Huấn luyện các khuôn mặt chưa biết

# 1. Giới thiệu

Trong phần này, chúng ta sẽ xây dựng 1 chức năng để xử lý việc nhận diện sai trong hệ thống. Để thực hiện điều này, bạn sử dụng một loạt các phương thức và biến trong mã dưới đây.

* Lưu ý: Cần cài đủ python và các thư viện liên quan
  + Như bên dưới tôi dùng python bản 12.2
  + Các thư viện liên quan: (Để cài được dlib thì cần cài cmake trong môi trường visual studio hay còn gọi là C++)

| pip install opencv-python  pip install dlib  pip install face-recognition  pip install setuptools |
| --- |

# 2. Biến

* **private List<string> imagePath**: Một danh sách đường dẫn của các hình ảnh.
* **private string pythonScriptPath**: Đường dẫn đến tập lệnh Python cần thực thi.

| private List<string> imagePath = new List<string>();  private string pythonScriptPath = Path.Combine("..", "..", "Python", "main.py"); |
| --- |

# 3. Phương thức

* **WrongIdentificationForm()**: Phương thức khởi tạo của lớp WrongIdentificationForm, chứa các thiết lập ban đầu.
* **dtpkDate\_ValueChanged(object sender, EventArgs e)**: Xử lý sự kiện khi giá trị của DateTimePicker thay đổi.
* **ResizeImage(Image image, int maxWidth, int maxHeight)**: Thay đổi kích thước của một hình ảnh.
* **AddFilesToDataGridView(string[] files)**: Thêm các tệp vào DataGridView.
* **DisplayDataForSelectedDate(DateTime selectedDate)**: Hiển thị dữ liệu cho ngày được chọn.
* **DisplayDataForAll()**: Hiển thị dữ liệu cho tất cả các ngày.
* **ClearDataGridViewImages()**: Xóa các hình ảnh trong DataGridView.
* **btnShowAll\_Click(object sender, EventArgs e)**: Xử lý sự kiện khi nút "Hiển thị tất cả" được nhấp.
* **dtgvReport\_CellContentClick(object sender, DataGridViewCellEventArgs e)**: Xử lý sự kiện khi một ô trong DataGridView được nhấp.
* **RunPythonScript(string cmd, string args)**: Chạy một tập lệnh Python từ mã C#.
* **RunEncodeImagesInDataset(string name)**: Chạy tập lệnh Python để mã hóa các hình ảnh trong dataset.

# 4. Chi tiết mã nguồn

## 4.1. WrongIdentificationForm:

| public WrongIdentificationForm()  {  InitializeComponent();  // Xử lý sự kiện khi ngày được chọn thay đổi  dtpk.ValueChanged += new EventHandler(dtpkDate\_ValueChanged);  // Hiển thị dữ liệu ban đầu với ngày được chọn mặc định là ngày hiện tại  imagePath = DisplayDataForSelectedDate(DateTime.Today);  } |
| --- |

## 4.2. dtpkDate\_ValueChanged(object sender, EventArgs e):

| private void dtpkDate\_ValueChanged(object sender, EventArgs e)  {  // Lấy ngày được chọn từ DateTimePicker  DateTime selectedDate = dtpk.Value;  // Hiển thị dữ liệu cho ngày được chọn  imagePath = DisplayDataForSelectedDate(selectedDate);  } |
| --- |

## 4.3. ResizeImage(Image image, int maxWidth, int maxHeight):

| private Image ResizeImage(Image image, int maxWidth, int maxHeight)  {  float aspectRatio = (float)image.Width / image.Height;  // Tính toán kích thước mới dựa trên tỷ lệ và giới hạn kích thước  int calculatedWidth = Math.Min(maxWidth, (int)(maxHeight \* aspectRatio));  int calculatedHeight = Math.Min(maxHeight, (int)(maxWidth / aspectRatio));  // Kiểm tra nếu kích thước mới vượt quá kích thước ban đầu  if (calculatedWidth > image.Width || calculatedHeight > image.Height)  {  calculatedWidth = image.Width;  calculatedHeight = image.Height;  }  Bitmap resizedBitmap = new Bitmap(calculatedWidth, calculatedHeight);  using (Graphics g = Graphics.FromImage(resizedBitmap))  {  g.DrawImage(image, 0, 0, calculatedWidth, calculatedHeight);  }  return resizedBitmap;  } |
| --- |

