of our overall thesis.

Chapter 5 contains the conclusion and future work.

Chapter 2

Literature Review

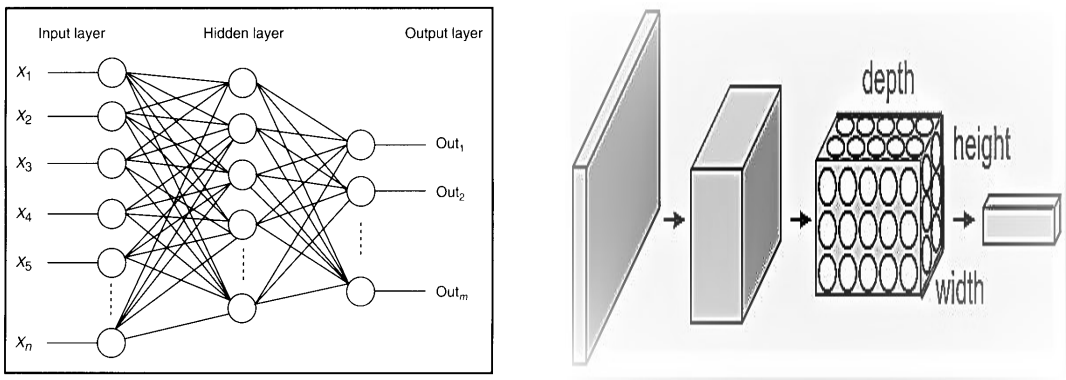
This chapter contains the background study required for this research work. Our findings include understanding about Convolutional Neural Network , Image pre-processing and compiling the models for our thesis

2.1 Convolutional Neural Network

A convolution is a mathematical operation which represent a signal passing through a LTI(Linear and Time-Invariant) system or filter. Suppose a signal s(t) is passing through a system with response h(t). the output is the convolution of s(t)with h(t). The convolution is simply the integral of the product of the two functions,where one is reversed [2].The equation is:

$$(s * h)(t) = \int_{-\infty}^{\infty} s(\tau)h(t-\tau)d\tau \tag{2.1}$$

Convolution neural networks work with this convolution theory. They are almost like ordinary Neural Networks. There are some neurons and all the neurons have learnable weights and biases. Some inputs are received by the neurons, and then a dot product is performed and optionally follows it with a non-linearity. The entire network still expresses a single differentiable score function: from the raw image pixels on one finish to class scores at the other [3] neural network receive an input



(a) Neural Network (b) Convolutional Neural Network

Figure. 2.1: NN vs CNN [3]

which is a single vector and the input is transformed by a series of hidden layers. In each hidden layers consists some neurons and each neuron is fully connected to all the neurons in previous layer and neuron functionality in a single layer is completely independent and don't share any connections. The last fully-connected layer is termed as the output layer. Regular neural network don't scale well to full images. Suppose the image size is 224×224 so the neurons will have 224×224×3=150528 weights (3 is for the color channel RGB). This full connectivity is wasteful and also the huge range of parameters would quickly cause over fitting [10]. In this case, CNN works more efficiently. CNN extract the feature of an image and convert it into lower dimension without changing or losing its characteristics. Unlike the neural network, the layers of CNN arranged in 3 dimensions: width, height and depth. For the previous example, instead of applying neural network, if we use CNN the input dimension can be reduced in 1×1 ×10 (Suppose it has 10 classes). It means only 10 neuron is needed in first layer.

2.2 How CNN Works

There are several layers of CNN [15]. They are:

1. Input Layer
2. Convolution Layer
3. ReLu(Rectified Linear Unit) Layer
4. Pool Layer
5. Fully Connected Layer
6. Output Layer

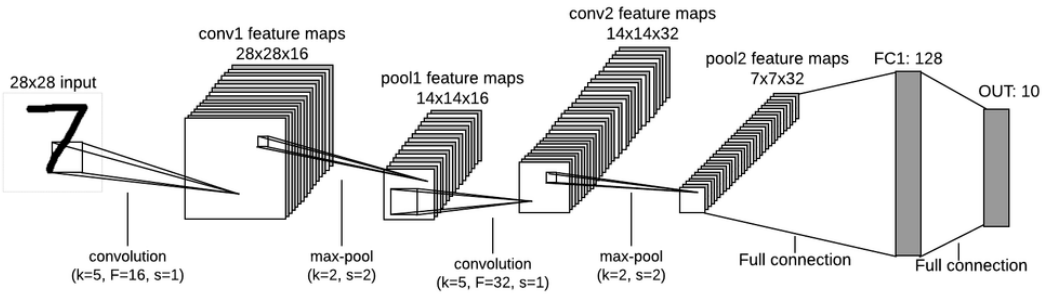


Figure. 2.2: CNN Architecture [29]

Input Layer

This layer contains the image data which is represented by three dimensional matrix. If we have image of 224×224 dimension it is needed to be converted into 50176×1. If the training sample size is m, the dimension of input will be(50176,m) for this case.

Suppose the input is X,the CNN should detect and categorised all the samples that looks like X.

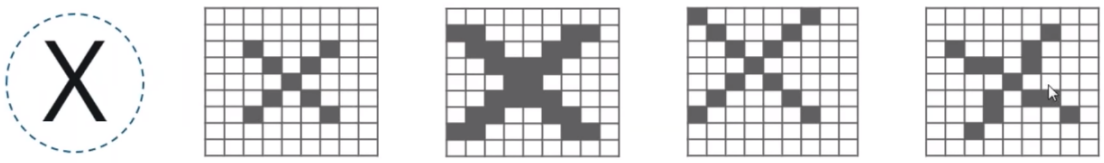


Figure. 2.3: Sample Input [36]

Convolution Layer

Convolution layer is sometimes called feature extractor layer because features of the image are get extracted within this layer. This layer compare the images pieces by pieces and the pieces what is looked for are called the features [4]. It will calculate the output neurons that are connected to local regions.By finding rough feature matches, in roughly the same position in two images, it will select one or some features from the photo and create one or multiple matrix and dot product with the image matrix and after the whole procedure finally give a result which will be the convolution layer. This layer has parameters and additional hyperparameters. If we have $N \times$N image size and $F \times$F filter size then after convolution result will be:

$$(N \times N) \times (F \times F) = (N - F + 1) \times (N - F + 1) \tag{2.2}$$

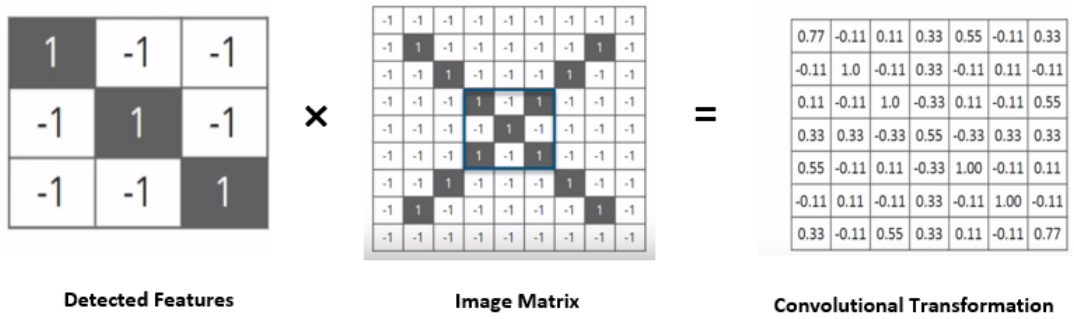


Figure. 2.4: Convolution Layer [36]

ReLu Layer

After getting the result from the convolution layer, the RELU layer will turn all the negative number into 0. If the number is positive, it will remain same. This layer don't have any parameters and additional hyperparameters. More details about it has been described in Chapter 3 Section 4.

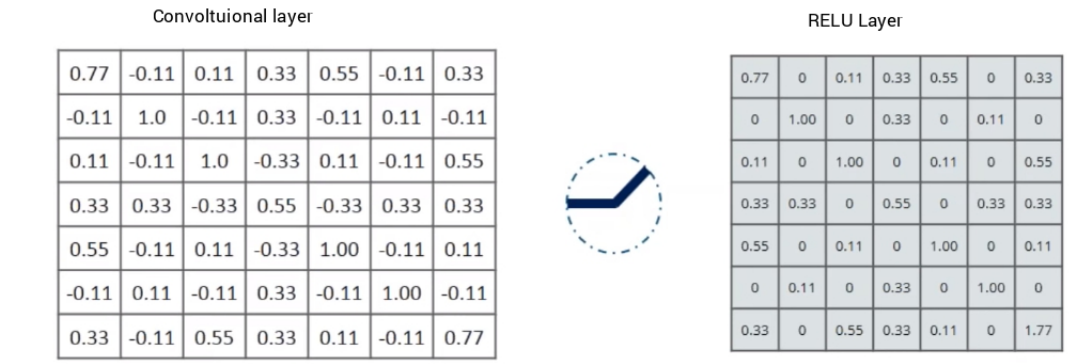


Figure. 2.5: ReLu Layer [36]

Pooling Layer

Pooling layer is used to reduce the spatial volume of input image after convolution. It do not has any parameters but it has additional hyperparameters. Two hyperparameters are: Filter(F) and Stride(S) [35].In general, if we have input dimension W1 x H1 x D1, then

$$W2 = \frac{W1 - F}{S + 1} \tag{2.3}$$

$$H2 = \frac{H1 - F}{S + 1} \tag{2.4}$$

$$D2 = D1 \tag{2.5}$$

Where W2, H2 and D2 are the width, height and depth of output.In the following picture, at first the RELU layer is converted into 4×4 matrix.

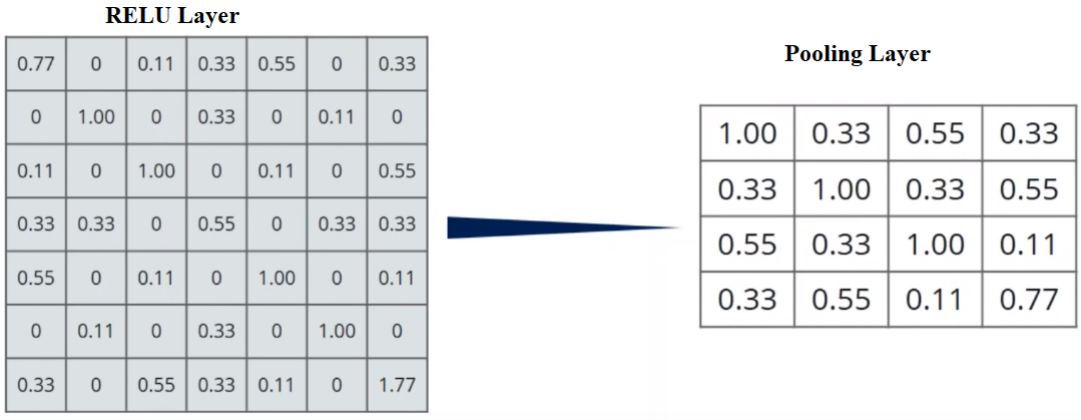


Figure. 2.6: Pool Layer [36]

The whole procedure happens multiple time and shrink the image data into a 2×2 small

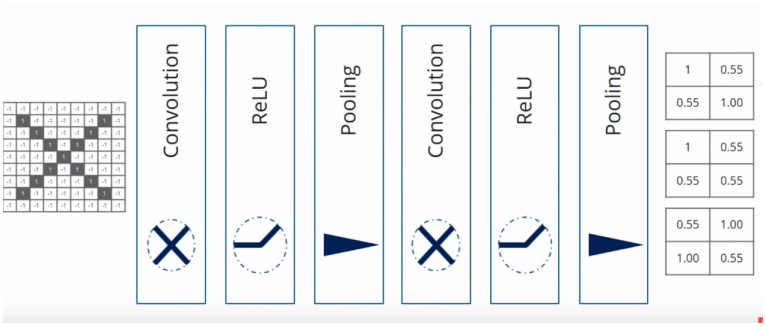


Figure. 2.7: Converted 2×2 matrix [36]

matrix with the given image with the modified values.

Fully Connected Layer

Fully Connected Layer will compute the class scores leading to the column of size 1×1×12 for the following picture as there are 3 features selected and for each feature,one matrix has been generated in pooling layer. If the number of selected features were 2, it would have generated a matrix 1×1×8 for the same image. Like convolution layer, it also has has parameters and additional hyperparameters.

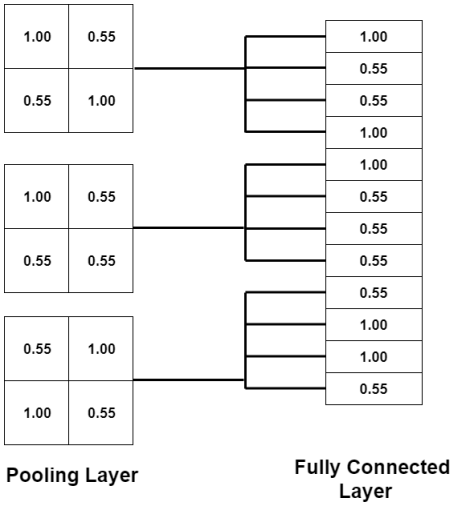


Figure. 2.8: Fully Connected Layer [36]

Output Layer

Output layer contains the label which is in the form of one-hot encoded. All the data will be saved as the X and if another new image is given it will check how much similarities it has with the saved data and then it will detect is it X or not.

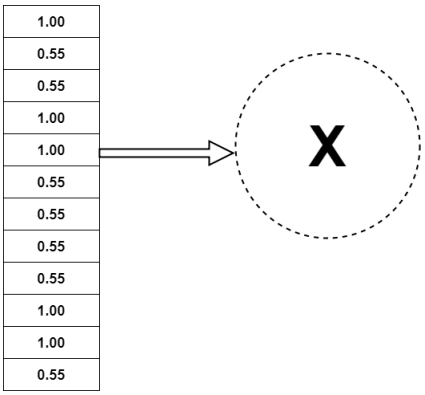


Figure. 2.9: Output Layer [36]

In this way, CNN transforms the original image from original pixel values into the final class scores.

2.3 Previous Works

In this dynamic digital world where we expect efficiency in every case it is very expected that researchers already have works to make bone age assessment more efficient . To accustom with the procedures of existing works, we examined a great deal of papers identified with our point. As bone age classification is an uprising issue in forensic science and medicine, as well as in physical anthropology for human identification purpose and biological profiling so there was no uncertainty that Scholars would come up with the possibility of digitally supported determination to classify bone age. The most widely recognized calculation in this methodology comprises of a few firm advances: image per-processing, segmentation, feature extraction, classification and evaluation.
Vladimir I. Iglovikov et al. portrayed a profound learning way to deal with the issue of bone age appraisal utilizing information from the 2017 Pediatric Bone Age Challenge composed by the Radiological Society of North America [31]. This dataset comprised of 12,600 radiological pictures. Every radiograph in the dataset was a picture of a left hand marked with bone age and sex of a patient. Their methodology presented a far reaching preprocessing convention based on the positive mining system. They used pictures of entire hands also as explicit hand parts for both preparing and expectation. This enabled them to measure the significance of explicit hand bones for mechanized bone age examination. One of the key commitments of their work was thorough preprocessing pipeline. To keep the display from taking in false relationship from artifacts in the picture, they first expelled foundation by segmenting the hand. At that point they normalized contrast and detect key points. Then they applied affine transforms to enroll fragmented pictures in a basic organize space. Other than improving the nature of information, this progression permits us to precisely recognize distinctive areas of the hand. They trained a few profound systems utilizing distinctive pieces of hand pictures to survey how extraordinary hand bones added to the models' execution crosswise over four major skeletal improvement stages. At long last, they analyzed regression and classification, sex-specific and sex agnostic models, and assess in general execution of their methodology. By utilizing diverse hand zones, they found that BAA should be possible just for carpal

bones or for metacarpals and proximal phalanges with around 10– 15% increment in error contrasted with the entire hand. Their methodology outflanks other regular techniques for bone age evaluation.
Marjan Mansourvar et al. proposed an automated approach for Bone Age Assessment based totally on combined method [11].The manual methods are prone to variability of observation, time consuming and limited to objective decisions. These were big motivations for automatic method for bone age assessment. Their method tried to overcome the problems of conducting BAA in manual methods. They used both quantitative and qualitative methods to collect data. Three distinct tests were carried out to dissect and assess the usability of bone age assessment BAA systems. The integrated BAA system that combined the use of the hand and the clavicle bones was conducted on separate set of data. The aftereffects of programming assessment showed this framework as a solid programmed technique for bone age appraisal with high exactness. The catchy part of their work was it was easy to use and it could be implemented in web environment but they did not show any accuracy percentage.
Shuqiang Wang et al. recommended an automated skeleta dulthoo popularit system. First the detected the distal radius and ulna (DRU) regions from hand and wrist X-ray images by means of a quicker area-based totally convolutional neural network model [34].A well-tuned convolutional neural community classification model was applied to estimate the bone ages. They discussed the model performance in line with various community configurations. Their Detection task was based on Caffe framework, and classification task was based on Theano, Lasagne Deep learning framework.In the training process, they used the Adam optimization algorithm and minimized the multiple classification cross-entropy loss function.They also added that a kid's skeletal development is essential for keep tracking for skeletal disorders during growth and To detect scoliosis, endocrine and metabolic disorders we need good information of bone age. Their proposed version finally completed 92% and 88% accuracy for radius and ulna, respectively.
Ewa Pietka et al. started with a preprocessing function yielding epiphyseal/ metaphyseal regions of interest (EMROIs). These regions were subjected to a feature extraction function. Extracted features describe the stage of skeletal development more objectively than visual comparison [5].Accuracy of diameters extraction had been performed by evaluating their intersection with bones (evaluation of location) and location of end points (evaluation of distance measures). Their Image analysis was applied to 200 clinical hand wrist radio-graphs selected from the normally developed population. In boys, the age was limited to 14; in girls, to 12. All of them were in the upright position. Accuracy had been measured independently at three stages of the image analysis. In different stages of analyzing the accuracy was in between 75-90%.
Liqiang Lu et al. illustrated that quick Winograd calculation can significantly diminish the arithmetic complexity and improve the execution of CNNs on FPGAs [23].First of all, they proposed a novel engineering for executing Winograd calculation on FPGAs. Their structure utilized line buffer structure to reuse the feature map data among different tiles. The computation of Winograd algorithm involved a mixed matrix transformation of general purpose matrix multiplication (GEMM) and element-wise multiplication (EWMM). They additionally successfully pipelined the Winograd PE motor and started various PEs through parallelization. Moreover, they proposed an expository model to anticipate the asset utilization and reason

about the execution. At that point, they utilized the model to manage a quick structure space investigation. Investigations utilizing the state of-the-art CNNs exhibited the best execution and vitality effectiveness on FPGAs. Here they demonstrated that fast Winograd algorithm can be used to derive efficient algorithms for CNNs with small filters. Their experiments were performed on Xilinx ZC706. And their performance prediction was very accurate. On average, the prediction error was 15.4% and 13.7% for filters 3 3 and 5 5, respectively. They achieved an average 1006.4 GOP/s for the convolutional layers and 854.6 GOP/s for the overall AlexNet and an average 3044.7 GOP/s for the convolutional layers and 2940.7 GOP/s for the overall VGG16 on XilinxZCU102 platform.

