Intel Internship Program - 2025

Project Title:

Image Sharpening using Knowledge Distillation

Team Name:

FOV (Field of Vision)

Submitted By:

Name	Registration Number	Email ID
Eesha Thottempudi	VU22CSEN0300417	ethottem@gitam.in
Lakshmi Aishani Tetali	VU22CSEN0300403	lpetali@gitam.in
Aithi Mouleendra	VU22CSEN0100340	aaithi@gitam.in

Mentor:

Dr. Sheik Khadar Ahmad Manoj - ksheik@gitam.edu
8886854522

Institution:

GITAM
Department of Computer Science and Engineering
Visakhapatnam, Andhra Pradesh

Date:

July, 2025

I. Project Summary

Problem Statement:

Image Sharpening using Knowledge Distillation

In the age of hybrid communication, image clarity during video conferencing is often compromised due to low bandwidth or unstable internet connections. Our objective is to develop a lightweight yet effective deep learning model that can enhance image sharpness in real time making communication more visually effective and less tiring.

To achieve this, we use a Teacher-Student Knowledge Distillation approach, where a powerful deep neural network first learns to deblur images, and then a compact student model is trained to mimic its performance making the solution suitable for deployment in constrained environments.

Architecture Overview:

1. Teacher Model: Enhanced U-Net

The teacher model utilized in this project is an **Enhanced U-Net** architecture, a well-established model in image-to-image translation tasks due to its ability to preserve spatial information through skip connections. This model forms the foundation for high-quality image deblurring, offering a balance between detailed feature extraction and efficient reconstruction.

Key Architectural Features:

- ***Encoder-Decoder Structure***: The model follows the classic U-Net design comprising a contracting path (encoder) to capture context and an expansive path (decoder) for precise localization.
- ***Skip Connections***: Feature maps from the encoder layers are concatenated with corresponding decoder layers, facilitating gradient flow and preserving fine-grained image details.
- ***Standard Conv2D Layers***: All convolutional operations are performed using standard 2D convolutions with ReLU activations and batch normalization.
- ***Loss Function***: A custom hybrid loss combining Mean Squared Error (MSE) and Structural Similarity Index Measure (SSIM):
  Combined Loss = 0.5×MSE+0.5×(1-SSIM)

This configuration ensures that the model not only minimizes pixel-level errors but also maintains perceptual similarity, which is crucial in deblurring tasks.

Suitability:
The enhanced U-Net provides high-capacity learning and is well-suited for acting as a "teacher" in a knowledge distillation framework. It achieves strong performance in terms of SSIM and MAE, thereby setting a high benchmark for the student model to mimic.

2. Student Model: Lightweight U-Net Variant

The student model is a **compact variant of the U-Net**, specifically designed to reduce computational complexity without compromising output quality. It was designed to be lightweight, faster in inference, and suitable for resource-constrained environments, such as mobile or edge devices.

Key Architectural Modifications:

- ***SeparableConv2D Layers***: Depthwise separable convolutions replace standard Conv2D layers, significantly reducing the number of trainable parameters and computation.
- ***Skip Connections via Additive Fusion***: Instead of concatenating encoder and decoder features, the student model uses element-wise addition, thereby lowering memory overhead.
- ***Dilated Convolution in Bottleneck***: A dilated convolution layer is used in the bottleneck to expand the receptive field without increasing parameters, capturing more contextual information.
- ***Compact Parameter Size***: The entire model contains approximately 20,000 parameters, making it highly efficient.
- ***Custom KD Loss***: A three-part loss function combining MSE with ground truth, MSE with teacher output, and SSIM with ground truth:
  $\textbf{KD Loss} = \boldsymbol{\alpha} \cdot \textbf{MSEteacher} + \boldsymbol{\beta} \cdot \textbf{MSEGT} + \boldsymbol{\gamma} \cdot \textbf{(1-SSIMGT)}$

Where $\alpha = 0.7$, $\beta = 0.3$, and $\gamma = 0.1$ were chosen to balance fidelity to the teacher with generalization to the true output.

Suitability:
The student model maintains high perceptual similarity and acceptable reconstruction accuracy despite its significantly reduced capacity. This makes it suitable for deployment in scenarios where memory, latency, and power constraints are critical considerations.

