1. **Introduction**

A Skeletal bone age assessment is a procedure used in pediatric radiology for both diagnostic and therapeutic investigations of endocrinology problems, children growth and genetic disorders.

A bone age study helps doctors estimate the maturity of a child's skeletal system. It's usually done by taking a single X-ray of the left wrist, hand, and fingers. It is a safe and painless procedure that uses a small amount of radiation. The bones on the X-ray image are compared with X-rays images in a standard atlas of bone development, which is based on data from large numbers of other kids of the same gender and age. The bone age is measured in years.

A difference between a child's bone age and his or her chronological age might indicate a growth problem. Therefore it is important to determine the bone age

As we know, deep learning has been applied to computer vision tasks and achieved drastic performance improvement. We propose a method which learns real latent features of hand X-ray images and facilitates the feature capture to perform Skeletal Age Determination.

**1.1 Problem Statement**Traditional bone age assessment methods normally require doctors to physically examine the x-ray with reference to a huge dataset of x-rays to look for even the smallest discrepancies. This is a very tedious task that requires a huge amount of time by the doctor.Since, the discrepancies in the X-rays are what helps us determine the bone age and since these discrepancies are really small, the margin of error is relatively high. This means that the results obtained from traditional methods have a high chance of being inaccurate.

1. **Literature Survey**

| No. | TITLE | USAGE | RESULTS |
| --- | --- | --- | --- |
| 1 | SUNG JOON SON , YOUNGMIN SONG , NAMGI KIM , YOUNGHAE DO , NOJUN KWAK , (Member, IEEE), MU SOOK LEE, AND BYOUNG-DAI LEE  ***TW3-Based Fully Automated Bone Age Assessment System Using Deep Neural Networks***  Available: [https://ieeexplore.ieee.org/document/8660640?denie0d=](https://ieeexplore.ieee.org/document/8660640?denied=) | Proposes a complete end-to-end BAA system to automate the entire process of the Tanner–Whitehouse 3 method, starting from localization of the epiphysis–metaphysis growth regions within 13 different bones and ending with estimation of the corresponding BA. Specific modifications to the CNNs and other stages are proposed to improve results. | The proposed system is shown to outperform a commercially available Greulich–Pyle-based system, demonstrating the potential for practical clinical use. |
| 2 | Wei Tang∗ , Gang Wu† , Gang Shen∗ ∗School of Software Engineering †Tongji Hospital Huazhong University of Science and Technology, Wuhan, China  ***“Improved Automatic Radiographic Bone Age Prediction with Deep Transfer Learning”***  Available: [ttps://ieeexplore.ieee.org/document/8965906/authors#authorsh](https://ieeexplore.ieee.org/document/8965906/authors#authors) | This paper uses a high definition dataset to apply a CNN on in order to come up with a model that accurately predicts skeletal age | This system essentially uses the same classification methods that are commonly used (TW method),but, the unique datasets used here are of significantly higher resolution than the traditional RSNA dataset. The results obtained in this system confirm the fact that images of higher resolution produce more accurate results even if the sample dataset is relatively smaller. |
| 3 | Monika Pahuja, M.Tech (scholar),  Dr. Naresh Kumar Garg (HOD)  Department of Computer Science Engineering GianiZail,  Singh Campus College of Engineering and Technology,  Bathinda, Punjab  ***“Skeleton Bone Age Assessment Using Optimized Artificial Neural Network”***  Available:  [https://nebula.wsimg.com/c03972155fbf4db9b87a37978fc0c1dd?AccessKeyId=DFB1BA3CED7E7997D5B1&dispositiworigin=1](https://nebula.wsimg.com/c03972155fbf4db9b87a37978fc0c1dd?AccessKeyId=DFB1BA3CED7E7997D5B1&disposition=0&alloworigin=1)  [on=0&all](https://nebula.wsimg.com/c03972155fbf4db9b87a37978fc0c1dd?AccessKeyId=DFB1BA3CED7E7997D5B1&disposition=0&alloworigin=1) | This paper focuses on algorithm development and testing on a database of 12,000 X-rays to determine the future height of an infant based on their sex. | Unique Feature Fetch with the help of feature  extraction algorithm and region of interest  calculated.    Implements the Optimized ANN method to classify  the knowledge base in the image processing. |

1. **Requirement Analysis**

**3.1 Functional Requirements**

In this research, as a convolutional network, the following modules are used respectively: a previously traced Convolution layer is considered as a black and white version. Also, CNNs are mostly used in classifying work; Bone Age Evaluation is a regression study. In this study, both the performance and the regression and classification method were evaluated to evaluate the best method. Both models have similar structure and protocols, and they differ only in the last two layers. This model contains a convolutional network with regression output.

**3.1.1 Hardware Requirements/Interface**

* Core i7-4790 or Ryzen 3 3200G.
* RTX 2080 Super, RTX 3070 (or RX 6800)
* 16GB RAM.
* 8GB VRAM.
* 50GB SSD storage.

**3.1.2 Software Requirements/Interface**

* Windows 10 64-bit.
* Python
* Anaconda, Jupyter Notebook (or) WebBrowser for Google Colab.
* In this system, there will be five libraries: KERAS, TENSORFLOW, PYPLOT in MATPLOTLIB, SIGMOID function from NUMPY and MATLAB.

**3.2 System Features**

This project plans to improve the BA prediction accuracy to the level of professional radiology doctors.

These techniques provide innovations in the design of deep neural networks by enabling deeper networks with a smaller number of weights, thus leading to higher classification accuracies while avoiding the overfitting problem.

The evaluation of skeletal bone age is the most clinical application for the study of endocrinology, genetic disorders and growth in young people.

This project examines the use of deep learning for medical imagery, especially for the evaluation of auto bone age using X-ray images.

**3.3 Non functional requirements**

**Reliability :** Reliability is one of the most important non functional requirements. Since this project is going to be used in the medical field, it is crucial that the application does not crash or perform poorly. In order to combat this, we have given the deep learning model a plethora of sample datasets to ensure there is no discrepancy in the application.

**Speed :** Speed is one of the major factors for the creation of this project, as this application is supposed to reduce time in calculating skeletal age. In order to achieve this the model is trained with plenty of datasets on a relatively powerful GPU to ensure fast and smooth operation

**Accuracy:** Accuracy is the other major factor next to speed which led to the creation of this project. To ensure accuracy, regions of interest of the X ray of the left hand are marked and then classified into sub categories based on these regions of interest. The input X ray is then given a score that will help the model classify it into the categories previously created. This ensures proper accuracy of the result obtained.

**Availability:** Since this project is GPU intensive, it’s availability is on a first come, first serve basis i.e the first X ray input will be analyzed first. This is due to the fact that only one X-ray can be analyzed at a time.

