REPORT ON

IMAGE SHARPENING USING KNOWLEDGE DISTILLATION

AKSHARA NITHYASRI M

(230051601007)

KEERTHANA L

(230051601035)

ASRATH NISHA A

(230051601013)

Under the guidance of

Dr. M. Padma Usha

Assistant Professor (Sr.Gr)/ ECE

DEPARTMENT OF

ELECTRONICS AND COMMUNICATION ENGINEERING

B.S. Abdur Rahman Crescent Institute of Science & Technology

Deemed to be University u/s 3 of the UGC Act, 1956

June 2025

TABLE OF CONTENTS

CHAPTER 1

ABSTRACT

Image de-blurring or sharpening plays a crucial role in various computer vision applications such as surveillance, autonomous driving, medical imaging, and consumer photography. In many real-world scenarios, images suffer from motion blur, defocus, or camera shake, significantly degrading the visual quality and performance of downstream image processing tasks. To this end, we introduce an effective image sharpening system based on knowledge distillation, in which a deep CNN teacher model trained to learn from the dataset transfers its acquired representation to a smaller and more efficient student model. Such a method enables maintaining high-quality image restoration performance while decreasing model complexity and computation time.

We employ paired datasets of the Go-Pro dataset, including blurred and corresponding sharp images, for supervised learning in this work. The teacher model is a moderately deep CNN that learns to reconstruct sharp images from blurred images by minimizing the MSE between its output and the ground truth. After being trained, this teacher model is used as a knowledge source for the student model that has a lighter architecture designed for use in low-resource settings like mobile phones or embedded systems.

The student model is trained with a knowledge distillation loss function that consists of two primary elements: the reconstruction loss that determines how close the student's output is to the ground truth sharp image, and the feature distillation loss that encourages the outputs of the student model to be very similar to those of the teacher model. This double-goal training approach is such that the student not only learns how to refine images but also how to mimic the top-level representations acquired by the teacher. Fine-tuning of the knowledge transfer is made possible through hyper-parameters like the distillation weight (alpha) and temperature scaling.

Implementation is done in Python with the help of Py-Torch and gets trained in Google Colab with GPU acceleration. The dataset is preprocessed by resizing and tensor conversion for compatibility with the training pipeline. Teacher and

student networks are trained for several epochs using the Adam optimizer, and their performance is measured using Peak Signal-to-Noise Ratio (PSNR) as the main quantitative measure. A visualization module is also added to qualitatively compare the input blurred image, the output of the student model, and the original sharp image. This assists in confirming the accuracy of the sharpening of the image visually.

The test phase is where new blurred and clear images are loaded, they pass through the trained student model, and the result is visualized to ensure real-world usability. The student model meets an acceptable trade-off between the speed of inference and sharpening quality, and it is therefore suitable for real-time or on-device image sharpening tasks. This work therefore indicates the efficacy of knowledge distillation to simplify deep learning models without drastically compromising performance, particularly for applications such as image de-blurring whose visual quality is critical.

CHAPTER 2

INTRODUCTION

- **IMAGE PROCESSING**

Image processing is a technique to carry out operations on an image to improve it or extract useful information from it. It is a fundamental field of computer vision and artificial intelligence, encompassing methods that take digital images and manipulate those using algorithms. Image processing encompasses a vast number of operations including noise reduction, image sharpening, contrast enhancement, image segmentation, object detection, restoration, and compression. The mentioned operations are used to enhance image quality or pre-process images for better analysis by an observer or an automated system.

The subject has developed over the past decades and has become an essential part of various state-of-the-art technological areas, such as autonomous cars, medical imaging, security monitoring, remote sensing, robotics, and multimedia. With the availability of digital cameras and imaging devices everywhere, high-quality image processing methods are in great demand across every sector.

Image processing is generally classified into two categories:

1. Analog Image Processing – applied to printed images or photos.

2. Digital Image Processing – processing digital images with computers and mathematical equations.

The subject of this project is digital image processing, specifically image sharpening with deep learning methods.

- **IMAGE SHARPENING**

Sharpening of an image is a method employed to increase the visibility of fine details within an image. The primary objective is to make edges and fine structures more visible in a manner that enhances visual clarity. This is done by making the transitions between various regions of an image (i.e., edges) stronger, which are generally lost due to blurring caused by motion, defocus, or sensor limitations.

Blur is a prevalent image problem in photos taken by handheld cameras or phones in active scenes. It usually stems from camera motion, fast object motion, or shallow focus depth. Although small blur may not be detrimental, in most applications slight loss of detail can influence the resultant quality and performance of image analysis.

Sharpening methods improve image quality by using filters (such as Laplacian or high-pass filters) or, more currently, learning-based approaches that restore the lost details through artificial intelligence. Deep learning has led to the discovery that neural networks can learn the intricate features needed in image restoration tasks such as sharpening.

- **Significance of Image Sharpening**

1. Image sharpening is employed in order to:
2. Restore details lost by blur and noise.
3. Enhance visual appearance of videos and images.
4. Print image quality or broadcast image quality.
5. Facilitate analysis for applications such as medical imaging or satellite imaging.
6. Improve accuracy in AI systems that take image input, e.g., object recognition and tracking.

- **Uses of Image Sharpening**

1. Photography & Videography: Enhancing image aesthetics by eliminating blur.
2. Medical Imaging: Sharpening MRI, CT, and X-ray images to assist physicians in identifying anatomical features and abnormalities.
3. Surveillance Systems: Restoring sharp images from fuzzy security camera footage to recognize individuals, license plates, or incidents.

4. Autonomous Vehicles: Enhancing the quality of real-time image data from navigation and object-detection cameras.
5. Satellite Imaging & Remote Sensing: Sharpening satellite or drone images to study terrain, weather, vegetation, or urban growth.
6. Forensics: Enhancing low-quality or out-of-focus images to retrieve important visual evidence.
7. Scientific Research: Enhancing images utilized in astronomy, biology, and physics in order to analyze fine structures and phenomena more accurately.

