Image Deblurring using Knowledge Distillation

This project implements an image deblurring pipeline using deep learning. It uses a pretrained DnCNN model as a teacher to train a lightweight student model using knowledge distillation, aimed at achieving faster inference with minimal loss in accuracy.

# 📌 Project Highlights

* Simulates blurry images by downscaling and upscaling high-resolution images.
* Trains a student network using both ground truth sharp images and outputs from a pretrained DnCNN teacher model.
* Evaluates results using SSIM (Structural Similarity Index) and Inference FPS (Frames Per Second).
* Visual comparison of blurry vs predicted vs ground truth images.

# 🏗️ Architecture

- Teacher Model: Deep CNN with 17 layers (DnCNN)  
- Student Model: Shallow CNN with 3 convolutional layers  
- Loss Function: Combined MSE from ground truth and teacher output  
- Framework: PyTorch

# 🗂️ Project Structure

intel\_unnati.py # Full pipeline: model training, testing, evaluation  
dncnn.pth # Pretrained teacher model weights (downloaded via gdown)  
/high\_res\_images # Folder with high-resolution clean images  
/image\_sharpening\_dataset  
 /train  
 /input # Blurry training images  
 /target # Corresponding sharp images  
 /test  
 /input # Blurry test images  
 /target # Corresponding sharp images

# 🚀 Getting Started

1. Clone the repository

git clone https://github.com/<your-username>/<your-repo>.git  
cd <your-repo>

2. Install Dependencies

pip install torch torchvision scikit-image gdown pillow matplotlib tqdm

3. Prepare Dataset

Place your high-resolution images inside a folder called `/content/high\_res\_images`. The script will automatically generate synthetic blurry images and split them into train/test sets.

# 🏃‍♂️ Run Training

python intel\_unnati.py

This will:  
- Train the student model for 15–20 epochs.  
- Calculate SSIM on the test set.  
- Display input, output, and ground truth images.  
- Report inference FPS.

# 📈 Results

- Average SSIM: ~0.85 (depends on data and training)  
- Inference Speed: 20–40 FPS on GPU  
- Lightweight student model suitable for real-time applications.

# 🤝 Acknowledgements

- DnCNN: Denoising Convolutional Neural Network  
- PyTorch, scikit-image, and PIL for image processing.

# 📜 License

This project is licensed under the MIT License. See the LICENSE file for details.

# 📬 Contact

For any queries or suggestions, feel free to reach out to Your Name (your-email@example.com) or open an issue.