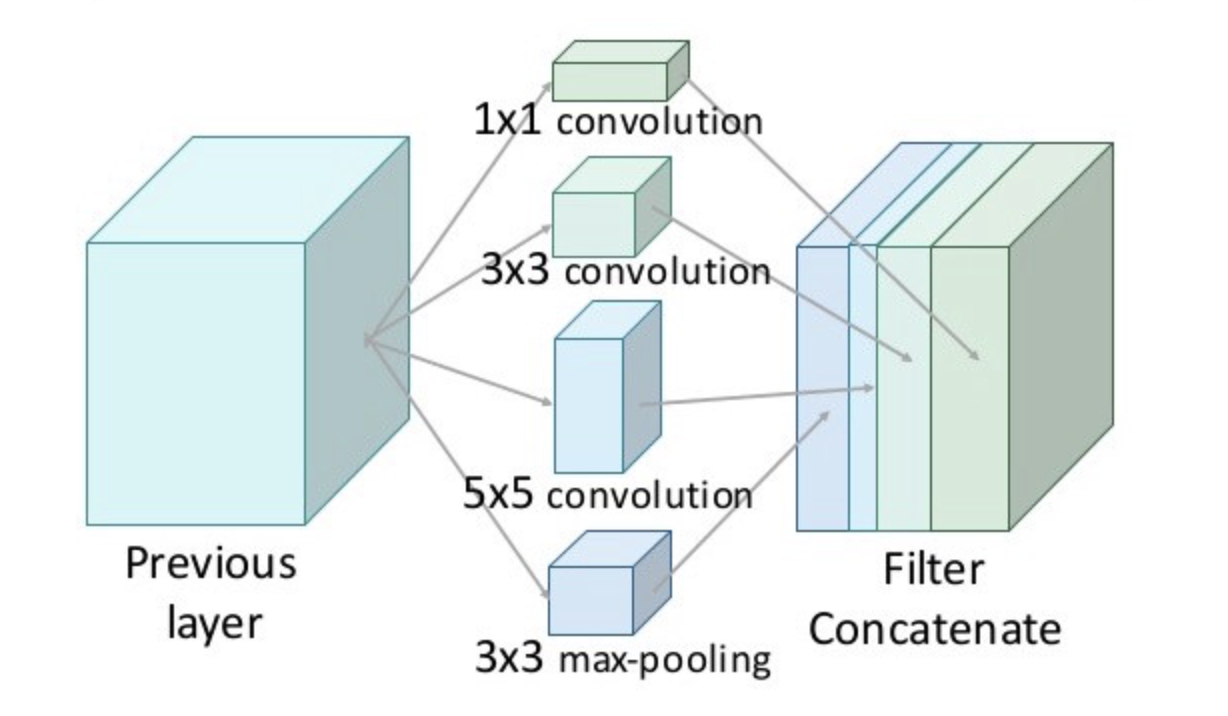
**Maintainability :** All the functions and features in this project are implemented in the form of small modules. Therefore, additional modules can easily be added to this if new features are to be implemented. If any changes are to be made to existing modules, these changes will not affect the other modules.

1. **Design Methods**

**4.1 Algorithm:**

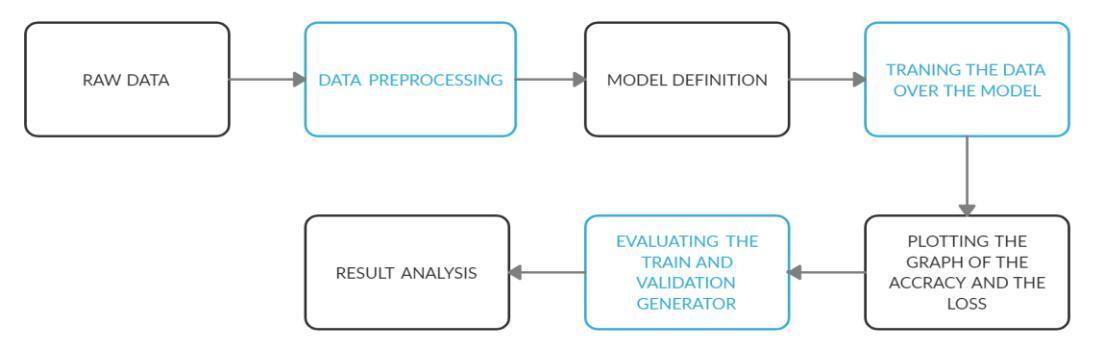
* Making a dataset by fetching data from multiple sources.
* Data preprocessing.
* Model definition.
* Training the data over the defined model.
* Draw out the accuracy and the loss plot of the train and validation datasets.
* Evaluate the train and validation generator.
* Result analysis.