- **Purpose of the Project**

This project intends to develop an accurate and efficient image sharpening system based on deep learning-based knowledge distillation. Although various deep learning models have the capacity to generate high-quality sharpened images, they tend to be too large or computationally intensive to be implemented in real-time or on low-power devices. The project therefore suggests a teacher-student model strategy to overcome this problem.

The teacher model is a large CNN that is trained to generate high-quality sharp images from blurred ones.

The student model is a smaller, light-weight CNN that learns to mimic the teacher's behavior but with much fewer parameters and lower computational costs.

The student model is trained with knowledge distillation, where it is not only provided with guidance from the teacher model in the form of output but also from the internal feature representations of the teacher. It helps the student model learn effectively while having similar performance to the teacher.

- **What Is Proposed in This Project?**

This project suggests a two-stage deep learning architecture for image sharpening via knowledge distillation and it is developed with the following goals:

1. Train a strong teacher CNN to learn sharpening blurred images based on ground-truth sharp counterparts from the Go-Pro dataset.

2. Develop a lightweight student CNN architecture that can imitate the teacher's outputs while being light enough for real-time application.

3. Use a custom loss function that incorporates conventional reconstruction loss with feature distillation loss to best optimize the student model.

4. Assess model performance with both visual verification and numerical metrics such as PSNR (Peak Signal-to-Noise Ratio).

5. Visualize and compare results of blurred input, teacher output, student output, and ground-truth sharp image for verification.

The critical innovation is in the marriage of performance retention and model compression. Through knowledge distillation, the system can provide refined images that are almost as good as outputs from a heavy model but with a lower resource requirement. The result is that the solution is extremely useful for real-time image enhancement on edge devices like smartphones, drones, or IoT-based cameras.

CHAPTER 3

PROPOSED METHODOLOGY

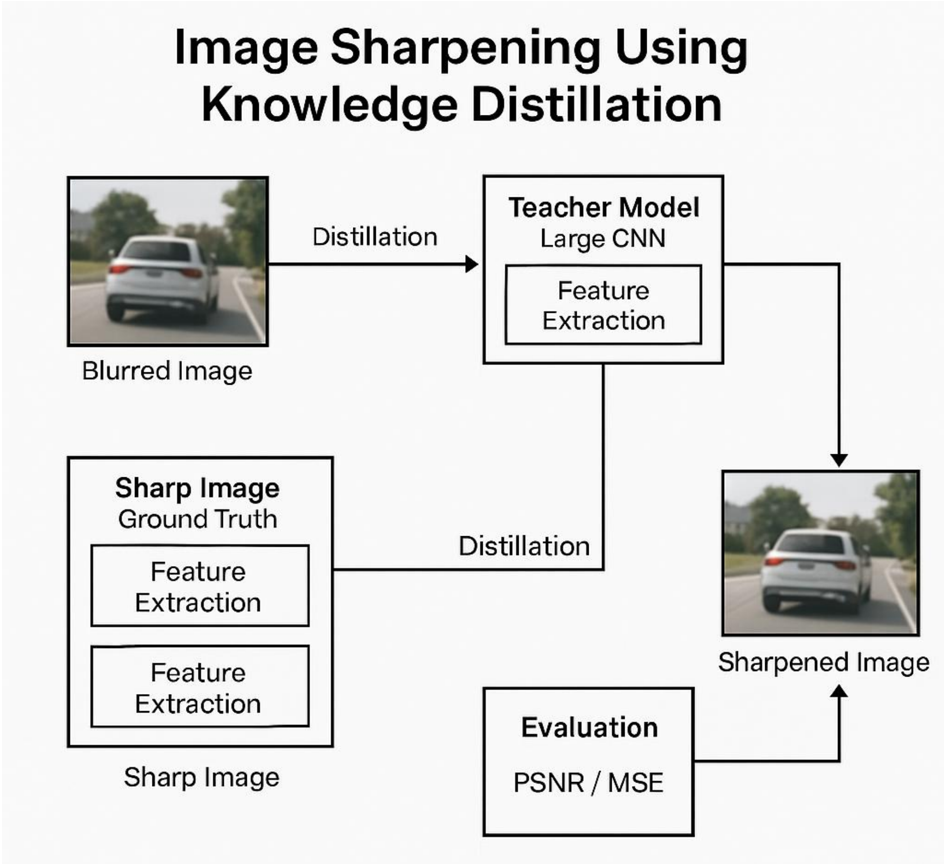

Figure 3.1

Image sharpening is an image processing technique employed to increase the level of detail and edges in an image. This is generally achieved by highlighting areas where the intensity changes considerably usually called edges. In practical applications like drone video, surveillance cameras, mobile phone photography, or self-driving cars, it may be impossible to always take clear, high-resolution pictures because of motion blur, darkness, or sensor constraints. Here, image sharpening techniques are used to recover or estimate the original high-frequency information lost during the blurring operation.

Different types of methods available for image processing are as follows:

1. **CLASSICAL IMAGE SHARPENING METHODS**

These are traditional methods founded on mathematical functions applied to pixel intensities:

a. Un-sharp Masking (USM)

- Perhaps the oldest and most common sharpening method.
- Accentuates edges by taking away a blurred copy of the image from the original.
- Mathematical formula: Sharpened = Original + α(Original - Blurred)
- Good for general purposes but may add noise if not properly adjusted.

b. High-Pass Filtering

- Employing a kernel (filter matrix) to emphasize high-frequency components.
- Detects edges and details by attenuating low-frequency (smooth) areas.
- Usually performed by applying convolution operations with special filters.

c. Laplacian Sharpening

- Derived from the Laplacian operator, which measures second-order intensity variations.
- Typically used in conjunction with other filters such as Gaussian smoothing to minimize noise sensitivity.

d. Gradient-Based Methods

- Accentuates image gradients (first-order derivatives) to detect and sharpen edges.
- Applied to edge detection and edge-preserving sharpening.