Hyunkwang Lee et al. offered a fully automated deep learning pipeline to segment an area of interest, institutionalize and preprocess input radiographs, furthermore, perform BAA [22].Their models utilized an Image Net pre trained, fine-tuned convolutional neural system (CNN) to accomplish 57.32% and 61.40% accuracies for the female and male associates on their held-out test pictures. Female test radiographs were doled out a BAA inside 1 year 90.39% and inside 2 years 98.11% of the time. Male test radiographs were doled out 94.18% inside 1 year and 99.00% inside 2 years. Utilizing the input occlusion strategy, attention maps were made which uncover what features the prepared model used to perform BAA. Moreover, their methodology exploits transfer learning with a pre-trained deep CNN to consequently extricate critical highlights from all bones on a ROI that was consequently divided by a detection CNN. Their hospital's radiology reports incorporate the patient's chronological age and the bone age with reference to the models of Greulich and Pyle. Finally, their BAA framework could be conveyed in the clinical condition by showing three to five reference pictures from the G&P chart atlas with an indication of their automated BAA for radiologists to make the last age assurance with one click, organized report generation. They have also added that their system will take approximately 10 MS to finish one BAA with the preprocessed image. Though, it takes averagely 1.71 s to crop, segment, and preprocess an image prior to classification.

Shuqiang Wang et al. presented an automated skeletal maturity recognition system based on DRU assessment system that took a solitary hand radiograph as an input and lastly yield the bone age prediction [33].Radiographic images of the DRU was detected and extracted directly by Faster R-CNN model. Their proposed framework took input pictures and produced classification results specifically. It first precisely distinguished the distal radius and ulna regions from the hand and wrist X-ray pictures by a quicker area based convolutional neural system (CNN) demonstrate. At that point, a very much tuned CNN classification display was connected to gauge the bone ages. In the trial segment, they utilized an informational collection of 1101 hand and wrist radiographs and directed far reaching investigates the proposed framework. Moreover they talked about the model execution as indicated by different network configurations, multiple optimization algorithms, and diverse preparing test sums. After parameter streamlining, the proposed model was finally accomplished 92% and 90% classification correctness's for radius and ulna grades, individually. Taking everything into account, the primary commitments of their investigation include: (1) to propose a completely automated and fast skeletal maturity estimation framework which is created dependent on profound CNNs; and (2) to improve the models' execution with data sample balance, data augmentation

and favored optimization algorithms.
Jianlong Zhou et al. examined the utilization of Deep Convolutional Neural Networks (DCNNs) for the automatic bone age appraisal [28]. As there exists no extensive scale commented on restorative picture dataset tantamount to ImageNet for medical image analysis, they transferred the learned ImageNet loads as starting loads for the DCNN, and fine-tuned these pre-prepared nonexclusive network to perceive hand bone pictures and improved bone age classifications. Their dataset was made of around 1390 left hand radiological scans crosswise over 19 years (018) of ages. The pre-trained VGGNet in view of ImageNet was fine-tuned with the bone age X-ray picture patches. They kept the prior layers of DCNNs fixed what's more, just tweak some more elevated amount bit of the network. Their paper utilized transfer learning inside DCNNs to perform bone age classifications making full utilization of favorable circumstances of DCNNs. They characterized different Regions of Interest (ROIs) based on domain knowledge for every one of which a neighborhood bone age classification display was accomplished by fine-tuning the pre-trained VGGNet with corresponding ROI patches. A last bone age arrangement was gotten by melding numerous regional models. Two measures were utilized to assess the execution of the classification models: classification accuracy and Mean Absolute Error (MAE) which was the mean absolute difference between the anticipated bone age and the genuine bone age. The outcomes demonstrated that the proposed approach outperformed the present best in class grouping strategies in BAA with little dataset. They have achieved classification accuracy of 45% and MAE of 72%.
Xiangyu Zhang et al. intended to quicken the test-time computation of convolutional neural systems (CNNs), particularly very deep CNNs that have significantly affected the computer vision network [18]. Their strategy considered the nonlinear units. They first pro-posed a response reconstruction method that took into account the nonlinear neurons and a low-rank constraint. They built up a compelling solution for the subsequent nonlinear optimization issue without the need of stochastic gradient descent (SGD). Their nonlinear technique empowered a deviated reproduction that decreased the quickly collected error when different (e.g., ¿=10) layers were approximated. Moreover they presented a rank selection method for adaptively determining the speeding up of each layer for an entire model, in view of their excess. In trials, they showed the impacts of the nonlinear solution, topsy-turvy recreation, and entire model speeding up by controlled experiments of a 10-layer show on ImageNet classification. For the generally utilized extremely profound VGG-16 model, their strategy accomplished an entire model speedup of 4 with just a 0.3% expansion of best 5 error in ImageNet classification. Their 4 quickened VGG-16 demonstrated additionally demonstrate an effortless exactness debasement for object detection when plugged into the Fast R-CNN detector.
C. Spampinato et al. assessed, Skeletal bone age assessment was basically performed on left hand by using Greulich and Pyle (GP) method or the Tanner-Whitehouse (TW) one to growth disorders in children [27]. Because of limitation of those methods they proposed several deep learning approaches. The outcomes demonstrated a normal error between manual and programmed assessment of about 0.8 years, which is cutting edge execution. Besides, this is the principal robotized skeletal bone age appraisal work tried on an open dataset and for all age extents, races and sexual orientations, for which the source code is accessible, in this manner speaking to a comprehensive the benchmark for future research in the field. They had tried a

few existing pre-prepared convolutional neural systems (OverFeat, GoogLeNet and OxfordNet) on a dataset of around 1,400 X-ray pictures and demonstrated that profound learning arrangements, even prepared on general symbolism, can adapt adequately to every conceivable instance of automated skeletal bone age evaluation

Chapter 3

Workflow and Feature Extraction

3.1 Workflow

This chapter gives a detailed description about our proposed models and the whole working process. The following figure 3.1 shows the visualization of our whole working process.
At the very beginning we have collected the data set. After that, we attach the data image path to the csv file. Then we have split it into the raw training data, and texting data. Then again we split the data into training and validation data. We pre-processed all the images into data by data generator. After the image data generation has been completed, we have created the pre-trained model vector that are given in the keras, we also have created the gender vector and then concentrate the vectors into our pre-trained model to get a proper training model. Then we have send our testing data into the model to check how much accuracy our pre-trained model is giving.

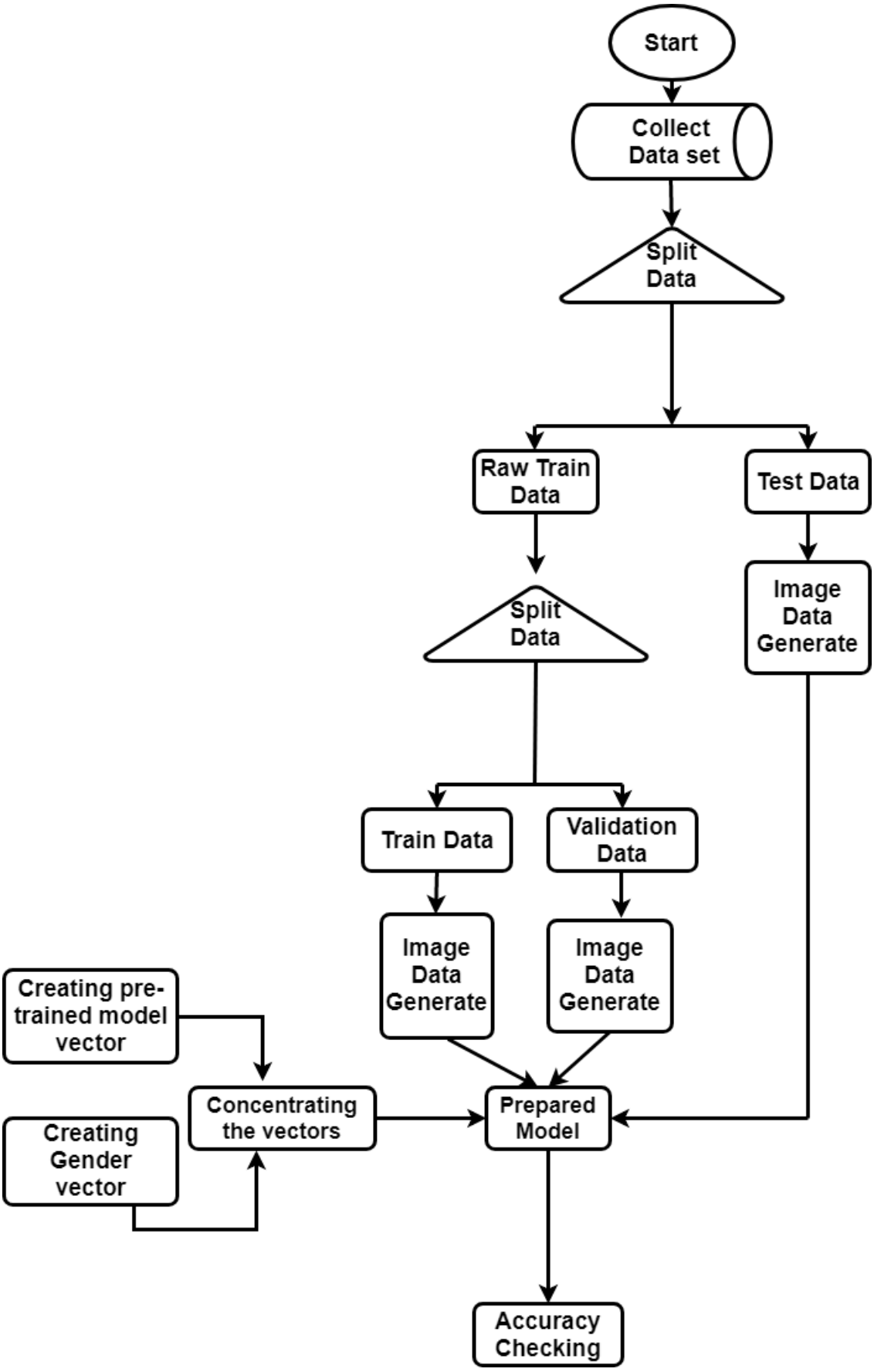


Figure. 3.1: Proposed System

3.2 Data-set Description

The dataset we are using for our thesis is a secondary dataset. We got our dataset from kaggle [32]. The dataset is known as RSNA bone age and primarily used for "Pediatric Bone Age Challenge". The challenge was to develop an algorithm to determine the age from the given data, gender, age and image. This dataset was obtained from Stanford children's hospital and Colorado children's hospital [24].
This dataset is divided into two parts: Image data and the csv file. This dataset contains total 12611 data. The image data contains the x-ray image of the left hand of the both male and female and the csv file contains the bone-age of several patients. The csv file contains the sex of the person, bone-age and the id number. Id number is used for connecting the dataset with the images.
The following table is shows the sample of the csv file

Table 3.1: Sample of CSV

id	boneage(months)	male
1377	180	FALSE
1378	12	FALSE
1379	94	FALSE
1380	120	TRUE
1381	82	FALSE
1382	138	TRUE
1383	150	TRUE
1384	156	TRUE
1385	36	TRUE

The id number is the patient id number, the boneage is given in month and the male column true means the patient is male, otherwise that patient is female.
The following image shows the image sample of the dataset.

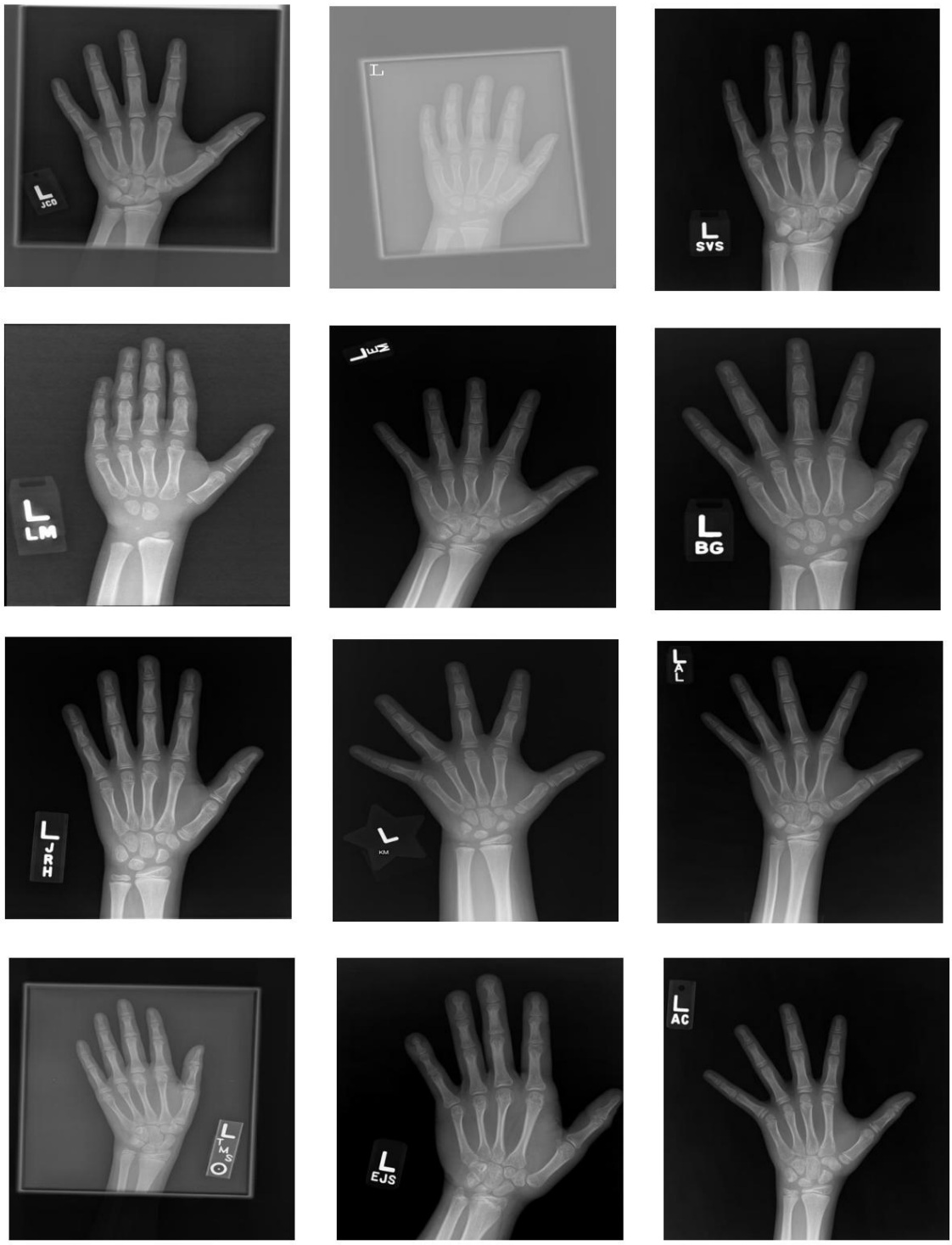


Figure. 3.2: Image Dataset

Male category has 6833 data and female category has 5778 data. The bone-age in male category are from 1 month to 228 months. All data are divided into 131 categories according to the age. In the boneage category, most data are found for 168 months and the amount is 783 and the lowest amount of data is only one x-ray image and 40 category fall into this criteria.The following table shows the sample of the category of bone age in of the dataset in male section.

Table 3.2: Boneage Male

boneage (months)	data found
168	738
156	725
150	595
162	491
138	428
132	395
120	357
108	310
96	248
84	225
72	224
180	214
204	173
60	140
186	127
144	115
192	112

In the female category the bone-age are from 4 months to 216 months and the data are divided into 100 categories. For female category the highest amount of data has been found for 132 months which is 689 data and the lowest amount of data is 1 image data and 33 categories are in this section.The following table shows the sample of the category of bone age in of the dataset in female section.

Table 3.3: Boneage female

boneage (months)	data found
132	689
120	635
144	542
94	490
106	474
156	388
82	380
180	204
162	191
69	190
168	154
60	138
126	124
138	101
50	95
150	83
36	68

The following figure shows the different categories of the male. Top 17 category with highest amount of data are given here.

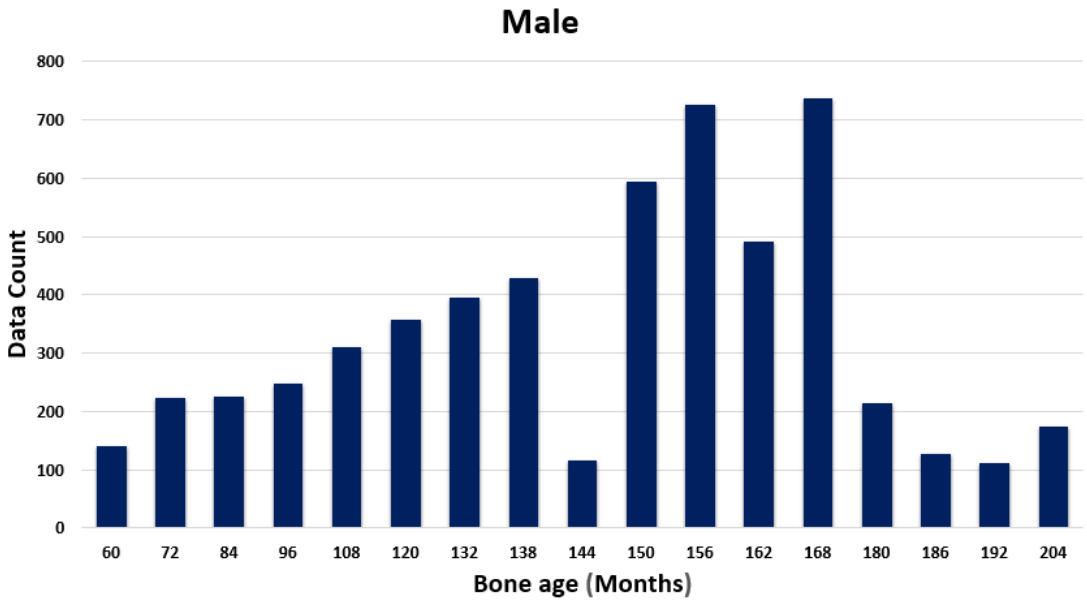


Figure. 3.3: Male data-set

The following figure shows the different categories of the female. Top 17 category with highest amount of data are given here.

