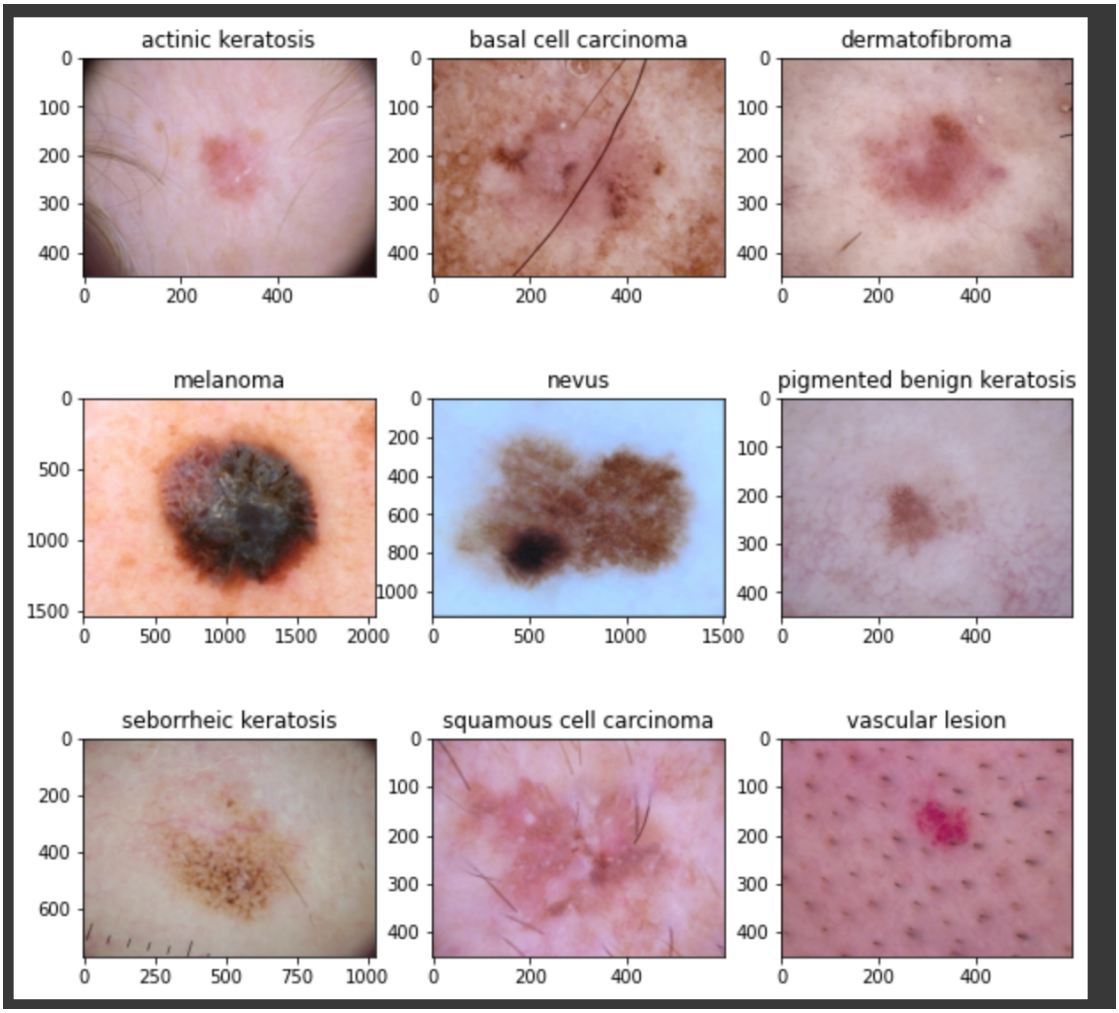
Melanoma Assignment Todo answer_Thao Tran Chi

