```
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.preprocessing import image
import matplotlib.pyplot as plt
import tensorflow as tf
import numpy as np
import cv2
import os
```

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mour
```

```
ls
```

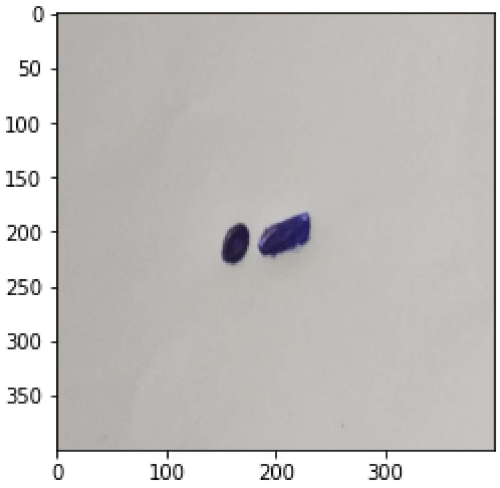
```
 DLD_528.pdf
 dogscats/
'Exp.2 Projection of Lines.gdoc'
'Exp.2 Projection of Lines.pdf'
 final.gsheet
 final.xlsx
'Getting started.pdf'
'Google Keep Document.gdoc'
'HANDS ON MACHINE LEARNING.pdf'
 IMG20200116133002.jpg
 IMG_20220912_104846.jpg
 IMG_20220912_130736.jpg
 IMG_20220912_130741.jpg
 IMG_20220912_142953.jpg
'IMP ASPECTS BEFORE TOUR.gdoc'
 kagglecatsanddogs_3367a/
 K.RAVITEJA.jpg
'KSPK '/
 Machine-Learning-Tom-Mitchell.pdf
 mit-deep-learning-book-pdf-master.zip
 NANNA_ARGUMENTS/
 PC2_528.pdf
 PC-2.pdf
'Ravi_Resume_1.0 (1).pdf'
'Ravi_Resume_1.0 (2).pdf'
'Ravi_Resume_1.0 (3).pdf'
'Ravi_Resume_1.0 (4).pdf'
 Ravi_Resume_1.0.pdf
'Ravi_Resume (1).pdf'
'Ravi_Resume (2).pdf'
'Ravi_Resume_New (1).pdf'
'Ravi_Resume_New (2).pdf'
'Ravi_Resume_New (3).pdf'
 Ravi_Resume_New.pdf
 Ravi_Resume.pdf
 RaviResume.pdf
```

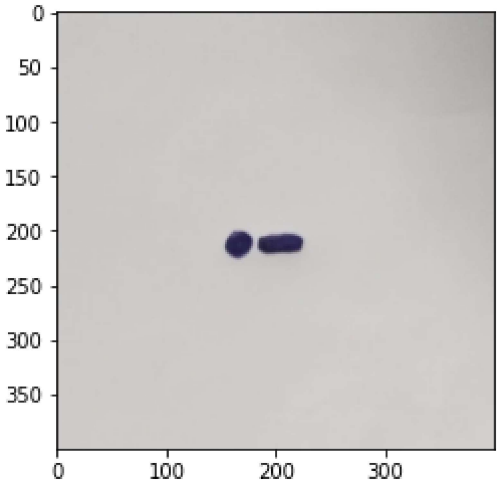
```
'RAVITEJA (1).pdf'
 RAVITEJA.pdf
 signature.jpg
 test_data.npy
'THINGS MATTER.gdoc'
 TTVC_19761A0528.pdf
 UC-75e981bd-6a96-4189-a723-59acb1a35c34.pdf
 UNIT-IV.doc
 Untitled0.ipynb
'Untitled form (1).gform'
'Untitled form (2).gform'
'Untitled form (3).gform'
'Untitled form (4).gform'
'Untitled form.gform'
 VID_20170619_052056_4.mp4
 Wipro
 X_ai.pkl
 X_dl.pkl
 X.pkl
 y_ai.pkl
 y_dl.pkl
 y.pkl
```

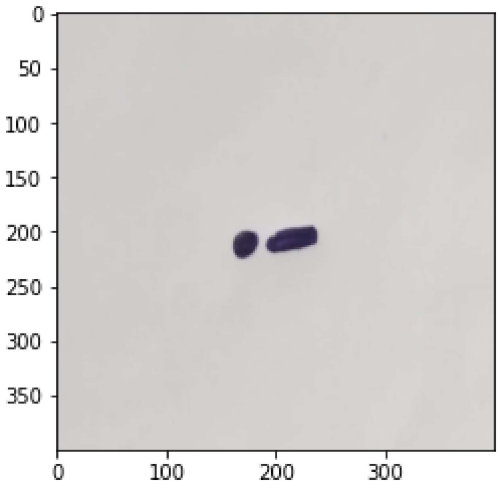
```
cd drive/MyDrive
```

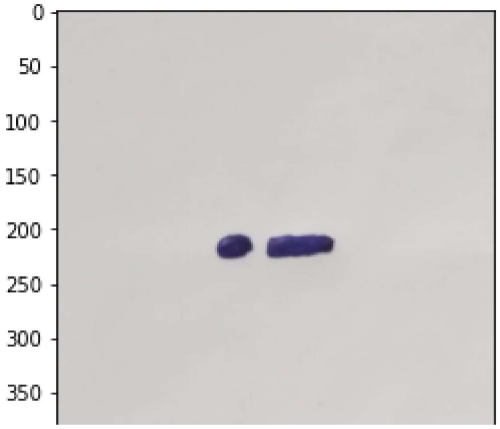
```
[Errno 2] No such file or directory: 'drive/MyDrive'
/content/drive/MyDrive
```

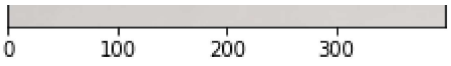
```
dir_path1 = 'ComputerVision/training/A'