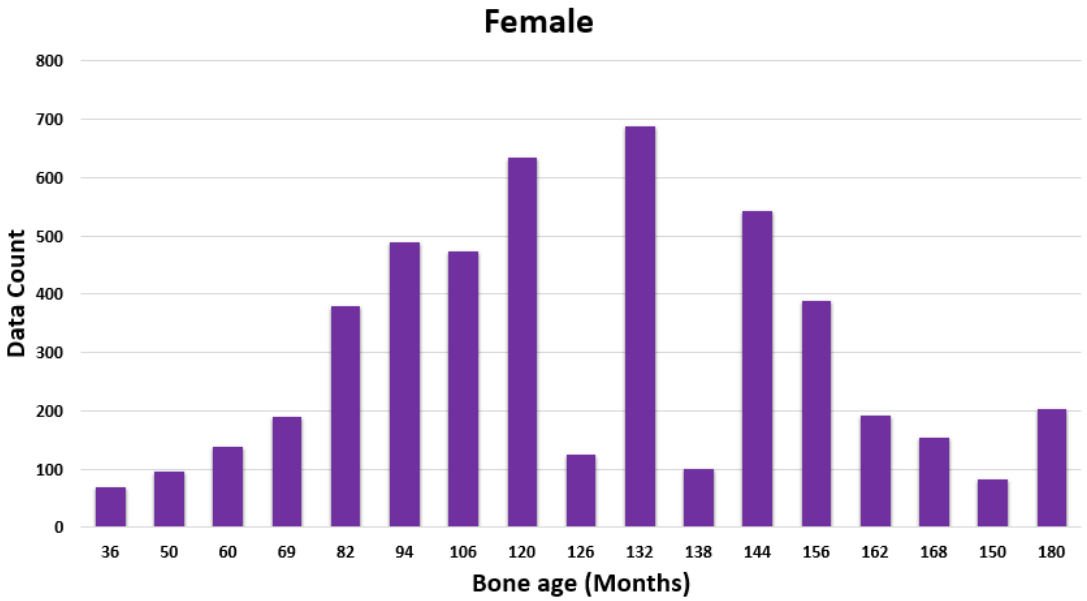


Figure. 3.4: Female data-set

We have randomly split the whole data for testing,training and validation. We took 10000 data for the training, 1009 data for validation and 2523 data for testing our pre-trained model. We are going to use both age and gender features for our research because female bone are approximately 2.5 years mature than male.

3.3 Pre-prossesing

3.3.1 Image processing technique

Image processing is a technique what we use to perform some operation on image to associate degree increased image or to extract some helpful data from the image. Image processing is a sort of signal dispensation that takes the input as image sort of a video frame and photograph and therefore the output will be image or characteristics related to that given image. Usually the images are treated as two dimensional signals when the set signal processing methods are being applied to them.

Purpose of image processing

There are several reasons for the image processing. Mainly they are divided into five groups.

1. **Visualization:** Observation of the objects that are not visible.
2. **Observation of the objects that are not visible.:** To make a more robust image.
3. **Image retrieval:** Explore the image of interest.
4. **Measurement of pattern:** Calculates various objects in an image.
5. **Image Recognition:** Differentiate the objects in an image.

Types of image processing

Analog image processing and digital image processing are two types of variation on the image processing [6]. To analyze the hard copies like printout or photographs Analog or visual techniques are used where as for the digital image processing helps to manipulate the digital images given to the computer by some techniques. Our processing is digital image processing as we are providing the image data to the pre-trained models.

3.3.2 Image pre-processing with keras

Keras Image Preprocessing is the image data preprocessing and data augmentation module of the Keras deep learning library and we have used it to process our image. This image generator generate batches of tensor image data with real-time data augmentation [1].
The image we got in our dataset were about resolution 2000×1500 pixels greyscale format. We have converted it into resolution 224×224 pixels 8 bit RGB format as it is the standard input for the pre-trained models of CNN in keras. The following figure is the parameters that are used in converting the images into data. We have flipped the image horizontally but not vertically. As all of the image given are left hand images, horizontal flip will help to acquire the knowledge of the right hand also. We have used 15% height shift and width shift and zoom the images 25% to get enhanced images to train our models more accurately. After doing a little research we found those parameters to be optimal. We have used the filling mode

```
ImageDataGenerator(samplewise_center=False,
samplewise_std_normalization=False,
horizontal_flip = True,
vertical_flip = False,
height_shift_range = 0.15,
width_shift_range = 0.15,
rotation_range = 5,
shear_range = 0.01,
fill_mode = 'reflect',
zoom_range=0.25)
```

Figure. 3.5: Parameters used in ImageDatagenrator() method

'reflect' it means after doing the horizontal and shift is the points are outside of the boundary it will be filled by reflection. Which means if the original data is "abcd" if the points of the left are outside the boundary it will be like "abcddcba" and if the points of the right are outside of the boundary it will be like "dcbaabcd". We didn't use 'constant' as our image is black and white, it can hamper the accuracy of the result. Similar goes for the 'nearest'. But we can use the 'wrap' parameter as it has some similarities with the 'reflect'. We have used a rotation of 5 degree for our training shearing the image. By doing this whole procedure our classifier got many altered version of the same image which forced the network to find out the features which are essential and images of a hand can be easily acquired by with varying zoom, rotation, position as well as the image is left hand's image or right hand's image. This factors factors do not impact our skills as radiologists to investigate the image.
All this data is going to be saved in a directory. After all the data are generated we generates batches of data from the images in the directory for training, testing and validation data. We did it by flow by data method. The following figure is the visual representation of the method. Class mode determines how the data is going

```
flow_from_directory(
base_dir,
class_mode = 'sparse',
batch_size = 10,
shuffle=True)
```

Figure. 3.6: Parameters used in `flow_from_directory()` method

to be returned. We have used 'sparse' so that the the gender must be included with the images as string and it will return a 1D integer label array. The batch size determines the amount of samples circulated over the network. For the training set we have set the batch size 10 which means from the 10000 training data the algorithm took take first 10 samples from 1st to 10th from the training dataset and trained the network. Then it took the next 10 samples and it kept doing it until all the samples are done. But for the testing and validation we included full batch data as it fluctuated more than the full batch gradient and for the testing and the

validation data that is not required.
For training we have used a very small batch for data because of some advantages:

- It took less memory. Since the network used fewer samples, the overall training procedure.
- Network can be trained faster with mini-batches because it updated the weights after each propagation. In our training we have propagated 1000 batches and after each of them the network's parameter has been updated. If we had used one parameter it would make only one update for the network's parameter.

3.4 Compiling model

After doing the pre-processing of the image we had compiled our model.In our model, input shape is the image shape which is for our data 224×224×3.And for the gender the model will take as input of arrays of shape (*, 1) as gender is only two input, 0 or 1(0 for female, 1 for male).

```
img = Input(shape = IMG_SHAPE)
gender = Input(shape=(1,))
cnn_vec = Model(input_shape = IMG_SHAPE,
        include_top = False, weights = 'imagenet')(img)
cnn_vec = GlobalAveragePooling2D()(cnn_vec)
cnn_vec = Dropout(0.2)(cnn_vec)
gender_vec = Dense(32,activation='relu')(gender)
features = Concatenate(axis=-1)([cnn_vec,gender_vec])
dense_layer = Dense(1024, activation = 'relu')(features)
dense_layer = Dropout(0.2)(dense_layer)
dense_layer = Dense(1024,activation='relu')(dense_layer)
dense_layer = Dropout(0.2)(dense_layer)
output_layer = Dense(1, activation = 'linear')(dense_layer)
bone_age_model = Model(inputs=[img,gender],outputs=output_layer)
```

Figure. 3.7: Compiling the Deep Model

3.4.1 Inactivating fully connected layer

By making the include top false the fully-connected layer at the top of the network has been disabled. The fully connected layers at the end can only take fixed size of input however its works the image input but not ideal for images. Fully connected networks works well with the whole images, but if a cropped image is given the actual object can't be classified easily with the fully connected network. For a better result the individual features of the object should be identified such as what types of lines and curves are there in that image so that the objects can be classified with the partial information. Activating the fully connected layers can lead the whole model into overfitting. Which means it will perform well in known instances, but performs badly in unknown instances. And the specific size of images only can be given when the fully connected network is false otherwise it will be converted to the size which is fixed by the models.

3.4.2 Weight selection

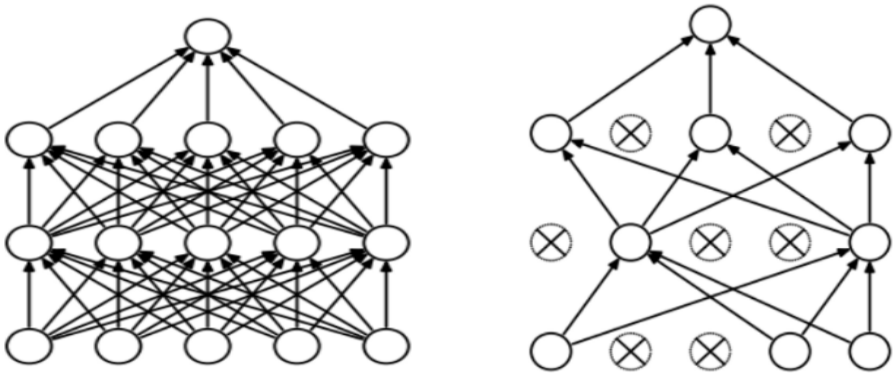
Weights that are used in our research is imagenet. The other weights can be used is null or the path to the weight file is loaded. As we are using pre-trained models to compare, imagenet has been used so that it will take the weight from the pre-trained models that are already done to find which one works well.

3.4.3 Pooling layer selection

For the pooling layer average pooling 2D is used. We haven't specified any parameters for our average pooling so that it will take the pool size 2*2 which is default in keras models. The reason of using 2D is because the model we are using is 2D, if we have used 1D or 3D it can hamper the result. We could have used the max pooling here, but in average pooling the average of the matrix is taken, where in max pooling only the take the maximum value from the matrix. A big chunk of data is rejected in max pooling and it retains at max 1/4th whereas average pooling do not reject all of it and more information can be retained from average pooling compare to max pooling and that's why it is believed to lead to a better result.

3.4.4 Dropout selection

Dropout of 0.2 has been selected for our model. Dropout is a form of regularization in CNN and it is mainly used for the CNN to prevent the overfitting of data. The main concept of dropout is ignoring the randomly selected neurons during training [13]. Dropout forces a neural network to learn more robust features and it also



(a) Standard Neural Network (b) After Applying Dropout

Figure. 3.8: CNN Dropout [13]

lessen the training time. The 0.2 dropout means it will ignore 20% of neurons during training. The reason we have used this dropout because after some research that happened on CIFAR-10 data set, we found that 0.2 is giving the more accuracy, so we have decided to use this rate for our thesis.

3.4.5 Activation function

We have created a gender network which has took the information "male" or "female" which has been converted into binary form earlier(0 for female and 1 for male)

and converted it into a 32-densely connected neurons. The purpose of using this neuron for balancing relative contribution of each input into final decision (All the pixels and gender). And the activation function "ReLu" has been used. The main function of the activation function is to convert an input signal of a node in a NN output signal [26]. The output signal can be used as input for the next layer. Most popular types of activation functions are:

1. Sigmoid or Logistic Function
2. Tanh Function
3. ReLu Function

Sigmoid Function

The mathematical term of sigmoid function is [19]

$$SF: \quad f(x) = \frac{1}{1 + exp(-x)} \tag{3.1}$$

Its range is between 0 and 1 ans it is a s-shaped curve. Applying this activation function is comparatively easy but it has vanishing gradient problem and its output isn't zeros centered as a result the gradient updates go too far in different direction which makes optimization harder.

Tanh Function

The mathematical term of tanh is [19]

$$Tanh: \quad f(x) = \frac{1 - exp(-2x)}{1 + exp(-2x)} \tag{3.2}$$

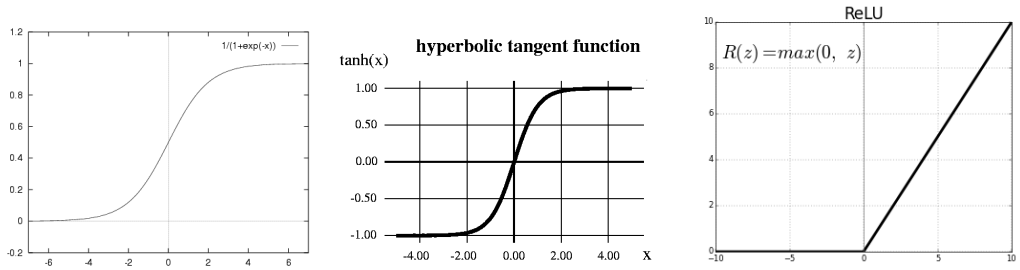
Its output is zero centered which gives a better result than sigmoid function but still it suffers from vanishing gradient problem.

ReLu Function

The mathematical fuction of relu is [19]

$$ReLu: \quad R(x) = max(0, x) \quad \begin{cases} R(x) = x & \text{if } x \geq 0 \\ R(x) = 0 & \text{if } x < 0 \end{cases} \tag{3.3}$$

Using this function is very simple and efficient and it avoids the vanishing gradient problem. Because of this reason we have used this activation function for our models. But it has a problem, it can be only applied in the hidden layers but comparing with the other two activation functions it seems better.



(a) Sigmoid (b) Tanh (c) ReLu

Figure. 3.9: Activation Functions [19]

3.4.6 Concatenation and output layer

Output of the both layers had been merged or concatenated with the concatenation method. This method takes as input a list of tensors.All of the same shape except for the concatenation axis and the output is a single tensor, where both the gender and the image has been merged [37]. The concatenated layer was then fed with two additional dense layer of 1024 and dropout layer of 0.2 and then the final layer connected to a single output neuron with the linear activation which predicts the age. The final layer consists of a single neuron with linear activation which predicts the age and this is the output layer. This ratio has been chosen because we did not want to bias the network too significantly based on gender input, instead of it we wanted to give it the ability of overall prediction.

3.5 Applying the prepared model for testing

After out model is fully prepared, we have text and check the dataset to find out which one is giving more accurate result for our data. Details about it has been described in Chapter 4.

Chapter 4

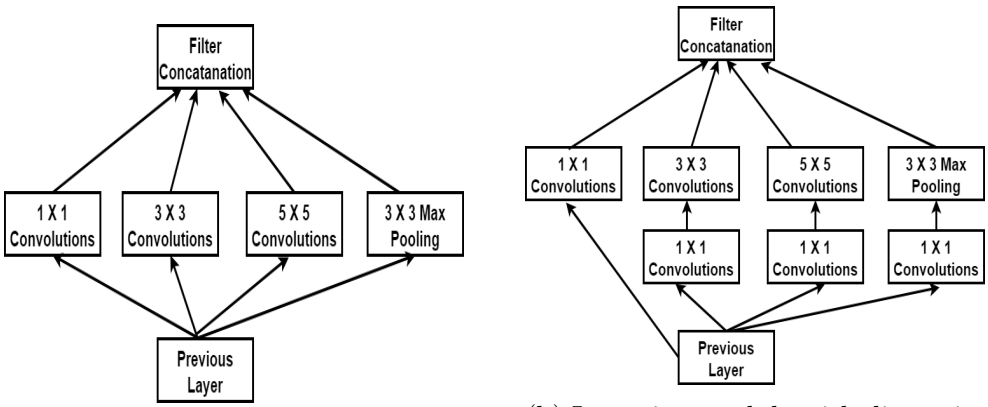
Implementation and result

This chapter gives full description about the models we have applied for our thesis also the outcomes of the thesis.

4.1 InceptionV3

4.1.1 Inception concept

The core idea of the Inception architecture is to consider how an ideal neighborhood inadequate structure of a convolutional vision system can be approximated and covered by dense components that are readily available. The first inception model named Inception-V1 which was first introduced by Google [14]. The first model that



(a) Inception module,naive version

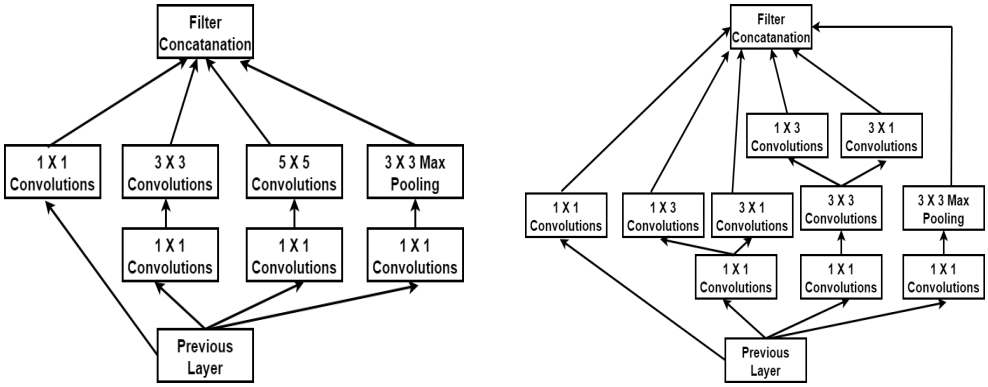
(b) Inception module with dimension reduction

Figure. 4.1: InceptionV1

is proposed performs convolution on an input with 3 different sizes filters 1×1, 3×3 and 5×5. Then max pooling is performed and outputs are concatenated and sent to the next inception module. The main problem of this model it requires a high amount of operation. The problem becomes even more once the pooling layer is added to the mix. To reduce this problem an additional convolution of size 1×1 has been added which helps the dimension reduction and performs fast that the naive inception model.

4.1.2 InceptionV3

To reduce the dimensionality factorization was introduced, which helped to reduce the overfitting problem [17]. For example, for 1 layer 5×5 filter, total number of parameter are 5×5=25. Now instead of 5×5 layer, if two 3×3 layer is used, total



(a) InceptionV1 (b) InceptionV3

Figure. 4.2: InceptionV1 vs InceptionV3

parameters will be 3×3+3×3=18 and we can see the parameters has been reduced by 28%. Again for the 3×3 filter, total number of parameters are 3×3=9 if we use the 3×1 and 1×3 filters, number of parameters will be 3×1+1×3=6 Number of parameters has been reduced by 33% The whole factorization procedure the number of parameters has been reduced and the network can go deeper.

4.1.3 Applying InceptionV3 in our dataset

After preparing our model we have fit our train data and validation data to the models, where we have used the 10, 20, 100 and 1000 steps with 10 and 20 epochs to compare the output that our model is giving. Validation data is for evaluating the loss at the end of each epoch. The model is not be trained on this data. after running several iteration we have found that the amount of steps per epochs have a

Table 4.1: InceptionV3 10 epochs result

Steps per epochs	Mean absolute error	Explained Variance Score	Value loss
10	37.04	0.04	1.2488
20	22.25	0.52	0.4921
100	14.76	0.79	0.2102
1000	13.49	0.86	0.1375

significant affect on the accuracy. The more number of steps the accurate the result is. The reason behind this is the more steps per epochs,the more steps happens in the inception,so that,it is giving the more accuracy. Number of epochs also have an effect in the output. The models are saving only the best value so that the more epochs happens, the chances of getting accurate result is being increased. However,it does not have the significant chance as the steps per epochs, it still have some effect int the output result.The output age of the test result is shown in figure 4.3-4.10.

