Git hub: <https://github.com/redwankarimsony/project-tomato>

* **File không cần**

Enhancements.py

Inference.py

* **Thứ tự file py**

Utils.py

Dataset\_preparation.py

Dataset.py

Download\_dataset.py

Model.py

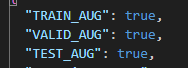
Train.py

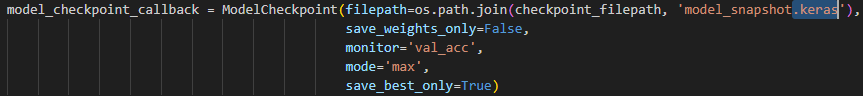
Evaluate.py

**Thực thi**

1. Chuẩn bị: (đưa các file từ github qua project của mình)

* **utils.py** 
  + Dùng để **lưu trữ thông tin huấn luyện** và **vẽ đồ thị** tóm tắt hiệu quả huấn luyện của mô hình

**Chú ý:   
1.** chỉnh sửa các biến “TRAIN\_AUG”, “VALID\_AUG”, “TEST\_AUG” thành “True” để kích hoạt tăng cường dữ liệu trên các mục train, val và test  
  
  
  
2. Chỉnh sửa “**model\_snapshot.ckpt**” thành “**model\_snapshot.keras**” của “**ModelCheckpoint**” của function “**load\_callbacks(config)**”



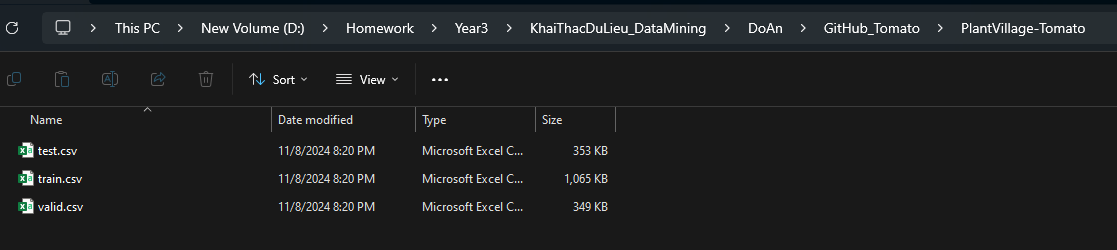
* **config.json**
  + Dùng để dễ dàng thay đổi các tham số mà không phải chỉnh sửa mã nguồn trực tiếp
* **saved\_models (folder)**
  + lấy mô hình retrain (MobileNet)

1. **Dataset\_preparation.py**  
   mục đích của file này dùng để tạo ra 3 file csv:
   * train.csv
   * val.csv
   * test.csv

nội dung của csv sẽ bao gồm 3 mục “filepath”, “label”, “label\_tag” dùng để bước 2 thực hiện chia file. Thực thi thông qua file **“class\_mapping.json ”**

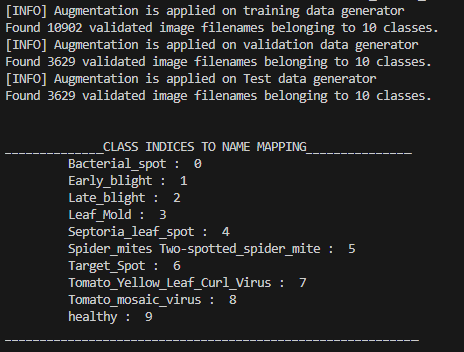
NOTE: class\_mapping.json ở github không có nên ta tiến hành skip bằng cách copy

* + train.csv
  + val.csv
  + test.csv

Vào thư mục có tên “**PlantVillage-Tomato**” bên đồ án của mình  


1. **dataset.py**

* Dùng để thiết lập và trả về các bộ sinh dữ liệu (data generators)
* Phục vụ cho quá trình huấn luyện, xác thực, và kiểm tra mô hình học máy
* Kết quả thực thi (kiểm tra là chính)



(những con số nghĩa là gán nhãn cho từng cây)

1. **download\_dataset.py** (tải dataset về máy)

Trước khi qua bước này, cần làm trước điều sau:

Copy thư mục “**splits**” bên githud sang project của mình (giải thích sau)

