Problem Statement:

Text recognition and extraction from images for a sample dataset

To complete this task, we implemented OCR (Optical Character Recognition). We took help of EasyOCR framework. It is an open-source user-friendly framework, written in Python and built on top of PyTorch. The EasyOCR framework allows researchers to quickly showcase their OCR models without the need to pre-process and post-process their data, greatly simplifying the model training and evaluation process.

Methodology:

To use this framework, we performed the following steps.

1.Install libraries:

Command:
```
pip install opencv-python
pip install easyocr
pip install pillow==9.5.0
```

2. Import libraries:

Command:
```
import easyocr
import cv2 as cv
import os
```

3. Load the Model (Optional):

Before performing OCR, we have the option to load the model into memory. We can load the model by specifying the languages we want to read. We can also choose which detectors and recognizers to use in this step. These models can be independently trained and then plugged into the framework with ease.

Command:
```
reader = easyocr.Reader(['en'])
```

In this example, the reader will be able to read English text and will use the default model (CRAFT) for detection and their standard recognition model.

4. Perform OCR:

To extract text from an image, use the `readtext()` function of the `Reader` object you created. Pass the image path or an OpenCV image object (numpy array) to this function.

Command: result = reader.readtext('path/to/your/image.jpg')

The `result` variable will store the output of OCR, which will be a list of text detections along with their bounding boxes and confidence levels.

5. Process the Output:

The output of `readtext()` will be in a list format, with each item representing a text detection. Each detection is a tuple containing the bounding box coordinates, the detected text, and the confidence level. We can access this information and process the text as needed.

6. Optional Configurations:

EasyOCR provides some optional configurations, such as setting `detail=0` for simpler output without bounding boxes and confidence levels or specifying `gpu=False` to run the model in CPU-only mode.

Output: The output is an image with bounding boxes around the text identified in the image and the text in the command line output.

Thus, we successfully were able to extract text from a sample dataset using OCR.

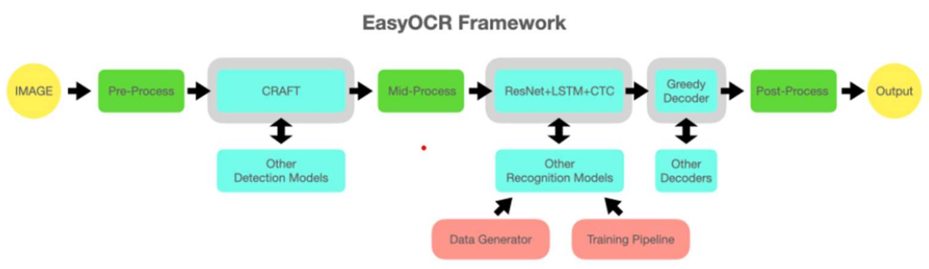
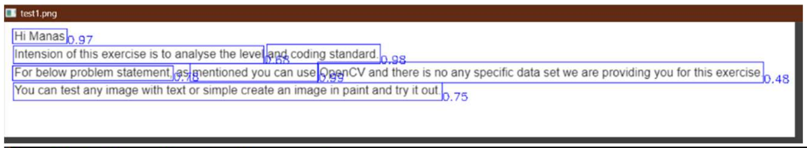


Fig: EasyOCR framework: The detection model, recognition model and decoders (in light blue) can be replaced with customized models and decoders trained by the user.

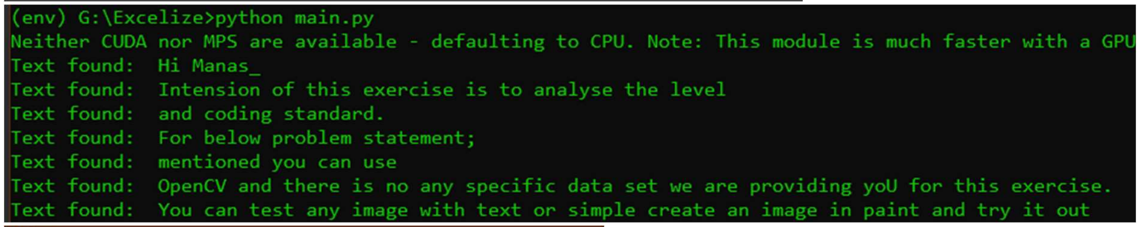
Sample Output:

1)

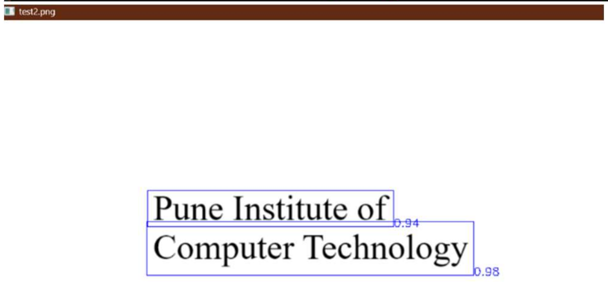


Hi Manas
Intension of this exercise is to analyse the level and coding standard.
For below problem statement, as mentioned you can use OpenCV and there is no any specific data set we are providing you for this exercise
You can test any image with text or simple create an image in paint and try it out

2)



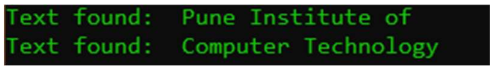
```
(env) G:\Excelize>python main.py
Neither CUDA nor MPS are available - defaulting to CPU. Note: This module is much faster with a GPU
Text found:  Hi Manas_
Text found:  Intension of this exercise is to analyse the level
Text found:  and coding standard.
Text found:  For below problem statement;
Text found:  mentioned you can use
Text found:  OpenCV and there is no any specific data set we are providing yoU for this exercise.
Text found:  You can test any image with text or simple create an image in paint and try it out
```



Pune Institute of
Computer Technology

3)

4)



```
Text found:  Pune Institute of
Text found:  Computer Technology
```

Steps to run the project:

1. **Install requirements:** Use pip command to install all necessary dependencies.

   pip install -r requirements.txt

2. **Run main.py file:** Use python command to run the main.py file

   Python main.py

.