for i in os.listdir(dir_path1):
  img = image.load_img(dir_path1 + '//' + i,target_size=(400,400))
  plt.imshow(img)
  plt.show()
```

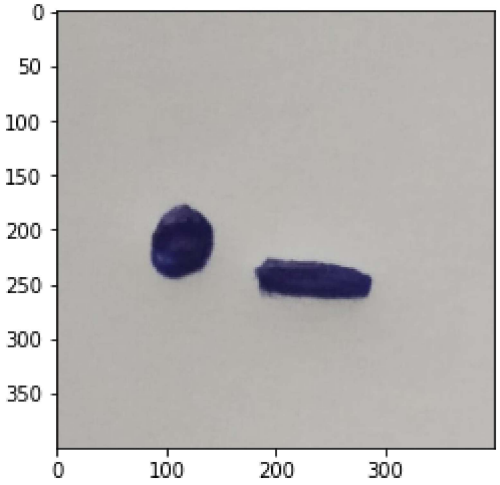

0
50
100
150
200
250
300
350
0
100
200
300

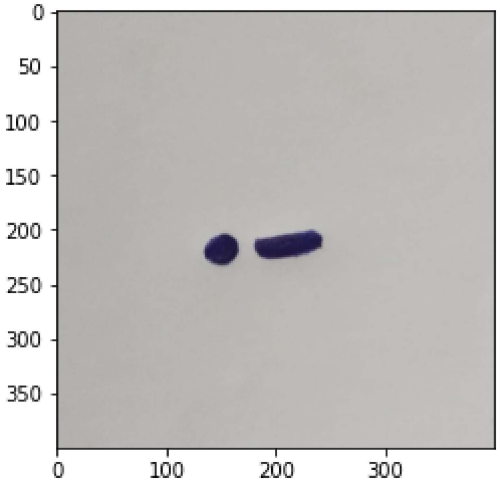

0
50
100
150
200
250
300
350
0
100
200
300

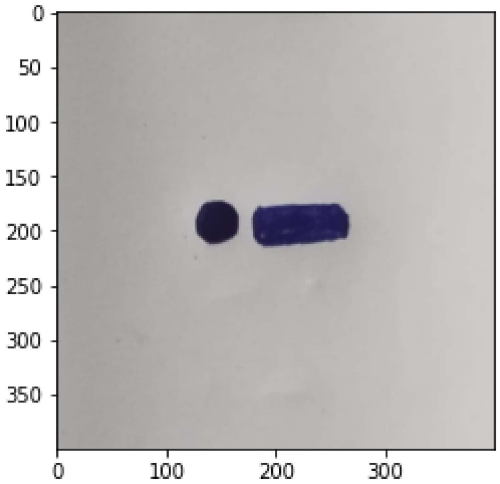

0
50
100
150
200
250
300
350
0
100
200
300

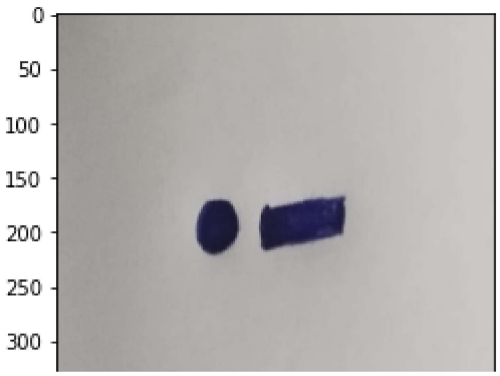

0
50
100
150
200
250