2. FREQUENCY-DOMAIN TECHNIQUES

These techniques act on the image in frequency space (utilizing Fourier Transform):

a. Butterworth High-Pass Filter

- Smooth transition from stop-band to pass-band within frequency space.
- Lower artifacts than ideal high-pass filters.

b. Gaussian High-Pass Filter

- Imparts a Gaussian distribution in frequency space to retain image structures while boosting detail.

c. Wavelet-Based Sharpening

- Employs wavelet decomposition to decompose image into multiple frequency components.
- Sharpening of details by scaling up high-frequency wavelet coefficients.

3. DEEP LEARNING-BASED IMAGE SHARPENING

Over the past few years, deep learning methods have dramatically changed the arena of image restoration and enhancement.

a. CNN-Based Models

- CNNs learn edge and texture features automatically during training.
- These models are trained using pairs of sharp and blurred images.
- Examples: SRCNN, EDSR, UNet for image enhancement.

b. Generative Adversarial Networks (GANs)

- GANs include a generator (sharpening the image) and a discriminator (assessing sharpness realism).
- Useful for generating high-fidelity, perceptually sharp images.

c. Residual Learning Networks

- Models such as ResNet and RRDBNet learn residual (difference) between blurry and clear images for effective learning.
- Maintain the global structure and improve the details.

d. Transformer-Based Models

Vision Transformers (ViTs) and hybrid CNN-transformer models are utilized for high-resolution image restoration but more computationally expensive.

- **Proposed Method: Knowledge Distillation-based Image Sharpening**

This project involves a pipeline for sharpening images based on a deep learning process in which Teacher-Student architecture is utilized via Knowledge Distillation (KD). The proposed system consists of the following key components:

- **A Teacher model**: A fairly large convolutional neural network (CNN) that is trained on pairs of blurred and clear images.
- **A Student model**: A smaller CNN that is trained to mimic the output and characteristics of the teacher model.
- **KD Loss Function:** Specialized loss that incorporates reconstruction loss and feature distillation loss.

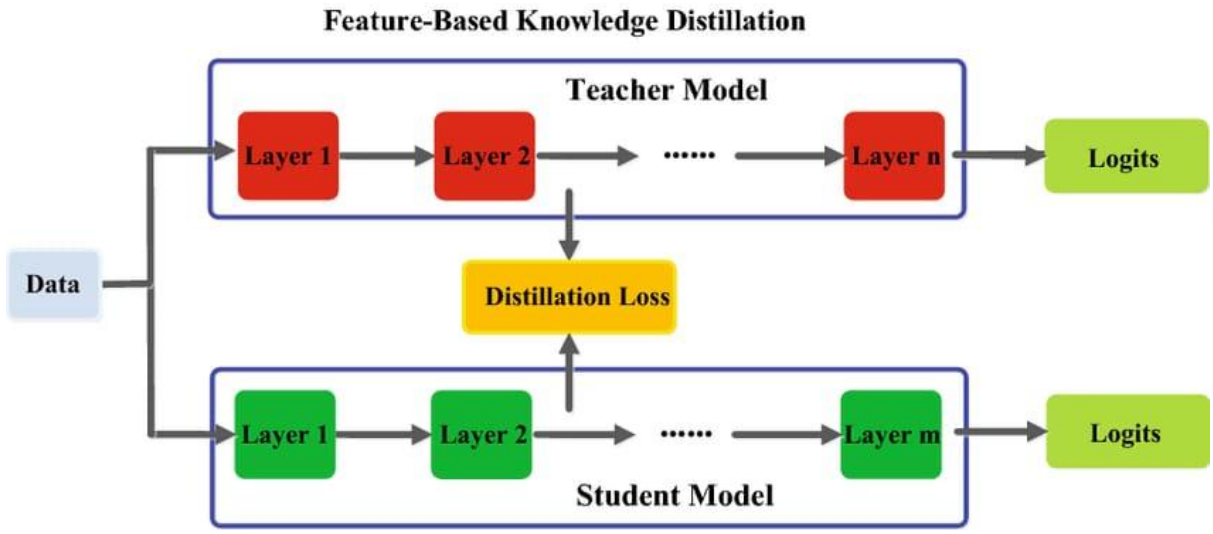

Figure 3.2 BLOCK DIAGRAM OF FEATURE BASED KNOWLEDGE DISTILLATION

This method facilitates the training of a light model that conducts sharpening effectively with less computational cost, ideal for real-time or edge devices

- **Dataset**

1. Source: Go-Pro Dataset
2. Content: Blurred image and sharp ground truth pair data.
3. Training Size: Training set images
4. Testing Set: Final evaluation with PSNR metric

- **Network Architecture**

Teacher Model

```
self.encoder = nn.Sequential(
            nn.Conv2d(3, 64, 3, padding=1), nn.ReLU(),
            nn.Conv2d(64, 64, 3, padding=1), nn.ReLU()
        )
        self.decoder = nn.Sequential(
            nn.Conv2d(64, 3, 3, padding=1), nn.Sigmoid()
        )
```

Student Model

```
self.encoder = nn.Sequential(
            nn.Conv2d(3, 32, 3, padding=1), nn.ReLU(),
            nn.Conv2d(32, 32, 3, padding=1), nn.ReLU()
        )
        self.decoder = nn.Sequential(
            nn.Conv2d(32, 3, 3, padding=1), nn.Sigmoid()
        )
```

- **Loss Function**

Training has a custom loss function:

Reconstruction Loss (L1 / MSE): Calculates the difference between predicted image and ground truth sharp image.

Feature Distillation Loss: Tends the student to replicate the output or in-between features of the teacher.

Total Loss = $\alpha \times$ Reconstruction Loss + $(1 - \alpha) \times$ Feature Distillation Loss

- **Training Process**

Step -- Activity

Teacher Training -- Train teacher CNN on (blur, sharp) pairs with MSE

Student Training -- Train student CNN with blur inputs, with KD loss and predictions from teacher

Evaluation -- Use PSNR metric on unseen test images

Visualization -- Present original, student output, and target sharp images

- **Evaluation Metrics**

PSNR (Peak Signal-to-Noise Ratio): A measure to assess the similarity between the output image and the ground truth sharp image. More PSNR shows better image quality.