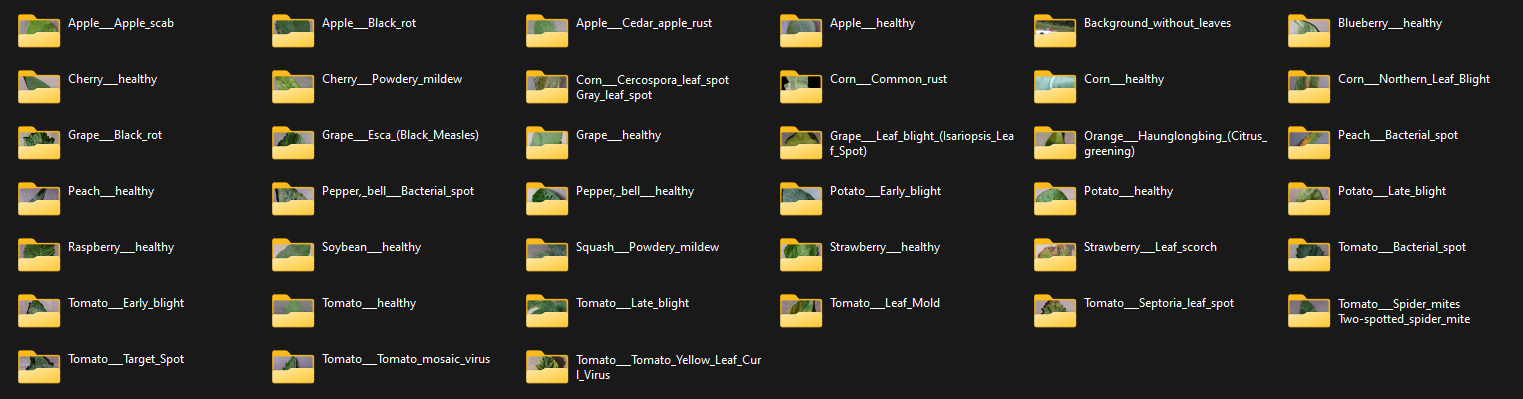
* **downloadFile (datasetUrl= datasetUrl, fileName=fileName)**
  + Màn hình tải



* + Sẽ tải về file zip có tên (PlantVillage-Tomato.zip)

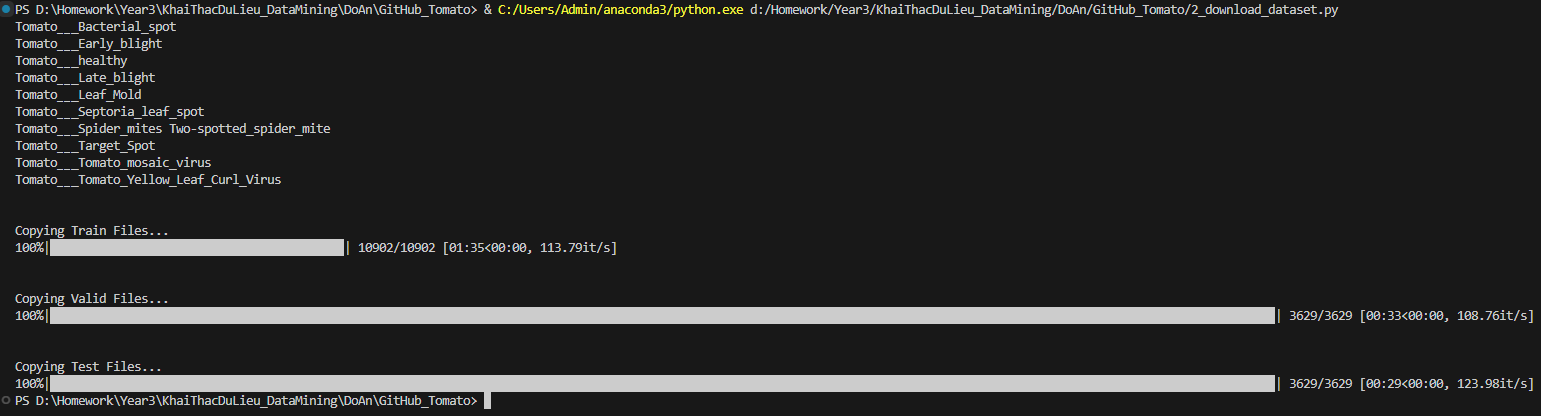


* **extractZip** **(filename="PlantVillage-Tomato.zip")**
  + Tạo ra 1 folder có tên (Plant\_leave\_diseases\_dataset\_without\_augmentation) 