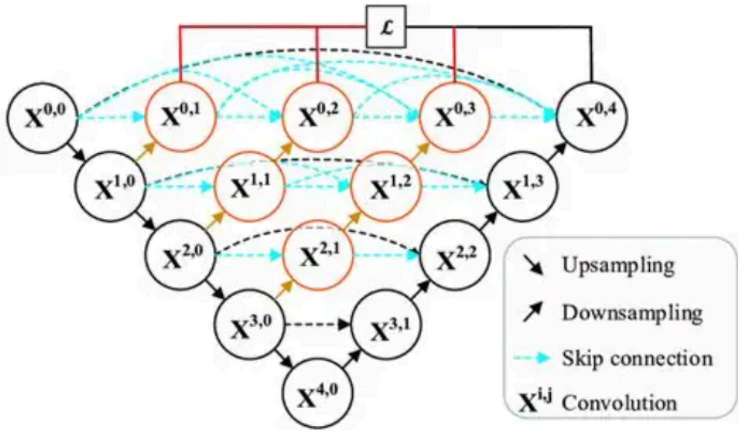
3. Final Statement

Although we initially explored various teacher-student architecture pairings like **EDSR-FSRCNN** and **SwinIR-MobileNetV2**, we ultimately opted for a more tailored approach. Our teacher model is based on an **enhanced U-Net** architecture optimized for image deblurring using a **combined MSE and SSIM loss**. The student model is a **lightweight U-Net variant** with **SeparableConv2D layers**, **reduced skip complexity**, and a **compact design** (~20K parameters). This custom pairing provided a balance of **performance and efficiency**, while maintaining **high perceptual quality**, well-suited for real-time or edge deployment scenarios.

II. Teacher Model - Enhanced U-Net++ Based Image Sharpening

1. Architecture Overview

The teacher model employed in this project is a deep convolutional neural network inspired by the U-Net++ architecture, specifically designed for the task of single-image deblurring. U-Net++ enhances the traditional U-Net by utilizing nested and dense skip connections, although in this implementation, a customized version of U-Net was used to balance complexity and performance.



The architecture consists of the following components:

- ***Double Convolutional Blocks***: Each block includes two convolutional layers followed by Batch Normalization and LeakyReLU activation functions.
- ***Encoder-Decoder Structure***: The model follows a symmetric downsampling (encoder) and upsampling (decoder) pipeline.

- ***Skip Connections***: Feature maps from the encoder are directly added to the corresponding decoder layers, facilitating the preservation of spatial features.
- ***Dropout Layers***: Incorporated in the bottleneck region to prevent overfitting and improve generalization.
- ***Transpose Convolution Layers***: Used for upsampling during the decoding phase to reconstruct high-resolution outputs.

This configuration results in a model with approximately **469,155 trainable parameters**, offering a good balance between expressiveness and efficiency.

2. Input-Output Flow

- ***Input***: A blurry RGB image of size **128×128**.
- ***Output***: A sharpened RGB image of the same dimension (**128×128**), aiming to closely match the original unblurred version.

3. Data Preparation

To create realistic training data, face images from the **CelebA dataset** were used. The preparation process involved the following steps:

- ***Blur Generation***: Each image was processed using either Gaussian blur or Motion blur, applied randomly.
- ***Image Pairs***: For each blurred image, the corresponding original image was preserved as the ground truth.
- ***Dataset Size***: A total of 8,000 blurry-sharp image pairs were generated.
- ***Data Split***: The dataset was divided into 80% for training and 20% for validation.

This approach helped simulate real-world blurring effects and ensured robustness in model performance.

4. Loss Function and Optimization

The teacher model was trained using a **custom combined loss function** designed to optimize both pixel-level accuracy and perceptual quality:

$\text{Loss}=0.5\times\text{MSE}+0.5\times(1-\text{SSIM})$

- ***Mean Squared Error (MSE)*** penalizes pixel-wise differences between the predicted and ground truth images.
- ***Structural Similarity Index Measure (SSIM)*** evaluates perceptual similarity, helping the model preserve structural details.

Additional training configuration:

- ***Optimizer***: Adam optimizer was used for its adaptive learning rate capabilities.

- ***Metrics***: Mean Absolute Error (MAE) and SSIM were tracked during training.

