**Assignment – Classification of Defected Manufactured Automobile Part using AI**

**Tools / Programming languages used:**

Python

Keras (open source high level Deep Learning Library)

OpenCV (Image Processing Library)

Powershell (scripting language for Automation)

For classification of defected part we have used Convolutional Neural Network Using Keras library which is high level python library built upon Tensorflow Framework.

Image Classification involved various stages –

Image Preprocessing

Designing Artificial Neural Network in python

Evaluating our model and parameter optimization

**NOTE:** As we have only approx. 110 images per class (per type of image like Defected or ok) it is impossible to build an A.I Neural Network that have good accuracy for test images. Still to overcome this I have used Image Augmentation.

Same model can be used with more number of images in test data to help it learn patterns in images.

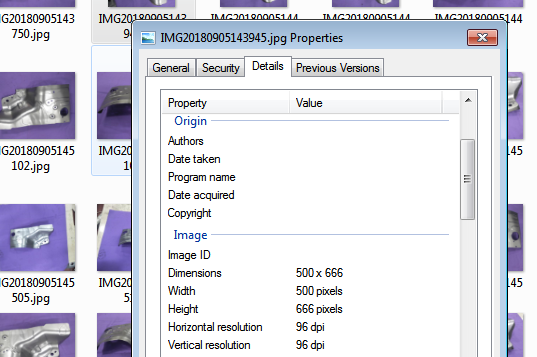
More the number of images in test data better A.I model will learn.

**Conversion of images to low resolution**

As our Dataset consists of images of size 4 mb (approx) we need to reduce the size of the images to fit it to our Neural Network as number of pixels in image is directly proportional to the number of neurons in convolutional neural networks.

For this particular task only I have used python programming language.

Images are converted to size 500x500 with 96 dpi

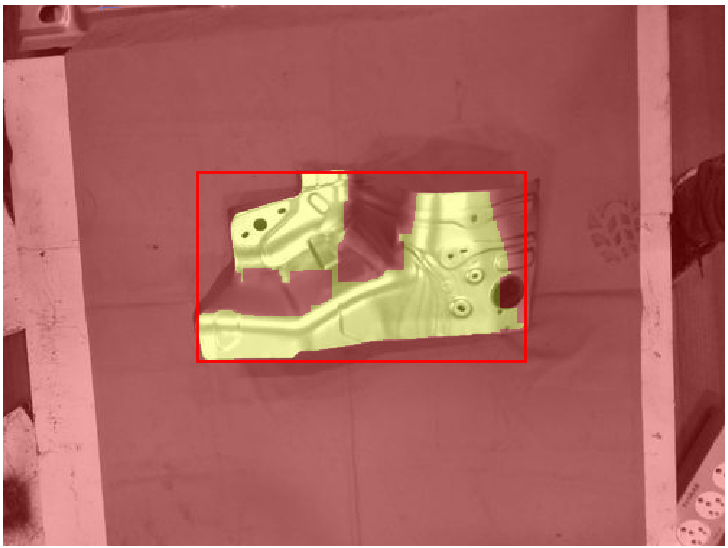
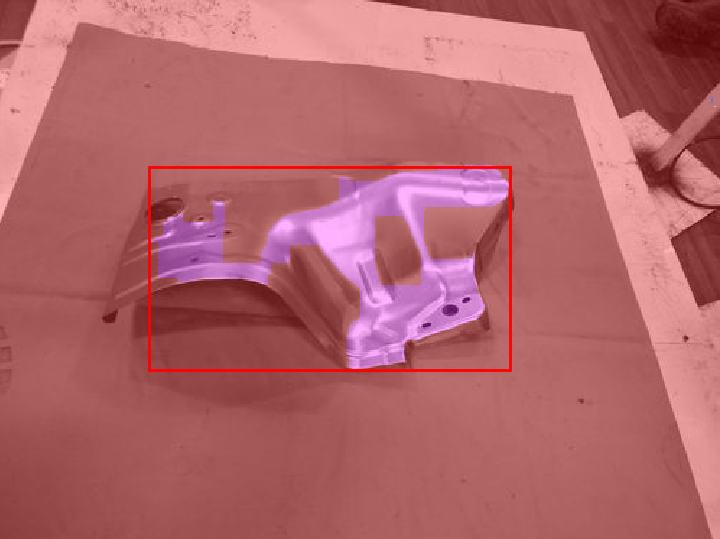


**Contour detection and selection of only the product**

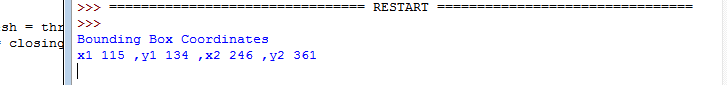
Now skimage contour detection feature is used to detect exact bounding box of desired object.