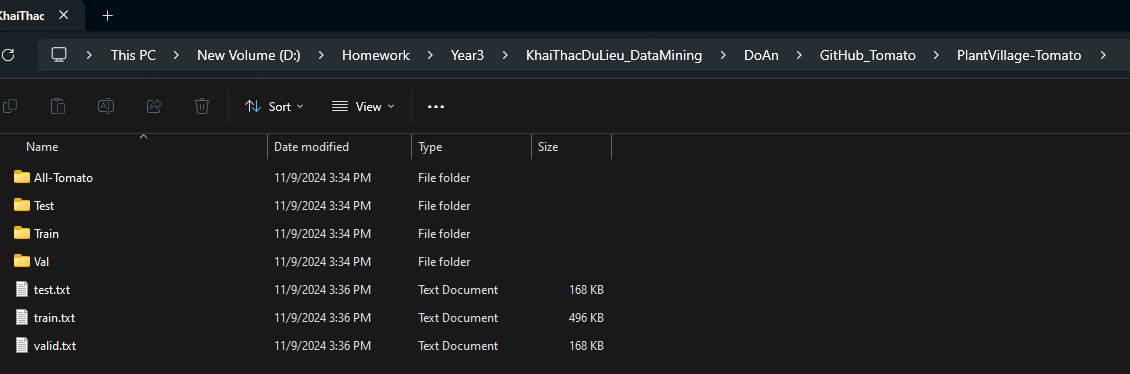


* **scanOldDataset()**
  + duyệt qua ảnh trong các thư mục “Train”, “Val”, “Test” trong thư mục “PlantVillage-Tomato”.
  + Từ bước trên, Tạo ra “train.txt”, “valid.txt”, “test.txt” vào thư mục tên là “splits” để thuận lợi trong việc duyệt
  + (Dữ liệu đã có sẵn từ trc)
* Hàm này mục đích để load lại dữ liệu cũ (train, test, val) thành 3 file txt rồi đưa vào **splits** để push lên github vì github giới hạn dung lượng
* Thông qua hàm **arrangeDataset()** ta có thể tiến hành lấy lại 3 thư mục (train, val, test)
* Hàm này thực thi trên máy của Admin (người push lên github) không phải của người clone từ github
* **arrangeDataset()**
  + Đọc 3 file txt từ thư mục “**splits**” để tiến hành tạo ra 3 thư mục train, test, val trên thư mục “**PlantVillage-Tomato**” để tiến hành train

Kết quả thực thi hàm:



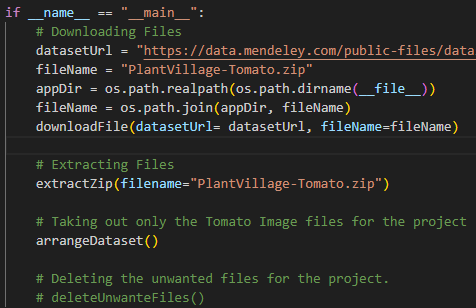
* Tạo thành công 3 mục train, test, val trên “**PlantVillage-Tomato**”



* **deleteUnwanteFiles ()**
  + Hàm này đơn giản chỉ là xoá file đã tải

**TÓM Lại ở file** download\_dataset.py

1. Copy thư mục “splits” bên githud sang project của mình
2. Rồi chạy các hàm sau:

****

*(deleteUnwanteFiles () enable hay ko điều đc)*



*(nếu có deleteUnwanteFiles ())*

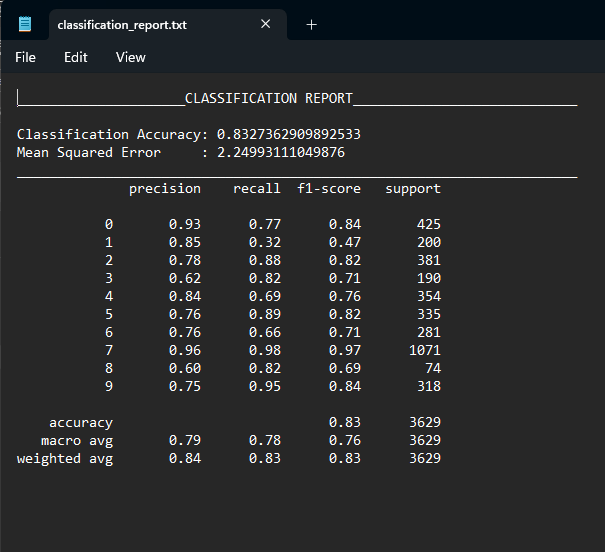
1. **model.py** (tải pretrained model \_ mobilenetv2)
2. **train.py** (train model)
   * **Chú ý:** Chỉnh sửa những file sau để tiến hành train model
     + **Dataset.py**