5. Training Configuration

- ***Epochs***: Maximum of 50, with EarlyStopping enabled.
- ***Batch Size***: 16
- ***EarlyStopping***: Patience set to 5, monitoring validation SSIM.
- ***Learning Rate Schedule***: ReduceLROnPlateau was used to reduce the learning rate when the validation SSIM did not improve, with a minimum threshold.

The model's training stopped at epoch 17, with the best performance recorded at **epoch 12**, after which no significant improvements were observed.

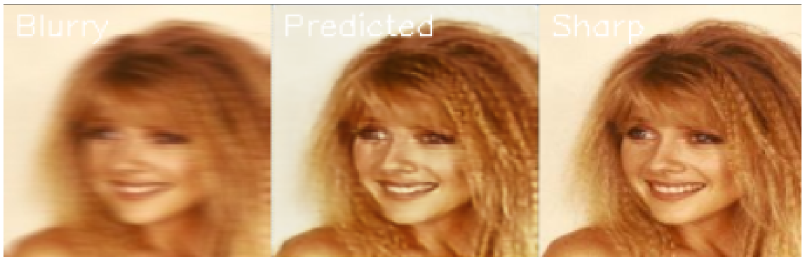
6. Final Evaluation Results

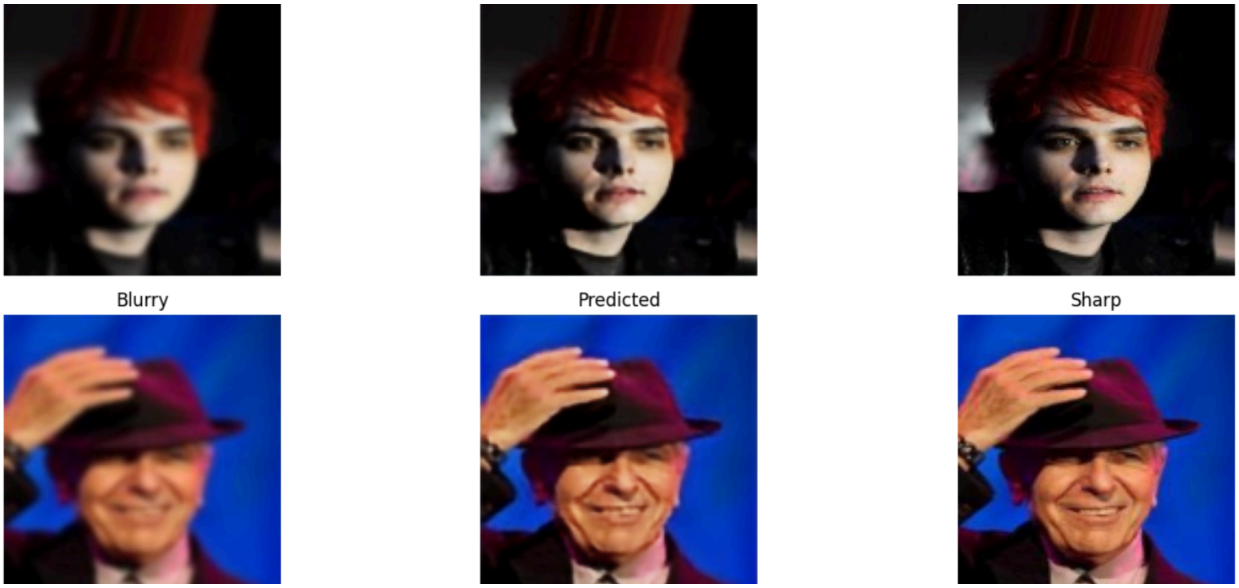
Metric	Value
SSIM	0.9296
MAE	0.0222
Parameters	~469,155
Best Epoch	12

These results confirm that the teacher model achieved high perceptual fidelity and low reconstruction error, establishing a strong reference for guiding the student model in the knowledge distillation process.

7. Sample Output

A visual comparison demonstrates the model’s effectiveness. The structure follows:





This illustrates the model's ability to generate perceptually sharp reconstructions from low-quality inputs.