**4.2 Architecture Diagram**

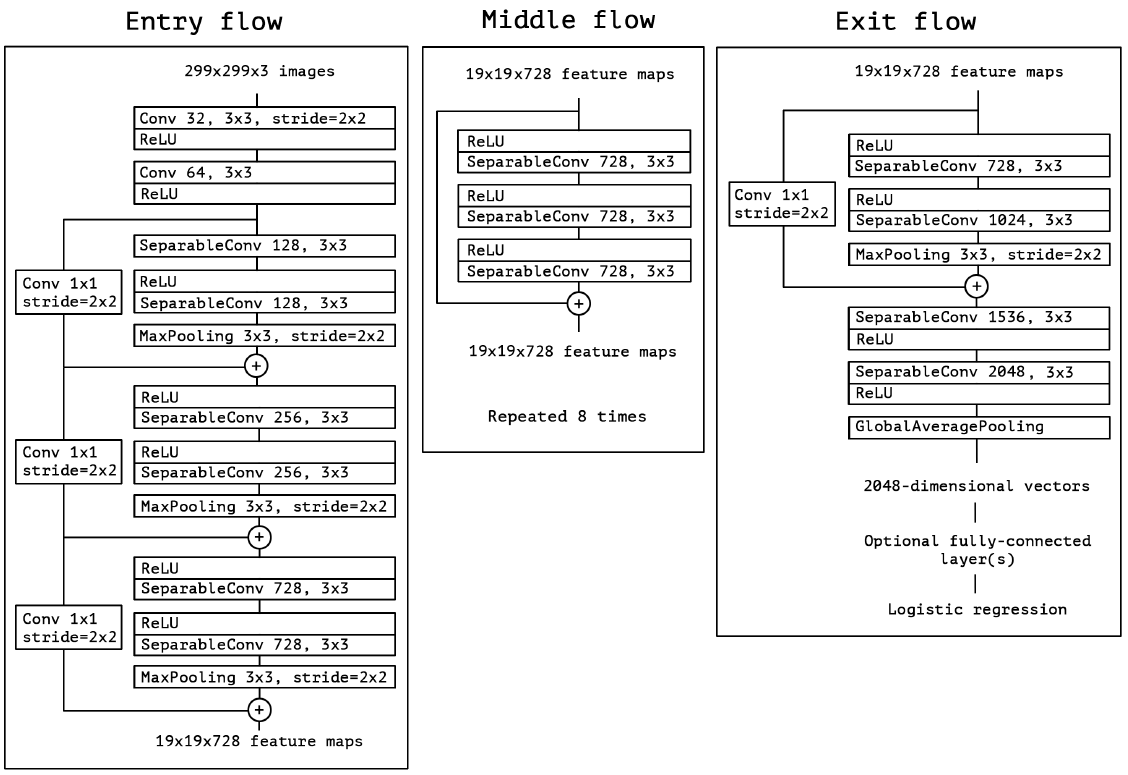
****

**Fig 4.2 Xception architecture diagram**

**4.3 Flowchart/DFD/UML Diagrams**

****

**Fig 4.3.1 Flowchart**



**Fig 4.3.2 Xception Architecture Flowchart**

1. **Project Breakdown**
2. **Creating the dataset by data preprocessing**

**The Dataset consists of two parts:**

● The first part consists of the labels for the dataset in the form of csv files

<https://www.kaggle.com/kmader/rsna-bone-age?select=boneage-test-dataset.csv>

<https://www.kaggle.com/kmader/rsna-bone-age?select=boneage-training-dataset.csv>

● The second part consists of the X-rays of the left hand in the form of png

<https://www.kaggle.com/kmader/rsna-bone-age?select=boneage-test-dataset>

<https://www.kaggle.com/kmader/rsna-bone-age?select=boneage-training-dataset>

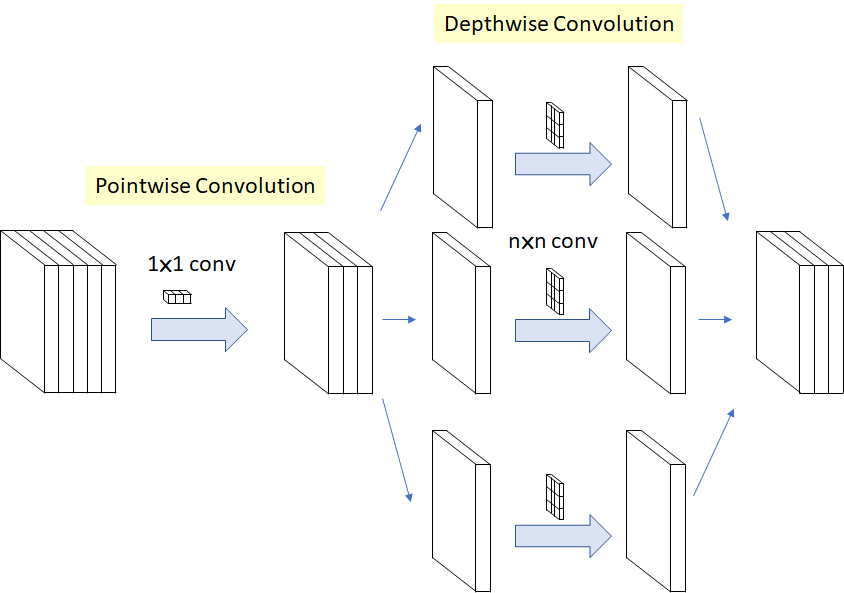
At first we take the training data in the form of **.png** and perform train-test split or evaluating the performance of a machine learning algorithm. ( 80% training data and 20% testing data.

Now we need to preprocess the training dataset. At first we need to check the split between the two sexes as boneage of a female is normally more than that of a male for a given age. We then calculate the mean and median ages in the dataset to determine which age occurs the most. This will help us focus on the model learning more from the most occurring age as the training data will be the largest here.

We then combine the information obtained above and split the dataset into 2 genders and find the most occurring age for both males and females individually so that we can focus the model to give more accurate results for each category(Male and Female).

After this, we need to convert all the pngs which are in grayscale format into RGB(A) format and then further transform them into pseudocolour format. This will help in feature detection of the hand.

1. **Training over Xception**



**Fig 5.2.1 Training over Xception Architecture**

The input is going to be x-ray of the left hand

Xception is the modified version of depthwise separable convolution where pointwise convolution is used followed by depthwise convolution

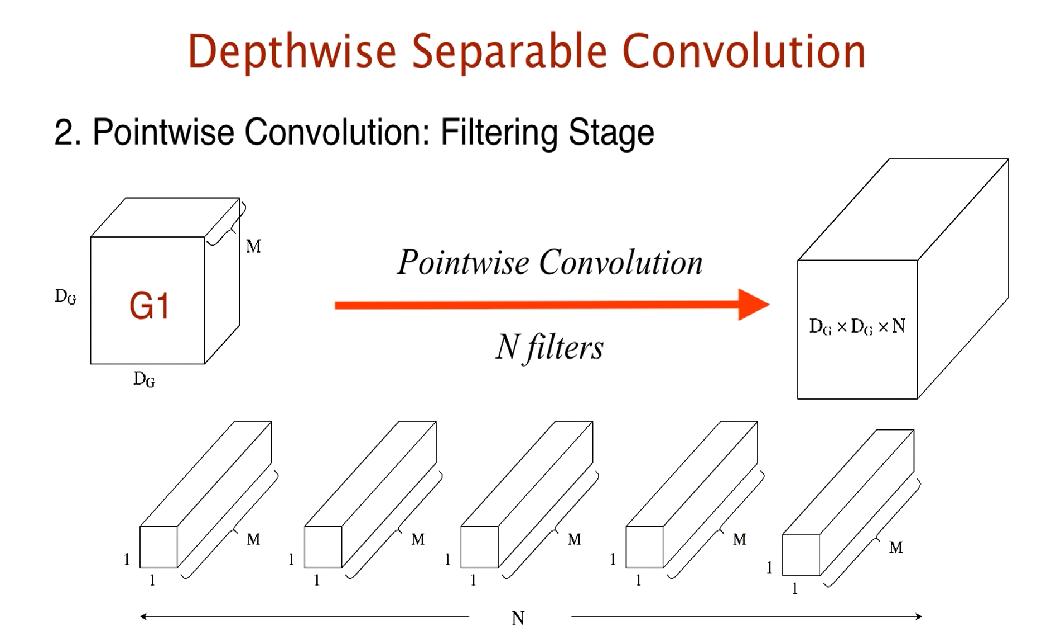
Depthwise separation is a computationally cheap alternative to traditional convolution that consists of 2 basic stages:

Depthwise Convolution

Pointwise Convolution

**Pointwise Convolution**

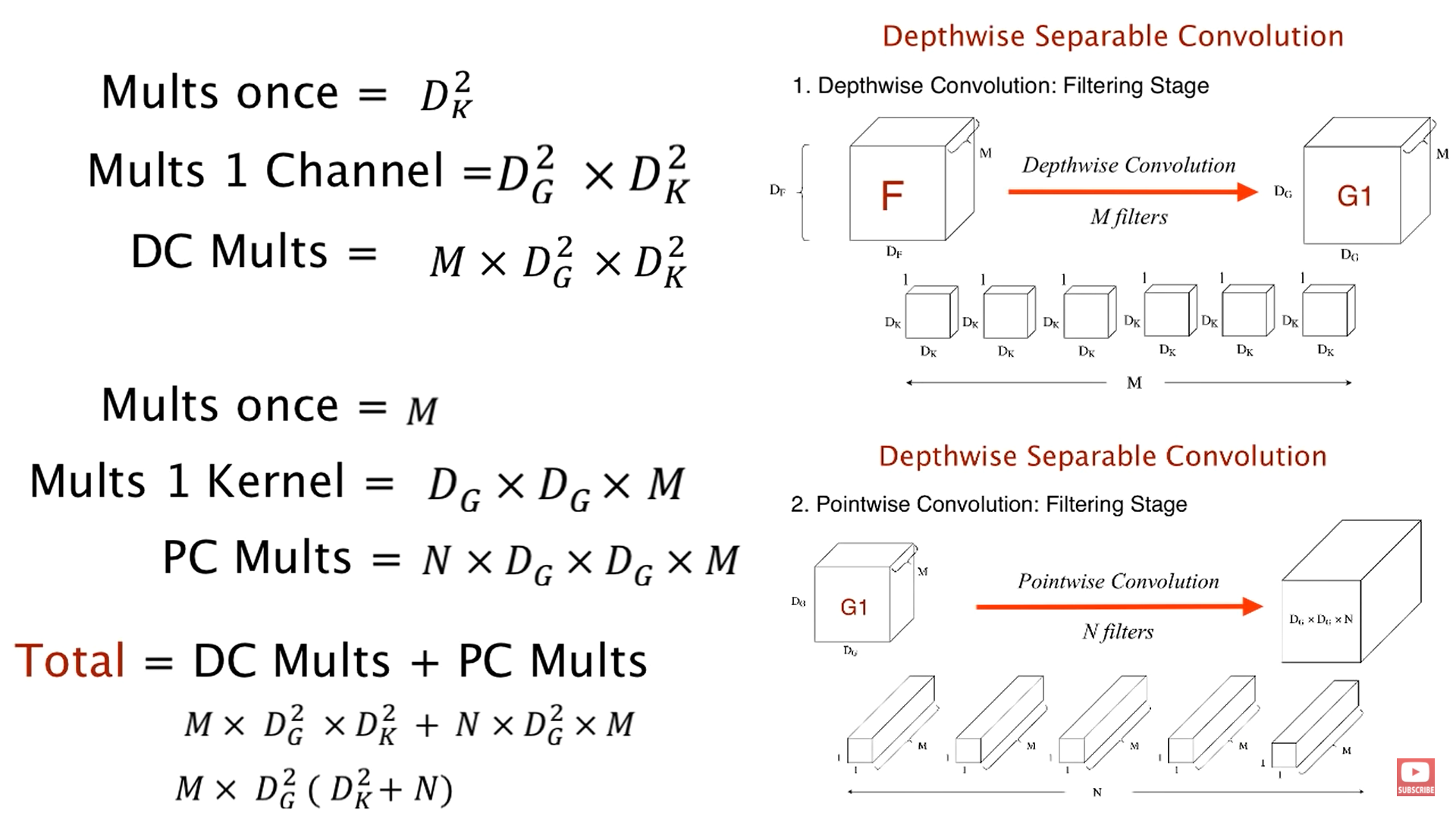
Pointwise Convolution is a type of convolution that uses a 1x1 kernel: a kernel that iterates through every single point. This kernel has a depth of however many channels the input image has. It can be used in conjunction with depthwise convolutions to produce an efficient class of convolutions known as depthwise-separable convolutions.



**Fig 5.2.2 Pointwise Convolution**

**Depthwise Convolution**

In depthwise convolution, we use an N x N x 1 filter. This is because we have already used a 1 x 1 x 3 filter to get the colour information in the previous step. The 3 in 1 x 1 x 3 corresponds to RGB colours. So now that we have got colour information, we need to get information of the features in a spacial sense. So we use the said N x N 1 filter across the image ( We are only using one colour channel as the information on all the channels were obtained in the previous step)



**Fig 5.2.3 Depthwise Convolution**

We then combine the information we got in the previous 2 steps to get the features.

**Max Pooling**

Maximum Pooling (or Max Pooling): Calculate the maximum value for each patch of the feature map.The result of using a pooling layer and creating down sampled or pooled feature maps is a summarized version of the features detected in the input. They are useful as small changes in the location of the feature in the input detected by the convolutional layer will result in a pooled feature map with the feature in the same location. This capability added by pooling is called the

model’s invariance to local translation.

**Dropout Layer**

Ensembles of neural networks with different model configurations are known to reduce overfitting, but require the additional computational expense of training and maintaining multiple models

A single model can be used to simulate having a large number of different network architectures by randomly dropping out nodes during training. This is called dropout and offers a very computationally cheap and remarkably effective regularization method to remove overfitting and improve generalization error in deep neural networks of all kinds.

**Flatten layer**

In between the convolutional layer and the fully connected layer, there is a ‘Flatten’ layer. Flattening transforms a two-dimensional matrix of features into a vector that can be fed into a fully connected neural network classifier.

Flattening is converting the data into a 1-dimensional array for inputting it to the next layer. We flatten the output of the convolutional layers to create a single long feature vector. And it is connected to the final classification model, which is called a fully-connected layer. In other words, we put all the pixel data in one line and make connections with the final layer. And once again.What is the final layer for? The classification of either ‘pf or normal’.

**Dense Layer**

A fully connected layer also known as the dense layer, in which the results of the convolutional layers are fed through one or more neural layers to generate a prediction.A dense layer feeds all outputs from the previous layer to all its neurons, each neuron providing one output to the next layer. It's the most basic layer in neural networks.

**Xception Architecture**

Xception Architecture is a convolutional neural network architecture based entirely on depthwise separable convolution layers.

It is composed of 36 convolutional layers forming the feature extraction base of the network

It is structured into 14 modules, all of which have linear residual connections around them, except for the first and last modules.

These 36 convolution layers consist of our depthwise separable convolution as an alternative to traditional convolution.