Table 4.2: InceptionV3 20 epochs result

Steps per epochs	Mean absolute error	Explained Variance Score	Value loss
10	33.53	0.04	1.0292
20	19.95	0.66	0.3689
100	14.25	0.82	0.2011
1000	10.10	0.91	0.1002

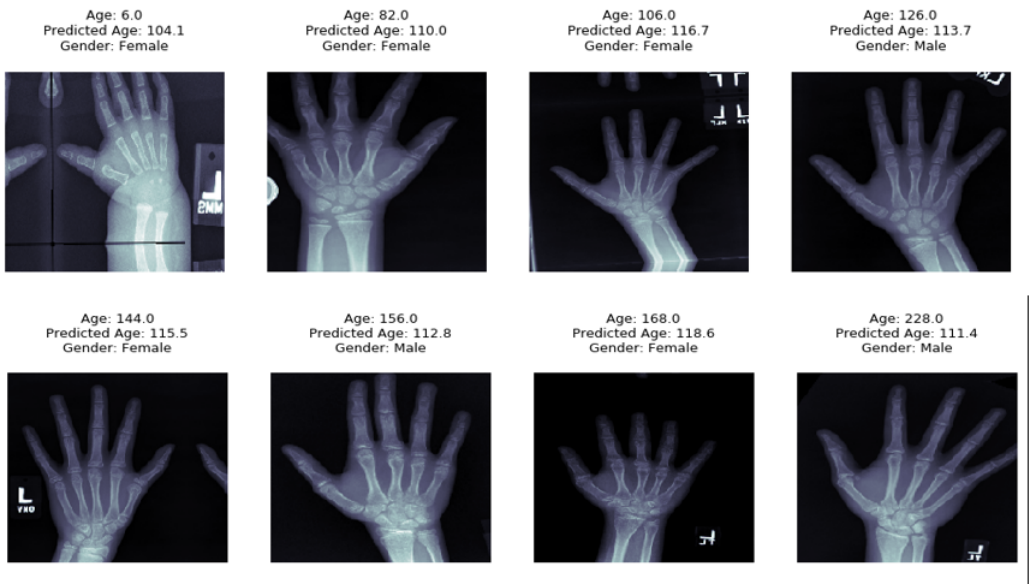


Figure. 4.3: InceptionV3 10 steps of epoch and 10 epoch

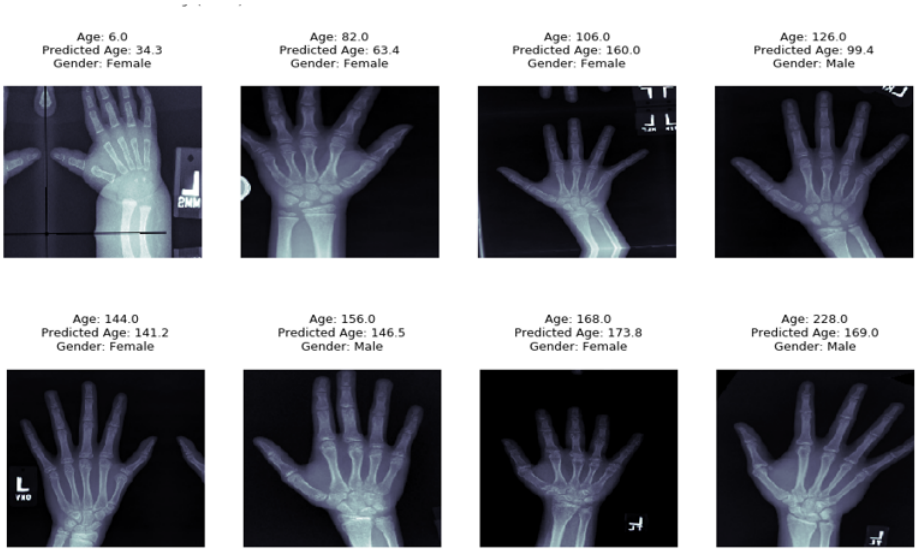


Figure. 4.4: InceptionV3 20 steps of epoch and 10 epoch

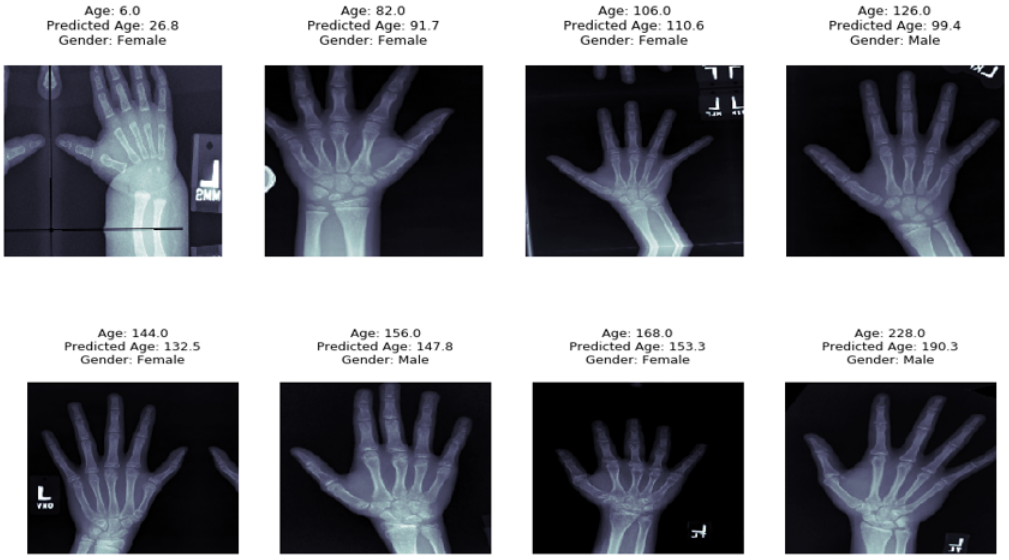


Figure. 4.5: InceptionV3 100 steps of epoch and 10 epoch

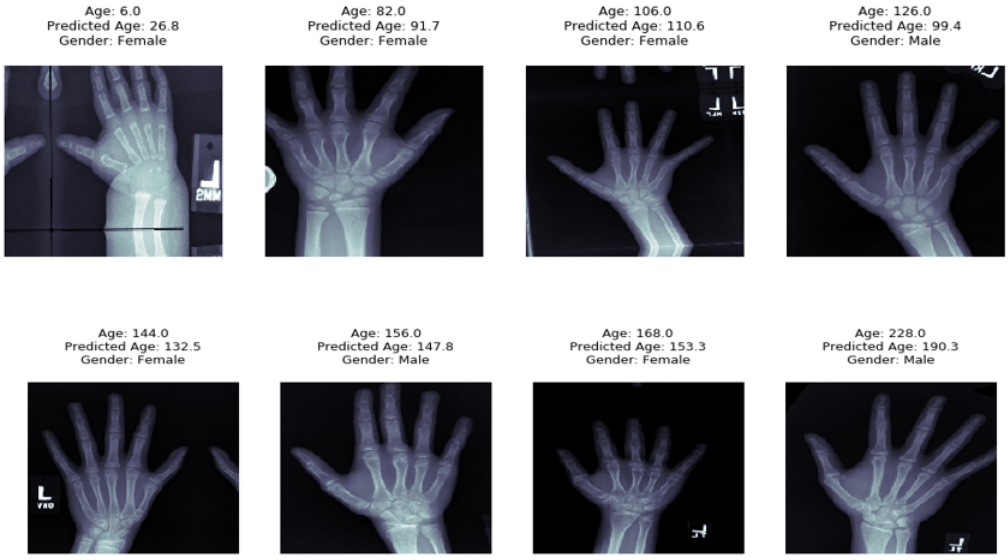


Figure. 4.6: InceptionV3 1000 steps of epoch and 10 epoch

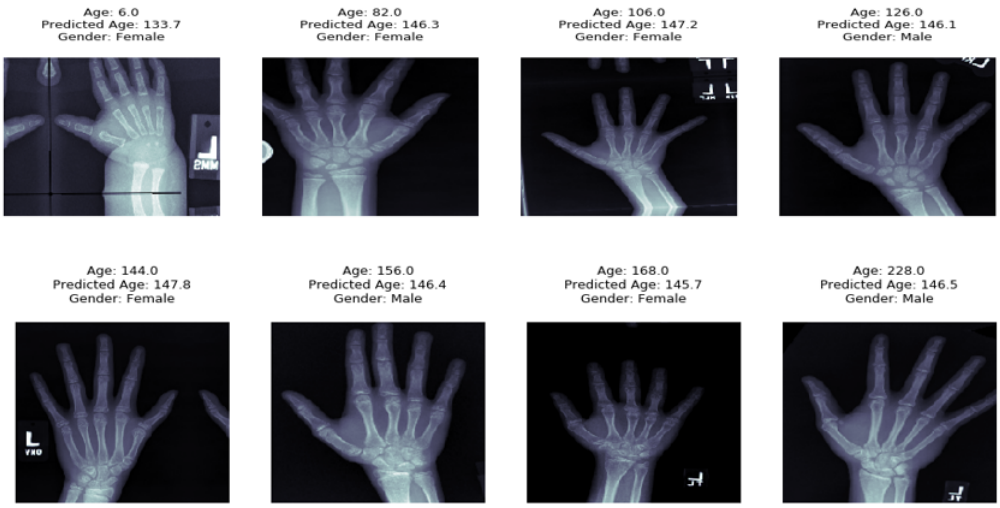


Figure. 4.7: InceptionV3 10 steps of epoch and 20 epoch

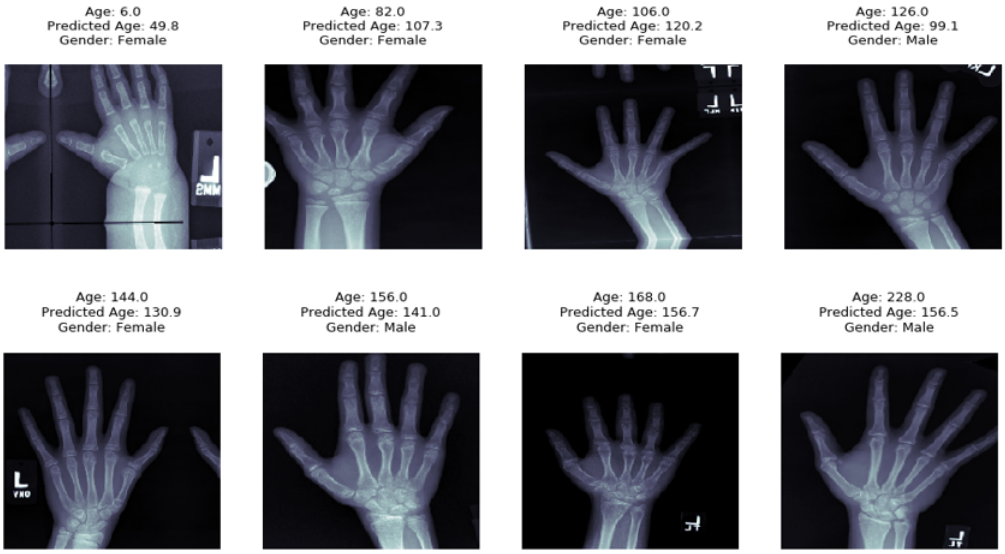


Figure. 4.8: InceptionV3 20 steps of epoch and 20 epoch

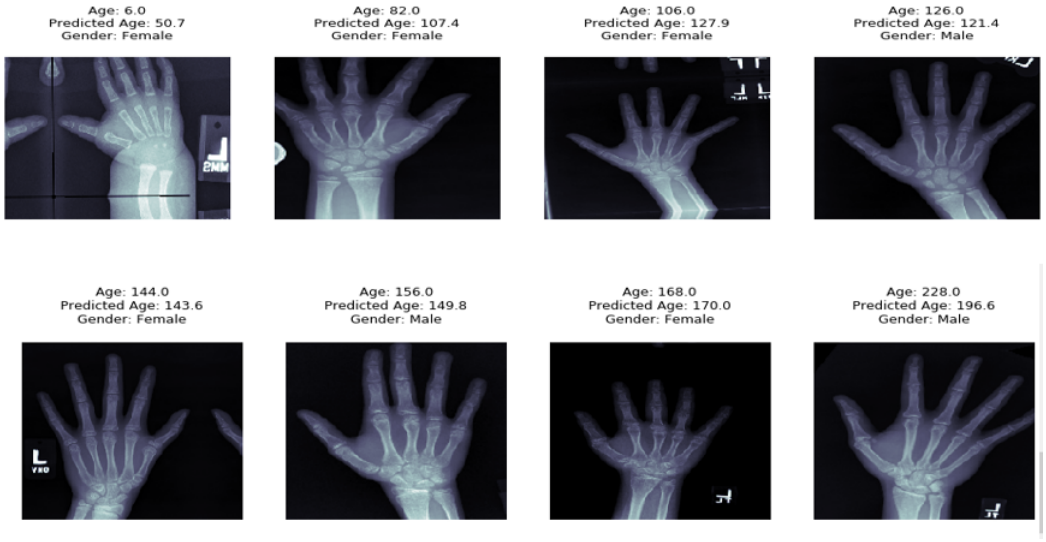


Figure. 4.9: InceptionV3 100 steps of epoch and 20 epoch

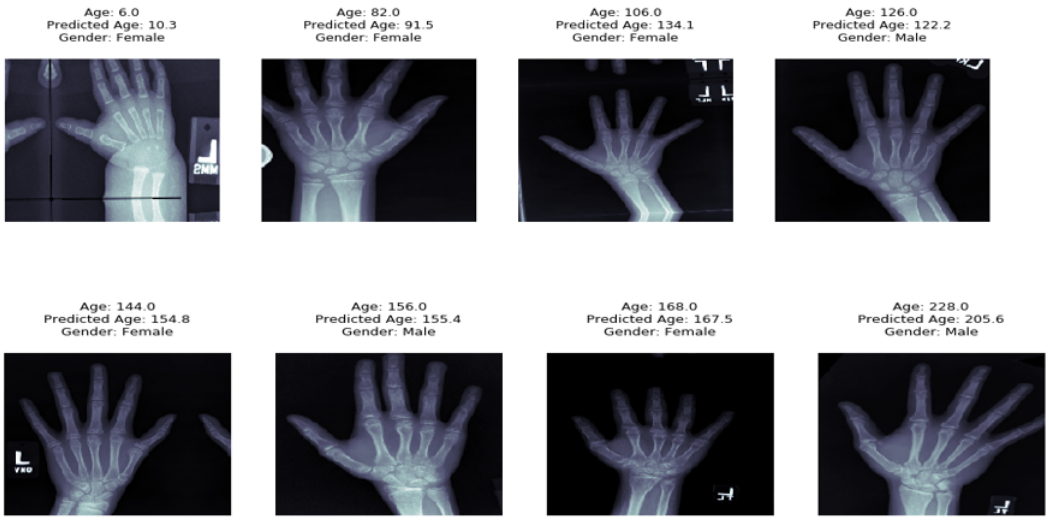
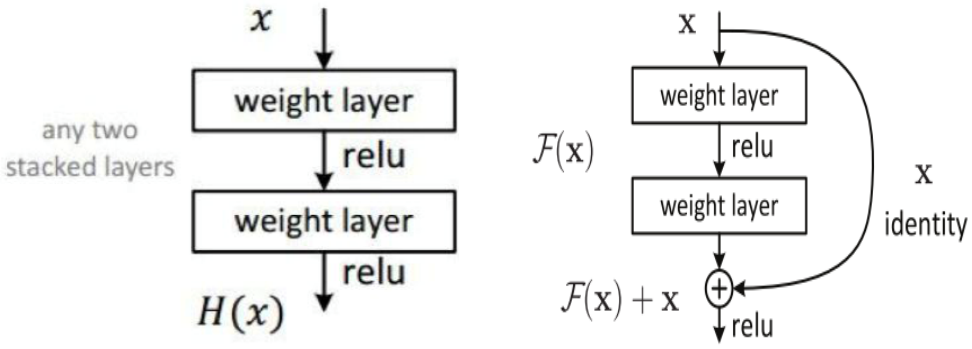


Figure. 4.10: InceptionV3 1000 steps of epoch and 20 epoch

4.2 ResNet50

4.2.1 ResNet concept

ResNet is the short form of residual net.In a deep convolutional neural network, several layers are stacked. The network learns several features at the end of its layers. But when the network goes deeper some difficulty is being faced and the accuracy starts saturating and it also degrades [20]. From CIFAR-10 and ImageNet it is found that the more layers are in it, the more error it is showing. The reason for this is the gradient cannot flow backwards because of the vanishing gradient problem and if the layer is very deep it cannot learn the features from the beginning layers. So the solution is being proposed to add an identity function x which will map the features from the previous layer directly to current layer and the gradient



(a) Plaint Net (b) Residual Net

Figure. 4.11: Plaint Net and Residual net [29]

can be transferred back easily [16]. So if the gradient is vanished because of the non-linearity. The gradient can directly flow to the input layer and in this way the network will learn better. This system is called residual net.

4.2.2 Applying ResNet50 in our dataset

Just like InceptionV3,the steps per epochs are affecting the result and the number of epochs are also affecting the result. The output of ResNet50 is being shown in the figure 4.12-4.19

Table 4.3: ResNet50 10 epochs result

Steps per epochs	Mean absolute error	Explained Variance Score	Value loss
10	39.61	-.052	2.0353
20	32.89	0.32	1.0292
100	15.78	0.76	0.2027
1000	12.87	0.85	0.1246

Table 4.4: ResNet50 20 epochs result

Steps per epochs	Mean absolute error	Explained Variance Score	Value loss
10	32.60	0.029	1.02780
20	29.75	0.135	0.90183
100	21.17	0.57	0.46710
1000	10.65	0.90	0.10130