trong hàm “**flow\_from\_dataframe**” của “**train\_generator**”, “**valid\_generator**”, “**test\_generator**” tiến hành đổi **class\_mode='sparse' -> class\_mode='categorical'**

Mục đích:

* + 1. “**spare**” nhận giá trị là float với nhãn mà nhãn là 1 giá trị phải là nguyên trong đoạn [1, 10])
    2. “**categorical**” các nhãn sẽ được chuyển thành one-hot encoding và giữ kiểu nguyên

1. evaluate.py (gốc)
   * **calc\_accuracy ()**
     + Tính toán dộ chính xác của model
   * **evaluate ()**
     + đánh giá model
     + xuất file classification\_report.txt



* + **plot\_confusion\_matrix ()**
    - lưu 4.confusion-matrix.pdf  
      
  + **find\_misclassified ()**
    - tạo thư mục misclassifiedd
      * Phát hiện các mẫu dữ liệu mà mô hình dự đoán sai.
      * Tạo trực quan các lỗi để dễ dàng phân tích và cải thiện mô hình.
      * Lưu trữ các **hình ảnh bị dự đoán sai** kèm thông tin rõ ràng về nhãn thực tế và nhãn dự đoán.

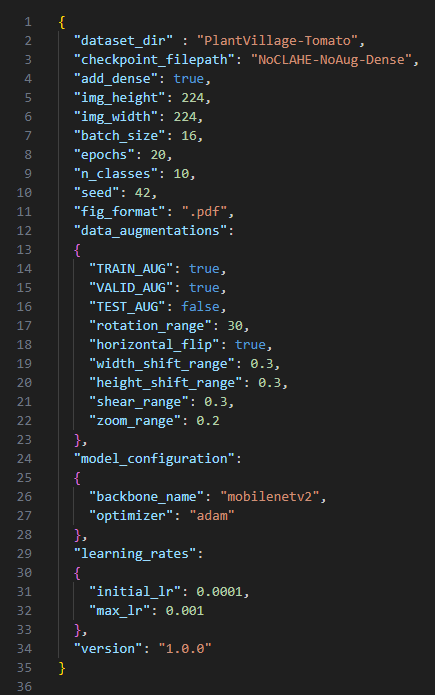




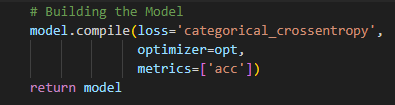
-------------------------------------------------- Xử lý train (cải thiện accurate) -----------------------------------------------

**1. Thay đổi file**

* **config.json**
  + kích hoạt "add\_dense": true
  + Thay đổi "data\_augmentations"
    - Tăng epochs: 20
    - "TRAIN\_AUG": true,
    - "VALID\_AUG": true
    - "rotation\_range": 30
    - "width\_shift\_range": 0.3
    - "height\_shift\_range": 0.3
    - "shear\_range": 0.3
    - Thêm tăng cương "zoom\_range": 0.2
  + Giảm "learning\_rates":
    - "initial\_lr": 0.0001
    - "max\_lr": 0.001



* **model.py**
  + ở dòng 83, loại bỏ **'mse'** ở metrics (mse không cần dùng, có mse có thể giảm độ chính xác)



* **train.py**
  + thêm code này trước hàm save\_training\_history(train\_history, config) để thích hợp với setting mặc định của file until.py (muốn thay đổi file train.py hơn là thay đổi file until.py)

history = train\_history.history

    max\_length = max(len(v) for v in history.values())

    for key, value in history.items():