300
350

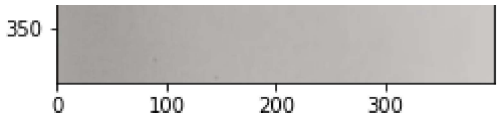

0
100
200
300

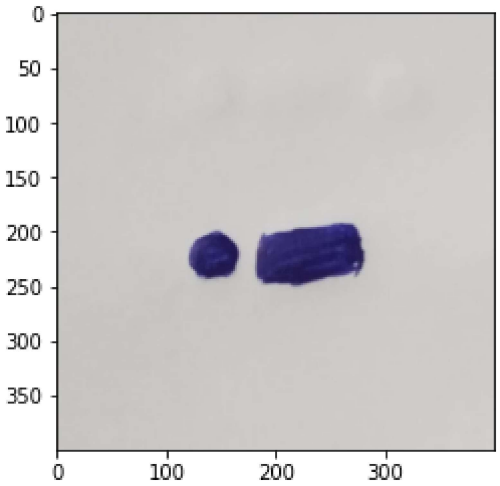

0
50
100
150
200
250
300
350
0
100
200
300

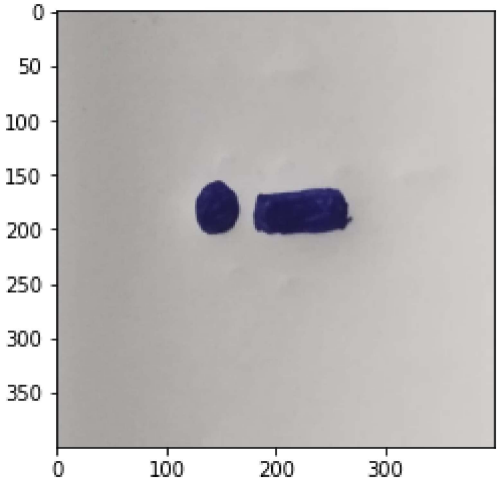

0
50
100
150
200
250
300
350
0
100
200
300

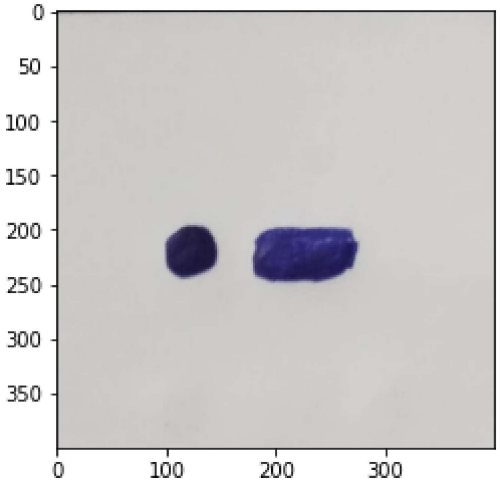

0
50
100
150
200
250
300
350
0
100
200
300


0
50
100
150
200
250
300

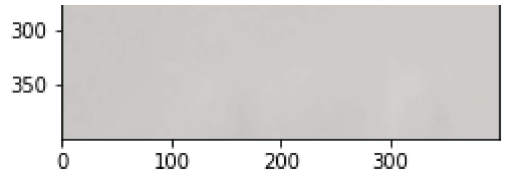

350
0
100
200
300

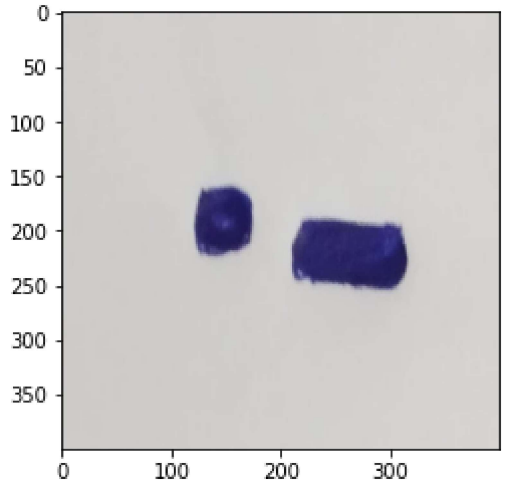