To summarise, Xception has depthwise convolution as a convolution operation which is validated and made non linear by an activation function(We have made 2 models, one that uses ReLU and the other that uses swish). This output is then maxpooled and fed into input of the next depthwise separable convolution layer. This is repeated 4 times. After this the output is fed to a depthwise convolution layer and an activation function but there is no max pooling. This is repeated 8 times. Finally the result is maxpooled and then sent to the final depthwise convolution layer. The output of this is then finally globally average pooled and flattened into 2048 vectors. This finally goes to the fully connected layers and output is finally obtained.

1. **Activation functions**

Activation functions are an essential part of an artificial neural network. They enable a neural network to be built by stacking layers on top of each other, glued together with activation functions. Activation function decides whether a neuron should be activated or not by calculating weighted sum and further adding bias with it. The purpose of the activation function is to introduce non-linearity into the output of a neuron.

The two activation functions used with our Xception Architecture are:

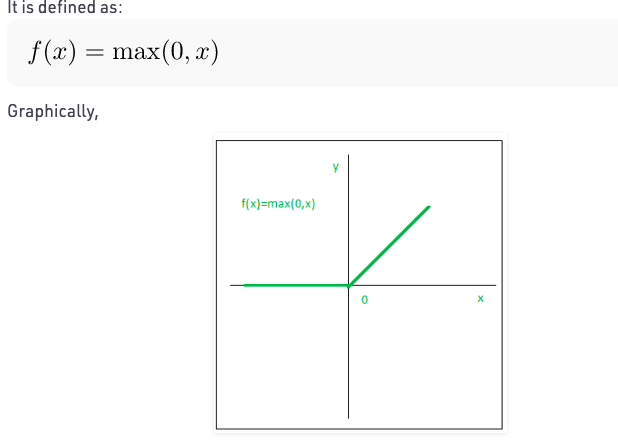
* ReLU
* Swish

**ReLU:**

ReLu is a non-linear [activation function](https://deepai.org/machine-learning-glossary-and-terms/activation-function) that is used in multi-layer [neural networks](https://deepai.org/machine-learning-glossary-and-terms/neural-network) or deep neural networks. This function can be represented as:

**f(x)= max(0,x)**

Where **x** is the input value



**Fig 5.3.1 Graph of ReLU function**

Graph of the ReLU function, showing its flat gradient for negative x.

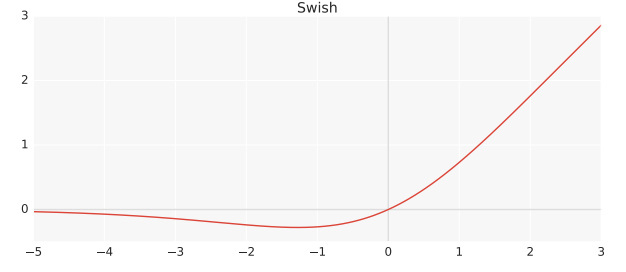
**Swish:**

Swish is essentially a modified sigmoid function which is simply

**f(x) = x · sigmoid(x).**

Experiments show that Swish tends to work better than ReLU on deeper models across a number of challenging data sets.

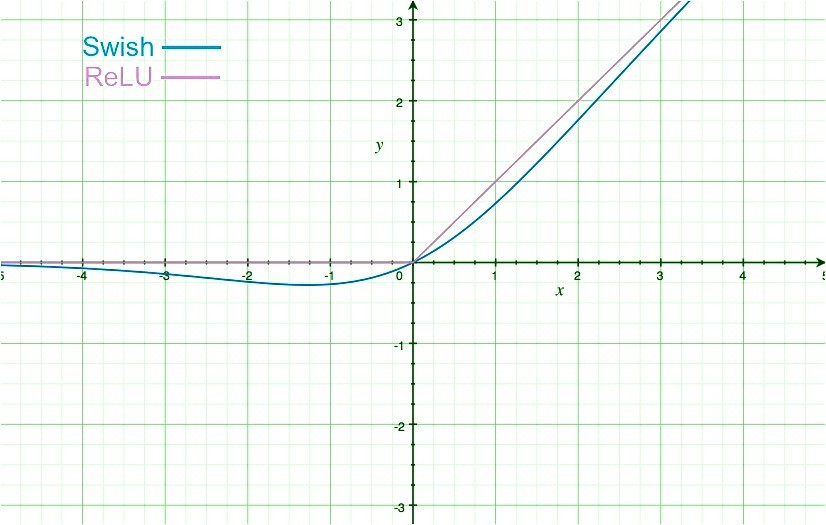
Swish is a smooth, non-monotonic function that consistently matches or outperforms ReLU on deep networks applied to a variety of challenging domains such as Image classification and Machine translation.



**Fig 5.3.2 Graph of Swish Function**

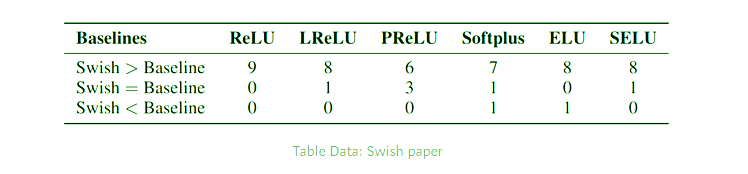
**ReLU vs Swish**

Having no bounds is desirable for activation functions as it avoids problems when gradients are nearly zero. The ReLU function is bounded above but when we consider the below region then being bounded below may regularize the model up to an extent, also functions that approach zero in a limit to negative infinity are great at regularization because large negative inputs are discarded. The swish function provides it along with being non-monotonous which enhances the expression of input data and weight to be learnt.



**Fig 5.3.3 Comparison between ReLU and Swish function**

The performance metric of Swish function over other activation functions like ReLU, SeLU, Leaky ReLU and others:

