भारतीय प्रौद्योगिकी संस्थान रूड़की
Indian Institute of Technology Roorkee

IMAGE DENOISING

By-VEDANT VERMA(21112120)

Chemical Engineering

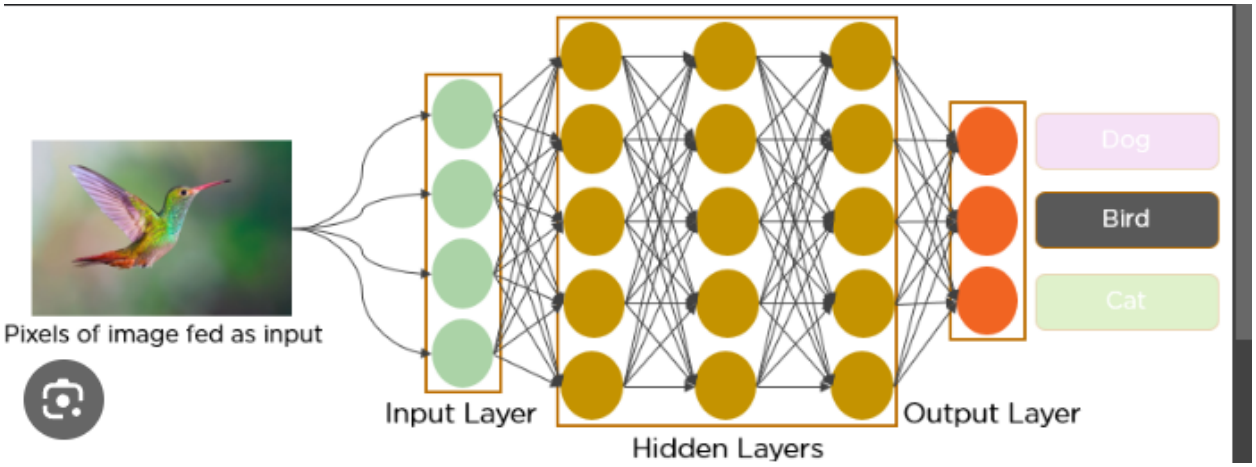
Introduction

Objective

The objective of this project is to develop and implement a deep learning-based approach for image denoising. Image denoising is the process of removing noise from images, which is a crucial step in various image processing applications.

Importance

Denoising is essential for enhancing image quality, which is critical in fields like medical imaging, photography, and remote sensing. This project utilizes Convolutional Neural Networks (CNNs) to achieve effective denoising of images corrupted by noise.



Data Preparation

Importing Libraries

The following libraries are used in this project:

- numpy: For numerical operations.
- matplotlib.pyplot: For plotting images and visualizations.
- tensorflow and keras: For building and training the neural network models.

```
import os
import cv2
import numpy as np
from sklearn.model_selection import train_test_split
import tensorflow as tf
from tensorflow.keras import layers, models
from torch.utils.data import Dataset, DataLoader
from torchvision import transforms
from PIL import Image
from skimage.metrics import peak_signal_noise_ratio as psnr
import matplotlib.pyplot as plt
```

Loading the Dataset

The dataset consists of pairs of noisy and clean images. The noisy images serve as the input to the model, while the clean images are the ground truth for training.

Preprocessing

- Normalization: The pixel values of the images are normalized to a range of 0 to 1 to facilitate faster and more effective training.
- Data Splitting: The dataset is split into training and testing sets. The training set is used to train the model, while the testing set is used to evaluate its performance.

```python
def load_images(directory, size=(128, 128)):
    img_list = []
    for file in os.listdir(directory):
        path = os.path.join(directory, file)
        image = cv2.imread(path)
        if image is not None:
            image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
            image = cv2.resize(image, size)
            image = image / 255.0
            img_list.append(image)
    return img_list
```

```python
# Preprocessing Function for TensorFlow
def image_preprocessing(image):
    return tf.image.convert_image_dtype(image, tf.float32)

# Create TensorFlow Datasets
train_low_ds = tf.data.Dataset.from_tensor_slices(train_low).map(image_preprocessing).batch(32)
train_high_ds = tf.data.Dataset.from_tensor_slices(train_high).map(image_preprocessing).batch(32)
test_low_ds = tf.data.Dataset.from_tensor_slices(test_low).map(image_preprocessing).batch(32)
test_high_ds = tf.data.Dataset.from_tensor_slices(test_high).map(image_preprocessing).batch(32)
```

Model Building

Model Architecture

A Convolutional Neural Network (CNN) is used for the denoising task. The model consists of the following layers:

- Convolutional Layers: These layers extract features from the input images.
- Batch Normalization Layers: These layers normalize the output of each layer to improve training stability and performance.
- Activation Layers (ReLU): These layers introduce non-linearity into the model, enabling it to learn complex patterns.
- Upsampling Layers: These layers restore the image to its original size after downsampling operations.

Model Summary

The model's architecture is summarized, showing the types and sizes of layers used, along with the total number of parameters.

Model Training

Training Process

The model is trained on the training dataset. During training, the loss and accuracy metrics are monitored to ensure the model is learning effectively. Key training parameters include:

- Epochs: The number of times the entire training dataset is passed through the model.
- Batch Size: The number of samples processed before the model's internal parameters are updated.

Model Evaluation

Evaluation Metrics:-

The model is evaluated using the following metrics:

- Peak Signal-to-Noise Ratio (PSNR): Measures the ratio between the maximum possible power of a signal and the power of corrupting noise.

Results:-

The evaluation results are presented, showing the PSNR

PSNR stands for Peak Signal-to-Noise Ratio. It is a metric used to measure the quality of reconstruction of an image or video after compression or transmission. Here's a breakdown of what it means:

```
# Predict with the Model
output_images = denoising_model.predict(test_low_ds)

# Calculate PSNR
psnr_scores = [psnr(test_high[i], output_images[i]) for i in range(len(test_high))]
mean_psnr = np.mean(psnr_scores)
print(f'Average PSNR: {mean_psnr}')
```

```
4/4 ━━━━━━━━━━━━━━━━━━━━ 1s 131ms/step
Average PSNR: 17.29026880820179
```