1. Todo, create a code to visualize one instance of all the nine classes present in the dataset

```
import matplotlib.pyplot as plt

### your code goes here, you can use training or validation data to visualize
import matplotlib.image as mpimg
plt.figure(figsize=(10,10))
for i in range(9):
  plt.subplot(3, 3, i + 1)
  image = mpimg.imread(str(list(data_dir_train.glob(class_names[i]+'/*.jpg'))[1]))
  plt.title(class_names[i])
  plt.imshow(image)
```



2. Todo: Create a CNN model, which can accurately detect 9 classes present in the dataset. Use layers.experimental.preprocessing.Rescaling to normalize pixel values between (0,1). The RGB channel values are in the [0, 255] range. This is not ideal for a neural network. Here, it is good to standardize values to be in the [0, 1]

```
from keras.layers import Dense, Dropout, Flatten, Conv2D, MaxPool2D, Activation, BatchNormalization, MaxPooling2D
num_classes = 9

model = Sequential([layers.experimental.preprocessing.Rescaling(1./255, input_shape=(img_height, img_width,3))])
model.add(Conv2D(16, 3, padding='same'))
model.add(Activation('relu'))
model.add(MaxPooling2D())

model.add(Conv2D(32, 3, padding='same'))
model.add(Activation('relu'))
model.add(MaxPooling2D())

model.add(Conv2D(64, 3, padding='same'))
model.add(Activation('relu'))
model.add(MaxPooling2D())
model.add(Dropout(0.15))

model.add(Flatten())
model.add(Dense(128))
model.add(Activation('relu'))

model.add(Dense(num_classes))
```

3. Todo: Write your findings after the model fit, see if there is an evidence of model overfit or underfit

There are some clear points indicating that the model is overfit

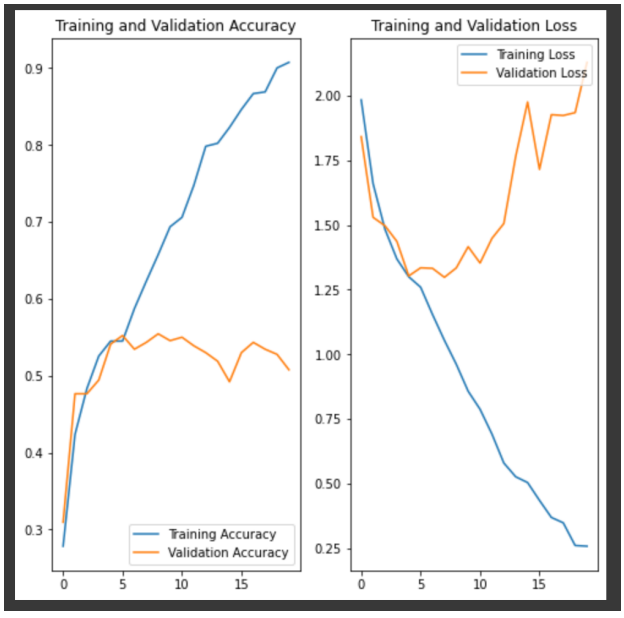
The training accuracy is way higher than validation.

The training loss is also very less compared to the validation loss.

Actually, the validation acuracy start dropping accuracy and increasing loss at 5th epochs.

```
[ ]  # Todo, after you have analysed the model fit history for presence of underfit or overfit, choose an appropriate data au
     # Your code goes here
     data_aug = tf.keras.Sequential(
       [
         layers.experimental.preprocessing.RandomFlip("horizontal",
                                                      input_shape=(img_height,
                                                                   img_width,
                                                                   3)),
         layers.experimental.preprocessing.RandomRotation(0.1),
         layers.experimental.preprocessing.RandomZoom(0.1),
       ]
     )
```

```
[ ]  # Todo, visualize how your augmentation strategy works for one instance of training image.
     # Your code goes here
     plt.figure(figsize=(12, 12))
     for images, labels in train_ds.take(1):
         for i in range(9):
             ax = plt.subplot(3, 3, i + 1)
             plt.imshow(data_aug(images)[i].numpy().astype("uint8"))
             plt.title(class_names[labels[i]])
             plt.axis("off")
```



4. Todo: Write your findings after the model fit, see if there is an evidence of model overfit or underfit. Do you think there is some improvement now as compared to the previous model run?