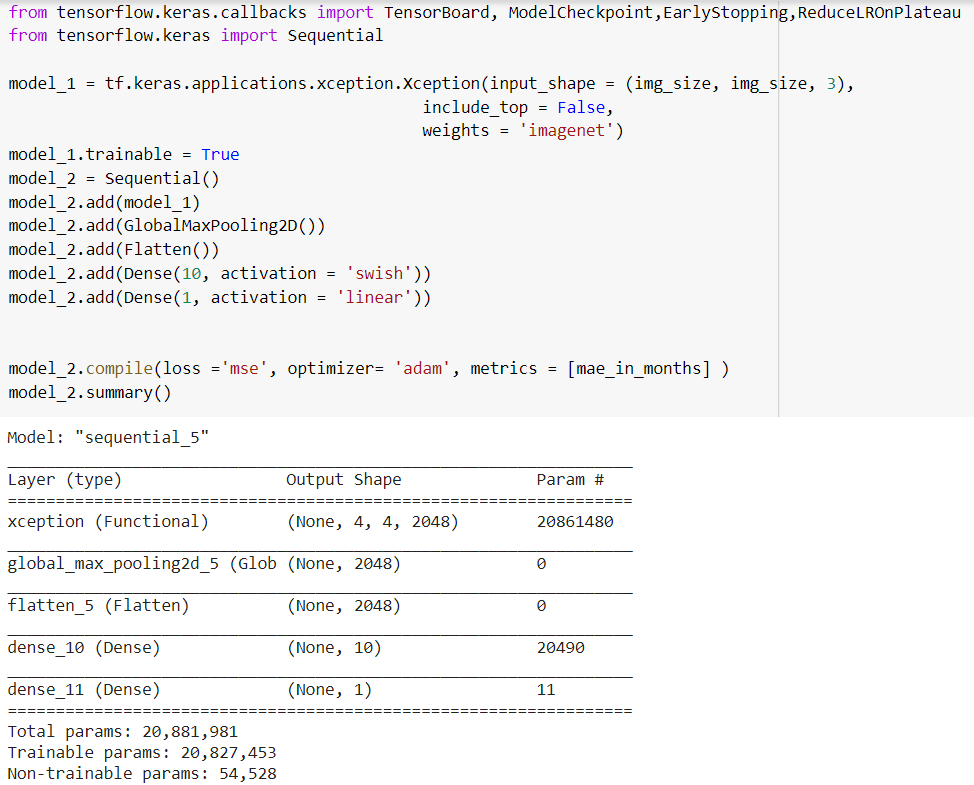


**Fig 5.3.4 Performance of Swish function over other activation functions**

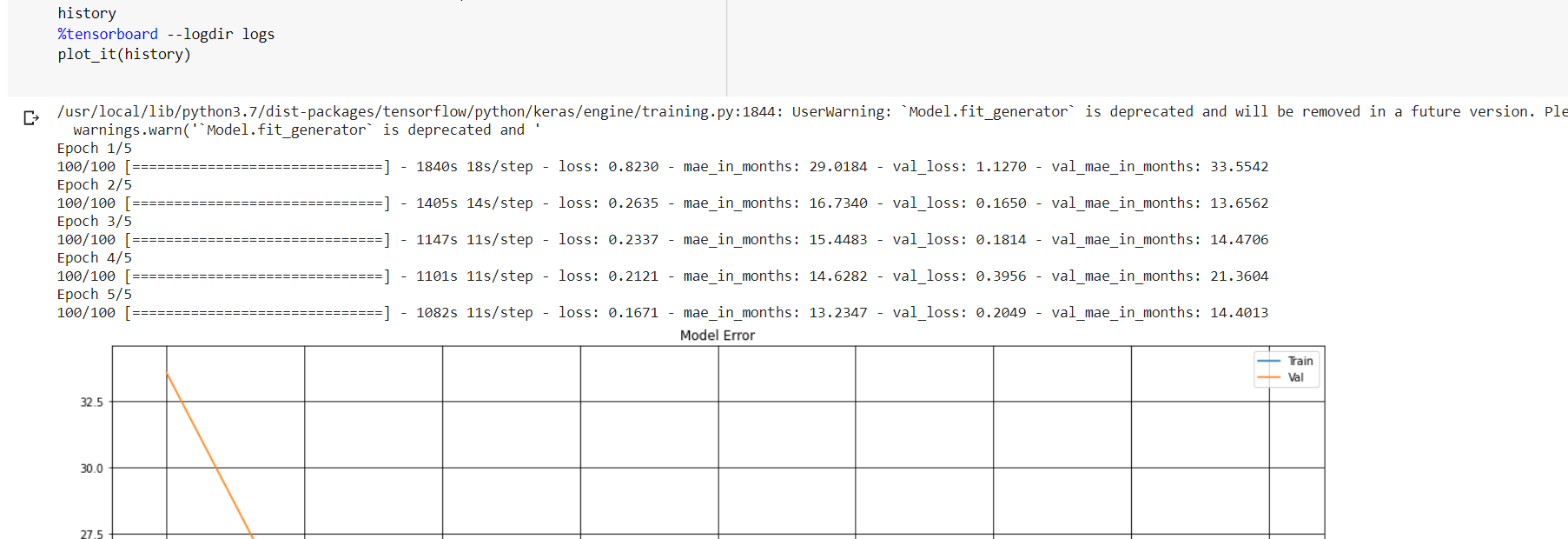
1. **Implementation**

**For Swish:**



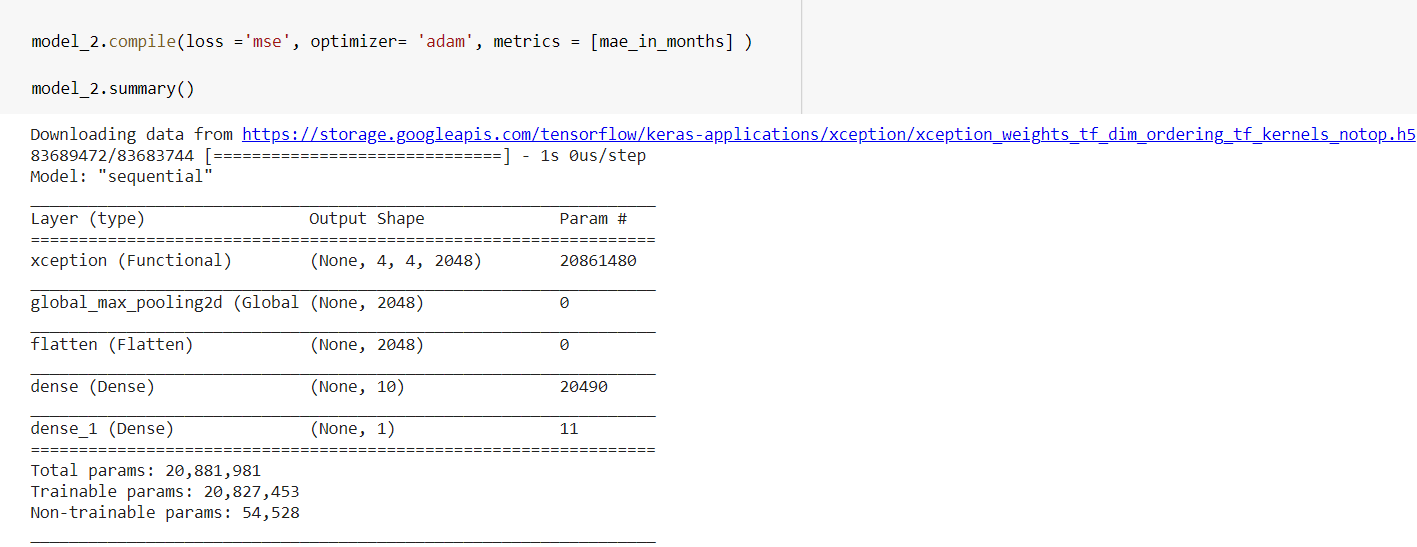




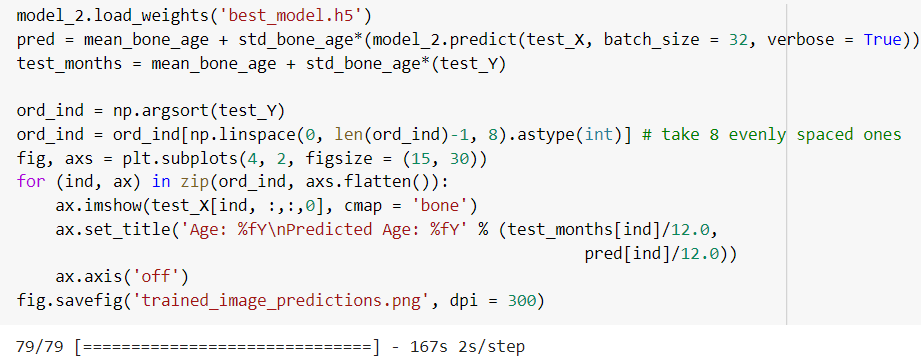