8. Training Insights

The teacher model was trained over multiple epochs, with the following highlights:

- ***Best Epoch***: Epoch 12
- ***Validation SSIM*** at Best Epoch: 0.9069
- ***Validation MAE*** at Best Epoch: 0.0230
- Learning Rate Adjustment: Triggered at Epoch 15 due to stagnation in performance

Epoch-wise Summary (Selected)

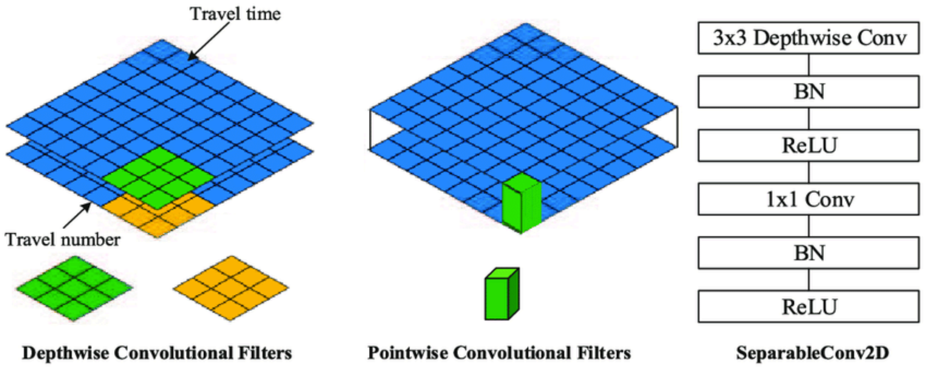
Epoch	Train Loss	Val Loss	Val SSIM	Val MAE	Notes
1	0.1374	0.1458	0.7173	0.0780	Initial convergence
2	0.0865	0.0697	0.8628	0.0320	Significant early improvement
4	0.0659	0.0570	0.8885	0.0345	Sharp gain in perceptual quality
7	0.0574	0.0521	0.8974	0.0257	Stable performance, SSIM rising

9	0.0537	0.0497	0.9022	0.0261	Peak quality approaching
12	0.0511	0.0472	0.9069	0.0230	Best Epoch (model saved)
15	0.0504	0.0515	0.8986	0.0258	Learning rate reduced to 0.0005
17	0.0479	0.0486	0.9040	0.0233	Training stopped, best weights restored

III. Student Model - Lightweight Knowledge Distillation Network

1. Architecture Overview

The student model, named LightStudentUNet_v2, is a compact convolutional neural network purposefully designed for image deblurring via knowledge distillation. It draws structural inspiration from U-Net but replaces heavy operations with lighter alternatives to significantly reduce parameter count while preserving spatial accuracy.



Depthwise Convolutional Filters

Pointwise Convolutional Filters

SeparableConv2D

Key architectural features include:

- ***SeparableConv2D Layers***: Replace standard convolutions in both encoder and decoder to reduce computational complexity and parameter count.
- ***Dilated Convolution in Bottleneck***: Expands receptive field without increasing parameters, enhancing the model's ability to restore fine-grained spatial information.

- ***Additive Skip Connections***: Skip connections are implemented via element-wise addition rather than concatenation, which helps conserve memory.
- ***LeakyReLU Activation (α=0.1)***: Used throughout the network to prevent dying ReLU issues.
- ***Sigmoid Output Activation***: Produces output in the normalized [0,1] range for pixel prediction.

The architecture follows the standard **Encoder → Bottleneck → Decoder → Refinement** pipeline and results in a model of approximately **20,190 trainable parameters**, a 23× reduction compared to the teacher model.

2. Input-Output Flow

- ***Input***: RGB blurry image of shape **128×128×3**
- ***Output***: RGB deblurred image of shape **128×128×3**
- ***Training Target***: Concatenated tensor of shape **128×128×6**, composed of:
  - Ground Truth sharp image (channels 1-3)
  - Teacher model prediction (channels 4-6)

3. Data Preparation

The training data used for the student model mirrors that of the teacher model:

- ***Dataset***: 8,000 sharp-blurry image pairs from the **CelebA** dataset
- ***Blur Simulation***: Gaussian and motion blur applied randomly to simulate real-world distortions
- ***Data Split***: 80% for training (14,400 samples), 20% for validation (3,600 samples)

To facilitate knowledge distillation, a custom data generator called **KDImagePairGenerator** was implemented. It performs the following steps for each training sample:

1. Loads a blurry input image
2. Loads the corresponding ground truth (GT) sharp image
3. Feeds the blurry image to the frozen **teacher model** to get predicted output
4. Concatenates the GT and teacher output to form a **combined training target**

This setup enables the student model to learn from both the ground truth and the intermediate representation provided by the teacher.

