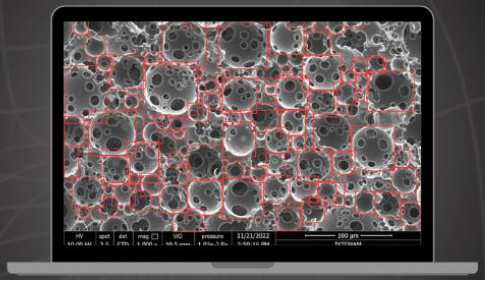
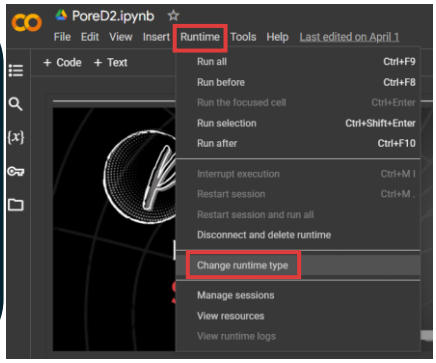
# PoreD² USER GUIDE

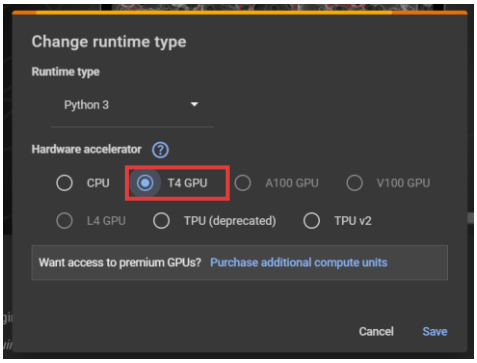


You can open the PoreD² app by clicking this box.

Firstly, be sure that you are using GPU as runtime type. To check this click on (1) “Runtime“ then (2) “Change runtime type“.

In openned pop-up page click on (3) “GPU“ option and save the preferences.



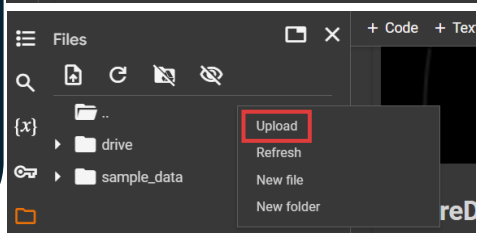

```
[ ] !pip install easyocr

[ ] import torch
    import torchvision.transforms as transforms
    from PIL import Image
    import openpyxl
    import pandas as pd
    import easyocr
    import matplotlib.pyplot as plt
    import cv2
    import openpyxl
    from openpyxl.utils import get_column_letter
    from openpyxl.styles import PatternFill
    import requests
    import os
    import glob

[ ] !git clone https://github.com/ultralytics/yolov5  # clone
    %cd yolov5
    %pip install -qr requirements.txt  # install
    from yolov5 import utils
    display = utils.notebook_init()  # checks
```

Then, run the first 3 cell (on the left) to install and import all required libaries.

During this process, you can upload your image folder as “.zip“ file to the folder section which is shown on the right. (On the folder section, right-click to uploading.)


```
[ ] url1 = 'https://github.com/ilaydakaraca/PoreD2/raw/main/weights/bestpore.pt?download='
    r1 = requests.get(url1, allow_redirects=True)
    open('bestpore.pt', 'wb').write(r1.content)

    url2 = 'https://github.com/ilaydakaraca/PoreD2/raw/main/weights/best_scale_bar.pt?download='
    r2 = requests.get(url2, allow_redirects=True)
    open('best_scale_bar.pt', 'wb').write(r2.content)

    url3 = 'https://github.com/ilaydakaraca/PoreD2/raw/main/weights/best_window.pt?download='
    r3 = requests.get(url3, allow_redirects=True)
    open('best_window.pt', 'wb').write(r3.content)

[ ] model = torch.hub.load('ultralytics/yolov5', 'custom', path='/content/yolov5/bestpore.pt')
    model2 = torch.hub.load('ultralytics/yolov5', 'custom', path='/content/yolov5/best_scale_bar.pt')
    model3 = torch.hub.load('ultralytics/yolov5', 'custom', path='/content/yolov5/best_window.pt')
```

Then, run the following 2 cells (shown on the left) for uploading and defining the trained weights.