Bounding box contour detection in skimage returns us multiple bounding boxes I have sorted them and selected one with highest rectangular area in order to detect exact location of our object in image

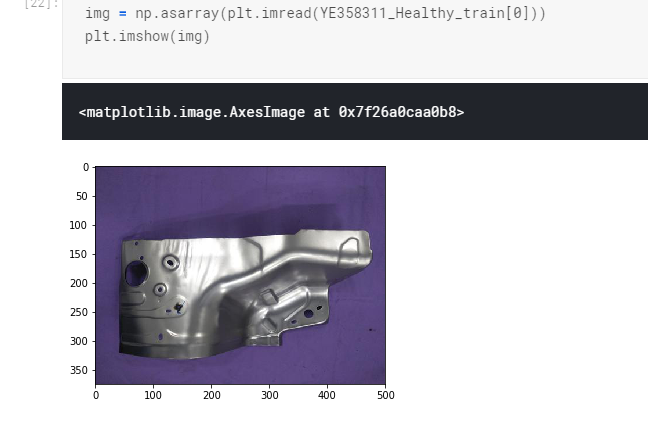
This gives us with coordinate values for our object

**Cropping image to save bounding box region to disk**



**After these 2 steps we have processed images which are free from unnecessary background noises.**



**Data Augmentation**

A.I is all about data and without data we cannot learn weights in neural networks and as I was only provided with very less number of images

Data Augmentation must be used.

It basically increases the number of images in test data by doing below mentioned tasks-

featurewise\_center

featurewise\_std\_normalization

rotation\_range

width\_shift\_range

height\_shift\_range

zoom\_range

horizontal\_flip

shear\_range

Keras provides Data Augmentation API Image\_Generater which we have used for this task

**VGGNet Deep Neural Network in keras**

Now I created Deep Neural Network in using keras library in python with Tensorflow backend.

Model consists of multiple layers of Convolution and pooling.

Relu is used as activation function

And number of epoch used are 5 as I do not have high end GPU for processing of such GPU intensive CNN.

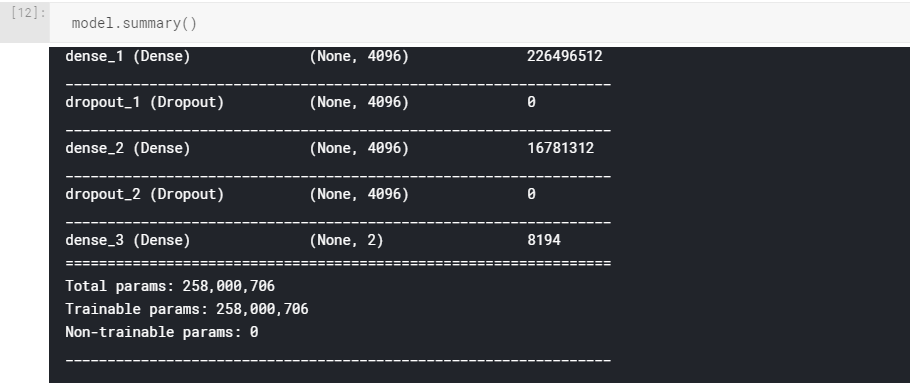
For testing and production phase implementation of this Image processing API

We can test and use more number of epoch till loss function is decreasing.

Data is divided in test and training sets

Train dataset is used to train the weights of our neural network and test set is used to test/validate model results.

As it is good practice to divide data in 80/20 format we have used same for our program.



**Detection of features**

This model when compiled and fitted to our data set will automatically learn patterns in images and will adjust weights of neurons in neural network model to identify classes (Defected,Normal)

**Prediction**

Now Model consists of learned weights and “.predict” function can be used to predict any new incoming images now

**Coming to the 2nd part of Question in Assignment – Make it work on Mobile platform**

We can create flask API in python and host it as a microservice on cloud.

Mobile application or any webpage can call this API with image as base64 string in URL and we can send the response in JSON or any desired format.

Leaving this part of assignment as it is fairly easy and I do not have exact requirements