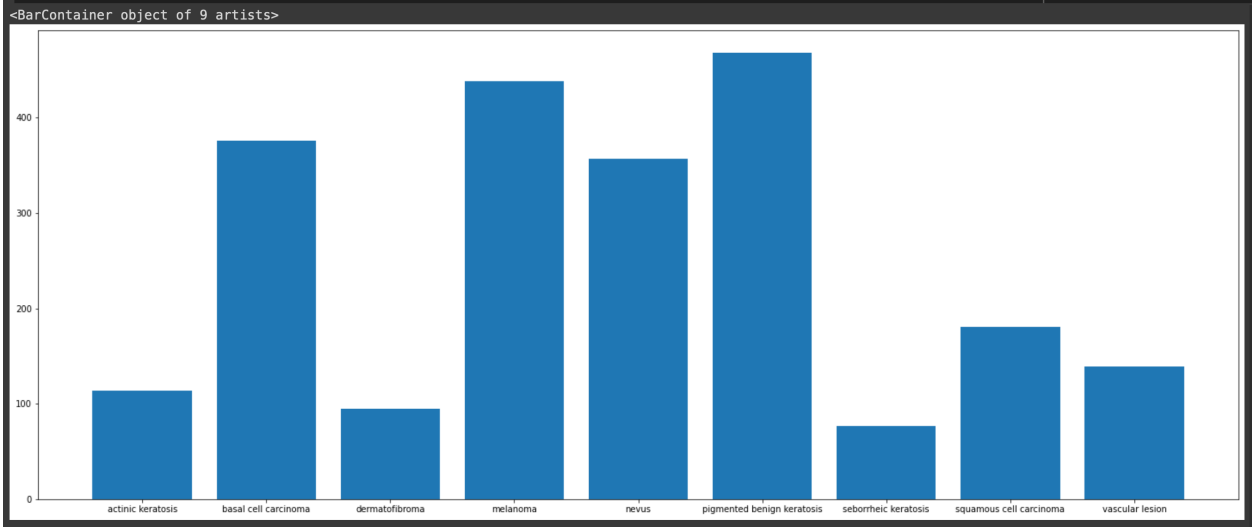
   Data augumentor helps a bit. The validation remains the same as previous model, but the training model also decrease significantly. However, the gap between training and validation now is much closers.

5. Todo: Find the distribution of classes in the training dataset.

```
## Your code goes here.
path_list=[]
lesion_list=[]
for i in class_names:

    for j in data_dir_train.glob(i+'/*.jpg'):
        path_list.append(str(j))
        lesion_list.append(i)
dataframe_dict_original = dict(zip(path_list, lesion_list))
original_df = pd.DataFrame(list(dataframe_dict_original.items()),columns = ['Path','Label'])
original_df
```

```
<BarContainer object of 9 artists>
```



6. Todo: Write your findings here:

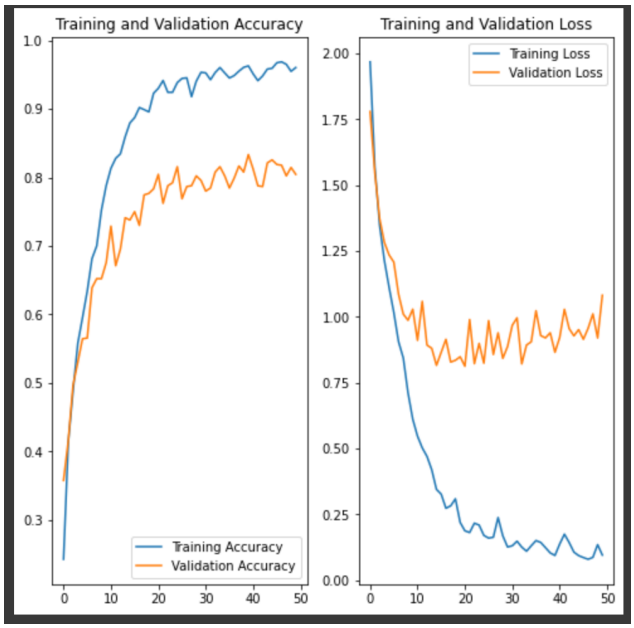
   - Which class has the least number of samples?