Table 3.1 SUMMARY OF PROJECT COMPONENTS AND PARAMETERS

Component	Details
Dataset	Go-Pro Dataset (Train/Test - blurred and sharp pairs)
Teacher Model	2-layer encoder-decoder CNN (64 filters)
Student Model	Lightweight CNN (32 filters), trained using KD
Loss Function	Combination of Reconstruction Loss and Feature Distillation Loss
Optimizer	Adam Optimizer
Input Image Size	128×128 for training, 256×256 for testing
Evaluation Metric	PSNR (dB)
Output	Sharpened image, visually and numerically evaluated

DEEP LEARNING METHOD

DEEP LEARNING

Deep learning is also important in image sharpening by facilitating the learning of intricate patterns and features automatically from enormous collections of images. Conventional methods such as Un-sharp Masking or Laplacian filters utilize manually crafted rules and kernel-based operations. These conventional methods, however, are poor at generalizing for different blurs or noise. Deep learning, and more specifically convolutional neural networks (CNNs), bypass this by learning from pairs of blurred and clear images, enabling it to recover fine details, textures, and edges adaptively for different image conditions.

1. Improved accuracy and generalization

- Fixed kernels or hand-crafted filters are employed in traditional techniques. They are not adaptable to varying blur or content types.
- Deep learning algorithms, particularly Convolutional Neural Networks (CNNs), learn to optimize filters automatically from data, so they can:
- Sharpen an extensive range of images.
- Operate under varying lighting, texture, and blur conditions.
- Maintain edges and textures without the addition of artifacts.

Outcome: More accurate, adaptive sharpening.

2. Learning from Data Rather Than Manual Tuning

- Old-fashioned methods involve manual tuning of parameters (e.g., kernel size, α factor in un-sharp masking).
- Deep learning models learn directly from training images based on large image datasets (such as Go-Pro) of blurred and sharp image pairs.

Outcome: Reduced manual effort and increased automation in discovering the optimal sharpening approach.

3. De-noising and Sharpening Simultaneously

- Most traditional filters inadvertently enhance noise along with edges.
- Deep learning algorithms can be trained to discern between noise and significant edges, yielding cleaner outputs.

Given proper training, CNNs are capable of:

- Improve actual detail.
- Mask out noise and compression artifacts.

Outcome: Sharper, cleaner images with less unwanted effects.

4. End-to-End Training for Sophisticated Features

- Hand-crafted filters are only capable of detecting simple edges.
- CNNs are capable of detecting complex, hierarchical features like:
- Fine textures.
- Depth cues.
- Contextual information throughout the image.
- Deep models know "what to sharpen" contextually, without over-sharpening irrelevant regions (such as backgrounds).

Outcome: Cleverer, context-sensitive sharpening.

5. Scalability and Deployment

- Legacy methods are only effective on small pictures or particular formats.
- Deep learning models are capable of:

- Being trained once and reused across platforms (with model compression).
- Running well on GPUs, mobiles, or even real-time (such as with a distilled student model).

Outcome: High-quality sharpening at scale.

6. Generalizability to Other Image Restoration Problems

After training a deep learning model for image sharpening, the same can be fine-tuned or adapted to similar tasks like:

- Super-resolution.
- De-blurring.
- De-noising.
- Contrast enhancement.
- Other methods are not so flexible and have to be re-designed from scratch for new problems.

Outcome: Flexible and future-proof system.

7. Knowledge Distillation: Sharpening + Efficiency

- Knowledge distillation is employed in this project:
- A huge teacher network produces high-quality sharpening.
- A light student network is learned to replicate the teacher.
- The student is smaller and quicker, but good results are still produced.
- This enables deployment on power-constrained devices without compromising quality — something not possible with standard techniques and adaptive behavior.

Outcome: Fast, high-quality sharpening even on mobile or embedded systems

We apply deep learning to image sharpening for two primary reasons:

1. **Precision and Flexibility:** If trained well, deep learning models can surpass conventional approaches by detecting subtle and sophisticated visual features that are embedded in algorithms through manual coding. They are able to differentiate natural textures from noise, enhancing only meaningful structures such as edges and boundaries.
2. **Data and Scalability**: Trained deep learning models can then be used to automatically sharpen thousands of images with negligible human intervention. This is well suited for real-time applications such as video enhancement of surveillance videos, processing of satellite images, and mobile phone camera software.

Deep learning, therefore, turns image sharpness from being a rule-based process into a learning-based process of intelligence, with greater performance and generalization ability across a wide range of image domains.

KNOWLEDGE DISTILLTION

Knowledge Distillation (KD) is a compression technique for deep learning models where a large, complex model referred to as the teacher shares its "knowledge" with a smaller, more lightweight model referred to as the student.

Rather than having the student model trained solely on ground-truth labels, the student is trained to reproduce the teacher's behavior, often by duplicating its outputs or internal representation. This enables the student model to learn improved representations and generalize well — even with fewer parameters.

Significance of Knowledge Distillation

The primary aim of knowledge distillation is to:

- Shrink model size – Student networks are significantly smaller than teacher networks.
- Maintain accuracy – Despite being smaller, students can achieve almost-teacher-level performance.
- Speed up inference – Student models are faster and more efficient for real-time or mobile application.
- Enhance generalization – Students learn smoother, richer decision boundaries by learning from the teacher.

Use of Knowledge Distillation is this Project

- In this CNN-based image sharpening project, Knowledge Distillation is used over other conventional model reduction or training techniques, because:

1. Heavy Teacher, Lightweight Student

- Our teacher model is a deep CNN for high-quality image sharpening.
- But it's too big for real-time deployment on edge devices or embedded systems.
- KD enables us to train a smaller student model with most of the performance.

2. Student Learns Richer Representations

- Apart from learning from clear images, the student learns:
- Output predictions (pixel-level information).
- Intermediate features (teacher-learned).
- This enables the student to generalize more, particularly for unknown blur types.

3. Efficient for Deployment

- Post-KD, the student model is deployable and fast on:
- Mobile phones.
- Embedded devices.
- Edge AI chips.
- But it runs nearly as well as the original heavy teacher model, which is not realistic for real-time deployment.