## 4.4. AddFilesToDataGridView(string[] files):

| private void AddFilesToDataGridView(string[] files)  {  int stt = 1;  foreach (string filePath in files)  {  // Thêm dòng vào DataGridView  Image image = Image.FromFile(filePath);  dtgvReport.Rows.Add(stt, ResizeImage(image, 512, 300), "Unknown", "", "Huấn luyện");  stt++;  image.Dispose();  }  } |
| --- |

## 4.5. DisplayDataForSelectedDate(DateTime selectedDate)

| private List<string> DisplayDataForSelectedDate(DateTime selectedDate)  {  // Xóa tất cả các dòng hiện tại trong DataGridView và giải phóng tài nguyên hình ảnh  ClearDataGridViewImages();  string directoryPath = @"./WrongIdentification";  string[] files = Directory.GetFiles(directoryPath);  List<string> filteredFiles = new List<string>();  foreach (string filePath in files)  {  // Lấy thông tin về thời gian chỉnh sửa của tệp tin  DateTime fileLastWriteTime = File.GetLastWriteTime(filePath);  // Kiểm tra xem tệp tin đã được chỉnh sửa vào ngày được chọn không  if (fileLastWriteTime.Date == selectedDate.Date)  {  filteredFiles.Add(filePath);  }  }  AddFilesToDataGridView(filteredFiles.ToArray());  return filteredFiles;  } |
| --- |

## 4.6. DisplayDataForAll():

| private List<string> DisplayDataForAll()  {  // Xóa tất cả các dòng hiện tại trong DataGridView và giải phóng tài nguyên hình ảnh  ClearDataGridViewImages();    string directoryPath = @"./WrongIdentification";  string[] files = Directory.GetFiles(directoryPath);  List<string> filteredFiles = new List<string>();  foreach (string filePath in files)  {  filteredFiles.Add(filePath);  }  AddFilesToDataGridView(filteredFiles.ToArray());  return filteredFiles;  } |
| --- |

## 4.7. ClearDataGridViewImages():

| private void ClearDataGridViewImages()  {  foreach (DataGridViewRow row in dtgvReport.Rows)  {  var imageCell = row.Cells["ImageIdentification"];  if (imageCell.Value != null && imageCell.Value is Image)  {  ((Image)imageCell.Value).Dispose();  }  }  dtgvReport.Rows.Clear();  } |
| --- |

## 4.8. btnShowAll\_Click(object sender, EventArgs e):

| private void btnShowAll\_Click(object sender, EventArgs e)  {  dtpk.Value = DateTime.Today;  string directoryPath = @"./WrongIdentification";  string[] files = Directory.GetFiles(directoryPath);  imagePath = DisplayDataForAll();  } |
| --- |

