Development of Computer Vision based POC for Engine part inspection using Deep Neural Network

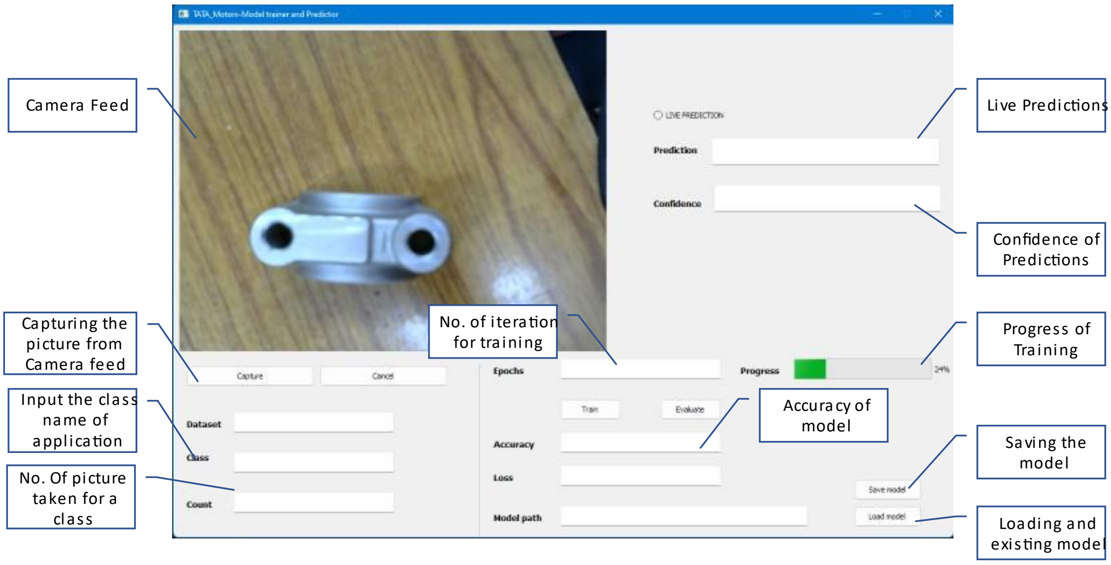
Configuration Process :-

1. To run python file download and install the following libraries with specified versions using python package manager (pip)

PyQt5	5.15.6
PyQt5-Qt5	5.15.2
PyQt5-sip	12.10.1
opencv-contrib-python	4.6.0.66
opencv-python	4.6.0.66
numpy	1.22.4

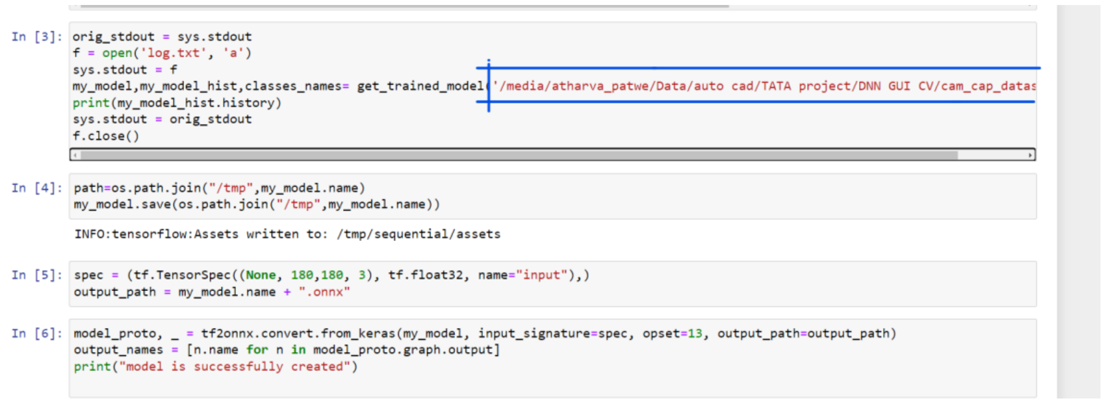
2. Connect camera to the PC/ Controller and run the file.

Developed GUI


TATA_Motors-Model trainer and Predictor
Camera Feed
○ LIVE PREDICTION
Prediction
Confidence
Live Predictions
Confidence of Predictions
Capturing the picture from Camera feed
No. of iteration for training
Progress of Training
Capture
Cancel
Epochs
Progress
24%
Input the class name of application
Dataset
Train
Evaluate
Accuracy of model
Class
Accuracy
Saving the model
No. Of picture taken for a class
Loss
Save model
Count
Model path
Load model
Loading and existing model