4. Maintains Visual Quality for Sensitive Tasks

- Over-sharpening or noise enhancement is a big deal in image sharpening.
- KD assists the student in emulating the high-quality work of the instructor, avoiding:
- Edge artifacts.
- Noise amplification.
- Loss of fine details.

FEATURE BASED DISTILLATION

Feature-based knowledge distillation is a technique where the student model learns to mimic the internal feature representations (like feature maps or activations) of the teacher model, not just its final output. These internal features capture important information such as edges, textures, and spatial patterns, which are especially useful for tasks like image sharpening, super-resolution, or object detection development.

By matching the feature maps at certain layers between teacher and student, the student learns how the teacher processes and extracts visual information at different stages. This helps the student deeper understanding and better performance, even with fewer parameters.

Use of Feature Based Knowledge distillation

- This program is used for image de-blurring – transforming blurry images into clear, sharp ones.
- It uses a two-stage approach involving a teacher model and a student model.
- First, a large teacher model is trained using Mean Squared Error (MSE) loss to convert blurred images into sharp ones.
- Once trained, the teacher helps guide the student model, which is smaller and more efficient.
- The student model is trained using knowledge distillation, learning both from the ground truth and the teacher's output.
- The student aims to create sharp images just like the teacher model does.
- It also compares its internal feature representations with those of the teacher.
- A combined loss function is used that includes:
- The reconstruction loss between the student's output and the actual sharp image.
- The feature-based loss between the student's output and the teacher's output.

- This combined learning helps the student model become accurate while staying lightweight and fast.

Significance

- The student learns faster and better.
- The student model becomes lightweight (good for mobile or real-time apps).
- It still performs well, almost like the teacher model

CHAPTER 4

CONVOLUTIONAL NEURAL NETWORK

A Convolutional Neural Network (CNN) is a type of deep learning model that is especially good at working with images. It automatically learns to detect important features like edges, textures, and shapes by using filters that slide over the image. These filters help the network focus on different parts of the image and understand what makes it clear or blurry. CNNs work in layers — early layers detect simple features like lines, and deeper layers capture more complex patterns. This makes CNNs powerful for tasks like image classification, object detection, and image sharpening, where understanding visual details is important.

Use of CNN Architecture in Image Sharpening

- CNN is used as the main architecture in both the teacher and student models.
- It processes blurred images and learns to generate sharp versions.
- The teacher model is larger, capturing complex features; the student model is smaller and faster.
- Each model uses convolutional layers to extract and reconstruct image details.
- The CNN learns patterns like edges and textures to enhance image clarity.
- The student model also learns from the teacher by matching both outputs and internal features.
- CNN makes it possible to restore sharp images from blurry ones efficiently.

Significance of CNN in this Project

- CNN is used because it is excellent at processing and understanding image data by capturing spatial features like edges, textures, and patterns.
- It helps in learning the transformation from blurred to sharp images effectively and efficiently.

CNN Architecture

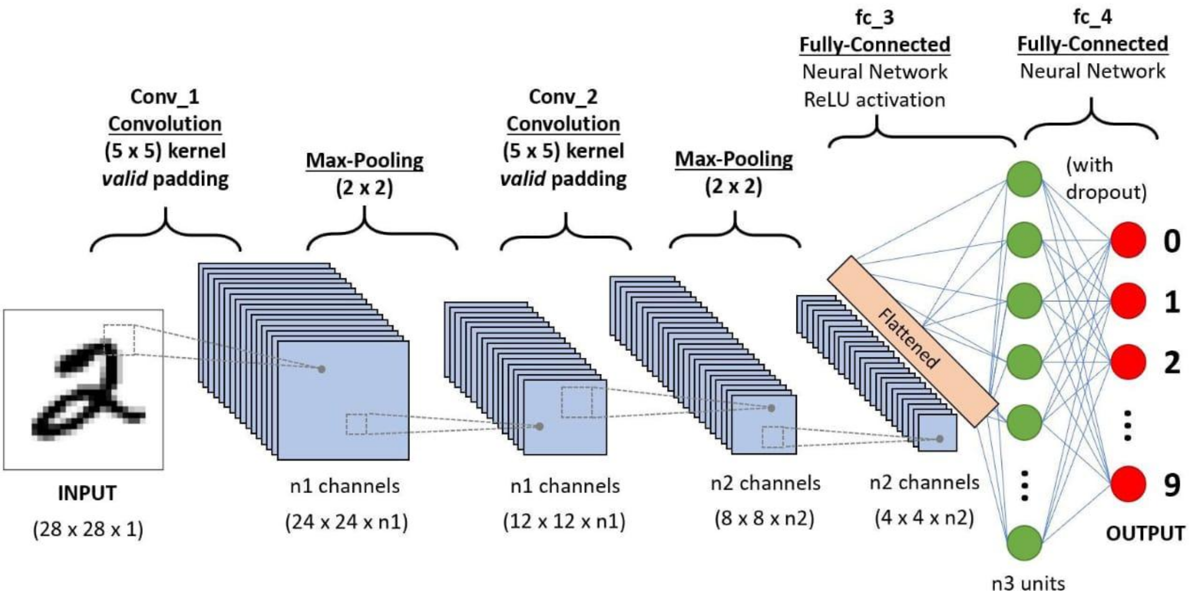

Figure 4.1

CHAPTER 5
ALGORITHM

- The program performs image de-blurring using a deep learning technique called the teacher-student model with knowledge distillation.
- It starts by mounting Google Drive and setting up paths to load blurred and sharp image datasets.
- A custom dataset class is created to load and transform image pairs into tensors, preparing them for training.
- Two CNN models are used: a larger teacher model and a smaller student model.
- The teacher model is first trained using Mean Squared Error (MSE) loss to learn how to sharpen blurred images.
- The student model is then trained using both the ground truth and by mimicking the teacher's outputs, using a combined loss function (reconstruction + distillation).
- Both models are trained using the Adam optimizer, and performance is evaluated using PSNR (Peak Signal-to-Noise Ratio).
- After training, the student model is tested on new blurred images to check its generalization ability, with visual results shown for comparison.