0
50
100
150
200
250
300
350
0
100
200
300

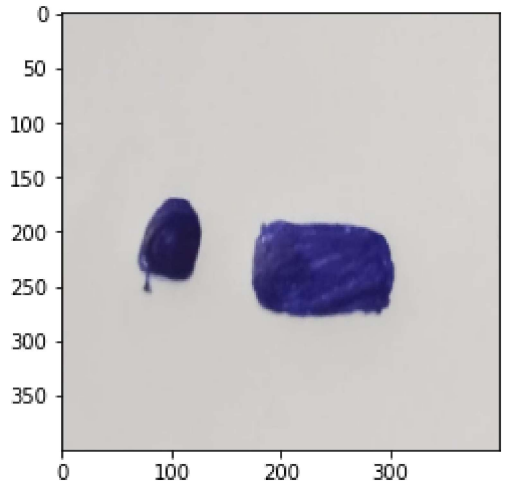

0
50
100
150
200
250
300
350
0
100
200
300

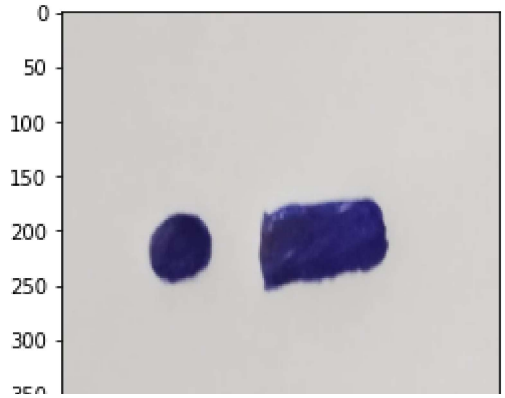

0
50
100
150
200
250
300
350
0
100
200
300


0
50
100
150
200
250

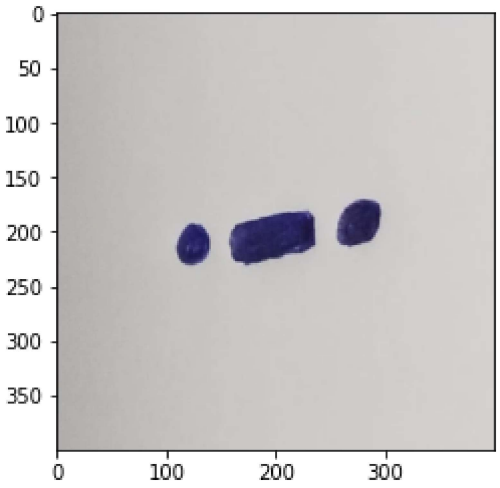


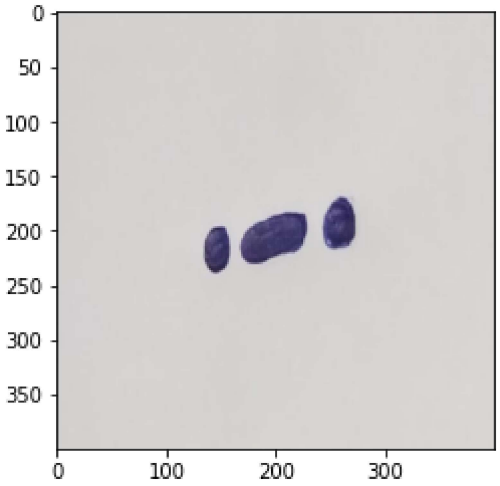


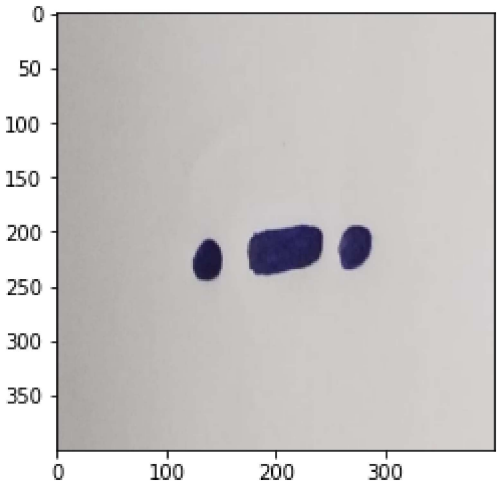


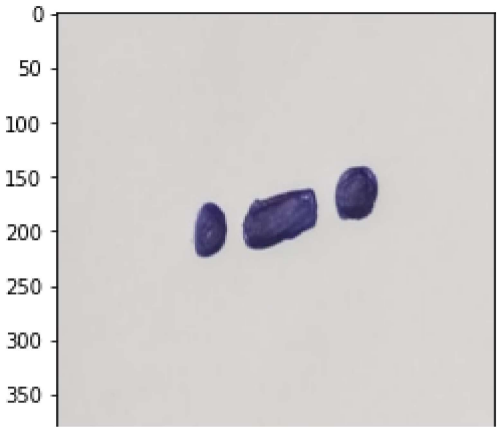


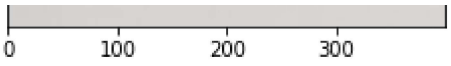

