การทำงานของฟังก์ชัน process_csv

ฟังก์ชันนี้รับพารามิเตอร์ 4 ตัว:

- csv_path: เส้นทางของไฟล์ CSV ที่ต้องการอ่านข้อมูล
- images_folder: โฟลเดอร์ที่เก็บรูปภาพทั้งหมด
- output_base_folder: โฟลเดอร์ปลายทางที่จะคัดลอกรูปภาพไปเก็บ
- has_labels: ตัวแปร boolean ที่บอกว่ามีการใช้ข้อมูลเพิ่มเติม (เช่น font, font_size, color) หรือไม่ ถ้า True จะใช้ข้อมูลเหล่านี้ในการจัดหมวดหมู่ไฟล์

ขั้นตอนในฟังก์ชัน:

1. อ่านไฟล์ CSV

   df = pd.read_csv(csv_path)

   ฟังก์ชันจะอ่านไฟล์ CSV ที่มีข้อมูลรูปภาพ โดยใช้ pandas เพื่อโหลดข้อมูลเข้าไปใน DataFrame ที่ชื่อ df

2. สร้างโฟลเดอร์ปลายทาง

   ```
   if not os.path.exists(output_base_folder):
       os.makedirs(output_base_folder)
   ```

   เช็คว่าโฟลเดอร์ปลายทาง (output_base_folder) มีอยู่หรือไม่ ถ้าไม่มีจะทำการสร้างโฟลเดอร์นี้ขึ้นมา

3. วนลูปผ่านแต่ละแถวของ CSV

   ```
   for index, row in df.iterrows():
       image_path = row['image_path']
   ```

   วนลูปผ่านข้อมูลในแต่ละแถวของ DataFrame เพื่อดึงข้อมูล เช่น image_path ที่เป็นเส้นทางของรูปภาพ

4. ตรวจสอบข้อมูลถ้ามีการใช้ Labels (ข้อมูลเพิ่มเติม) ถ้า has_labels เป็น True จะมีการดึงข้อมูลของฟอนต์ ขนาดฟอนต์ และสีของรูปภาพนั้นๆ

   ```
   if has_labels:
       font = row.get('font', 'unknown_font')
       font_size = row.get('font_size', 'unknown_size')
       color = row.get('color', 'unknown_color').replace("#", "")
   ```

ฟังก์ชันจะดึงข้อมูลฟอนต์ (font), ขนาดฟอนต์ (font_size), และสี (color) ถ้าข้อมูลนั้นไม่มี ฟังก์ชันจะกำหนดค่าเริ่มต้นเป็น 'unknown_font', 'unknown_size', และ 'unknown_color'

5. สร้างโฟลเดอร์ปลายทางตามหมวดหมู่

```
class_folder = os.path.join(
    output_base_folder,
    color,
    font,
    f"size_{font_size}"
)
if not os.path.exists(class_folder):
    os.makedirs(class_folder)
```

ถ้า has_labels เป็น True จะสร้างโฟลเดอร์ตามหมวดหมู่โดยใช้ค่า color, font, และ font_size แต่ถ้า has_labels เป็น False จะใช้โฟลเดอร์ปลายทางทั่วไป (output_base_folder)

6. คัดลอกรูปภาพจากโฟลเดอร์ต้นทางไปยังโฟลเดอร์ปลายทาง

```
source = os.path.join(images_folder, os.path.basename(image_path))
destination = os.path.join(class_folder, os.path.basename(image_path))

if os.path.exists(source):
    shutil.copy2(source, destination)
else:
    print(f"File {os.path.basename(image_path)} does not exist in {images_folder}")
```

ฟังก์ชันจะตรวจสอบว่าไฟล์รูปภาพที่มีอยู่ใน image_path ในโฟลเดอร์ต้นทาง (images_folder) มีอยู่หรือไม่ ถ้ามีจะคัดลอกไปยังโฟลเดอร์ปลายทาง แต่ถ้าไฟล์ไม่พบ จะพิมพ์ข้อความแจ้งเตือนว่าไฟล์นั้นไม่มีในโฟลเดอร์

การเรียกใช้งานฟังก์ชัน:

1. ประมวลผลไฟล์ train.csv ที่มีการใช้ labels

```
process_csv(train_csv_path, images_folder, train_output_base_folder, has_labels=True)
```

คัดลอกรูปภาพจากไฟล์ train.csv โดยมีการสร้างโฟลเดอร์ตามข้อมูล labels เช่น สี ขนาดฟอนต์ และฟอนต์

2. ประมวลผลไฟล์ test.csv ที่ไม่มี labels

process_csv(test_csv_path, images_folder, test_output_base_folder, has_labels=False)

คัดลอกรูปภาพจากไฟล์ test.csv โดยเก็บในโฟลเดอร์เดียวกันทั้งหมด เพราะไม่มีการใช้ข้อมูล labels