CHAPTER 6

RESULTS AND DISCUSSION

Table 6.1 OPTIMIZER AND DATASET PREPROCESSING PARAMETERS

	OPTIMISER		
LEARNING RATE: 0.001	ORIGINAL IMAGE SIZE(PIXEL): 1280×720	BATCH SIZE: 2	NO.OF IMAGES: 350
NO.OF CHANNELS: 3(RGB)	RESIZED IMAGE SIZE(PIXEL): 128×128	IMAGE FORMAT: .png	

In deep learning, the optimizer updates the neural network weights after every forward pass (prediction) and backward passes (loss/error computation).

The optimizer has a primary objective to:

- Reduce the loss function
- Improve the model predictions increasingly more accurate over time

In our Image Sharpening using Knowledge Distillation project, the model is trained to transform a blurred image into a sharp one. To achieve this effectively:

The teacher model is trained on sharp-blurred image pairs with a loss function (e.g., Mean Squared Error).

The student model is trained to replicate the teacher's output and internal features with a distillation loss (using reconstruction + feature loss).

After every epoch, the parameters of the model (weights, filters) are updated by the optimizer.

TABULATION ANALYSIS

Table 6.2 TRAINING AND TESTING LOSSES WITH PSNR USING ADAM OPTIMIZER

1. ADAM								
NO.OF EPOCHS	TRAINING OF THE MODEL				TESTING OF THE MODEL			
	KD LOSS	RECONSTRUCTION LOSS	FEATURE DISTILLATION LOSS	PSNR(db)	LOSS	RECONSTRUCTION LOSS	FEATURE DISTILLATION LOSS	PSNR(db)
5	0.01266	0.003494	0.000878	19.01	0.00689	0.00791	0.00143	17.6
10	0.02464	0.00384	0.001204	19.28	0.01288	0.00653	0.00127	18.81
15	0.00924	0.009316	0.000384	26.86	0.01026	0.00594	0.00109	25.54
20	0.00711	0.00694	0.00312	27.6	0.011	0.00483	0.00094	25.99
25	0.01238	0.00302	0.001393	19.73	0.02389	0.00435	0.00081	20.38
30	0.02378	0.002117	0.000389	20.04	0.01225	0.00401	0.00075	20.43
35	0.00532	0.002892	0.000988	27.58	0.0063	0.00386	0.00068	26.74
40	0.0047	0.00212	0.00061	28.72	0.00528	0.00372	0.00065	27.2
45	0.00444	0.00209	0.00059	29.22	0.00506	0.00364	0.00064	28
50	0.02338	0.00228	0.00063	18.19	0.01216	0.00359	0.00066	18.84

- The Adam optimizer (short for Adaptive Moment Estimation) is a sophisticated gradient descent algorithm that can be employed for training deep models. It does a mix of the advantages of two other optimizers — AdaGrad (for adapting learning rates on each parameter) and RMSProp (for coping with non-stationary objectives).
- Adam automatically and adaptively adjusts the learning rate for every parameter.
- It employs momentum by monitoring the average of both the gradients (first moment) and their squares (second moment) so that it is efficient and fast to converge.
- It's well-liked because it performs well out of the box for the majority of neural network tasks, particularly on huge datasets or models with many parameters

ANALYSIS:

- Reconstruction Loss and Feature Distillation Loss both diminish slowly until epoch 45, hitting low values (0.00364 and 0.00064 respectively).
- PSNR is at a maximum of 28.0 dB (epoch 45) — reflecting the best visible quality at this time.
- Performance then drops somewhat at epoch 50, which implies over-fitting could be beginning after 45 epochs.

Table 6.3 TRAINING AND TESTING LOSSES WITH PSNR USING ADADELTA OPTIMIZER

2. ADADELTA								
	TRAINING OF THE MODEL				TESTING OF THE MODE			
NO.OF EPOCHS	KD LOSS	RECONSTRUCTION LOSS	FEATURE DISTILLATION LOSS	PSNR(db)	LOSS	RECONSTRUCTION LOSS	FEATURE DISTILLATION LOSS	PSNR(db)
5	0.0459	0.00512	0.00166	13.63	0.022	0.00904	0.00189	13.85
10	0.04537	0.00462	0.00147	13.68	0.0224	0.00758	0.00163	13.73
15	0.04444	0.00413	0.00131	13.77	0.02222	0.00664	0.00144	13.78
20	0.04541	0.00381	0.00111	13.69	0.0229	0.00621	0.00137	13.69
25	0.045264	0.00342	0.00106	13.7	0.0217	0.00572	0.00125	13.89
30	0.0413033	0.00316	0.00098	13.85	0.02196	0.00528	0.00112	13.85
35	0.04554	0.00291	0.00091	13.68	0.02225	0.00503	0.00104	13.81
40	0.043665	0.00278	0.00087	13.75	0.020526	0.00485	0.00102	13.64
45	0.0450333	0.00261	0.00084	13.73	0.232311	0.00473	0.00098	13.6
50	0.042864	0.00257	0.00083	13.76	0.022	0.00465	0.00095	13.84

- Adadelta is an adaptive learning rate algorithm that builds on Adagrad by overcoming its biggest weakness: the drastic, monotonically decreasing learning rate.
- Adadelta adjusts the learning rate dynamically based on a sliding window of past gradients, instead of summing up all past gradients as Adagrad.
- It prevents the learning rate from decreasing too rapidly, which enables the model to keep learning even after countless training steps.

- It's most useful when you don't want to tune the learning rate manually and like an optimizer that improves over time without needing a specific schedule.

ANALYSIS:

- Reconstruction Loss declines from 0.00904 (epoch 5) to 0.00465 (epoch 50), indicating a good trend.
- Feature Distillation Loss goes down from 0.00189 to 0.00095, indicating better generalization.
- PSNR on test data varies from 13.6 to 13.89 and remains low throughout. The maximum PSNR (13.89) is achieved at epoch 25.