```
dir_path2 = 'ComputerVision/training/R'

for i in os.listdir(dir_path2):
  img = image.load_img(dir_path2 + '//' + i,target_size=(400,400))
  plt.imshow(img)
  plt.show()
```

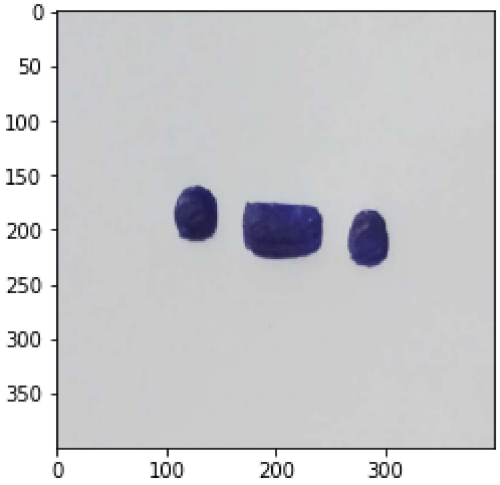

0
50
100
150
200
250
300
350
0
100
200
300

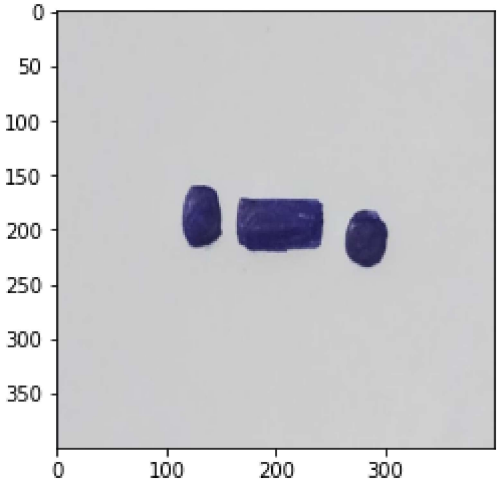

0
50
100
150
200
250
300
350
0
100
200
300

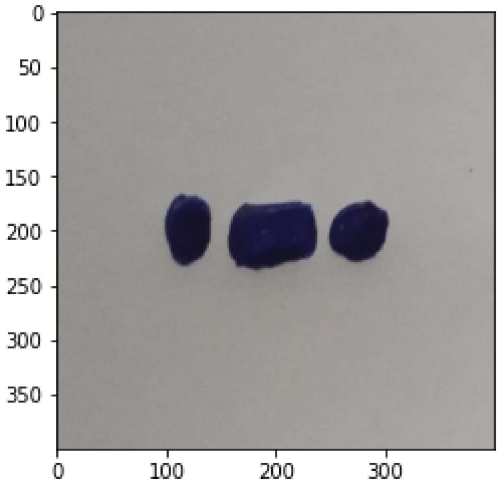

0
50
100
150
200
250
300
350
0
100
200
300

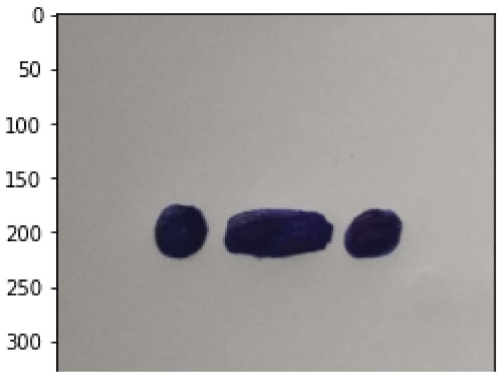

0
50
100
150
200
250
300
350

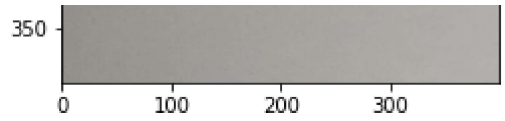

0
100
200
300

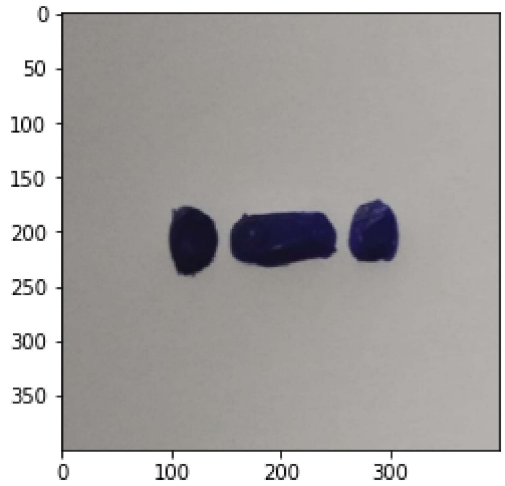

0
50
100
150
200
250
300
350
0
100
200
300

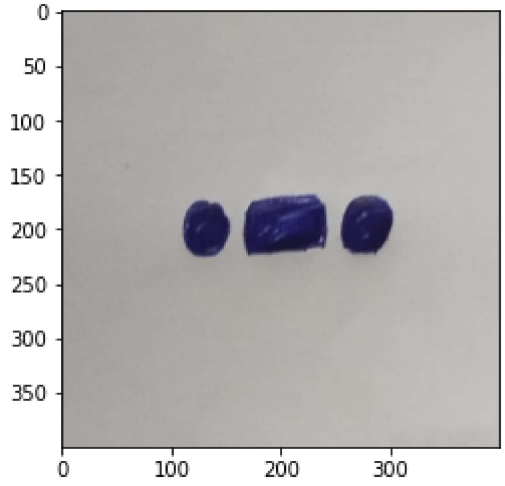

0
50
100
150
200
250
300
350
0
100
200
300

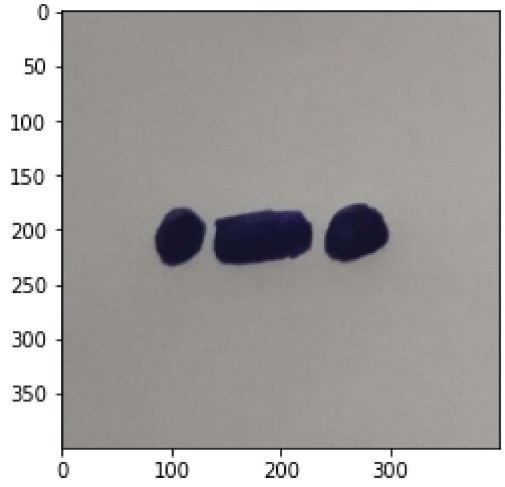

0
50
100
150
200
250
300
350
0
100
200
300


0
50
100
150
200
250
300


350
0
100
200
300


0
50
100
150
200
250
300
350
0
100
200
300


0
50
100
150
200
250
300
350
0
100
200
300


0
50
100
150
200
250
300
350
0
100
200
300


0
50
100
150
200
250