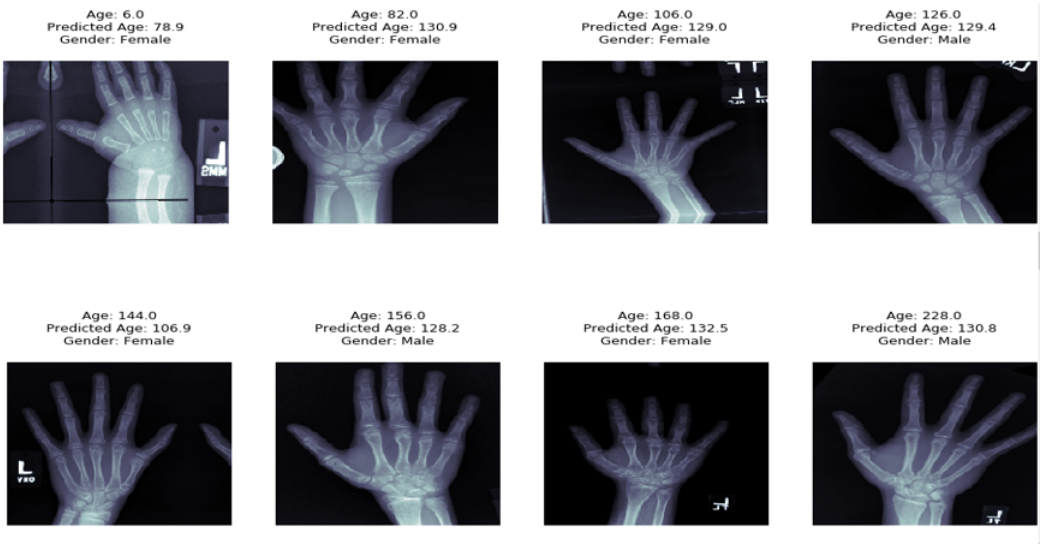


Figure. 4.12: ResNet50 10 steps of epoch and 10 epoch

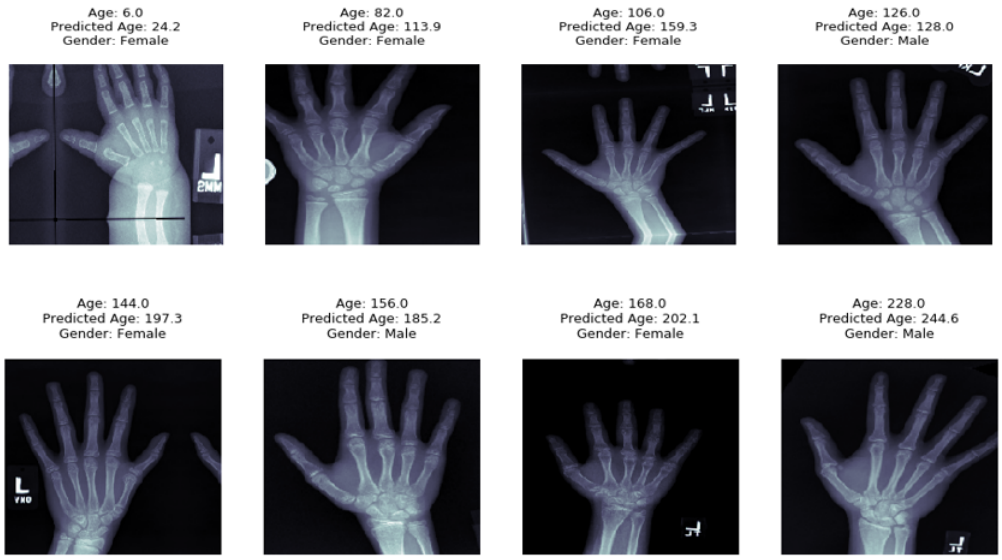


Figure. 4.13: ResNet50 20 steps of epoch and 10 epoch

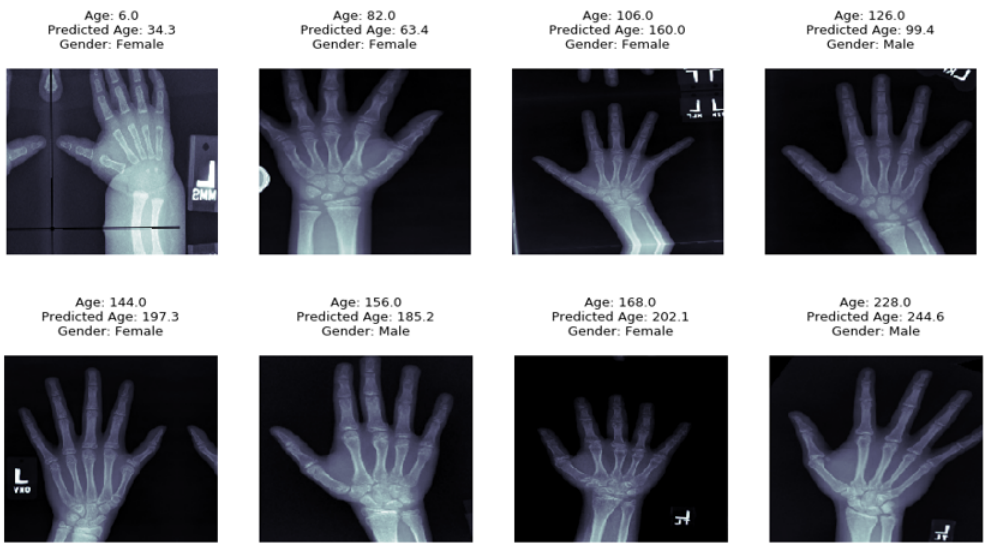


Figure. 4.14: ResNet50 100 steps of epoch and 10 epoch

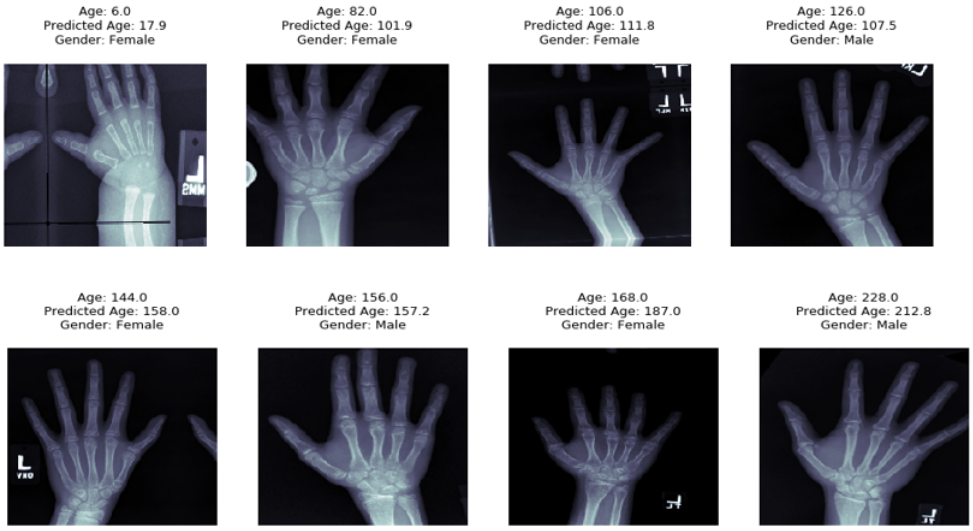


Figure. 4.15: ResNet50 1000 steps of epoch and 10 epoch

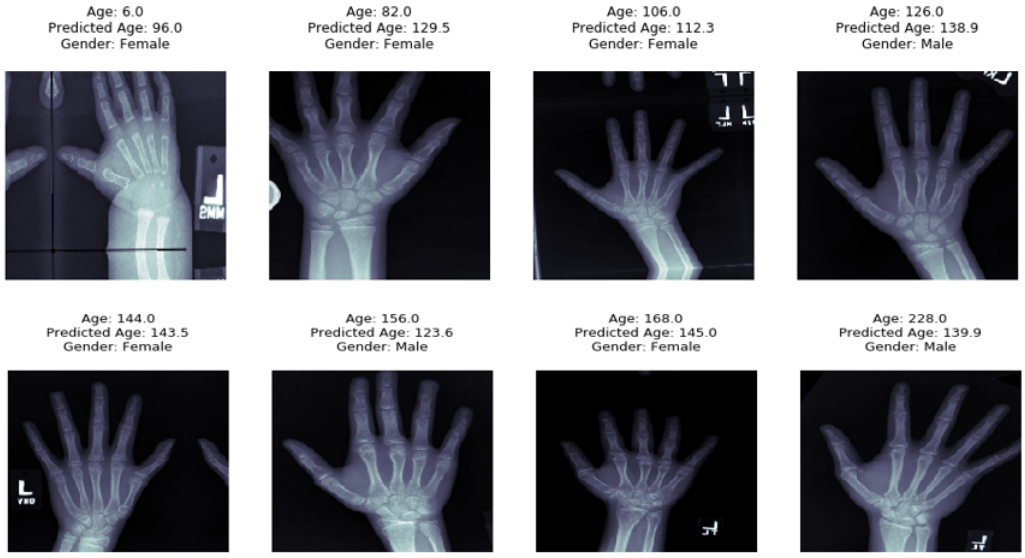


Figure. 4.16: ResNet50 10 steps of epoch and 20 epoch

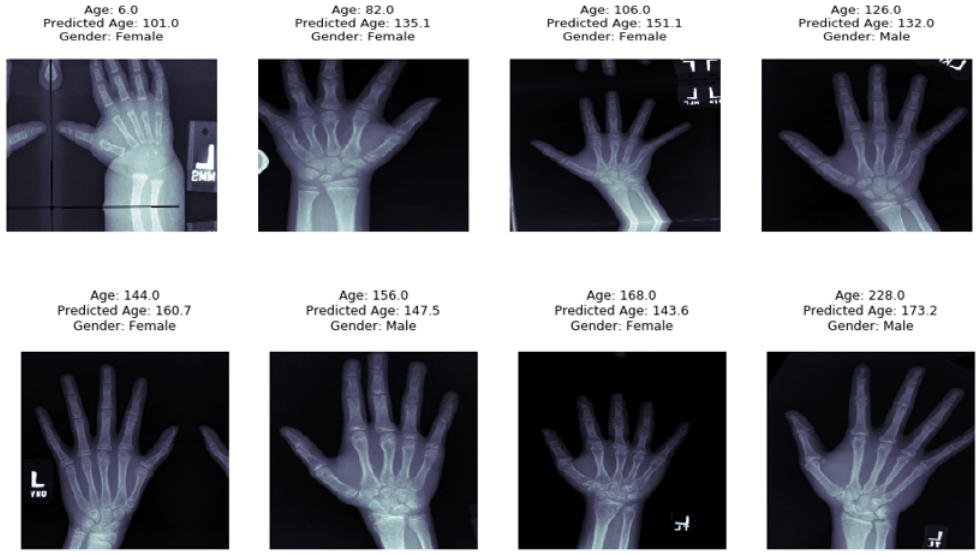


Figure. 4.17: ResNet50 20 steps of epoch and 20 epoch

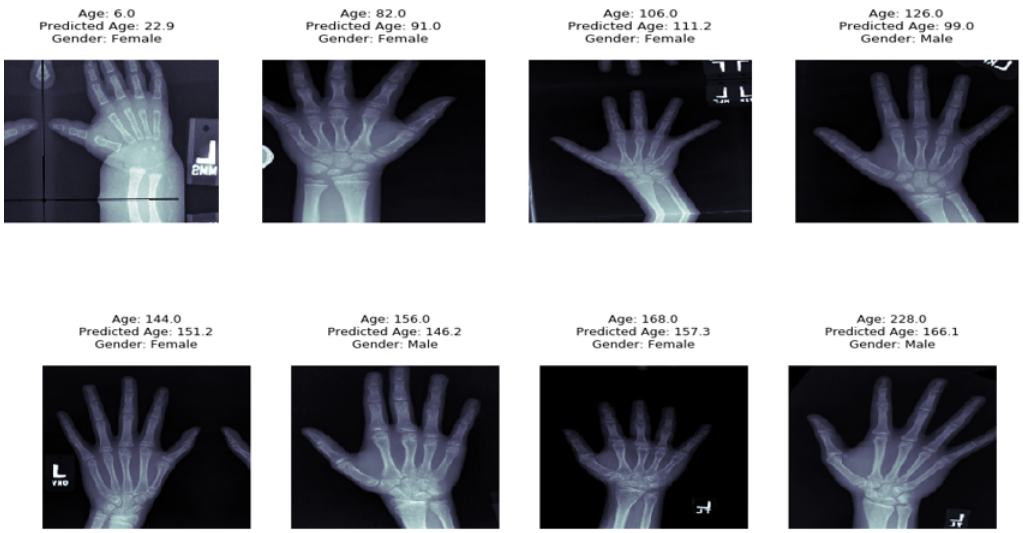


Figure. 4.18: ResNet50 100 steps of epoch and 20 epoch

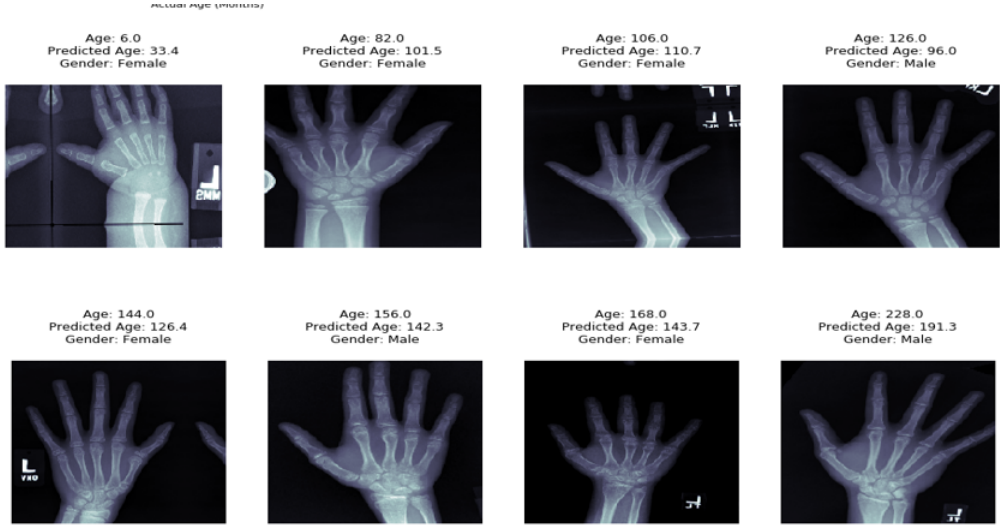


Figure. 4.19: ResNet50 1000 steps of epoch and 20 epoch

4.3 VGG16

4.3.1 VGG Concept

The VGG network architecture was introduced first at 2014 [12]. In training time the input of this algorithm is a fixed size 224×224 RGB image. The main preprocessing done is to subtract the mean RGB value, calculate the training set, from each pixel. Then the image is transferred through some convolutional layers where filters with very small receptive field: 3×3 is used (It is the least size to capture the notion of left/right, up/down or center). In one of the setups we additionally utilize 1×1 convolution channels, which can be seen as a linear transformation of the input channels. The convolution stride is fixed to 1 pixel; the spatial cushioning of conv. layer input is with the end goal that the spatial resolution is protected after convo-

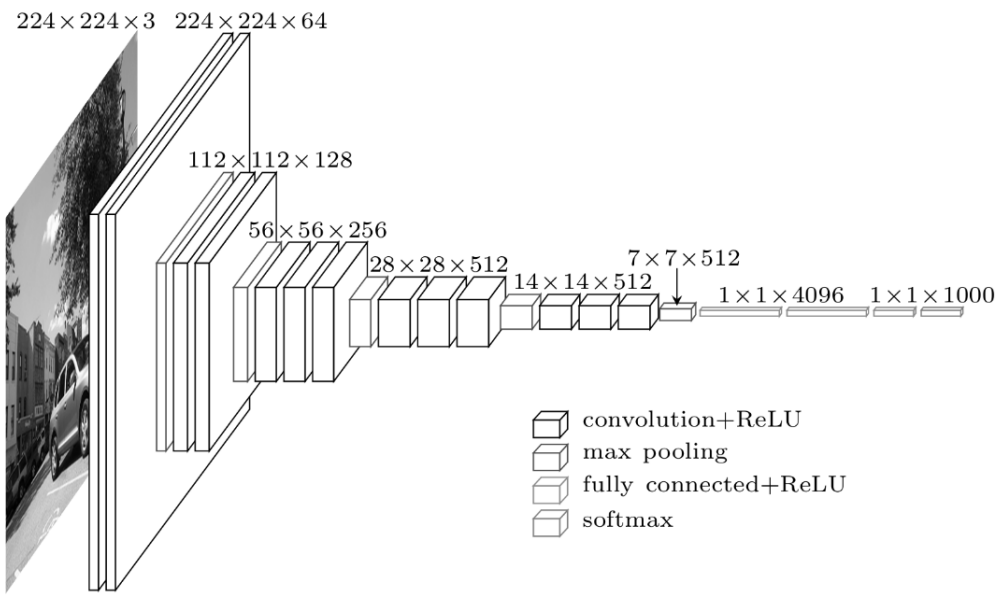


Figure. 4.20: VGG16 Architecture [29]

lution. (I.e. the padding is 1 pixel for 3×3conv. layers. Spatial pooling is carried out by five max-pooling layers, which follow some of the conv. layers. Max-pooling is performed over a 2×2pixel window, with stride2. A stack of convolutional layers is trailed by three Fully-Connected layers: the initial two have 4096 channels each, the third performs 1000-way ILSVRC arrangement and subsequently contains 1000 channels (one for each class). The last layer is the soft-max layer. The setup of the fully connected layers is the equivalent in all networks. Every shrouded layer are furnished with the rectification (ReLu) non-linearity.

4.3.2 Applying VGG16 in our dataset

After applying different steps per epochs and and different epochs the output of VGG16 does not vary much. so we can say that steps per epochs and epochs number do not play any significant role in the output.

Table 4.5: VGG16 10 epochs result

Steps per epochs	Mean absolute error	Explained Variance Score	Value loss
10	33.18	0.020	1.0312
20	32.58	0.032	1.0281
100	33.52	0.022	1.0329
1000	34.65	0.004	1.0351

Table 4.6: VGG16 20 epochs result

Steps per epochs	Mean absolute error	Explained Variance Score	Value loss
10	35.05	-0.052	1.0533
20	35.31	-0.058	1.0552
100	33.68	0.019	1.0418
1000	35.43	-0.032	1.0547

Figure 4.21-4.28 shows the output of the VGG16 model.