คลาส MultiAlphabetDataset

1. __init__(self, root_dir, transform=None)
    - ตัวสร้างของคลาสนี้ใช้เพื่อกำหนดค่าเริ่มต้น
        - root_dir: เส้นทางของโฟลเดอร์ที่เก็บรูปภาพ
        - transform: การแปลงข้อมูลรูปภาพ (เช่น การปรับขนาดหรือการทำ normalization) หากมีการกำหนดมา
    - การกำหนดค่าการแมป (Mapping):
        - self.color_mapping: แมปค่าของสี (รหัสสี) เข้ากับตัวเลข (เช่น 'FF6666' ถูกแมปเป็น 0)
        - self.font_mapping: แมปชื่อฟอนต์เข้ากับตัวเลข (เช่น 'Athiti-Regular' ถูกแมปเป็น 0)
        - self.font_size_mapping: แมปขนาดฟอนต์เข้ากับตัวเลข (เช่น 18 ถูกแมปเป็น 0)
    - การวนลูปเพื่อเก็บข้อมูลตัวอย่าง:

```
for color_dir in os.scandir(root_dir):
    if color_dir.is_dir():
        color = color_dir.name
        for font_dir in os.scandir(color_dir.path):
            if font_dir.is_dir():
                font = font_dir.name
                for size_dir in os.scandir(font_dir.path):
                    if size_dir.is_dir() and size_dir.name.startswith("size_"):
                        font_size = int(size_dir.name.split('_')[1])
                        for file in os.scandir(size_dir.path):
                            if file.is_file():
                                self.samples.append((file.path, font, font_size, color))
```

        - ใช้การสแกนไดเรกทอรี (ผ่าน os.scandir) เพื่อวนลูปในโฟลเดอร์สี (color_dir), ฟอนต์ (font_dir), ขนาดฟอนต์ (size_dir) และเก็บข้อมูลของไฟล์รูปภาพ (file.path, font, font_size, color) ลงใน self.samples
        - ข้อมูลที่เก็บไว้ใน self.samples จะประกอบด้วยพาธของไฟล์, ชื่อฟอนต์, ขนาดฟอนต์ และสี
    - การพิมพ์จำนวนตัวอย่างที่พบทั้งหมด:

```
print(f"Total samples found: {len(self.samples)}")
```

2. __len__(self)
    - เมธอดนี้ใช้เพื่อคืนค่าจำนวนตัวอย่างใน dataset โดยการคืนค่า len(self.samples) ซึ่งเป็นจำนวนข้อมูลที่รวบรวมจากการสแกนไดเรกทอรี
3. __getitem__(self, idx)
    - เมธอดนี้ใช้สำหรับดึงข้อมูลรูปภาพและป้ายกำกับตามดัชนีที่กำหนด (idx)
    - ดึงข้อมูลจาก self.samples:

      img_path, font, font_size, color = self.samples[idx]

        - ดึงข้อมูลพาธของไฟล์รูปภาพ (img_path), ฟอนต์ (font), ขนาดฟอนต์ (font_size), และสี (color) ตามดัชนีที่ระบุ (idx)
    - เปิดรูปภาพ:

      image = Image.open(img_path).convert('RGB')

        - เปิดรูปภาพจากพาธที่เก็บไว้ และแปลงเป็นรูปแบบ RGB

      if self.transform:
        image = self.transform(image)

        - ถ้ามีการส่ง transform เข้ามา จะนำไปใช้กับรูปภาพเพื่อทำการแปลง เช่น ปรับขนาดหรือ normalization
    - เข้ารหัสป้ายกำกับ (labels):

      font_encoded = torch.tensor(self.font_mapping.get(font), dtype=torch.long)
      font_size_encoded = torch.tensor(self.font_size_mapping.get(font_size), dtype=torch.long)
      color_encoded = torch.tensor(self.color_mapping.get(color), dtype=torch.long)

        - ป้ายกำกับต่างๆ (เช่น font, font_size, color) จะถูกแปลงเป็นค่าตัวเลขที่กำหนดโดยการแมป (self.font_mapping, self.font_size_mapping, self.color_mapping) แล้วแปลงเป็นเทนเซอร์ (tensor) เพื่อใช้ในโมเดลการเรียนรู้เชิงลึก (Deep Learning)
    - คืนค่า:

      return image, color_encoded, font_encoded, font_size_encoded

        - คืนค่ารูปภาพพร้อมกับป้ายกำกับที่เข้ารหัสเรียบร้อยแล้ว (color_encoded, font_encoded, font_size_encoded)

สรุป:

- คลาส MultiAlphabetDataset นี้ใช้เพื่อโหลดข้อมูลรูปภาพจากโฟลเดอร์ย่อยที่ถูกจัดเรียงตามสี ฟอนต์ และขนาดฟอนต์
- ฟังก์ชัน __getitem__ จะทำการเปิดรูปภาพจากพาธที่จัดเก็บ และคืนค่ารูปภาพพร้อมกับป้ายกำกับที่เข้ารหัสในรูปของเทนเซอร์