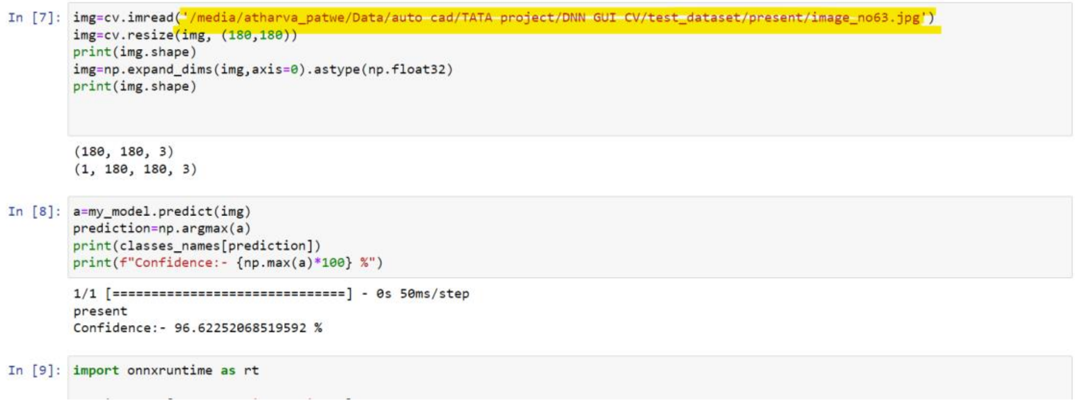
3. You should see the camera feed on the GUI
4. Type Dataset name for the object of interest
5. For classification type particular class name (presence/absence, Ok/NC, Good/Bad etc.)
6. Click on capture to capture the current frame. The pictures are then stored in the local directory of Dataset name.
7. Particular classes are stored in sub directories in the Dataset directory
8. Change the name of class to capture pictures of different class.
9. Copy the path of the dataset directory and put it in NN_model file for training.

10. Put the path of Directory in the function call highlighted in blue



```
In [3]: orig_stdout = sys.stdout
        f = open('log.txt', 'a')
        sys.stdout = f
        my_model,my_model_hist,classes_names= get_trained_model('/media/atharva_patwe/Data/auto cad/TATA project/DNN GUI CV/cam_cap_datas
        print(my_model_hist.history)
        sys.stdout = orig_stdout
        f.close()

In [4]: path=os.path.join("/tmp",my_model.name)
        my_model.save(os.path.join("/tmp",my_model.name))

        INFO:tensorflow:Assets written to: /tmp/sequential/assets

In [5]: spec = (tf.TensorSpec((None, 180,180, 3), tf.float32, name="input"),)
        output_path = my_model.name + ".onnx"

In [6]: model_proto, _ = tf2onnx.convert.from_keras(my_model, input_signature=spec, opset=13, output_path=output_path)
        output_names = [n.name for n in model_proto.graph.output]
        print("model is successfully created")
```

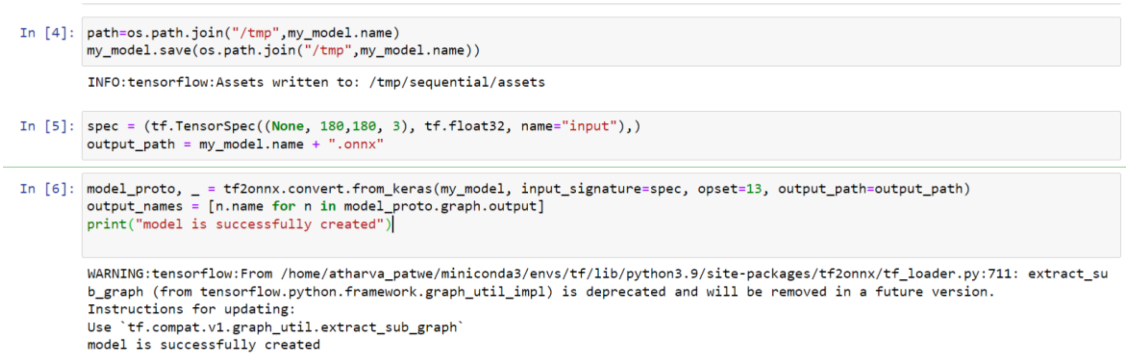
11. Put number of iterations to train for and batch size or leave it default
12. For inference put the input path to the function call highlighted in yellow



```
In [7]: img=cv.imread('/media/atharva_patwe/Data/auto cad/TATA project/DNN GUI CV/test_dataset/present/image_no63.jpg')
        img=cv.resize(img, (180,180))
        print(img.shape)
        img=np.expand_dims(img,axis=0).astype(np.float32)
        print(img.shape)

        (180, 180, 3)
        (1, 180, 180, 3)

In [8]: a=my_model.predict(img)
        prediction=np.argmax(a)
        print(classes_names[prediction])
        print(f"Confidence:- {np.max(a)*100} %")

        1/1 [==============================] - 0s 50ms/step
        present
        Confidence:- 96.62252068519592 %

In [9]: import onnxruntime as rt
```

13. Run model.predict(img) cell for inference

14. To convert the model into onnx format run following cells



```
In [4]: path=os.path.join("/tmp",my_model.name)
        my_model.save(os.path.join("/tmp",my_model.name))

        INFO:tensorflow:Assets written to: /tmp/sequential/assets

In [5]: spec = (tf.TensorSpec((None, 180,180, 3), tf.float32, name="input"),)
        output_path = my_model.name + ".onnx"

In [6]: model_proto, _ = tf2onnx.convert.from_keras(my_model, input_signature=spec, opset=13, output_path=output_path)
        output_names = [n.name for n in model_proto.graph.output]
        print("model is successfully created")

        WARNING:tensorflow:From /home/atharva_patwe/miniconda3/envs/tf/lib/python3.9/site-packages/tf2onnx/tf_loader.py:711: extract_su
        b_graph (from tensorflow.python.framework.graph_util_impl) is deprecated and will be removed in a future version.
        Instructions for updating:
        Use `tf.compat.v1.graph_util.extract_sub_graph`
        model is successfully created
```