4. Loss Function and Optimization

The student model is trained using a **custom Knowledge Distillation loss function (KDLoss)**, which integrates teacher supervision and perceptual accuracy:

KDLoss = 0.7×MSE(yteacher,y^)+0.3×MSE(ygt,y^)+0.1×(1−SSIM(ygt,y^))

Where:

- y^: Student model output
- ygt: Ground truth sharp image
- yteacher: Teacher model prediction

Training Metrics:

- ***Mean Absolute Error (MAE)***: Measured against ground truth
- ***SSIM (Structural Similarity Index Measure)***: Also measured against ground truth

Optimizer:

- **Adam** optimizer with an initial learning rate of 1e-3, adjusted dynamically via ReduceLROnPlateau

5. Training Configuration

Hyperparameters	**Value**
Epochs	50
Batch Size	32
Initial Learning Rate	0.001
EarlyStopping Patience	8 (based on validation SSIM)
Learning Rate Schedule	ReduceLROnPlateau
Best Epoch Restored	Epoch 46
Checkpoint Location	/content/StudentModel3/student_model_kd3.h5

6. Final Evaluation Results

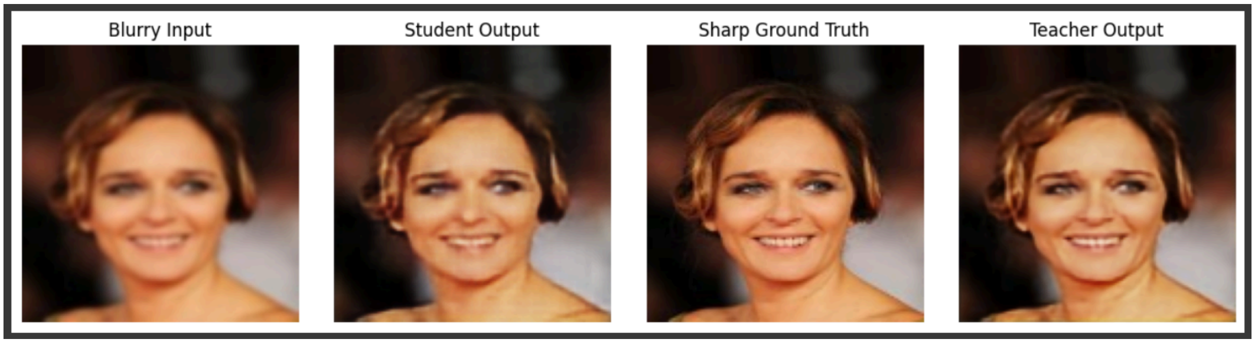
Metric	Teacher Model	Student Model
SSIM	0.9301	0.9138

MAE	0.0222	0.0251
Parameters	469,155	20,190

Despite having only **~4.3%** of the teacher model's parameters, the student retains **98% of its SSIM score**, showcasing its high efficacy and suitability for deployment in edge and resource-constrained environments.

7. Visual Comparison

A visualization function (visualize_student_vs_teacher) was implemented to compare outputs side-by-side. The layout includes:



This comparative visual helps assess perceptual quality and structure preservation between the models.

8. Training Insights

The training process was monitored closely through validation metrics, and the best weights were automatically restored from the epoch with highest SSIM.

Epoch-wise Performance (Selected)

Epoch	Loss	Val Loss	Val SSIM	Val MAE	Notes
1	0.0626	0.0347	0.7310	0.0706	Initial convergence
4	0.0196	0.0175	0.8379	0.0321	Rapid quality improvement
10	0.0142	0.0143	0.8672	0.0278	Stable convergence phase

46	0.0117	0.0122	0.8943	0.0242	Best Epoch (checkpoint)
50	0.0118	0.0120	0.8883	0.0241	Final epoch, LR reduced

9. Summary and Deployment Suitability

The **LightStudentUNet_v2** architecture demonstrates that knowledge distillation can effectively compress a complex deblurring model into a lightweight version with minimal performance degradation. The student model achieved a near-parity SSIM score while reducing parameter size by over **95%**.

Its low memory footprint and high perceptual performance make it highly suitable for real-time inference, mobile applications, and edge deployment scenarios.