```
train = ImageDataGenerator(rescale = 1/255)
validation = ImageDataGenerator(rescale = 1/255)
```

```
train_dataset = train.flow_from_directory('ComputerVision/training',
                                          target_size=(400,400),
                                          batch_size = 1,
                                          class_mode='binary')
```

```
Found 28 images belonging to 2 classes.
```

```
validation_dataset = validation.flow_from_directory('ComputerVision/validation',
                                                    target_size = (400,400),
                                                    batch_size = 1,
                                                    class_mode = 'binary')
```

```
Found 28 images belonging to 2 classes.
```

```
train_dataset.class_indices
```

```
{'A': 0, 'R': 1}
```

```
train_dataset.classes
```

```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1], dtype=int32)
```

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense,Conv2D,MaxPool2D,Flatten
```

```
def policy_network():
  model = Sequential()
  model.add(Conv2D(16,(3,3),activation='relu',input_shape=(400,400,3)))
  model.add(MaxPool2D(2,2))

  model.add(Conv2D(32,(3,3),activation='relu'))
  model.add(MaxPool2D(2,2))

  model.add(Conv2D(64,(3,3),activation='relu'))
  model.add(MaxPool2D(2,2))

  model.add(Flatten())
```

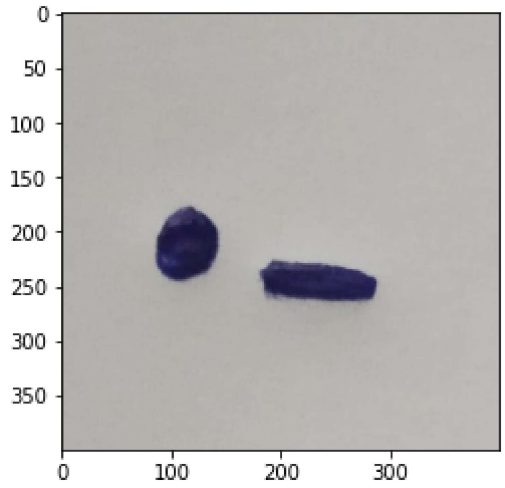
```
    model.add(Dense(512,activation='relu'))
    model.add(Dense(1,activation='sigmoid'))

    model.compile(loss='binary_crossentropy',optimizer='adam',metrics=['accuracy'])

    return model
```

```
model = policy_network()
model.fit(train_dataset,
          steps_per_epoch = 2,
          epochs = 100,
          validation_data = validation_dataset,
          shuffle=True)
```

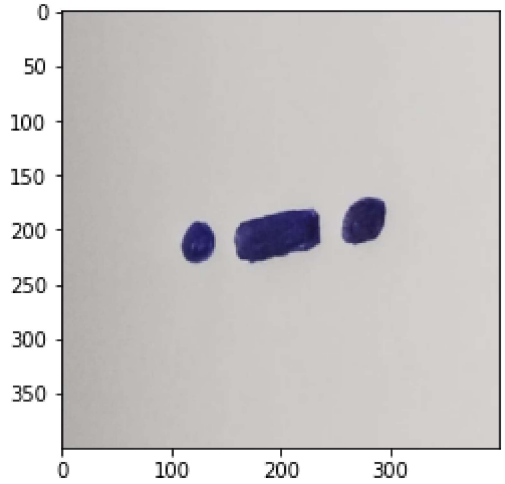
```
Epoch 73/100
2/2 [==============================] - 3s 3s/step - loss: 0.5780 - accuracy: 0.5000 -
Epoch 74/100
2/2 [==============================] - 3s 3s/step - loss: 0.4495 - accuracy: 0.5000 -
Epoch 75/100
2/2 [==============================] - 3s 3s/step - loss: 0.1815 - accuracy: 1.0000 -
Epoch 76/100
2/2 [==============================] - 3s 3s/step - loss: 8.4893e-04 - accuracy: 1.00
Epoch 77/100
2/2 [==============================] - 3s 3s/step - loss: 0.0603 - accuracy: 1.0000 -
Epoch 78/100
2/2 [==============================] - 3s 3s/step - loss: 0.5213 - accuracy: 0.5000 -
Epoch 79/100
2/2 [==============================] - 3s 3s/step - loss: 1.4828e-05 - accuracy: 1.00
Epoch 80/100
2/2 [==============================] - 3s 3s/step - loss: 0.5638 - accuracy: 0.5000 -
Epoch 81/100
2/2 [==============================] - 3s 3s/step - loss: 0.0069 - accuracy: 1.0000 -
Epoch 82/100
2/2 [==============================] - 3s 3s/step - loss: 0.0158 - accuracy: 1.0000 -
Epoch 83/100
2/2 [==============================] - 3s 3s/step - loss: 11.5422 - accuracy: 0.5000
Epoch 84/100
2/2 [==============================] - 3s 3s/step - loss: 0.0556 - accuracy: 1.0000 -
Epoch 85/100
2/2 [==============================] - 3s 3s/step - loss: 0.0150 - accuracy: 1.0000 -
Epoch 86/100
2/2 [==============================] - 3s 3s/step - loss: 0.0051 - accuracy: 1.0000 -
Epoch 87/100
2/2 [==============================] - 3s 3s/step - loss: 0.5600 - accuracy: 0.5000 -
Epoch 88/100
2/2 [==============================] - 3s 3s/step - loss: 0.4968 - accuracy: 0.5000 -
Epoch 89/100
2/2 [==============================] - 4s 3s/step - loss: 0.2230 - accuracy: 1.0000 -
Epoch 90/100
```

```
2/2 [==============================] - 3s 3s/step - loss: 0.7934 - accuracy: 0.5000 -
Epoch 91/100
2/2 [==============================] - 3s 3s/step - loss: 1.0313 - accuracy: 0.5000 -
Epoch 92/100
2/2 [==============================] - 3s 3s/step - loss: 0.5923 - accuracy: 0.5000 -
Epoch 93/100
2/2 [==============================] - 3s 3s/step - loss: 0.4990 - accuracy: 1.0000 -
Epoch 94/100
2/2 [==============================] - 3s 3s/step - loss: 0.6028 - accuracy: 0.5000 -
Epoch 95/100
2/2 [==============================] - 3s 3s/step - loss: 0.6089 - accuracy: 1.0000 -
Epoch 96/100
2/2 [==============================] - 3s 3s/step - loss: 0.3891 - accuracy: 1.0000 -
Epoch 97/100
2/2 [==============================] - 3s 3s/step - loss: 0.5573 - accuracy: 0.5000 -
Epoch 98/100
2/2 [==============================] - 3s 3s/step - loss: 0.5229 - accuracy: 0.5000 -
Epoch 99/100
2/2 [==============================] - 3s 3s/step - loss: 0.4314 - accuracy: 1.0000 -
Epoch 100/100
2/2 [==============================] - 3s 3s/step - loss: 0.5214 - accuracy: 0.5000 -
<keras.callbacks.History at 0x7f4ed1d46f10>
```