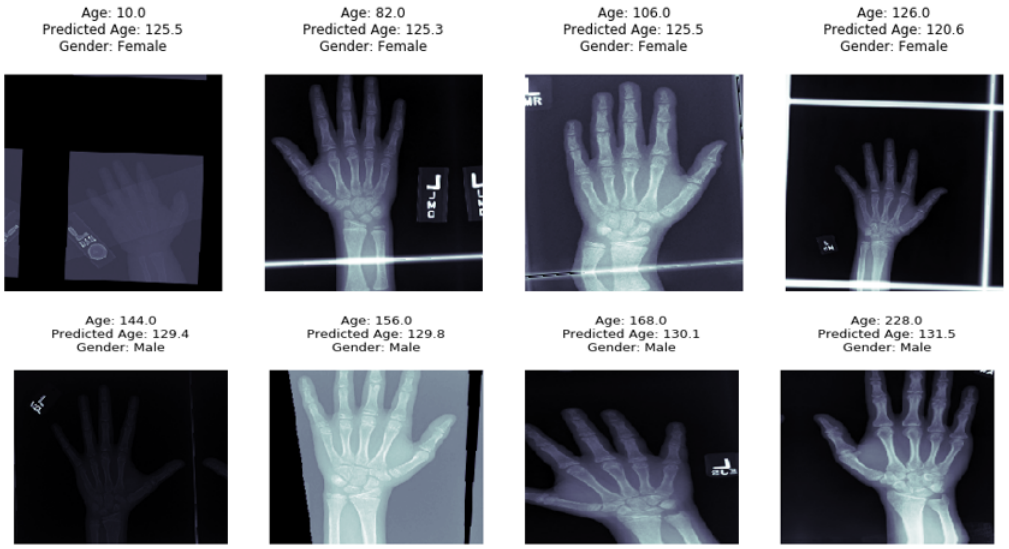


Figure. 4.21: VGG16 10 steps of epoch and 10 epoch

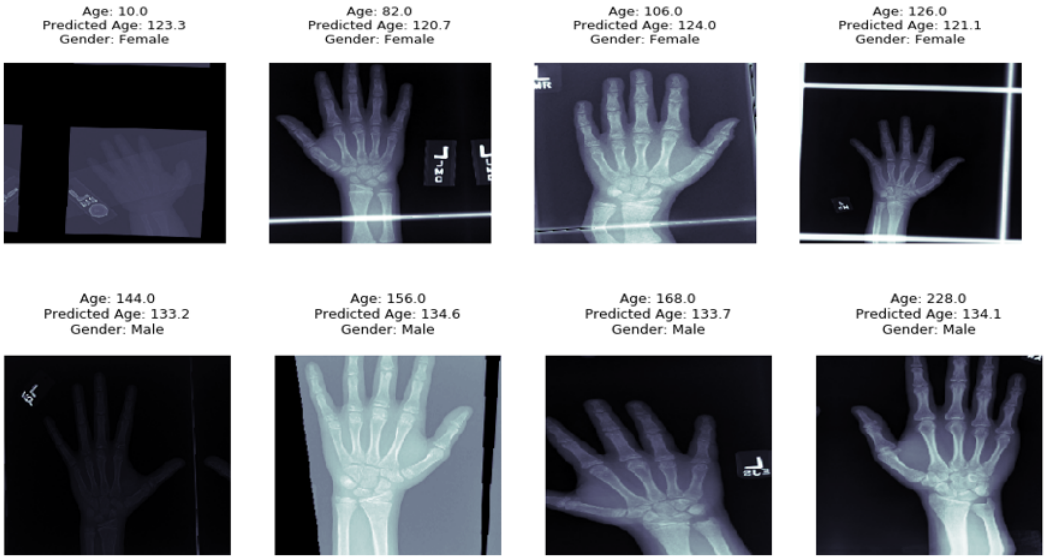


Figure. 4.22: VGG16 20 steps of epoch and 10 epoch

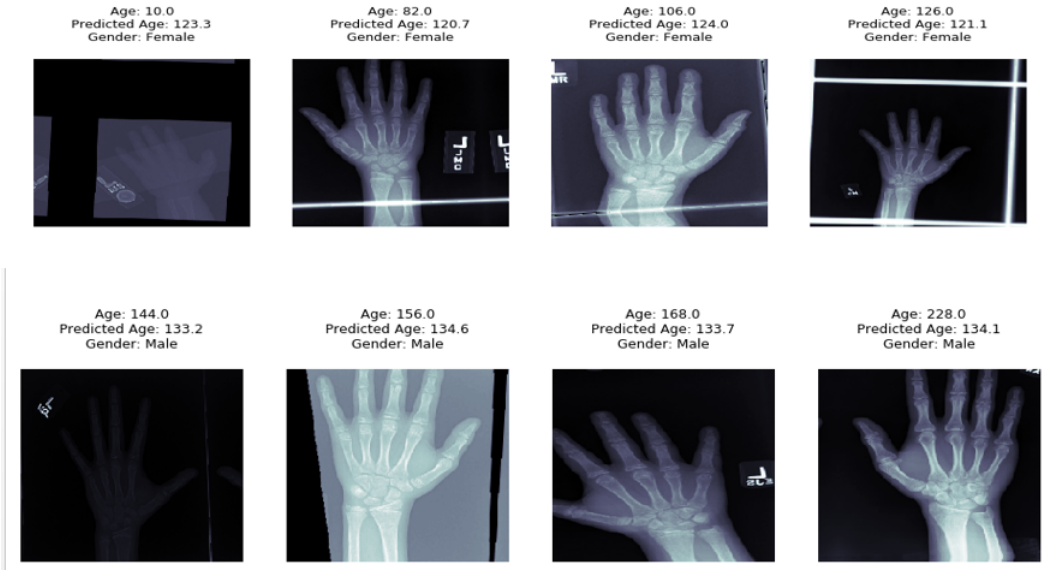


Figure. 4.23: VGG16 100 steps of epoch and 10 epoch

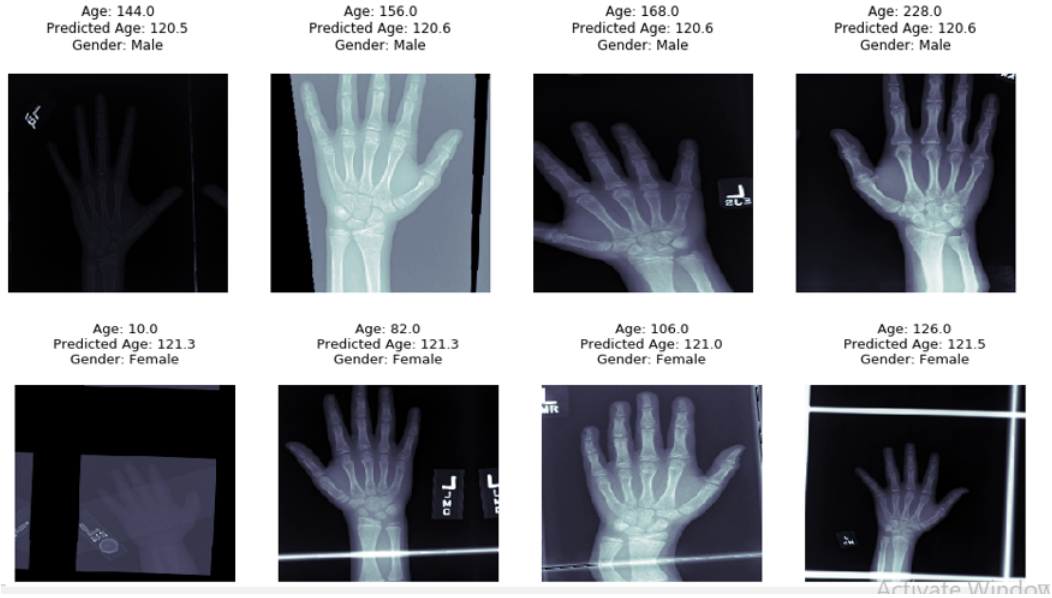


Figure. 4.24: VGG16 1000 steps of epoch and 10 epoch

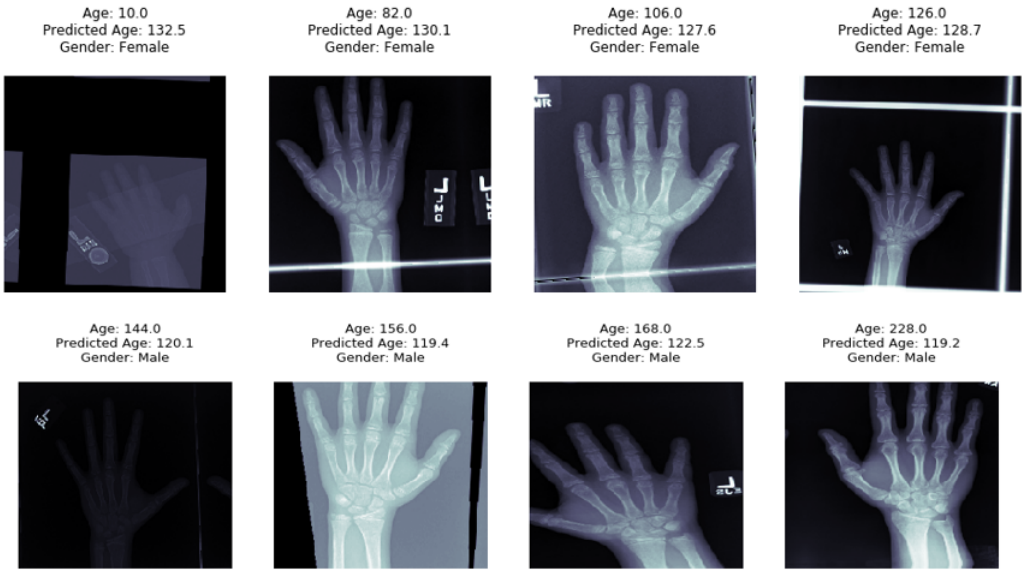


Figure. 4.25: VGG16 10 steps of epoch and 20 epoch

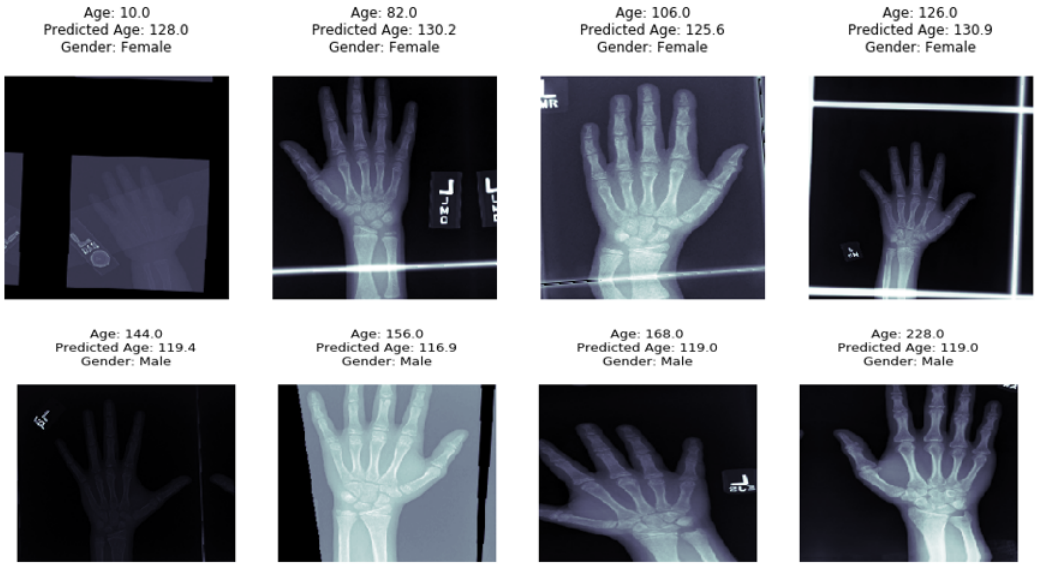


Figure. 4.26: VGG16 20 steps of epoch and 20 epoch

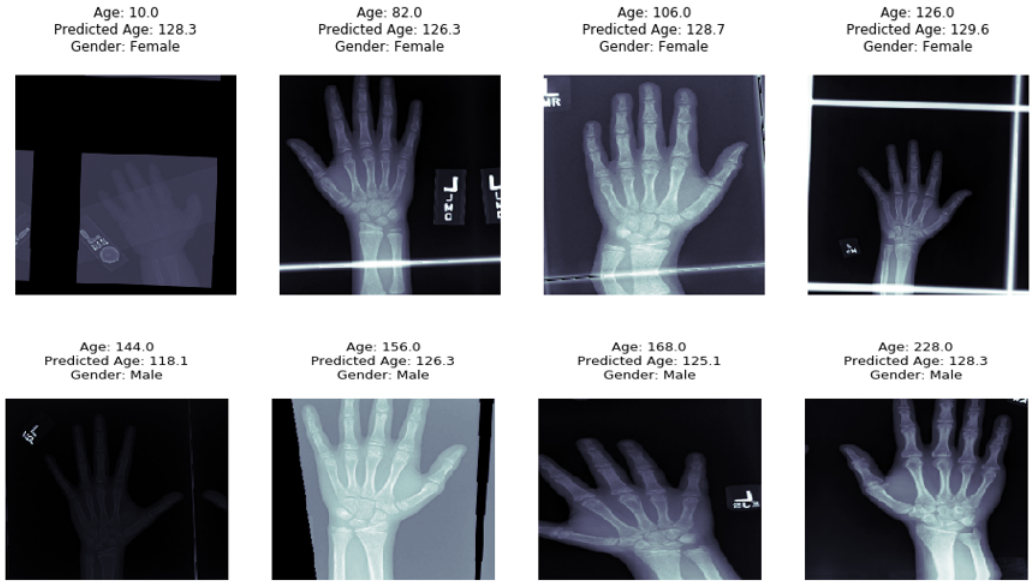


Figure. 4.27: VGG16 100 steps of epoch and 20 epoch

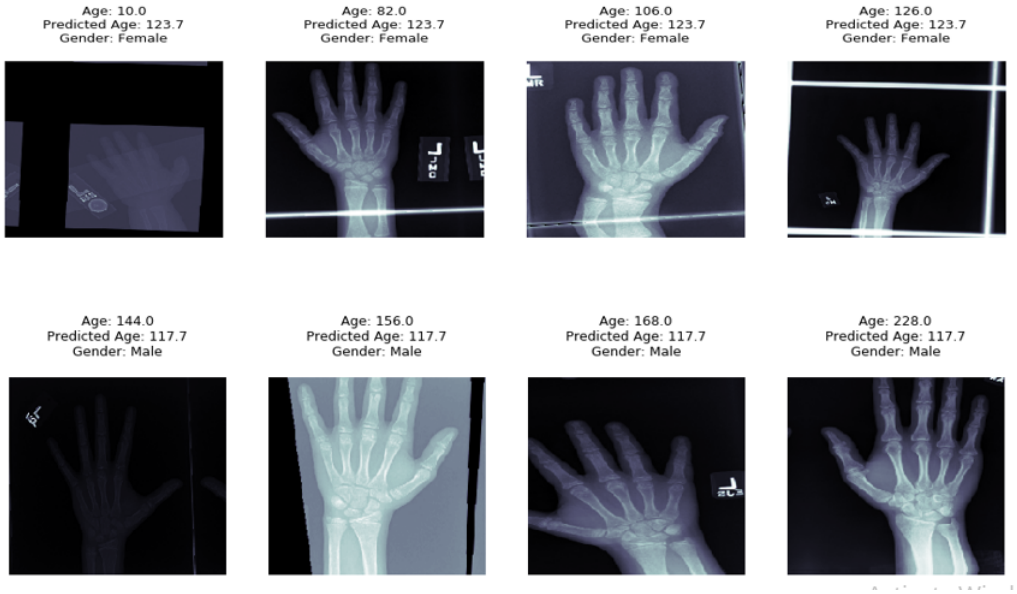
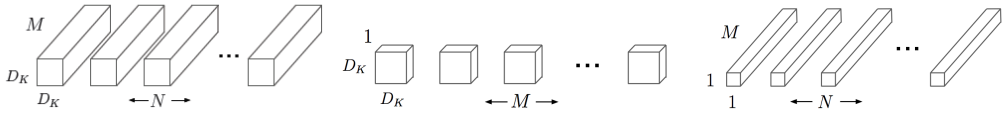


Figure. 4.28: VGG16 1000 steps of epoch and 20 epoch

4.4 MobileNet

4.4.1 MobileNet concept

Mobile Nets a class of lightweight deep convolutional that are famous for its short size and take less time for performance than other well-known models Classification, detection and other common tasks that convolutional neural networks are known for can be implemented by mobile nets models because of it's a class of small low latency low power models. That is the reason why mobile nets are good choice to apply deep learning models for mobile devices [21]. Compare to VGG16, mobile nets are 32



(a) Standard Convolution Filters (b) Depthwise Convolution Filters (c) Pointwise Convolution Filter

Figure. 4.29: Standard Convolution Filter and MobileNet Convolution Filter [21]

times smaller. The largest mobile nets models are 17MB on the other hand VGG16 553 MB in average size of course it is large difference especially when it is for mobile or running in a browser. The difference is for the parameters that are used in these networks. VGG16 used 138million parameters where mobile nets run on only 4.2 million parameters. But mobile nets are not as accurate as the other large models by the way it performs very well with smaller size. For mobile nets a single filters is applied by the depth wise convolution to each inputs then applies a 1×1 convolution to combine the outputs the depthwise convolution. For A standard convolution, the depthwise splits two layers of a separate layer for filtering and a separate layer for combining. Depthwise separable convolution are made up of two layers: depthwise convolutions and pointwise convolutions. Apply a single filter per each input to implement depthwise convolution and 1X1 convolution for pointwise. Suppose we have a convolutional kernel size of D_k×D_k×M ,so the depthwise convolution will have a computation cost of:

$$D_k.D_k.M.D_f.D_f \tag{4.1}$$

Depthwise convolution is very much same as standard convolution. The limitation is it can only filter input channels , unable to create new feature by combine them. For that we have to add an extra layer that figure a linear combination of the output of depthwise convolution.This is the pointwise convolution.After applying the depthwise convolution, the pointwise convolution is being applied. Then the two convolution is just summed up and the equation becomes

$$D_k.D_k.M.D_f.D_f + M.N.D_f.D_f \tag{4.2}$$

Whereas the standard convolution cost is

$$D_k.D_k.M.N.D_f.D_f \tag{4.3}$$

For this reason MobileNet performs faster and efficiently than the other pre-trained models.

4.4.2 Applying MobileNet in our dataset

Just like the InceptionV3 and ResNet,MobileNet has also shown different results for steps per epochs and epochs number.

Table 4.7: MobileNet 10 epochs result

Steps per epochs	Mean absolute error	Explained Variance Score	Value loss
10	21.31	0.63	0.4689
20	17.68	0.70	0.3065
100	11.30	0.88	0.1274
1000	9.47	0.91	0.0901

Table 4.8: MobileNet 20 epochs result

Steps per epochs	Mean absolute error	Explained Variance Score	Value loss
10	23.55	0.53	0.5407
20	16.55	0.76	0.2620
100	11.45	0.90	0.1197
1000	8.87	0.92	0.0809

Figure 4.29-4.26 shows the output of the MobileNet model.

