SAiDL Induction assignment – 4.1

- **Hrithik Nambiar**

Step 1: finding a phenomenon and a question to ask about it

Step 2: understanding the state of the art

Step 3: determining the basic ingredients

Step 4: formulating specific, mathematically defined hypotheses

Step 5: selecting the toolkit

Step 6: planning the model

Step 7: implementing the model

Step 8: completing the model

Step 9: testing and evaluating the model

Step 10: publishing models

This Assignment is based on a Project I have completed as I was learning Convolution Neural Networks using TensorFlow 2.0 and Keras. The code for the same can be found on my GitHub repository (**https://github.com/hrithik1729/Facial-key-point-detection).**

STEP 1:

Emotion AI, is one of the emerging areas of Artificial Intelligence where machines study non-verbal cues of humans like body language, facial expressions, gestures, and tonality of voice to detect their emotional state. This technology has started to find tremendous use in areas like advertising, customer service, healthcare, and others.

Facial Key point Recognition is critical in Image recognition. Facial Key points are also called Facial Landmarks which generally specify the areas of the nose, eyes, mouth, etc on the face, classified by 68 key points, with coordinates (x, y), for that face. With Facial Key

points, we can achieve facial recognition and emotion recognition.

In this project a Deep learning model to recognise the facial key points of the given images is created.

STEP 2:

Recent developments in Convolutional Neural Networks has led to great improvements in the field of Computer Vision. The deep structures of CNNs extract raw data into a high-level abstraction, which consists of various aspects of an input that keep the distinguished features but discard irrelevant information [1]. The CNN is replaced by Res-Net or deep residual neural network. Introducing an identity mapping shortcut connection [2], the Res-Net can prevent the problem degradation and allow to be trained on deeper networks. We furthermore adapt a recent technique, batch normalization [3], to avoid internal covariate shift.

1. Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (7553) (2015) 436–444.

2. K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, arXiv:1512.03385 (2015).

3. S. Ioffe, C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, arXiv:1502.03167 (2015).

STEP 3:

For this project a dataset consisting of 2140 images and the coordinates of their facial 29 key points is used. Then we augment this dataset by rotating these images, increasing brightness etcetera, so that it helps in training our model with a broader dataset. For this project we needed a neural-network library written in Python, hence Keras is used. This project is made using the TensorFlow 2.0. The knowledge about the working of CNN and Res-Net was an important pre-requisite.

We will be using the model to provide us with marked facial key points on the images inputted to it with a fair accuracy.

STEP 4:

Images are often represented as 3 dimensional arrays with height(h), width(w) and colour(c) as its dimensions. Because the spatial positions of pixels are important, convolutional layers in CNNs retain this spatial position information by performing a 2-dimensional convolution operation along h and w axes. The input and output of each convolutional layer constitute a feature map, which is a 3-dimensional array with the size of h × w × c. Each convolutional layer consists of n learnable kernels of size h × w × c representing height, width, and depth. The height h and width w are usually small to learn local features.

Convolution operation over a 2D image, where I(x, y) is a function of image, k(x, y) is a function of each kernel. Each element of the output feature map is the summation of the multiplied values between the kernel and the local feature map.

$$(I * k)(x, y) = \sum_{u,v} I(x, y) * k(x - u, y - v)$$

Finding the x, y positions of a facial key point can be implemented as a linear regression task. A conventional linear regression objective function used in facial key point detection is Mean Squared Error (MSE) as defined in Eq. (3), where N is the number of predictions, y is the ground truth target key points positions, and $\bar{y}$ is the predicted key points positions.

$$e = \frac{1}{N}\sum_{i=1}^{N}(y_i - \bar{y}_i\ )\text{^}2$$

STEP 5:

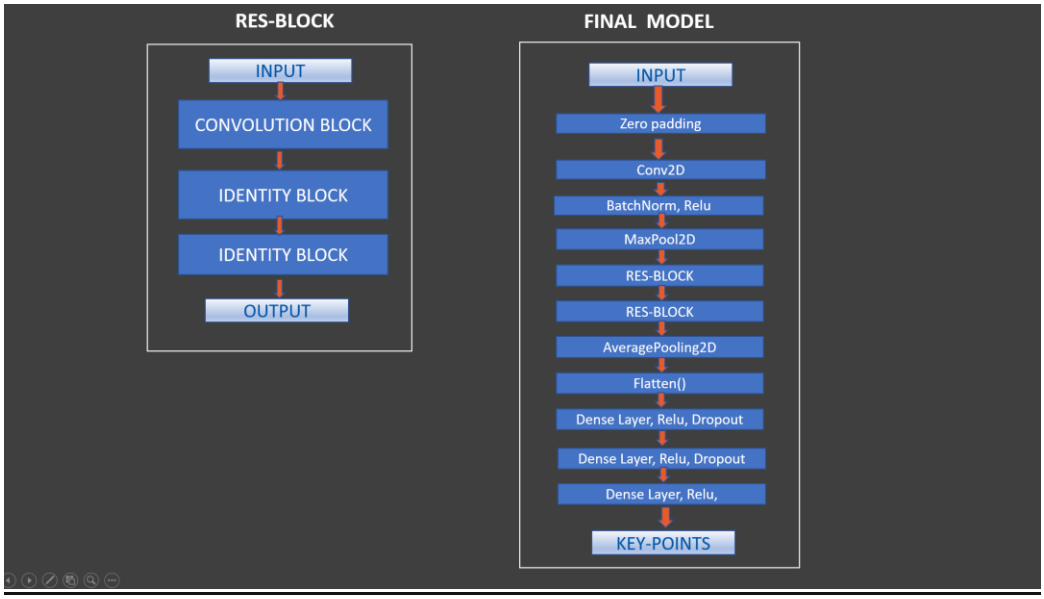
TensorFlow 2.0 and Keras have all the necessary toolkit to develop this model. The code is written in Python in a Jupyter notebook, so that it becomes easy to share the code with other people.

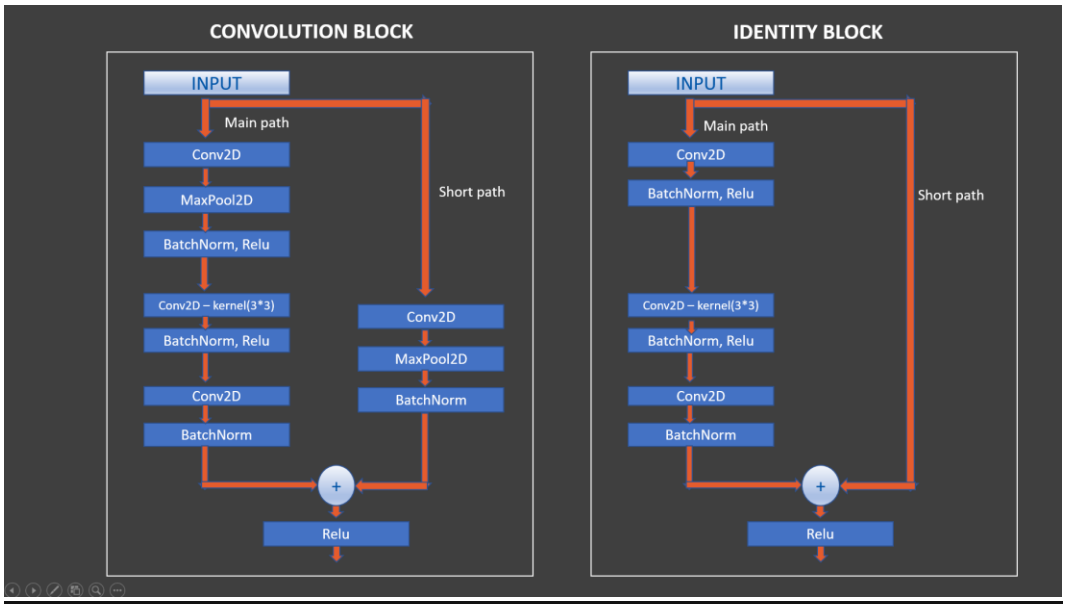
For calculations, the NumPy library is utilised. And for Data Analysis and cleaning

tasks the Pandas library is used. The matplotlib.pyplot library provides all the necessary toolkits for plots and visualisation.

STEP 6:

The model is then designed.





STEP 7 and 8:

The model shows an accuracy of 73.2 percent, which is quite acceptable and satisfa ctory.