- *Run the cell for Pore detection*

```
[ ] #!unzip "path_to_your_zip_file" -d "/content"
    output_folder = ''
    directory = 'path_to_your_folder'
    jpg_files = glob.glob(os.path.join(directory, "*.jpg"))
    image_names = [os.path.basename(file) for file in jpg_files]
    for image_name in image_names:
        image = os.path.join(directory, image_name)
        results = model(image)
        results.save(output_folder, "/content/yolov5/pore/image")
        output_path = os.path.join(output_folder, image_name)
        results.print()
        results.pandas().xywh[0].to_excel("/content/yolov5/pore/"+output_path + "_pore.xlsx")
```

Run the following 3 cells for pore, scale bar and window detection.

**Note1:** If you upload the image folder as “.zip“ file, delete the “Hashtag (#)“ sign and change the “path_to_your_zip_file“ part.
**Note2:** Do not forget the change the path for directory (which is path for your image folder (not “.zip“ file.))

- *Run the cell for Scale bar detection*

```
[ ] output_folder2 = ''
    for image_name in image_names:
        image = os.path.join(directory, image_name)
        results2 = model2(image)
        results2.save(output_folder2, "/content/yolov5/scalebar/image")
        output_path = os.path.join(output_folder2, image_name)
        results2.print()
        results2.pandas().xywh[0].to_excel("/content/yolov5/scalebar/"+output_path + "_scalebar.xlsx")
```

- *Run the cell for window detection*

```
[ ] output_folder3 = ''
    for image_name in image_names:
        # Construct the full path to the image
        image = os.path.join(directory, image_name)
        results3 = model3(image)
        results3.save(output_folder3, "/content/yolov5/window/image")
        output_path = os.path.join(output_folder3, image_name)
        results3.print()
        results3.pandas().xywh[0].to_excel("/content/yolov5/window/"+output_path + "_window.xlsx")
```

Up to this part, the object detection of the images has been completed, and after this, the editing of the obtained excel results into desired format and the text recognition part will be carried out.

Run the following cell for defining the excel files obtained from the detection.

- *Following cells are for defining files obtained from detection*

```
import os

# Define the input file directories
input_directory = "/content/yolov5/pore/"
input_directory2 = "/content/yolov5/scalebar/"
input_directory3 = "/content/yolov5/window/"

# List all Excel files in each directory
input_file = [os.path.join(input_directory, file)
              for file in os.listdir(input_directory)
              if file.endswith('.xlsx') or file.endswith('.xls')]


input_file2 = [os.path.join(input_directory2, file)
              for file in os.listdir(input_directory2)
              if file.endswith('.xlsx') or file.endswith('.xls')]

input_file3 = [os.path.join(input_directory3, file)
              for file in os.listdir(input_directory3)
              if file.endswith('.xlsx') or file.endswith('.xls')]
```

- *Run the following cell for loading the easyocr model for text detection*

```
reader = easyocr.Reader(['en']) # need to run only once to load model into memory
```

1st cell

- Run the following cell for performing text detection and saving as a '.txt' file
- Check if the output is correct

```
result_text = reader.readtext(image)
text = result_text[7][1]
print(text)
with open('text.txt', 'w') as f:
    print(text, file=f)
with open(text_file_path, 'r') as text_file:
    text_data = text_file.read()
```

2nd cell

```
100 um
```

- If the output of the previous cell is correct, continue without running the next cell.
- If is not, than wrote the correct information between the Quotation mark below. Then, delete the hashtag (#) symbol and run the cell.

```
#text_data = "50 um"
```

3rd cell

This part is for text recognition. Run the first cell for loading the easyocr model which will be use for the text recognition.