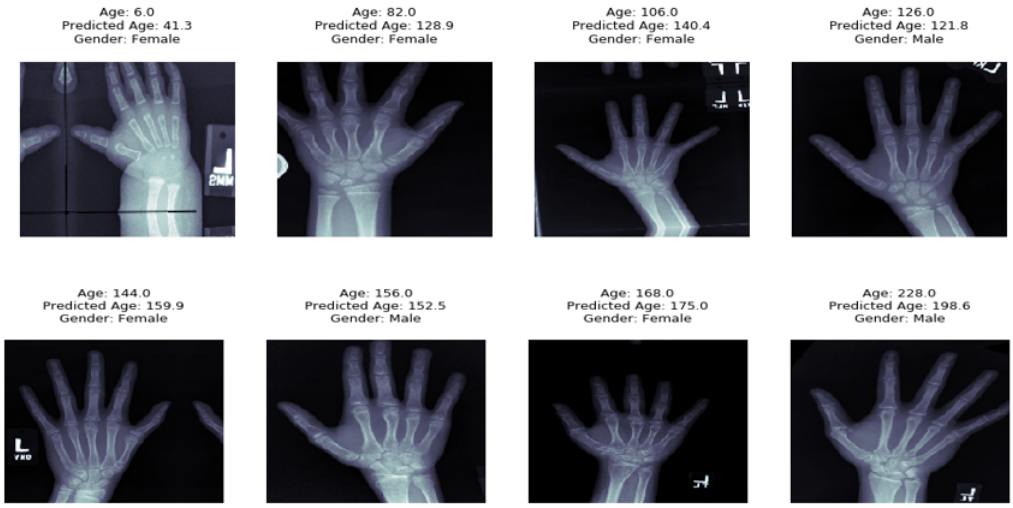


Figure. 4.30: MobileNet 10 steps of epoch and 10 epoch

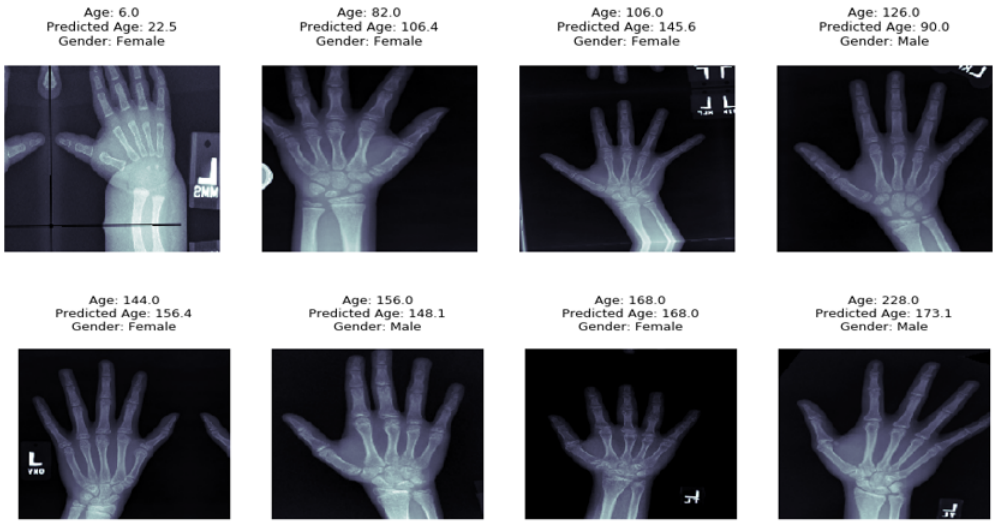


Figure. 4.31: MobileNet 20 steps of epoch and 10 epoch

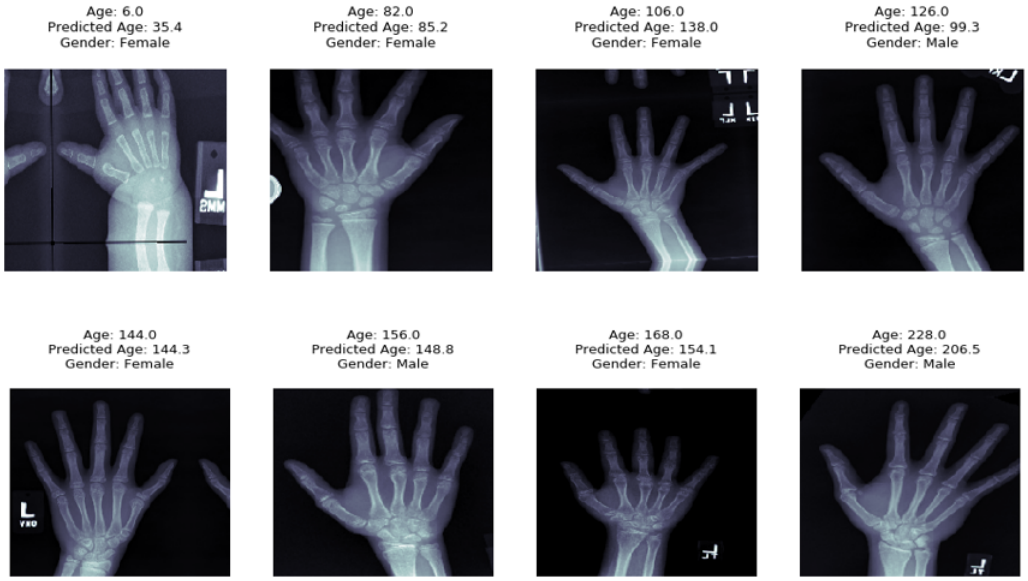


Figure. 4.32: MobileNet 100 steps of epoch and 10 epoch

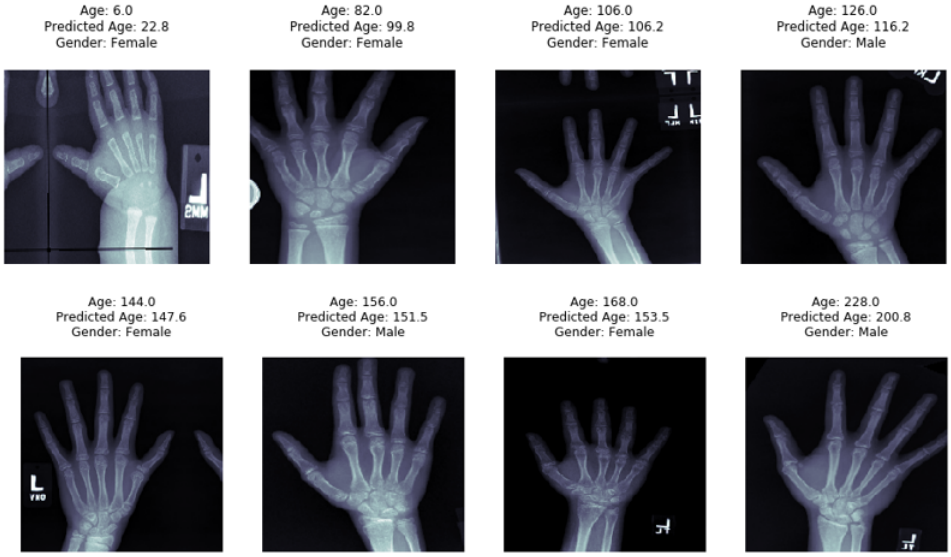


Figure. 4.33: MobileNet 1000 steps of epoch and 10 epoch

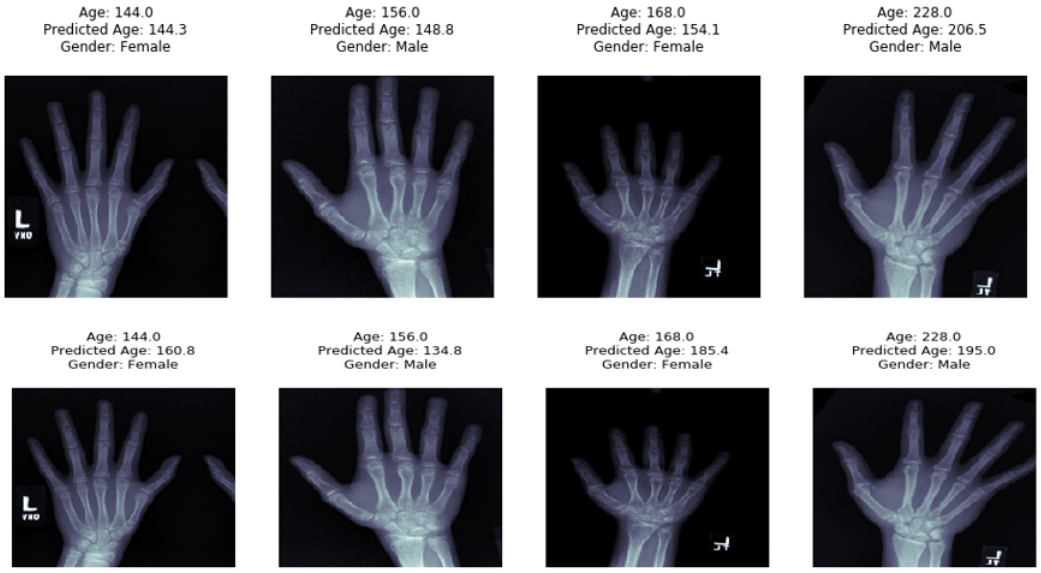


Figure. 4.34: MobileNet 10 steps of epoch and 20 epoch

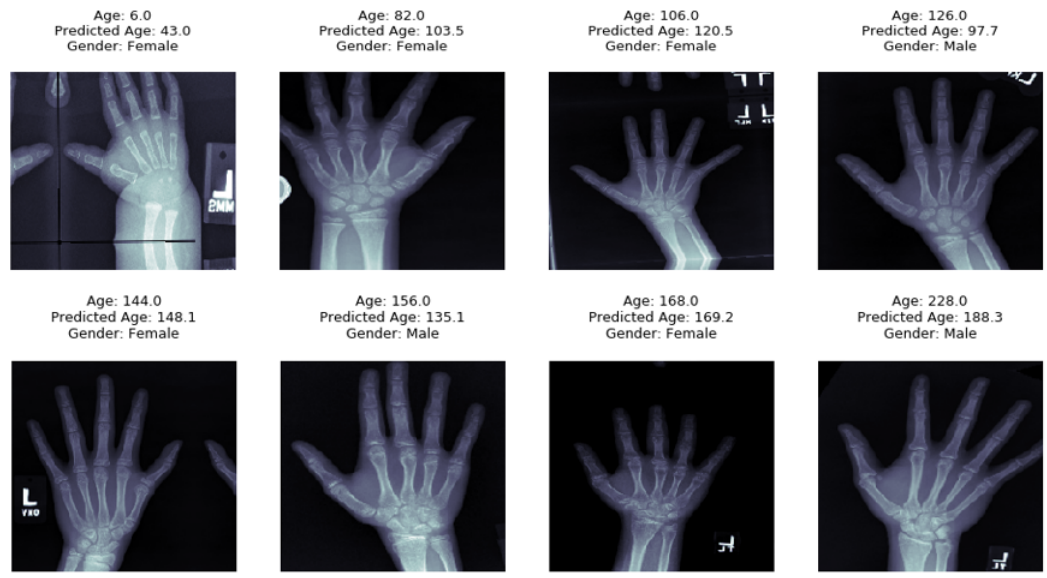


Figure. 4.35: MobileNet 20 steps of epoch and 20 epoch

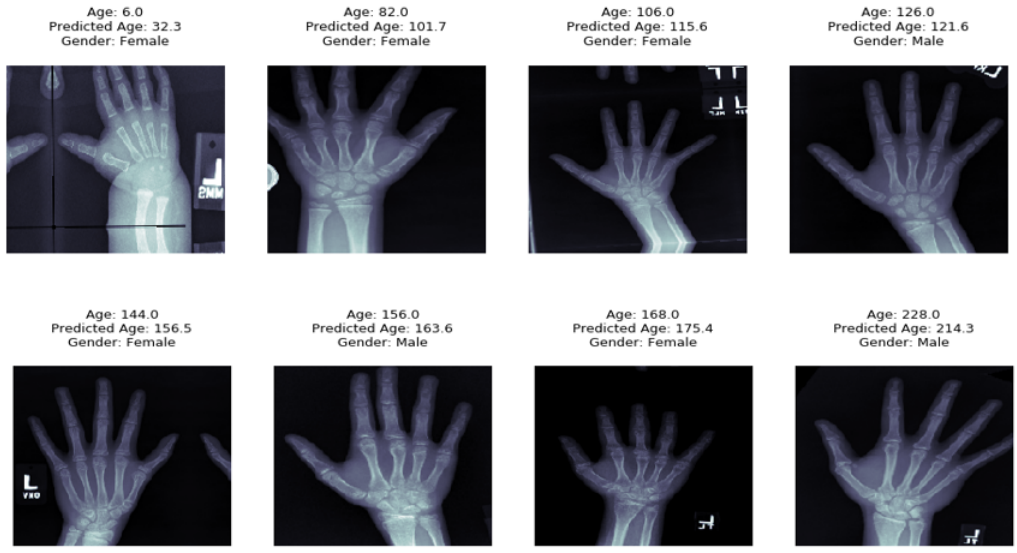


Figure. 4.36: MobileNet 100 steps of epoch and 20 epoch

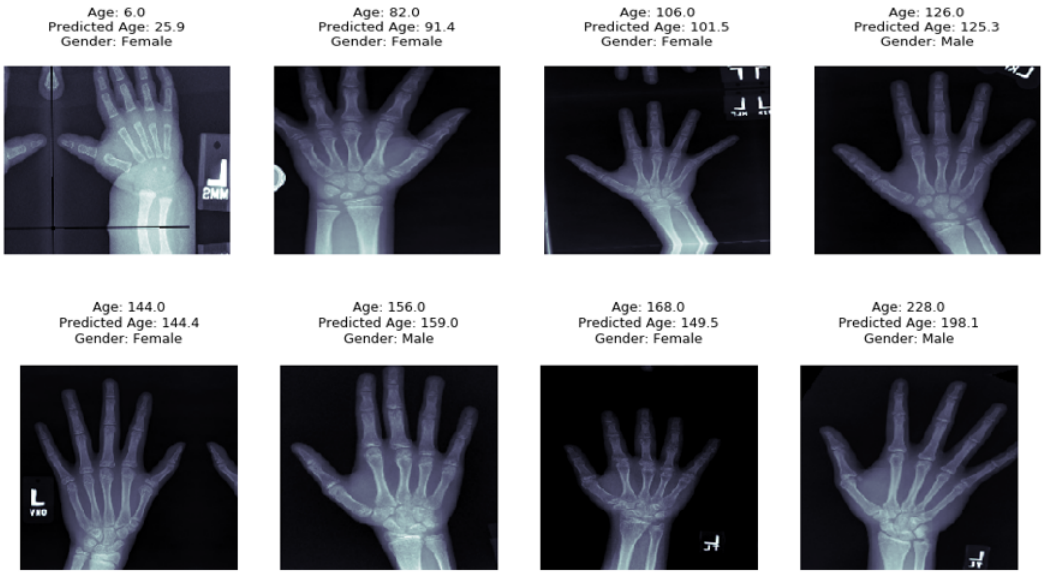


Figure. 4.37: MobileNet 1000 steps of epoch and 20 epoch

4.5 Comparison of CNN Models

VGG16 gave the less accuracy among this four pre trained models. From table 4.10 we can see than VGG16 has less parameters than InceptionV3 and ResNet and the convolution,concatenation and dense 2 parameters is lesser than other three models which cause a and the shape of the convolution and concatenation layer is also smaller which results in huge data loss and hamper the accuracy of the model as we know the more parameters can be trained in CNN,the more accurate result it can give.

InceptionV3 and ResNet gave same kind of result.The value loss is almost similar and the output shape is also similar. ResNet have more parameters than the InceptionV3 model, however the non-trainable parameters are higher than the non-trainable parameters of InceptionV3. So it cause InceptionV3 to perform slightly a better result. But if we increase the steps per epochs and the number of epochs the result may vary.

MobileNet gave the best accuracy in our dataset as we can see it from Table 4.9. It has a value loss of 0.08090 and the accuracy is 91.13%.The main reason behind this is depthwise convolution layers,the other CNN models are standard CNN models,so MobileNet performs well even if has less parameters than the other three models.This is mainly the speciality of depthwise models,it performs well with the lower parameters. also the convolution shape and the pooling layer shape is similar which is causing less data loss. So it is giving comparatively better result then the other models. Figure 4.38-4.39 is the graphical representation of the comparison between InceptionV3,ResNet50,VGG16 and MobileNet.

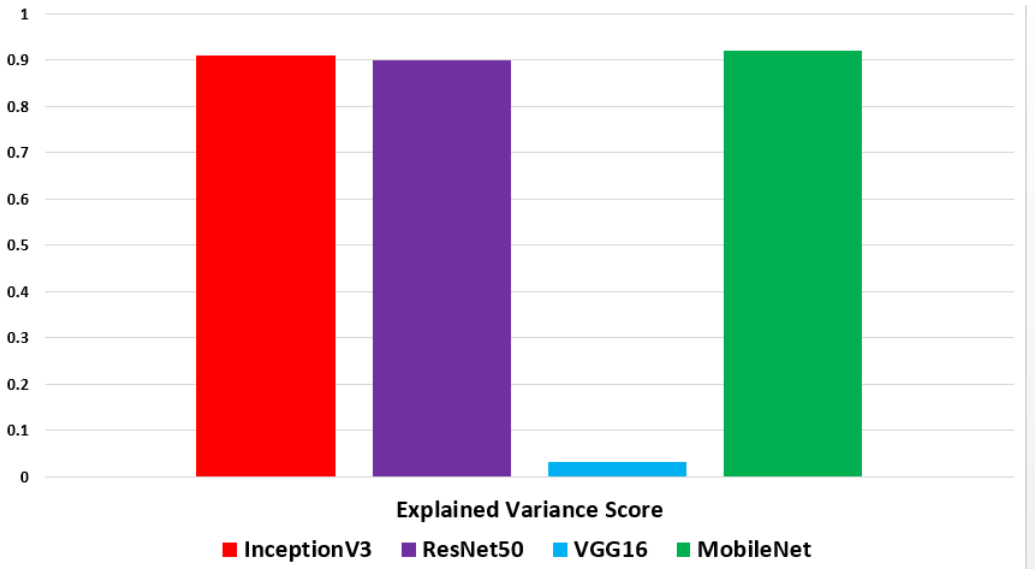


Figure. 4.38: Explained variance score of the pre-trained models

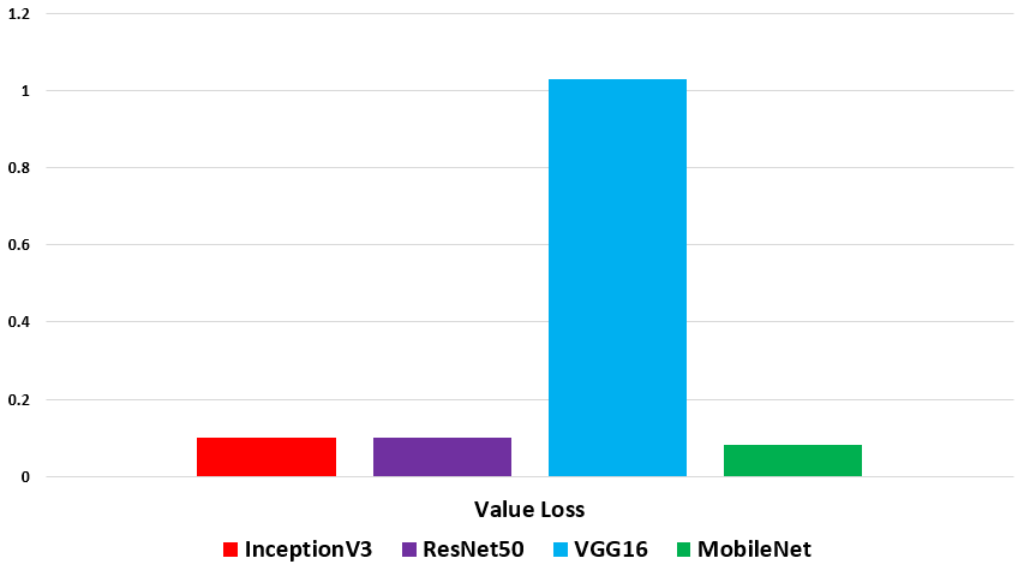


Figure. 4.39: Value loss of the pre-trained models

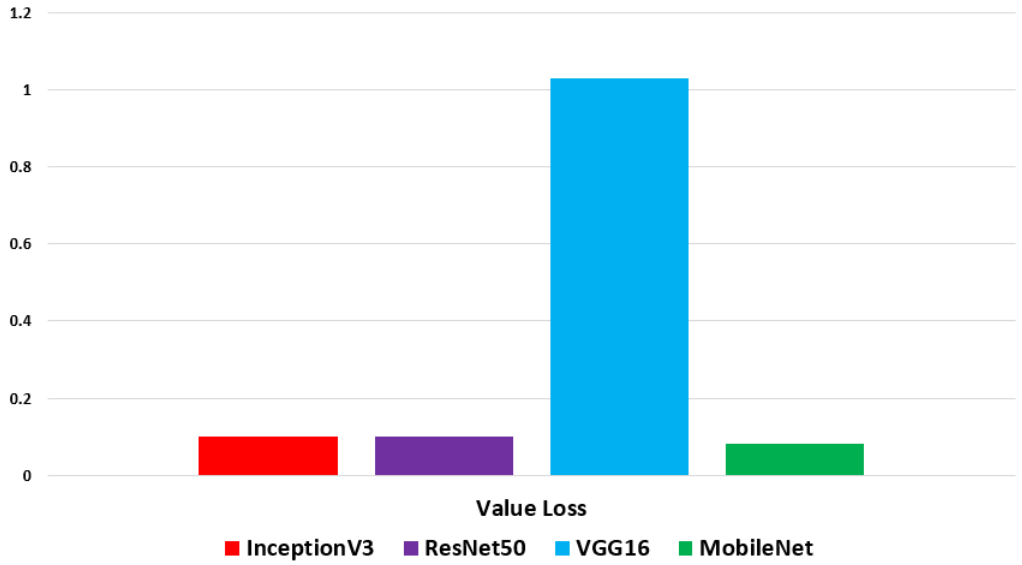


Figure. 4.40: Mean absolute error of the pre-trained models

Table 4.9: Comparison of the best result

Model Name	Mean absolute error	Explained Variance Score	Value loss
InceptionV3	10.10	0.91	0.1002
ResNet50	10.65	0.90	0.1013
VGG16	32.58	0.032	1.0281
MobileNet	8.87	0.92	0.0809

Table 4.10: Output shape and the parameters

Items	InceptionV3	ResNet50	VGG16	MobileNet
Input Shape	224,224,3	224,224,3	224,224,3	224,224,3
Model Convolution Shape	5,5,2048	7,7,2048	7,7,512	7,7,1024
Model Convolution Parameters	21802784	23587712	14714688	3228864
Pooling Shape	2048	2048	512	1024
Dense Gender shape	32	32	32	32
Dense Gender Parameters	64	64	64	64
Concatenate Output Shape	2080	2080	544	1056
Dense 2 Shape	1024	1024	544	1024
Dense 2 Parameters	2130944	2130944	558080	1082368
Dense 2 Shape	1024	1024	1024	1024
Dense 2 Parameters	1049600	1049600	1049600	1049600
Total Parameters	24984417	26769345	16323457	5362921
Trainable Parameters	24949985	26712225	16323457	5340033
Non-trainable Parameters	34432	53120	0	21888

4.6 MobileNet without the gender feature

As MobileNet gives the best result among the four pre-trained models in keras, we removed the gender feature from it to check if it changes the result. As we can see from table 4.11 and 4.12 the value loss has been increased from 0.08090 to 0.1246 and the accuracy has been decreased from 91.13% to 87.44%. From Figure 4.41-4.42 we can see how gender feature is important for more accurate result. By this analysis, we can say that gender feature have a impact on the accuracy of the model.

