Data Collection and Preprocessing Phase

Date	19 June 2025
Team ID	SWTID1749821186
Project Title	Enhancing Product Reliability: Leveraging Transfer Learning for Fault Detection
Maximum Marks	6 Marks

Preprocessing

The images will be preprocessed by resizing, normalizing, augmenting, denoising, adjusting contrast, detecting edges, converting color space, cropping, batch normalizing, and whitening data. These steps will enhance data quality, promote model generalization, and improve convergence during neural network training, ensuring robust and efficient performance across various computer vision tasks.

Section	Description
Data Overview	The dataset contains grayscale images of casting products labeled as either defective (def_front) or good (ok_front). It was sourced from Kaggle and includes 7348 images.
Resizing	All images are resized to a uniform target size of 224×224 pixels to match the input requirements of the VGG16 model.
Normalization	Pixel values are normalized to the range [0, 1] by dividing by 255. This helps the model converge faster during training.
Data Augmentation	Techniques such as horizontal flipping, rotation, zooming, and shifting are applied to artificially expand the dataset and improve generalization.
Denoising	Gaussian blur and median filters are optionally applied to reduce noise in the images and enhance feature clarity.
Edge Detection	Canny edge detection is used to highlight prominent edges and structural features in the casting images.

Color Space Conversion	Images are converted from grayscale to RGB format to be compatible with pre-trained models like VGG16.
Image Cropping	Cropping is applied to remove irrelevant background and focus on the casting region of interest.
Batch Normalization	Batch normalization layers are used in the neural network to stabilize and accelerate training by normalizing inputs to each layer.

Data Preprocessing Code Screenshots

Loading Data

```
os.environ["KAGGLE_KEY"] = userdata.get('KAGGLE_KEY')
os.environ["KAGGLE_USERNAME"] = userdata.get('KAGGLE_USERNAME')

!kaggle datasets download -d ravirajsinh45/real-life-industrial-dataset-of-casting-product

! unzip "real-life-industrial-dataset-of-casting-product.zip"
```

Resizing

```
def plot_sample_images(train_directory, test_directory):
    categories = ['def_front', 'ok_front']
    fig, axes = plt.subplots(2, 2, figsize=(3, 3))

    for i, category in enumerate(categories):
        train_path = os.path.join(train_directory, category)
        test_path = os.path.join(test_directory, category)

        train_images = os.listdir(train_path)[1:]
        test_images = os.listdir(test_path)[1:]

        for img_name in train_images:
            img_path = os.path.join(train_path, img_name)
            img = image.load_img(img_path, target_size=(224, 224))
            axes[i, 0].imshow(img)
            axes[i, 0].axis('off')
            axes[i, 0].set_title(f"Train - {category}")

        for img_name in test_images:
            img_path = os.path.join(test_path, img_name)
            img = image.load_img(img_path, target_size=(224, 224))
            axes[i, 1].imshow(img)
            axes[i, 1].axis('off')
            axes[i, 1].set_title(f"Test - {category}")

    plt.tight_layout()
    plt.show()
```

Normalization

```
train_datagen = ImageDataGenerator(rescale=1./255, zoom_range=0.2, rotation_range=40,
                                   fill_mode='nearest', horizontal_flip=True, shear_range=0.2)
test_datagen = ImageDataGenerator(rescale=1./255)
```

Data Augmentation	`train_datagen = ImageDataGenerator(rescale=1./255, zoom_range=0.2, rotation_range=40, fill_mode='nearest', horizontal_flip=True, shear_range=0.2)` `test_datagen = ImageDataGenerator(rescale=1./255)`