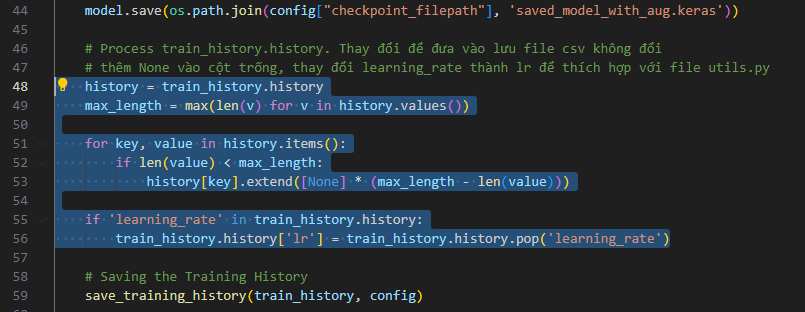
        if len(value) < max\_length:

            history[key].extend([None] \* (max\_length - len(value)))

    if 'learning\_rate' in train\_history.history:

        train\_history.history['lr'] = train\_history.history.pop('learning\_rate')

Ví dụ cụ thể:



* + Các thay đổi:
    - thêm None vào cột trống
    - thay đổi learning\_rate thành lr

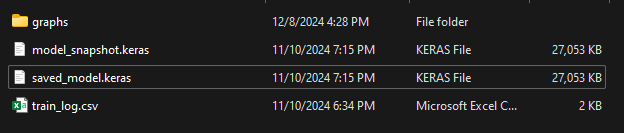
**2. Kết quả sau khi xử lý**

**2.1. Những gì cần trước khi train**

* Tại thư mục **NoCLAHE-NoAug-Dense** (nơi lưu file train.keras) thực thi: ****
  + Tạo mới 1 thư mục có tên là “**graphs**” (để lưu đồ thị). Không có thư mục này chạy code sẽ lỗi

**2.2 Kết quả train (train.py, evaluate.py)**

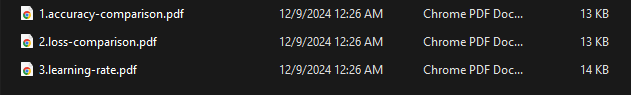
* Khi chạy xong train.py:
  + Sẽ có các thư mục và file sau:



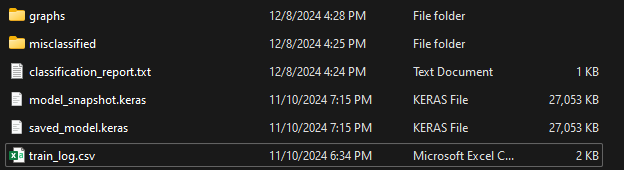
- Thứ tự tạo khi chạy (train.py)

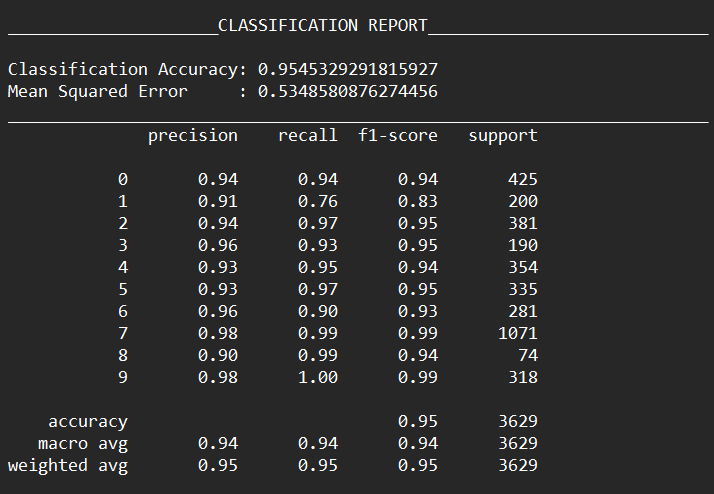
Model\_snapshot.keras -> saved\_model.keras -> train\_log.csv -> graphs (folder)