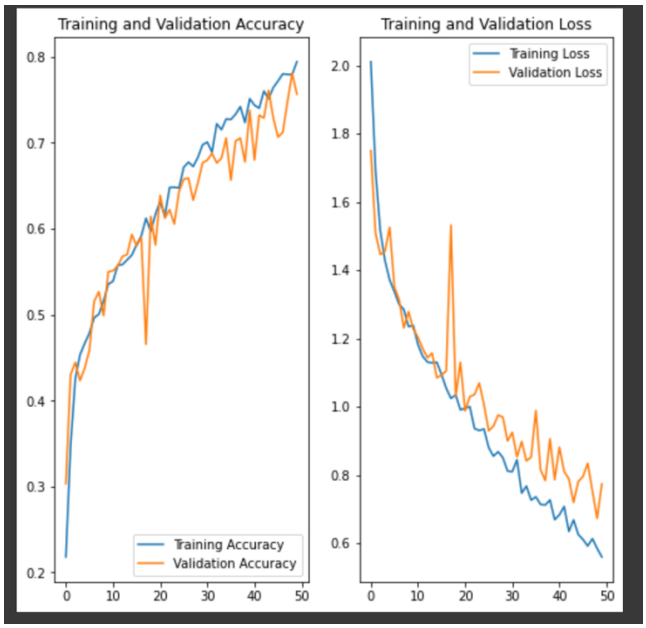
   It's very clear that class sebornheic keratois has the least number of samples

   - Which classes dominate the data in terms proportionate number of samples?

   and of course class pigmented benign keratosis has the most number of samples.

7. Todo: Visualize the model results





8. Todo: Analyze your results here. Did you get rid of underfitting/overfitting? Did class rebalance help?

The result gets better compared to previous model (CNN using data augmentation), however the loss in validation still increases after 20 epochs. Let's create a new model, which combines both data augmentation and augmented dataset.

Four models have been built:

- CNN without any tuning
- CNN with data augmentation
- CNN with augmentor dataset
- CNN with data augmentation and augmentor dataset

The last model gives the best results because accuracy and loss in both training and validation are similiar. It means the model is the most robust.

```
Predictions:
 [0 0 0 1 1 1 1 0 0 0 1 0 1 1 0 0 0 0 0 1 0 0 0 1 0 0 0 0 1 0 0 0 0 0 1 1 0
 0 1 1 1 1 1 0 0 1 1 0 0 0 1 0 1 0 1 0 0 0 1 1 1 0 0 1 0 0 0 0 1 0 0 0 1 1
 0 0 1 1 0 1 0 0 0 1 1 1 1 0 0 0 1 1 1 0 0 1 0 1 0 0 1 0 1 1 0 0 1 0 0 0 0
 1 1 1 1 0 0 0 1 1 0 1 1 1 1 0 0 0 0 0 1 1 1 0 0 0 1 1 1 0 0 1 0 1 0 0 1 1 0
 0 0 0 1 0 0 0 1 0 1 1 0 1 0 1 1 1 0 0 0 0 1 0 0 1 0 1 1 1 0 0 0 1 1 0 0 1
 0 0 1 0 0 1 0 0 1 1 0 1 0 0 1 1 1 1 0 1 0 0 0 0 0 1 1 1 1 0 0 0 0 0 1 1 1
 0 0 0 1 1 1 0 1 1 0 1 0 1 1 1 1 0 1 0 0 0 1 1 0 1 1 0 0 1 0 1 0 0 1 0 1 0
 0 0 1 1 1 0 0 1 0 1 0 0 0 0 1 1 1 0 0 0 1 0 0 1 1 1 1 1 0]
Labels:
 [3 0 7 8 4 2 0 5 1 4 2 3 0 5 5 1 2 5 1 3 0 5 2 3 4 0 5 7 1 2 2 2]
```