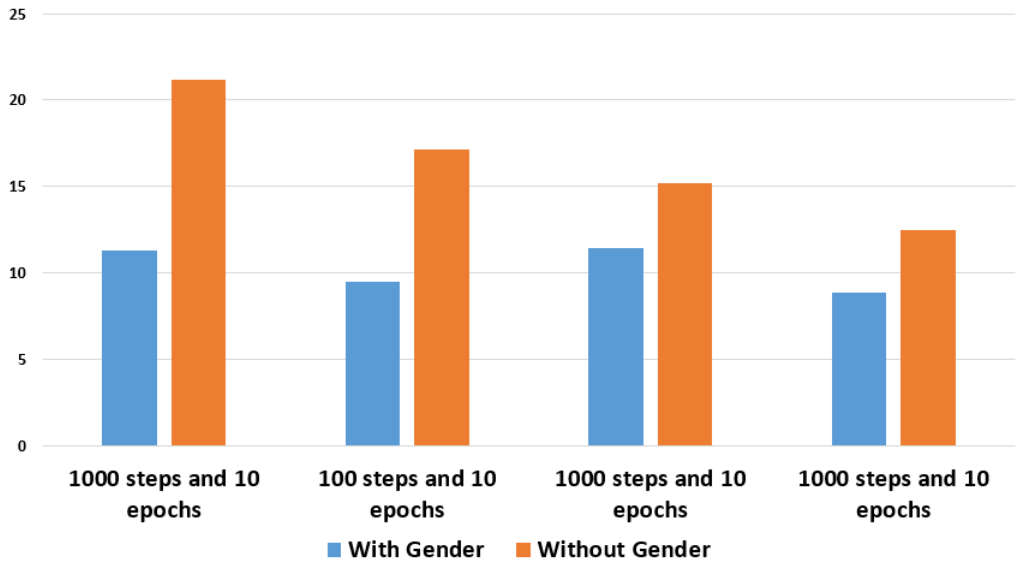


Figure. 4.41: Mean absolute error of MobileNet with and without gender feature

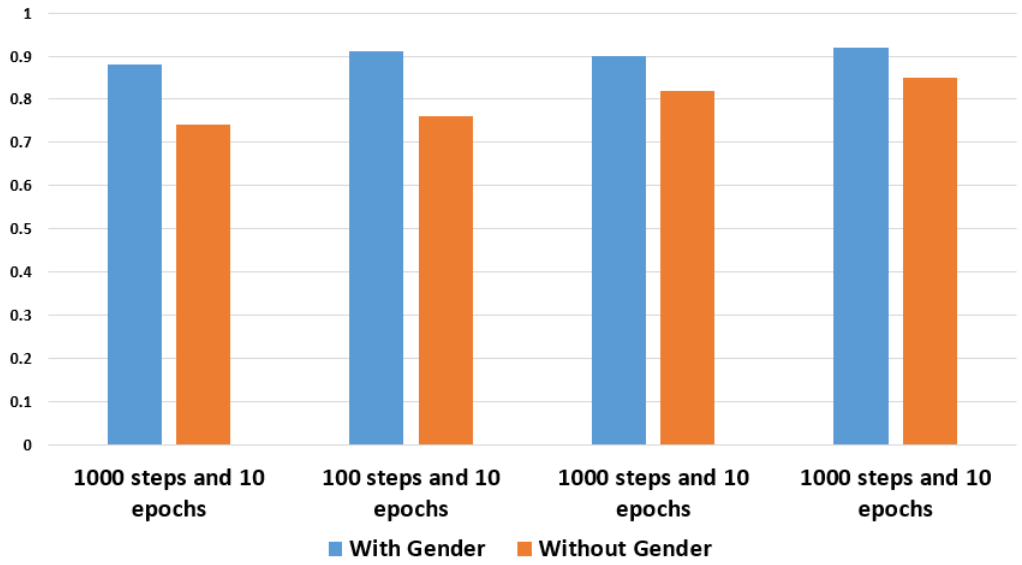


Figure. 4.42: Explained variance score of MobileNet with and without gender feature MobileNet

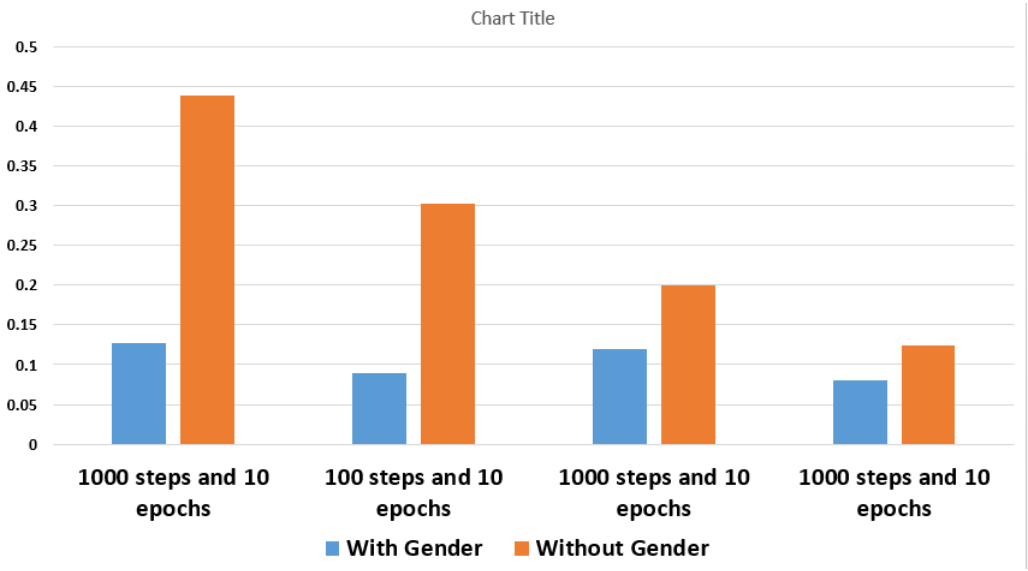


Figure. 4.43: Value Loss of MobileNet with and without gender feature

Table 4.11: MobileNet 10 epochs result without gender feature

Steps per epochs	Mean absolute error	Explained Variance Score	Value loss
100	21.15	0.74	0.4391
1000	17.11	0.757	0.3031

Table 4.12: MobileNet 20 epochs result without gender feature

Steps per epochs	Mean absolute error	Explained Variance Score	Value loss
100	15.21	0.823	0.2003
1000	12.44	0.85	0.1246

The figure 4.38-4.41 shows the output of the MobileNet without the gender feature.

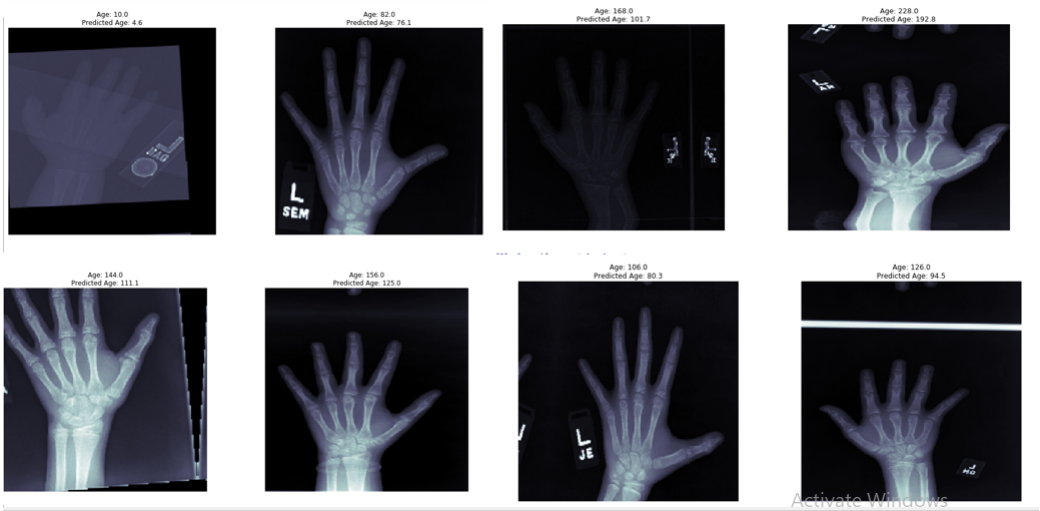


Figure. 4.44: MobileNet 100 steps of epoch and 10 epoch without gender

Figure. 4.45: MobileNet 1000 steps of epoch and 10 epoch without gender

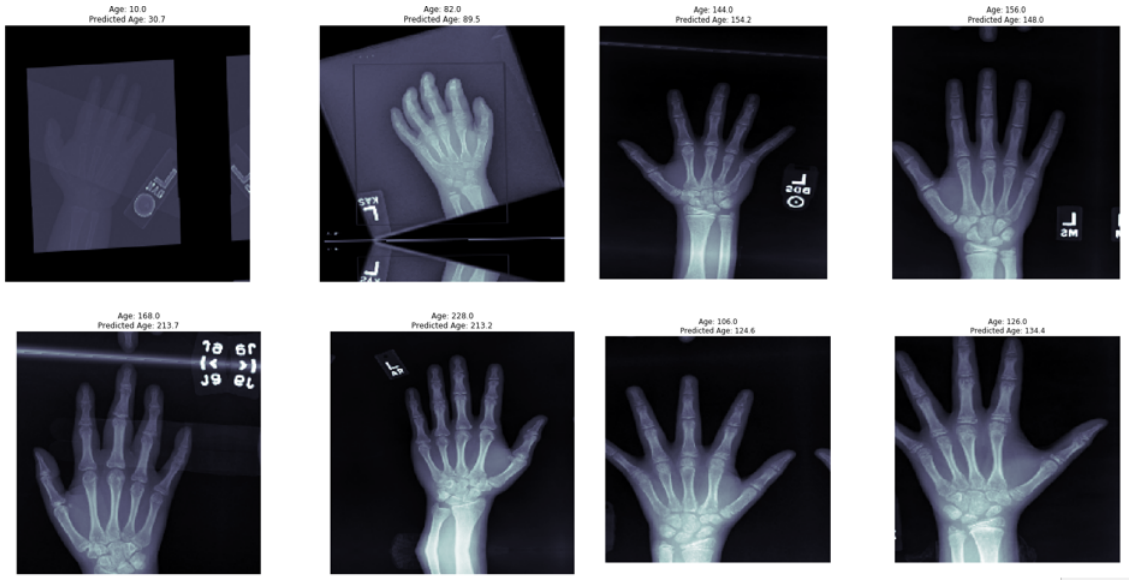


Figure. 4.46: MobileNet 100 steps of epoch and 20 epoch without gender

Figure. 4.47: MobileNet 1000 steps of epoch and 20 epoch without gender

Chapter 5

Conclusion

The focus of this thesis was to find a CNN model which works better with our current dataset. While researching for our thesis, we have found many great works in image recognition and classification which helped to guide us. By doing this thesis we came to understand how each model performs and the difference between them. VGG16 gave a low output and also the value loss is huge compare to MobileNet and the computation time of MobileNet is faster than VGG16. This is not the final deduction as there are many different changes can be made to this model like changing the dropout rate, increasing steps of epochs etc and we have only implemented four pre-trained models and there are other models that can be implemented to this dataset. Due to some limitations like low powered computer and unclear image the issue for getting less accuracy could be arise. Our Another concern is CNN requires a huge amount of time for training. We have predicted according to a fix dataset so it is very difficult to say we will get same amount of accuracy on the other data. Moreover, the bone-age images are all of the children's, so for the adult bone age,the models can give a wrong result.

5.1 Future Possibilities

Predicting bone age with the help of CNN system has a lot of opportunities to improve. As the world is changing fast, the way of solving any problem in automated method are becoming smarter and more efficient. If we want to cope up with the need of vast amount of people, many things need to be updated. We have tried only one type of algorithm which is CNN. CNN mainly extracts the feature automatically. We can try to extract the features in a manual process and can run it with other types of machine learning algorithms to check the performance.
In our work, VGG16 is giving the less accurate result. To overcome this problem our future work will be applying NASNetLarge and InceptionResNetV2 which are currently giving the highest accuracy on on the ImageNet validation dataset. Keras also provide to create own CNN model, so we will try to make our own CNN model and check if it is working well for our dataset.

References

[1] "Image preprocessing - keras documentation", *https://keras.io/preprocessing/image/*, Accessed on 03/12/2019.

[2] "Convolution - a tutorial with definition, interactive examples and properties", *http://www.onmyphd.com/?p=convolution*, (Accessed on 03/27/2019).

[3] C.-T. Lin and C. S. G. Lee, "Neural-network-based fuzzy logic control and decision system", *IEEE Transactions on computers*, vol. 40, no. 12, pp. 1320–1336, 1991.

[4] T. Roska and L. O. Chua, "The cnn universal machine: An analogic array computer", *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 40, no. 3, pp. 163–173, 1993.

[5] E. Pietka, A. Gertych, S. Pospiech, F. Cao, H. Huang, and V. Gilsanz, "Computer-assisted bone age assessment: Image preprocessing and epiphyseal/metaphyseal roi extraction", *IEEE transactions on medical imaging*, vol. 20, no. 8, pp. 715–729, 2001.

[6] R. Mary, "What is image processing : Tutorial with introduction, basics, types applications", *https://www.engineersgarage.com/articles/image-processing-tutorial-applications*, 2011.

[7] D. Cireşan, U. Meier, and J. Schmidhuber, "Multi-column deep neural networks for image classification", *arXiv preprint arXiv:1202.2745*, 2012.

[8] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks", pp. 1097–1105, 2012.

[9] G. Tur, L. Deng, D. Hakkani-Tür, and X. He, "Towards deeper understanding: Deep convex networks for semantic utterance classification", pp. 5045–5048, 2012.

[10] L. Deng, G. Hinton, and B. Kingsbury, "New types of deep neural network learning for speech recognition and related applications: An overview", pp. 8599–8603, 2013.

[11] M. Mansourvar, S. A. Kareem, M. A. Ismail, and F. H. Nasaruddin, "Automatic method for bone age assessment based on combined method", pp. 1–5, 2014.

[12] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition", *arXiv preprint arXiv:1409.1556*, 2014.

[13] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting", *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[14] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions", pp. 1–9, 2015.

[15] H. Greenspan, B. Van Ginneken, and R. M. Summers, "Guest editorial deep learning in medical imaging: Overview and future promise of an exciting new technique", *IEEE Transactions on Medical Imaging*, vol. 35, no. 5, pp. 1153–1159, 2016.

[16] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition", pp. 770–778, 2016.

[17] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision", pp. 2818–2826, 2016.

[18] X. Zhang, J. Zou, K. He, and J. Sun, "Accelerating very deep convolutional networks for classification and detection", *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 10, pp. 1943–1955, 2016.

[19] "Activation functions in neural networks – towards data science", *https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6*, Sep. 2017.

[20] S. Hershey, S. Chaudhuri, D. P. Ellis, J. F. Gemmeke, A. Jansen, R. C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, *et al.*, "Cnn architectures for large-scale audio classification", pp. 131–135, 2017.

[21] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications", *arXiv preprint arXiv:1704.04861*, 2017.

[22] H. Lee, S. Tajmir, J. Lee, M. Zissen, B. A. Yeshiwas, T. K. Alkasab, G. Choy, and S. Do, "Fully automated deep learning system for bone age assessment", *Journal of digital imaging*, vol. 30, no. 4, pp. 427–441, 2017.

[23] L. Lu, Y. Liang, Q. Xiao, and S. Yan, "Evaluating fast algorithms for convolutional neural networks on fpgas", pp. 101–108, 2017.

[24] "Rsna pediatric boneage challenge", *http://rsnachallenges.cloudapp.net/competitions/4#learn_the_details-news*, Oct. 2017.

[25] J. Russell, "Google's alphago ai wins three-match series against the world's best go player", *Retrieved November*, vol. 5, p. 2018, 2017.

[26] S. Sharma, *Activation functions in neural networks – towards data science*, (Accessed on 03/27/2019), Sep. 2017.

[27] C. Spampinato, S. Palazzo, D. Giordano, M. Aldinucci, and R. Leonardi, "Deep learning for automated skeletal bone age assessment in x-ray images", *Medical image analysis*, vol. 36, pp. 41–51, 2017.

[28] J. Zhou, Z. Li, W. Zhi, B. Liang, D. Moses, and L. Dawes, "Using convolutional neural networks and transfer learning for bone age classification", pp. 1–6, 2017.

[29] "Common architectures in convolutional neural networks.", *https://www.jeremyjordan.me/convnet-architectures/*, Apr. 2018.

[30] J. P. Dominguez-Morales, A. F. Jimenez-Fernandez, M. J. Dominguez-Morales, and G. Jimenez-Moreno, "Deep neural networks for the recognition and classification of heart murmurs using neuromorphic auditory sensors", *IEEE transactions on biomedical circuits and systems*, vol. 12, no. 1, pp. 24–34, 2018.

[31] V. I. Iglovikov, A. Rakhlin, A. A. Kalinin, and A. A. Shvets, "Paediatric bone age assessment using deep convolutional neural networks", pp. 300–308, 2018.

[32] "Rsna bone age — kaggle", *https:// www.kaggle.com/ kmader/ rsna-bone-age*, Jan. 2018.

[33] S. Wang, Y. Shen, C. Shi, P. Yin, Z. Wang, P. W.-H. Cheung, J. P. Y. Cheung, K. D.-K. Luk, and Y. Hu, "Skeletal maturity recognition using a fully automated system with convolutional neural networks", *IEEE Access*, vol. 6, pp. 29 979–29 993, 2018.

[34] S. Wang, Y. Shen, D. Zeng, and Y. Hu, "Bone age assessment using convolutional neural networks", pp. 175–178, 2018.

[35] "Convolutional neural network – towards data science", *https:// towardsdatascience.com/ covolutional-neural-network-cb0883dd6529*, Feb. 2019.

[36] "Deep learning tutorial — ai using deep learning — edureka", *https : // www . edureka . co / blog / deep - learning - tutorial ? utm_source = youtube & utm_campaign=deep-learning-180717-wr&utm_medium=description*, Feb. 2019.

[37] "Merge layers - keras documentation", *https:// keras.io/ layers/ merge/*, 2019.