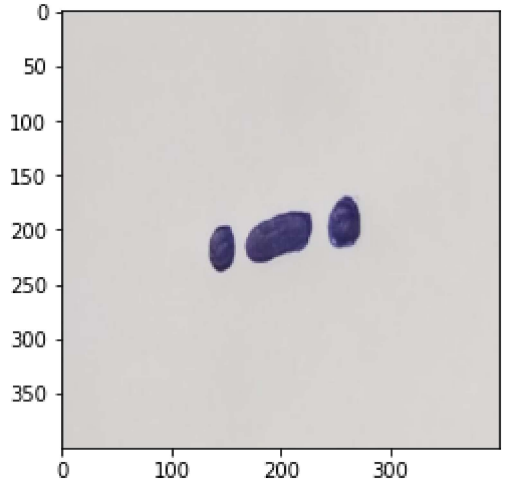
```
dir_path = 'ComputerVision/testing/'

for i in os.listdir(dir_path):
  img = image.load_img(dir_path + '//' + i,target_size=(400,400))
  plt.imshow(img)
  plt.show()

  X = image.img_to_array(img)
  X = np.expand_dims(X,axis=0)
  images = np.vstack([X])

  pred = model.predict(images)

  if(pred == 0):
    print("A")
  else:
    print("R")
```



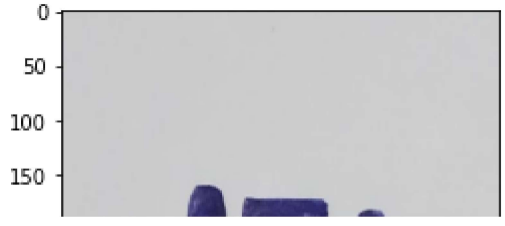
```
1/1 [==============================] - 0s 185ms/step
A
```

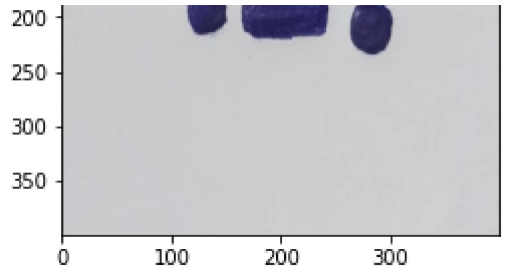


```
1/1 [==============================] - 0s 93ms/step
R
```

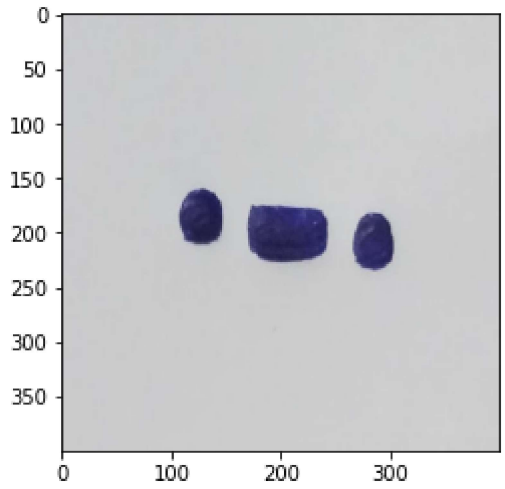


```
1/1 [==============================] - 0s 93ms/step
R
```

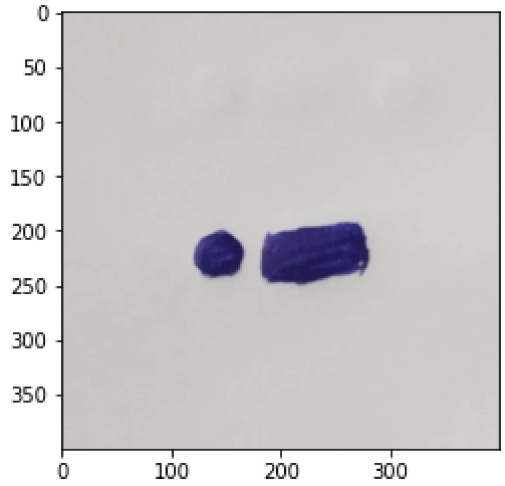




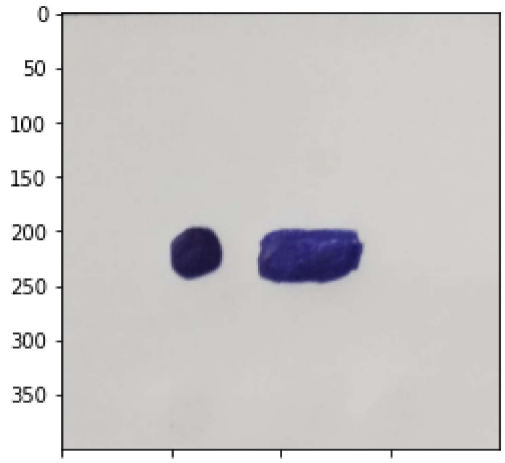

```
1/1 [==============================] - 0s 97ms/step
R
```

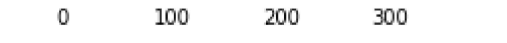



```
1/1 [==============================] - 0s 107ms/step
R
```