Table 6.4 TRAINING AND TESTING LOSSES WITH PSNR USING ADAGRAD OPTIMIZER

3. ADAGRAD								
	TRAINING OF THE MODEL				TESTING OF THE MODE			
NO.OF EPOCH	KD LOSS	RECONSTRUCTION LOSS	FEATURE DISTILLATION LOSS	PSNR(db)	KD LOSS	RECONSTRUCTION LOSS	FEATURE DISTILLATION LOSS	PSNR(db)
5	0.02846	0.00541	0.00159	18.8	0.02444	0.00897	0.00181	15.23
10	0.01807	0.00494	0.00141	22.69	0.02369	0.00742	0.00162	17.89
15	0.01993	0.00451	0.00127	23.84	0.02973	0.00678	0.00144	20.58
20	0.02208	0.00402	0.00117	24.63	0.03387	0.00618	0.00136	21.39
25	0.011184	0.00374	0.00103	25.1	0.015368	0.00572	0.00123	21.78
30	0.0245	0.00343	0.00098	25.37	0.01298	0.00531	0.00114	22.81
35	0.02754	0.00322	0.00093	25.87	0.03859	0.00509	0.00108	22.98
40	0.02724	0.00306	0.00088	25.95	0.0519	0.00491	0.00103	25.31
45	0.05075	0.00294	0.00084	23.25	0.05549	0.00481	0.001	23.4
50	0.006242	0.00289	0.0008	26.14	0.04328	0.00472	0.00097	24

- Adagrad (Adaptive Gradient Algorithm) is an algorithm for optimizing, which adapts the learning rate for every parameter separately and without asking and automatically during training.

- It provides bigger updates to rare parameters and tiny updates to common ones.
- That is beneficial for sparse features or data that occur infrequently.

ANALYSIS:

Nevertheless, its key weakness is that the learning rate continues to decrease in size over time, eventually halting learning.

- Reconstruction Loss decreases steadily, from 0.00897 to 0.00472, indicating better generalization.

  Feature Distillation Loss has a small decrease, from 0.00181 to 0.00097, ensuring consistent student-teacher agreement at inference.
- PSNR increases from 15.23 dB to a high of 25.31 dB at epoch 40, after which it settles around 24 Db

Table 6.5 TRAINING AND TESTING LOSSES WITH PSNR USING RMSPRO OPTIMIZER

4. RMSprop								
	TRAINING OF THE MODEL				TESTING OF THE MODEL			
NO.OF EPOCH	KD LOSS	RECONSTRUCTION LOSS	FEATURE DISTILLATION LOSS	PSNR(db)	KD LOSS	RECONSTRUCTION LOSS	FEATURE DISTLLATION	PSNR(db)
5	0.0561	0.00489	0.00148	17.33	0.01496	0.00785	0.00166	19.01
10	0.04318	0.00441	0.00125	16.93	0.01251	0.00632	0.00137	19.32
15	0.01222	0.00397	0.0011	25.89	0.01045	0.00557	0.00119	22.06
20	0.01014	0.00361	0.00096	24.81	0.00965	0.00496	0.00107	22.2
25	0.023956	0.00329	0.00087	20.02	0.00856	0.00439	0.00093	21.64
30	0.00757	0.00302	0.00083	25.9	0.006623	0.00407	0.00085	23.21
35	0.006834	0.00284	0.00078	26.37	0.00744	0.00392	0.00082	23.71
40	0.00648	0.00268	0.00074	28.94	0.00636	0.00374	0.000078	21.24
45	0.006687	0.00254	0.00072	26.86	0.00554	0.00361	0.00076	25.96
50	0.005422	0.00247	0.00069	28.75	0.00543	0.00353	0.00073	29

RMS-prop (Root Mean Square Propagation) is an adaptive learning rate optimizer that scales the learning rate per parameter by the recent average of squared gradients.

- It maintains a rolling average of the squared gradients to normalize the updates.
- This stabilizes training by avoiding the learning rate from growing too big or too small.
- It is particularly strong for non-stationary targets and performs optimally in recurrent neural networks (RNNs) or noisy data. analysis
- Reconstruction Loss goes down from 0.00785 to 0.00353, showing improved generalization on unseen blurred images.
- Feature Distillation Loss reduces from 0.00166 to 0.00073, confirming good teacher-student correspondence at inference.
- Testing PSNR rises steadily from 19.01 dB to a remarkable 29.00 dB at epoch 50, the best among all optimizers seen till now.

RESULT AND DISCUSSION

The results of the program demonstrate how effectively a convolutional neural network (CNN) can be used to restore sharpness in blurry images through a teacher-student learning approach. First, the teacher model, being larger and more powerful, learned to generate clear, sharp images from their blurred counterparts using traditional training methods. The output quality was visually impressive and backed by high PSNR values, which reflect good image reconstruction quality.

Next, the student model—although smaller and simpler—was trained using knowledge distillation. This means it didn't just learn from the original sharp images, but also from the behaviour of the teacher model. This dual-source learning helped the student model perform surprisingly well despite having fewer parameters, making it faster and more lightweight.

When tested, the student model produced outputs that were very close in quality to the teacher’s, showing it had successfully absorbed the knowledge. Visually, the images looked much clearer, with significant reduction in blur, and PSNR scores confirmed the improved image quality. This makes the student model a practical choice for real-world applications where speed and efficiency matter, like mobile devices or real-time video enhancement.