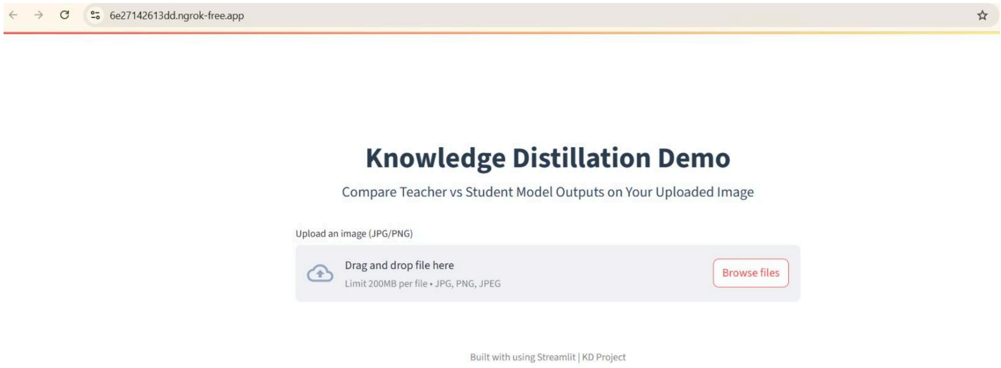
The final student model was exported and saved at:

/content/StudentModel3/stud_model_kd3.h5

This concludes the student model documentation section.

IV. Deployment Interface - Streamlit and Ngrok on Google Colab

To facilitate an interactive and user-friendly comparison between the outputs of the Teacher and Student models, a web-based interface was developed using Streamlit and temporarily hosted via Ngrok on Google Colab. This approach enables real-time inference and visualization using pre-trained models stored in Google Drive.



6e27142613dd.ngrok-free.app

Knowledge Distillation Demo

Compare Teacher vs Student Model Outputs on Your Uploaded Image

Upload an image (JPG/PNG)

Drag and drop file here

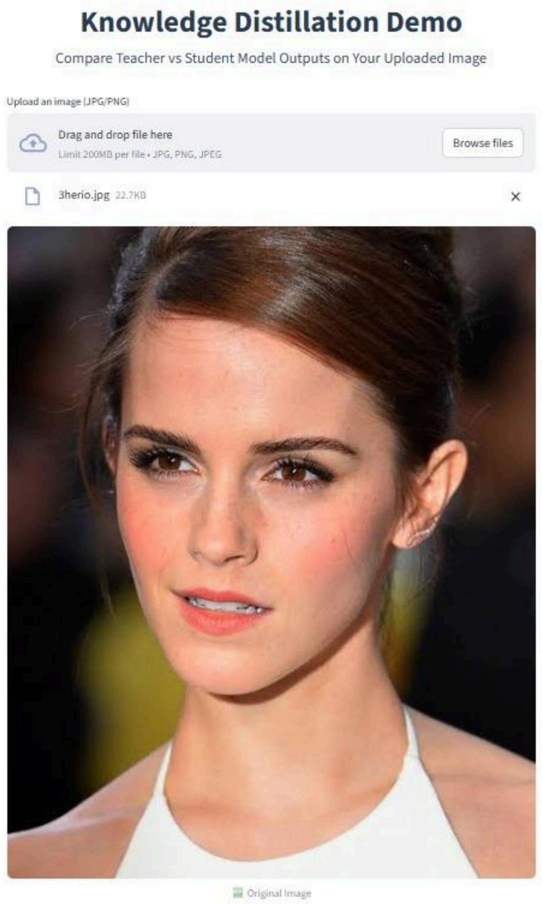
Limit 200MB per file • JPG, PNG, JPEG

Built with using Streamlit | KD Project

1. Interface Capabilities

The deployed interface supports the following features:

- Upload of any image file (JPG or PNG) from the user's local system.
- Visualization of side-by-side output from:
    - Teacher Model
    - Student Model
- Real-time prediction using models preloaded from Google Drive.
- Internal handling of image resizing, normalization, and post-processing.
- Organized UI with clearly separated columns and streamlined design for comparative analysis.



