**Project Report: Image Sharpening using Knowledge Distillation**

Submitted by – Parth Prakash

### **1. Problem Statement**

**Title**: Image Sharpening using Knowledge Distillation

**Objective**: To develop an image sharpening model suitable for real-time applications like video conferencing. The model should enhance image clarity under poor network conditions using a Teacher-Student knowledge distillation approach.

### **2. Prerequisites**

* Machine Learning Concepts
* Python Programming
* Deep Learning with CNNs

### **3. Approach**

#### **3.1 Knowledge Distillation Framework**

* **Teacher Model**: A high-performing pretrained model (Restormer) known for image restoration and deblurring.
* **Student Model**: A lightweight CNN model trained to mimic the outputs of the teacher model.

#### **3.2 Training Strategy**

* Simulated degraded images were generated using bicubic downscaling and upscaling, as instructed in the problem statement.
* Sharp images acted as ground truth, while the teacher model provided “soft labels.”
* The student model was trained using a combination of MSE loss and outputs from the teacher.

### **4. Dataset**

* **Source**: Custom dataset based on DIV2K
* **Total Images**: 900 image pairs (Blurry, Sharp, Teacher Output)
* **Categories Covered**: Text, Nature, People, Animals, Games
* **Preprocessing**:
  + All images resized to 256x256 during training
  + Bicubic downscaling used for generating blurry images

### **5. Model Description**

#### **5.1 Teacher Model (Restormer)**

* Transformer-based model for image restoration
* Pretrained on motion deblurring task
* High accuracy but computationally heavy

#### **5.2 Student Model (CNN)**

* Custom 3-layer convolutional neural network
* Designed for speed and reduced size
* Trained using PyTorch with MSE loss

### **6. Training and Evaluation**

#### **6.1 Training Setup**

* Epochs: 10
* Batch Size: 8
* Optimizer: Adam
* Loss Function: MSE Loss
* Device: NVIDIA GeForce RTX 3050 Laptop GPU

#### **6.2 Performance Metrics**

| Metric | Student Model | Teacher Model |
| --- | --- | --- |
| **SSIM** | 0.9320 | 0.9591 |
| **PSNR** | 22.37 dB | 23.62 dB |
| **FPS** | 387 | 3.5 |
| **Inference Time (900 images)** | 2.3 seconds | 252 seconds |

**PSNR Interpretation:**  
While the teacher model delivers slightly higher PSNR than the student model, it's important to note that PSNR can be affected by the level of blur in input images. High Gaussian blur removes fine details, making it difficult even for the teacher to exactly reconstruct the original image. This leads to lower pixel-wise similarity (i.e., lower PSNR) despite visually sharp outputs. Hence, PSNR alone may not fully reflect perceptual quality in such scenarios.

### 

### **7. Subjective Analysis**

* Visual comparisons show noticeable sharpening over blurry images.
* While the teacher performs slightly better in quality, the student offers real-time speed.
* Student output maintains structural similarity with the ground truth.
* A visual comparison of blurry input, student output, and teacher output is provided on the next page for reference.

**Visual Comparison:**

**Example 1** (Blurry → Teacher→ Student)



**BLURRY TEACHER OUTPUT STUDENT OUTPUT**

**Example 2** (Blurry → Teacher → Student)



**BLURRY TEACHER OUTPUT STUDENT OUTPUT**

### **8. File Structure**

imagesharpening/  
├── data/  
│ ├── blurry/  
│ ├── sharp/  
│ ├── student\_output/  
│ └── teacher\_output/  
├── src/  
│ ├── dataset.py  
│ ├── train\_student.py  
│ ├── test\_single\_image.py  
│ └── metrics/  
│ ├── psnr\_test.py  
│ ├── ssim\_test.py  
│ └── inference\_time.py  
├── restormer/ (Teacher codebase)  
└── checkpoints/  
 └── student\_model.pth

### **9. Conclusion**

* Successfully developed and trained a lightweight student model via knowledge distillation
* Met all performance benchmarks (SSIM > 90%, FPS > 30)
* Suitable for real-time image sharpening in video conferencing scenarios

### **10. References**

* Restormer: <https://arxiv.org/abs/2111.09881>
* DIV2K Dataset: <https://data.vision.ee.ethz.ch/cvl/DIV2K/>
* PyTorch Documentation: <https://pytorch.org/>