## 4.9. dtgvReport\_CellContentClick(object sender, DataGridViewCellEventArgs e):

| private void dtgvReport\_CellContentClick(object sender, DataGridViewCellEventArgs e)  {  // Kiểm tra nếu cột được nhấp là cột button và hàng được nhấp không phải là hàng header  if (e.ColumnIndex == dtgvReport.Columns["btnTrain"].Index && e.RowIndex >= 0)  {  // Thực hiện hành động khi nút được nhấp  // Ví dụ: lấy dữ liệu từ hàng được nhấp  int stt = Convert.ToInt32(dtgvReport.Rows[e.RowIndex].Cells["STT"].Value);  string wrongName = dtgvReport.Rows[e.RowIndex].Cells["WrongIdentificationName"].Value.ToString();  string correctName = dtgvReport.Rows[e.RowIndex].Cells["CorrectIdentificationName"].Value.ToString();  // Kiểm tra ô "Correct" có dữ liệu hay không  if (!string.IsNullOrEmpty(correctName))  {  string filePath = imagePath[stt - 1];  try  {  // Lấy hàng hiện tại  DataGridViewRow row = dtgvReport.Rows[stt - 1];  // Lấy tên tệp từ đường dẫn file  string fileName = Path.GetFileName(filePath);  // Lưu trữ và giải phóng ảnh hiện tại nếu có  if (row.Cells["ImageIdentification"].Value is Image currentImage)  {  currentImage.Dispose(); // Giải phóng ảnh hiện tại  }  Image newImage = Image.FromFile("./Processing.jpg");  row.Cells["ImageIdentification"].Value = newImage;  // Di chuyển tệp từ thư mục hiện tại sang thư mục dataset  string destinationPath = Path.Combine("..", "..", "./Datasets", fileName);  // Di chuyển tệp và xử lý ngoại lệ nếu có  try  {  File.Move(filePath, destinationPath);  RunEncodeImagesInDataset(correctName);  }  catch (IOException ex)  {  // Xử lý ngoại lệ IOException khi di chuyển tệp  MessageBox.Show($"Có lỗi IO khi di chuyển tệp {filePath} đến {destinationPath}: {ex.Message}");  }  newImage.Dispose();  }  catch (IOException ex)  {  // Xử lý ngoại lệ IOException  MessageBox.Show($"Có lỗi IO khi thực hiện thao tác trên tệp {filePath}: {ex.Message}");  }  catch (Exception ex)  {  // Xử lý các ngoại lệ khác  MessageBox.Show($"Đã xảy ra một ngoại lệ không xác định: {ex.Message}");  }  DateTime selectedDate = dtpk.Value.Date; // Lấy ngày được chọn từ DateTimePicker  if (selectedDate != DateTime.Today)  {  imagePath = DisplayDataForSelectedDate(selectedDate);  }  else  {  imagePath = DisplayDataForAll();  }  }  else  {  // Nếu ô "Correct" không có dữ liệu, thông báo người dùng  MessageBox.Show("Vui lòng nhập tên trước khi huấn luyện!");  }  }  } |
| --- |

## 4.10. RunPythonScript(string cmd, string args):

| private string RunPythonScript(string cmd, string args)  {  ProcessStartInfo start = new ProcessStartInfo  {  FileName = cmd,  Arguments = args,  UseShellExecute = false,  RedirectStandardOutput = true,  CreateNoWindow = true // Không hiển thị cửa sổ dòng lệnh  };  // Khởi chạy tiến trình  using (Process process = Process.Start(start))  {  // Đọc đầu ra tiêu chuẩn của tiến trình  using (StreamReader reader = process.StandardOutput)  {  string result = reader.ReadToEnd();  // Hiển thị đầu ra trên màn hình console  Console.WriteLine(result);  return result;  }  }  } |
| --- |

## 4.11. RunEncodeImagesInDataset(string name):

| private void RunEncodeImagesInDataset(string name)  {  string currentDirectory = Directory.GetCurrentDirectory();  // Lấy thư mục bin  DirectoryInfo binDirectory = Directory.GetParent(currentDirectory);  string binPath = binDirectory.ToString();  // Lấy thư mục Recognition  DirectoryInfo recognitionDirectory = Directory.GetParent(binPath);  string recognitionPath = recognitionDirectory.ToString();  string fullPathPythonScript = Path.Combine(currentDirectory, pythonScriptPath).Replace('\\', '/');  string datasetPath = Path.Combine(recognitionPath, "./Datasets").Replace('\\', '/');  string encodingFilePath = Path.Combine(recognitionPath, "./Models/encodings.txt").Replace('\\', '/');  string trainedPath = $"{datasetPath}/Trained";  RunPythonScript("python", $"\"{fullPathPythonScript}\" encode\_images\_in\_dataset \"{datasetPath}\" \"{encodingFilePath}\" \"{trainedPath}\" \"{name}\"");  MessageBox.Show("Đã huấn luyện xong", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Information);  } |
| --- |