Knowledge Distillation Demo

Compare Teacher vs Student Model Outputs on Your Uploaded Image

Upload an image (JPG/PNG)

Drag and drop file here

Limit 200MB per file • JPG, PNG, JPEG

3herio.jpg 22.7KB

Original Image

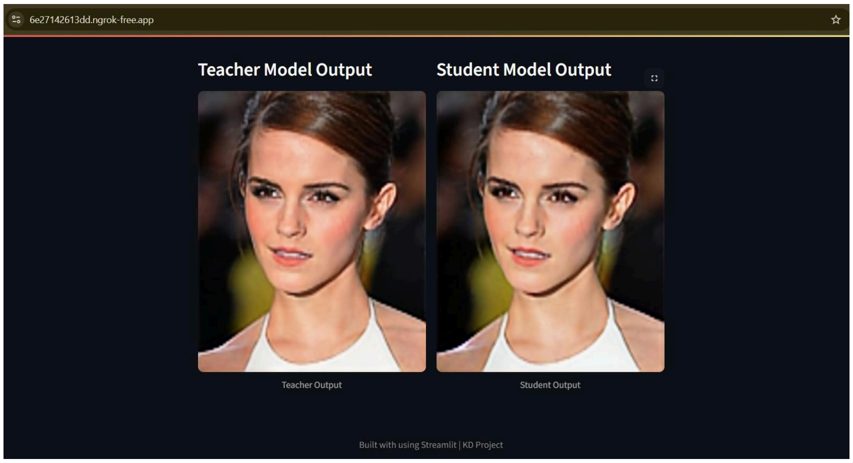
2. Technologies Utilized

Tool	Functionality Description
Streamlit	Web interface framework for Python applications
PyNgrok	Creates temporary public URLs to expose

	local services
Google Colab	Execution environment (GPU/CPU) for model inference
Google Drive	Persistent storage for trained **.h5** model files

3. Implementation Highlights

- The models (teacher_unet_alz.h5 and stud_model_kd3.h5) are stored on Google Drive and loaded into memory via the load_model() method.
- Image preprocessing includes resizing to 128×128 and normalization to [0, 1] range.
- Outputs are post-processed by scaling back to [0, 255] and converted into displayable images using PIL.
- Inference is triggered upon user upload, and results are displayed in two columns: one for each model.

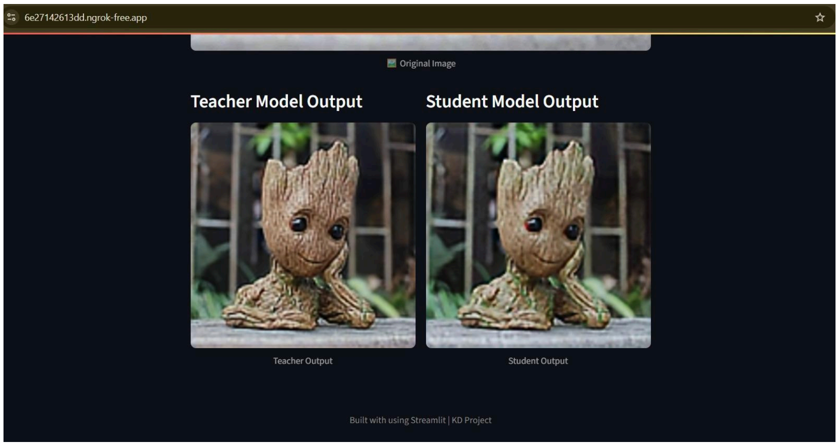


Built with using Streamlit | KD Project

4. Deployment Limitation: Session-Based Accessibility

Due to the use of **Ngrok within a Colab environment**, the deployed interface has the following constraints:

- The public link is valid only while the Colab notebook is active.
- The service becomes inaccessible if:
  - The runtime session is disconnected or idle.
  - The browser tab is closed, or execution is interrupted.
- As a result, this approach is not suited for long-term, production-level deployment.



6e27142613dd.ngrok-free.app

Original Image

Teacher Model Output

Student Model Output

Teacher Output

Student Output

Built with using Streamlit | KD Project

5. Technical Reasoning Behind Limitation

- Google Colab provides a temporary virtual machine (VM) for every user session.
- Once the session ends, the VM is terminated-shutting down both the local Streamlit server and the Ngrok tunnel.
- Since ngrok.connect() points to localhost:8501 on the Colab VM, the tunnel ceases to exist once the underlying host is no longer running.