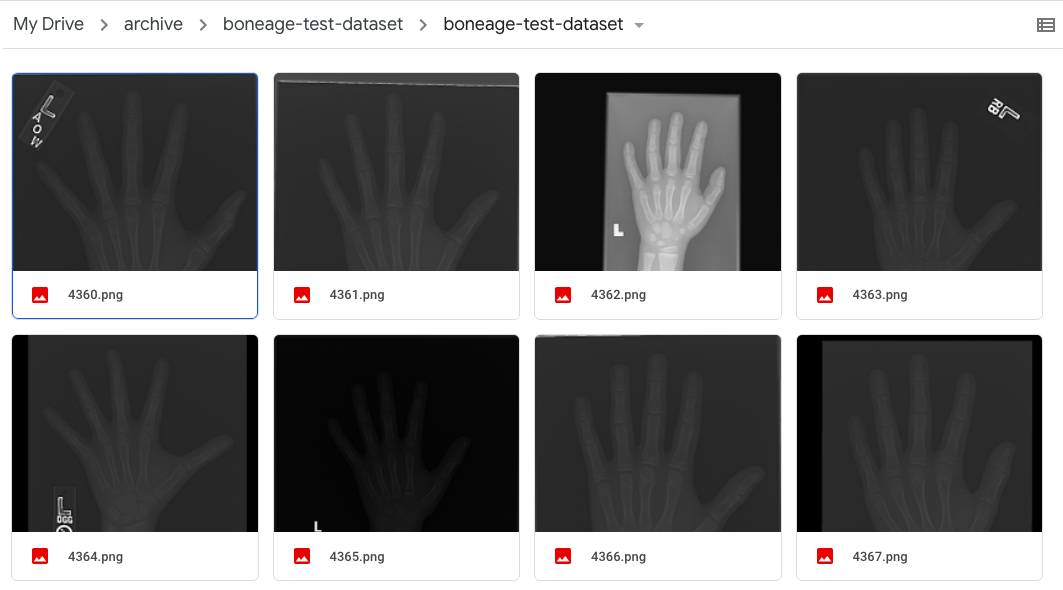
**For ReLU:**



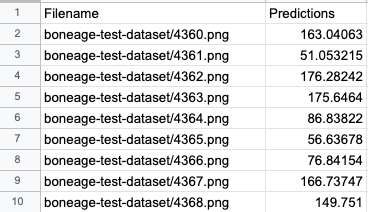


1. **Testing**





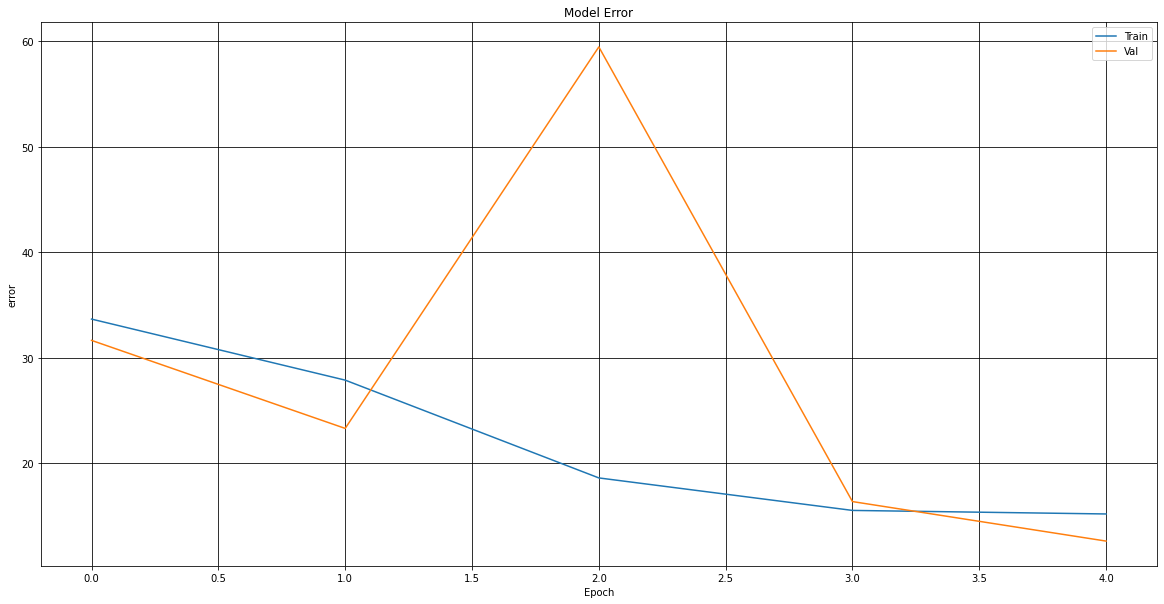
**Fig 7.1 Testing Dataset**



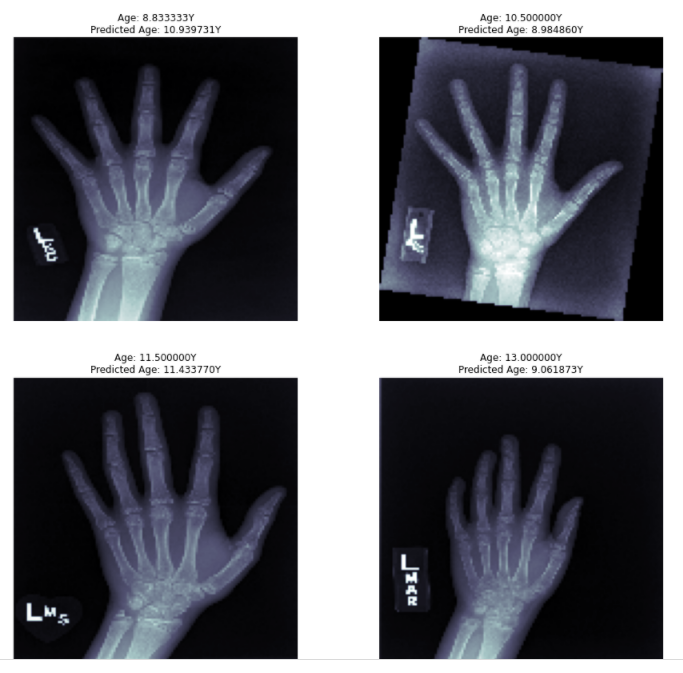
**Fig 7.2 Predicted value of Testing Dataset**