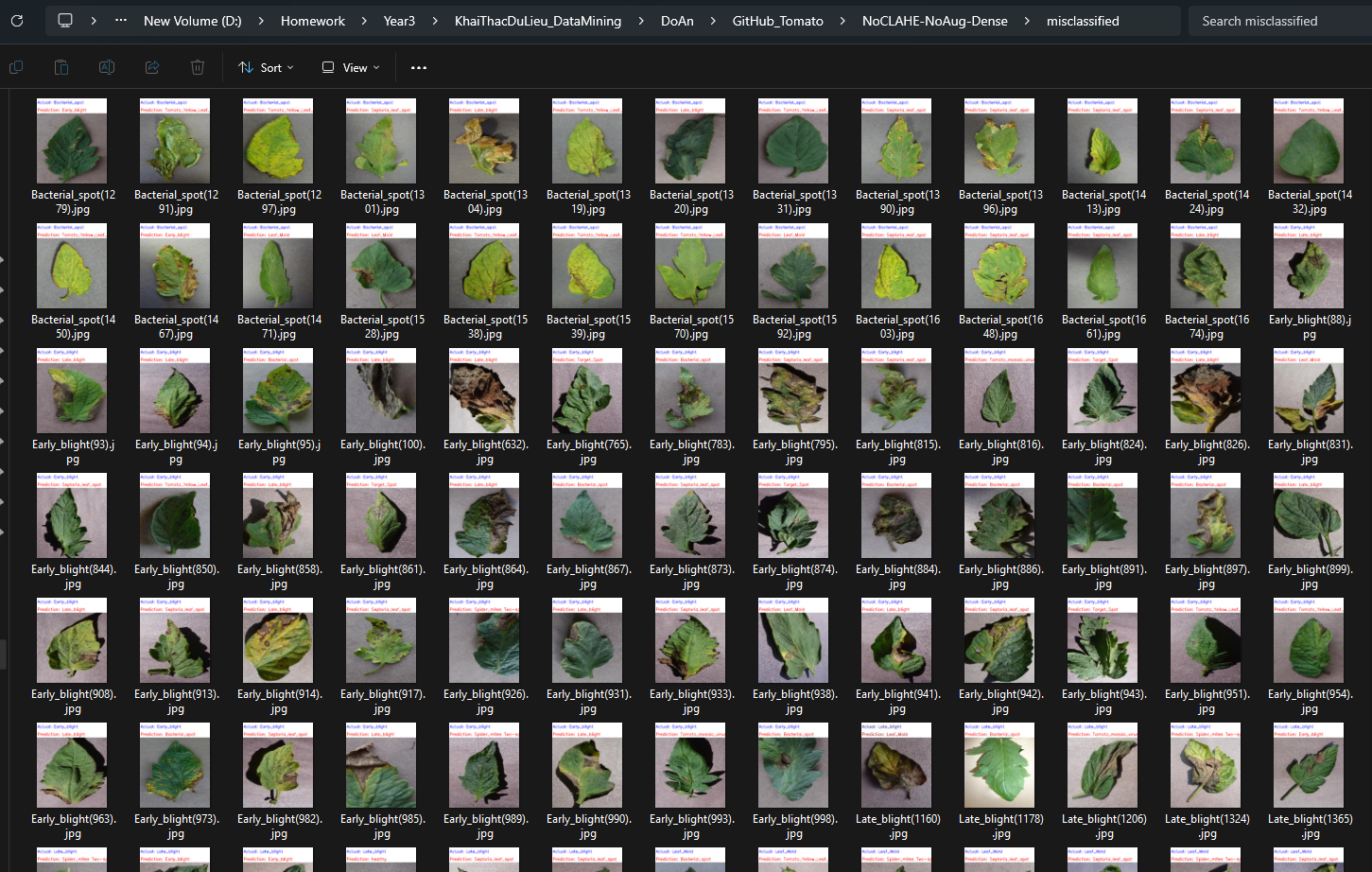
* + Train\_log.csv
    - Lưu các thông tin train
  + graphs (folder) bao gồm:



* Khi chạy xong evaluate.py:
  + Tạo thêm classification\_report.txt và thư mục misclassified và confusion-matrix.pdf vào thư mục graphs



* + - classification\_report.txt 
    - lưu 4.confusion-matrix.pdf  
      
  + thư mục misclassifiedd
    - Phát hiện các mẫu dữ liệu mà mô hình dự đoán sai.
    - Tạo trực quan các lỗi để dễ dàng phân tích và cải thiện mô hình.
    - Lưu trữ các **hình ảnh bị dự đoán sai** kèm thông tin rõ ràng về nhãn thực tế và nhãn dự đoán.





* Kết quả chạy (chỉ là tạm chứ ko phải cuối cùng):