Then run the second cell for the recognition. After running this cell, the output of the recognition will be demonstrated at the bottom (shown in red box). Check the output, and if it is correct continue without running the 3rd cell. However, if it is not correct, delete the “Hashtag sign (#)” and replace the “50 µm” part with the correct value and run the 3rd cell.

Run the following cell to getting the size of the scale bar in px from obtained excel files after detection.

```
def load_workbook(file_path):
    try:
        wb = openpyxl.load_workbook(file_path)
        ws = wb.active
        return wb, ws
    except Exception as e:
        print(f"Error loading workbook from {file_path}: {e}")
        return None, None

for file_path in input_file2:
    input2_wb, input2_ws = load_workbook(file_path)
    if input2_wb and input2_ws:
      input_value = input2_ws['D2'].value
```

Run the following two cells starting with #for pore and #for window to obtain desired excel files. (which includes ex. average window size, average pore size, Degree of interconnectivity)

```
#for pore
for file_path in input_file:
    input_wb, input_ws = load_workbook(file_path)
    if input_wb and input_ws:


      input_ws['N1'] = 'Avg. Pore Size (px)'
      input_ws['O1'] = 'Scale Bar Size (px)'
      input_ws['P1'] = 'Avg. Pore Size (µm)'
      input_ws['Q1'] = 'Avg. Pore Size (µm) w/cf'
      input_ws['R1'] = 'Scale Bar Text'
      input_ws['S1'] = 'Scale Bar #'
      input_ws['T1'] = 'Scale Bar Unit'

      target_cell = input_ws['R2']  # Replace 'A1' with the desired cell reference
      target_cell.value = text_data

      total_rows = input_ws.max_row
      for row_number in range(total_rows, 1, -1):
          d_value = input_ws.cell(row=row_number, column=4).value  # D column (4th column) value
          e_value = input_ws.cell(row=row_number, column=5).value  # E column (5th column) value

          # Compare the values and get the larger one
          if d_value and e_value:
              larger_value = max(d_value, e_value)

              # Write the larger value to column K
              input_ws.cell(row=row_number, column=11, value=larger_value)
```

```
#for window
for file_path in input_file3:
    input3_wb, input3_ws = load_workbook(file_path)
    if input3_wb and input3_ws:


      input3_ws['M1'] = 'Avg. Window Size (px)'
      input3_ws['N1'] = 'Scale Bar Size (px)'
      input3_ws['O1'] = 'Avg. Window Size (µm)'
      input3_ws['P1'] = 'Scale Bar Text'
      input3_ws['Q1'] = 'Scale Bar #'
      input3_ws['R1'] = 'Scale Bar Unit'

      target_cell = input3_ws['P2']  # Replace 'A1' with the desired cell reference
      target_cell.value = text_data

      total_rows = input3_ws.max_row
      for row_number in range(total_rows, 1, -1):
          d_value = input3_ws.cell(row=row_number, column=4).value  # D column (4th column) value
          e_value = input3_ws.cell(row=row_number, column=5).value  # E column (5th column) value

          # Compare the values and get the larger one
          if d_value and e_value:
              larger_value = max(d_value, e_value)

              # Write the larger value to column K
              input3_ws.cell(row=row_number, column=11, value=larger_value)
```

Run the following cell to obtain all the results in a “.zip“ file. The results will be saved as output.zip file which can be found under the folder section of Colab.

```
import os
from zipfile import ZipFile

!zip -r /content/pore ///content/yolov5/pore/
!zip -r /content/scalebar ///content/yolov5/scalebar/
!zip -r /content/window ///content/yolov5/window/

# Create a ZipFile Object
with ZipFile('/content/outputs.zip', 'w') as zip_object:
   # Adding files that need to be zipped
   zip_object.write('/content/pore.zip')
   zip_object.write('/content/scalebar.zip')
   zip_object.write('/content/window.zip')


# Check to see if the zip file is created
if os.path.exists('/content/outputs.zip'):
   print("ZIP file created")
else:
   print("ZIP file not created")
```