1. **Result Output**

**Result From ReLU:**

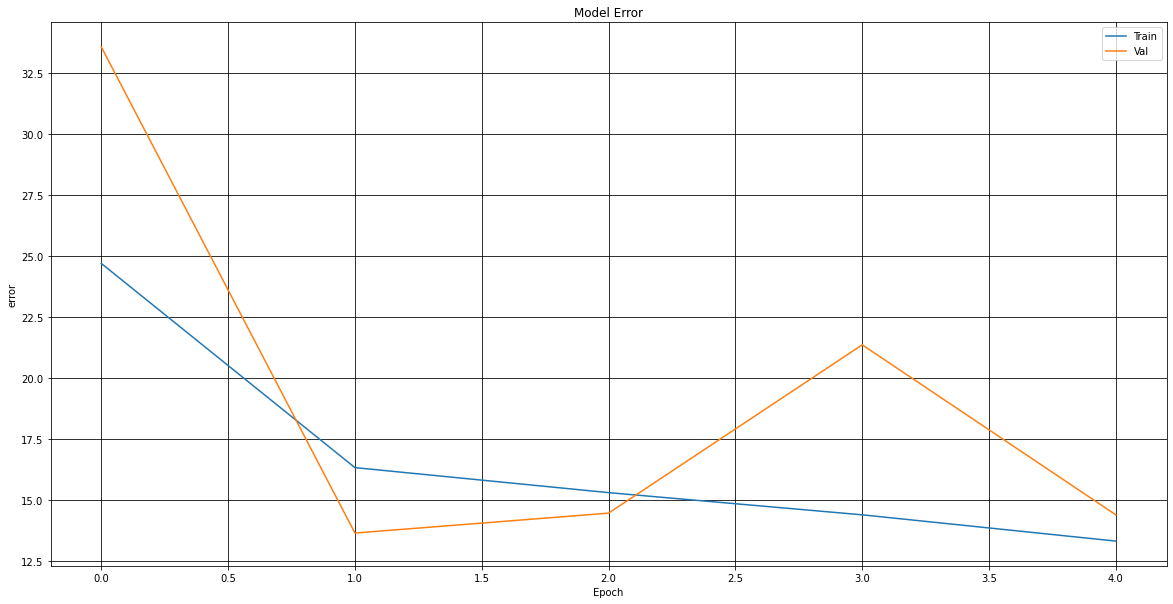


**Fig 8.1 Result from ReLU**

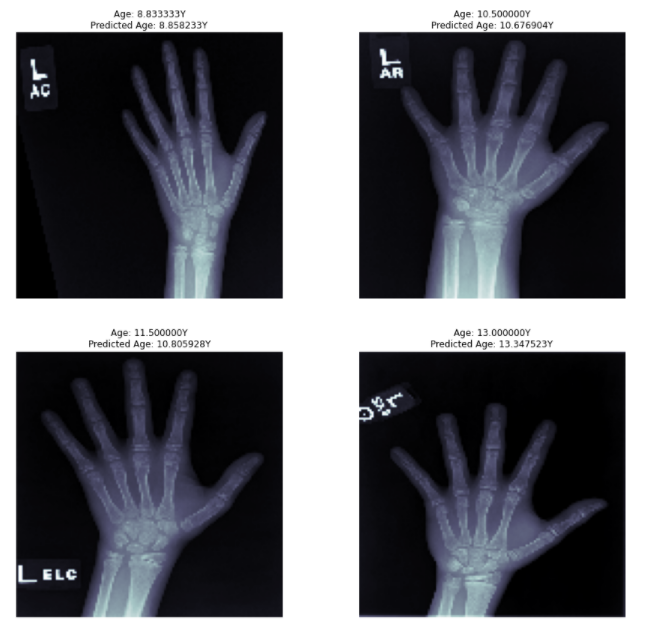


**Fig 8.2 Predicted age using ReLU**

**Result from Swish:**



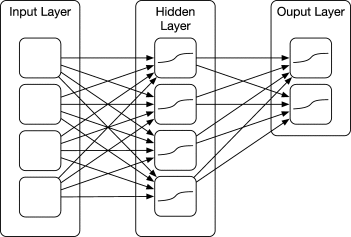
**Fig 8.3 Result from Swish**



**Fig 8.4 Predicted age using Swish**

1. **Prototype**

This application provides a simple way of determining the skeletal age of a child. All the user has to do is upload an X- ray of the patient’s left hand of acceptable resolution. The application then calculates the skeletal age in a relatively short amount of time and provides an accurate result.



**Fig 9.1 Prototype**

1. **Conclusion and future work**

From the above results provided, we can confidently say that Skeletal Age of a left wrist X-Ray can be efficiently achieved with the help of Convolutional Neural Network (CNN) following the Xception Architecture. Providing a competent alternate way to manual TW2 method, which is susceptible to human error.  
We can also see how much effect an activation function has on the predicted value. In this project we show how changing the activation function from ReLU to Swish increases the accuracy of Skeletal Age prediction drastically.

Future Work for this project involves comparing a dataset with images of lower resolution and a dataset with lesser samples; To determine which case provides more accurate predictions. This will help determine which case to follow in situations where computing power isn’t sufficient to run the original dataset without resizing.

**References**

* <https://machinelearningmastery.com/how-to-define-your-machine-learning-problem/>
* <https://machinelearningmastery.com/what-is-deep-learning/>
* <https://machinelearningmastery.com/save-load-keras-deep-learning-models/>
* <https://keras.io/api/optimizers/>
* <https://machinelearningmastery.com/image-augmentation-deep-learning-keras/>
* <https://machinelearningmastery.com/tutorial-first-neural-network-python-keras/>
* <https://towardsdatascience.com/a-simple-guide-to-the-versions-of-the-inception-network-7fc52b863202>
* <https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks/>
* <https://arxiv.org/abs/1710.05941>