CHAPTER 7

OUTPUTS

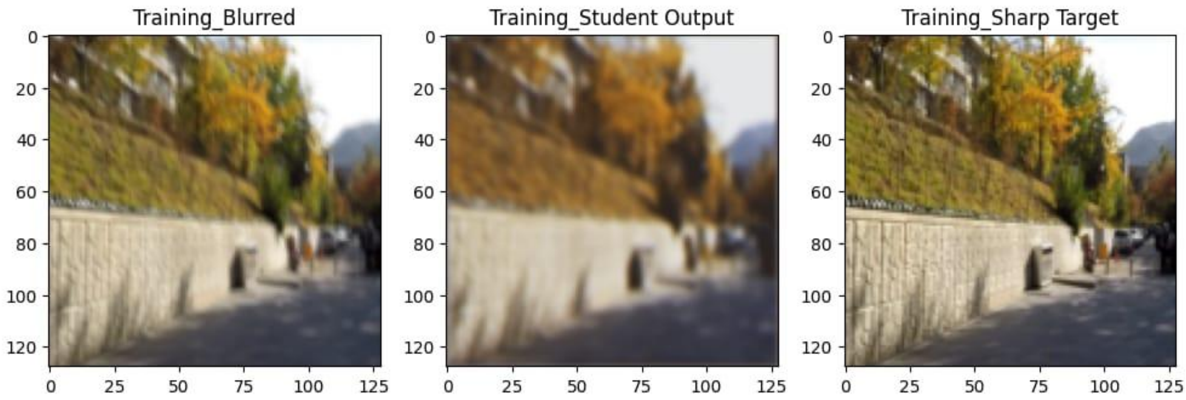

Figure 7.1 TRAINING OUTPUT OF THE MODEL

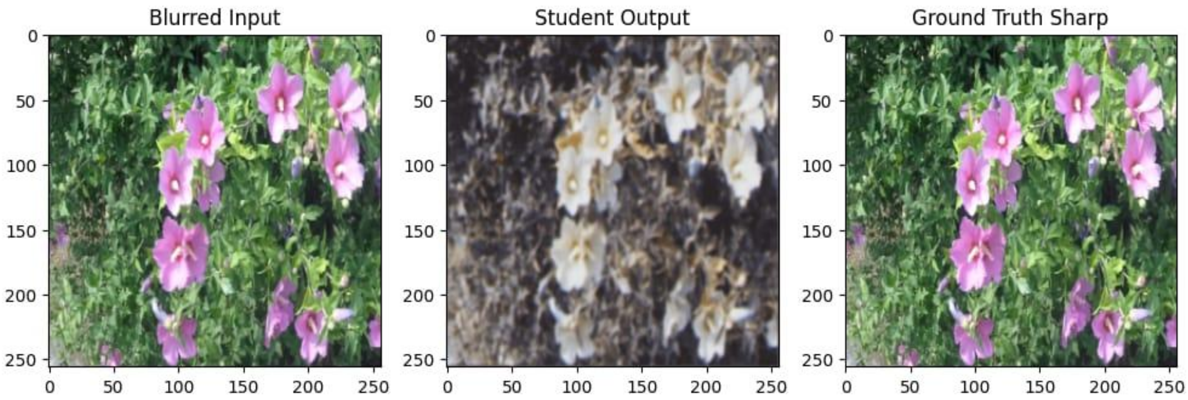

Figure 7.2 TESTING OUTPUT OF THE MODEL

CHAPTER 8

CONCLUSION

This project successfully demonstrated the effectiveness of using Convolutional Neural Networks (CNNs) for image de-blurring through a teacher-student learning approach. By first training a powerful teacher model and then guiding a lightweight student model using knowledge distillation, we achieved sharp image reconstruction with reduced computational cost. The student model was able to replicate the teacher's performance to a great extent, producing high-quality results even on unseen blurry images. This shows that we can design efficient and accurate models suitable for real-time or resource-constrained applications. Overall, the project highlights the potential of deep learning and model compression techniques in solving real-world image enhancement problems.

CHAPTER 9

FUTURE SCOPE OF THIS PROJECT

- Real-Time Deblurring: The student model, being lightweight, opens opportunities for real-time deployment on mobile devices, drones, or embedded systems where processing power is limited.

- Advanced Architectures: Future improvements can include using more sophisticated CNN architectures like U-Net, ResNet, or even transformer-based models to enhance deblurring accuracy further.

- Video Deblurring: Extending this work to video frames can help create smoother, clearer videos in low-light or fast-motion scenarios, which is useful in surveillance, cinematography, and AR/VR.

- Self-Supervised Learning: Exploring self-supervised or unsupervised approaches could reduce the need for large amounts of paired blurred and sharp images.

- Cross-Domain Application: The trained models could be adapted or fine-tuned for other domains like medical imaging, satellite image enhancement, or historical photo restoration.

CHAPTER 10

REFRENCES

1. Zamir, S. W., Arora, A., Khan, S., Hayat, M., Khan, F. S., & Yang, M.-H. (2021). Restormer: Efficient Transformer for High-Resolution Image Restoration. arXiv.
2. Tao, X., Gao, H., Liao, R., Wang, J., & Jia, J. (2018). **Scale-Recurrent Network for Deep Image Deblurring.** *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 8174–8182.
3. Zhang, K., Zuo, W., & Zhang, L. (2017). **Learning Deep CNN Denoiser Prior for Image Restoration.** *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 3929–3938.
4. Hinton, G., Vinyals, O., & Dean, J. (2015). **Distilling the Knowledge in a Neural Network.** *arXiv preprint arXiv:1503.02531.*
5. Romero, A., Ballas, N., Kahou, S. E., Chassang, A., Gatta, C., & Bengio, Y. (2014). **FitNets: Hints for Thin Deep Nets.** *arXiv preprint arXiv:1412.6550.*
6. Liu, P., Zhang, H., Zhang, K., Lin, L., & Zuo, W. (2020). **Multi-level Wavelet-CNN for Image Restoration.** *IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 773–782.
7. GoPro Dataset for Deblurring. https://seungjunnah.github.io/Datasets/gopro
8. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., ... & Chintala, S. (2019). **PyTorch: An Imperative Style, High-Performance Deep Learning Library.** *NeurIPS*, 8024–8035.
9. Wang, Z., Bovik, A. C., Sheikh, H. R., & Simoncelli, E. P. (2004). **Image Quality Assessment: From Error Visibility to Structural Similarity.** *IEEE Transactions on Image Processing*, 13(4), 600–612.
10. Redmon, J., & Farhadi, A. (2018). **YOLOv3: An Incremental Improvement.** *arXiv preprint arXiv:1804.02767.* (for real-time CNN performance references)
11. Goodfellow, I., Bengio, Y., & Courville, A. (2016). **Deep Learning.** MIT Press.