15

Our second task is to create our own CNN model, we trained our model by using the data from our train dataset, it is the same dataset used in vgg16 experiment. To see its performance against VGG16 model.

16

Here is our data preparation process, we read each image from disk, standardize their size by increasing their size by scale to 330 times 330 pixels to prevent any pixel losses, and then convert them to a numpy array.

17

For image labels, we convert string labels to numbers, and then create 2 arrays for training purposes, one array is images and another one is the names of the gem classes.

18

Here are 10 random plots showing the gem images and labels. We use this step to validate if our data preprocess outcome is as expected.

19

The final step of our data processing is to crop edges of gemstone images. We use the canny function from openCV to detect edges of gems. Then crop the image and resize them to 220 \* 220 pixels

20

Here you can see the cropping process. The picture on the left is the original image, the second picture is the edges detected by the canny function. The third one shows the bounding box’s location. The last image is the cropped gem image that is ready to be used in our model for training. All data processing is finished after this step

21

We use sequential from keras models as our model, it means a linear stack of layers. To compare our own CNN model with the VGG16 network, we use a very similar architecture. As you can see on the left, it has 9 layers. In the next slide, there will be model comparison.

22

VGG16 model is on the left, and our own CNN model is on the right. Our CNN model is pretty much a mini version of the vgg16 architecture, it reduces the numbers of layers of each size. Therefore, it only has about 3million trainable parameters while VGG16 has more than 100million.

23

After we create our model, it is time to fit data into the model. In this picture, we ran our model with the option of 10 epochs. The accuracy is increasing as the epoch reaches 10. It is because epoch stands for how many times the algorithm looks at the whole training data.

24

After 18 epochs, we have the accuracy number 0.49. The accuracy is the number of right predictions divided by the total number of predictions. It is likely we would have better results with more epochs, but we cannot do it in google colab. We will talk more about this in our conclusion section

25

Here is our F1 score,. F1 score is also known as F measure, it shows the accuracy of the test in a balanced way, by calculating both precision and recall value to determine the accuracy. The outcome is about 0.5 is very close to the original accuracy score. In addition, the confusion matrix shows a dense line across the diagonal and most non diagonal areas are filled with 0, so we think our result is credible.