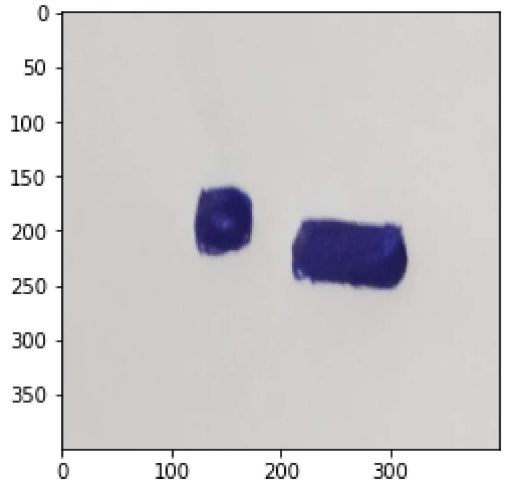



```
1/1 [==============================] - 0s 97ms/step
A
```

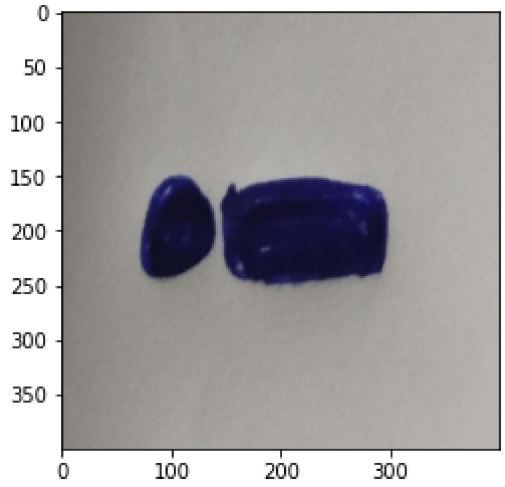




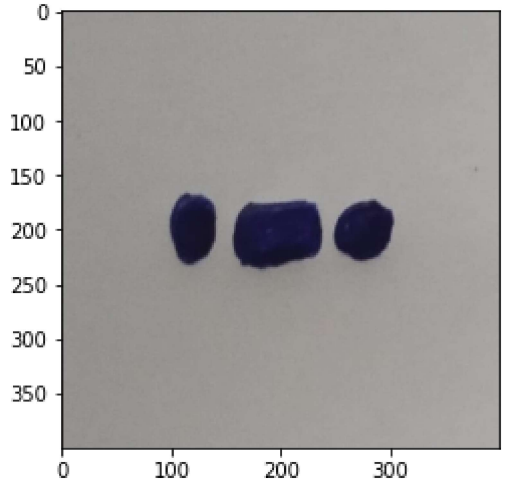

```
1/1 [==============================] - 0s 96ms/step
A
```



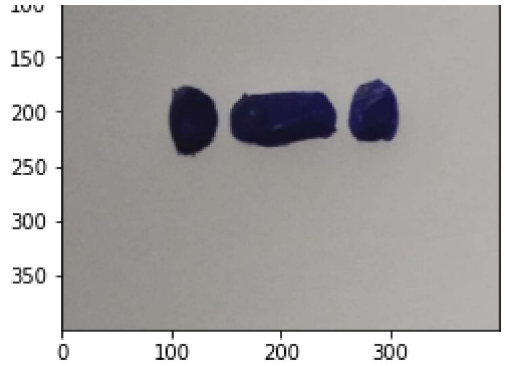

```
1/1 [==============================] - 0s 94ms/step
A
```



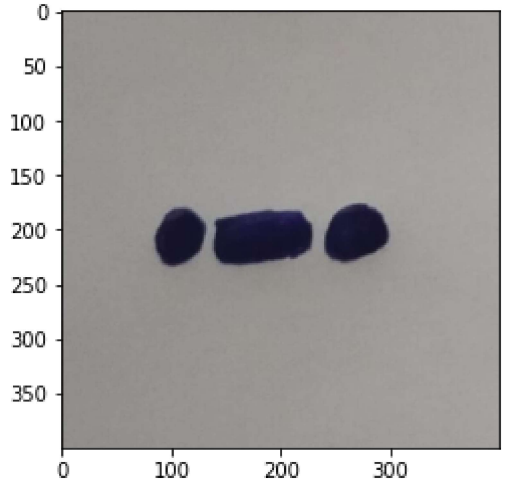

```
1/1 [==============================] - 0s 99ms/step
A
```



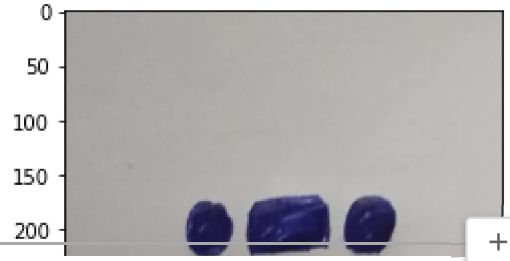

```
1/1 [==============================] - 0s 98ms/step
R
```



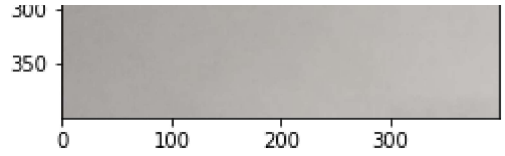

```
1/1 [==============================] - 0s 97ms/step
R
```




```
1/1 [==============================] - 0s 96ms/step
R
```



+ Code + Text




```
1/1 [==============================] - 0s 97ms/step
R
```

Colab paid products - Cancel contracts here

✓ 0s completed at 12:11 PM