Synopsis of

Image Sharpening using Knowledge Distillation

A Project Synopsis Submitted

In Consideration for Selection in the Summer Training Program

By

Himanshu Ranjan
(Enrollment Number: 20814802823)

Phone Number: 9718499168

and

Moksh Kaushish
(Enrollment Number: 21514802823)

Phone Number: 8920636055

Semester: IV
Batch: 2023–2027

Team Name: ***Synergy AI***

उद्यमेन हि सिध्यन्ति
कार्याणि न मनोरथैः

Maharaja Agrasen Institute of Technology
New Delhi, India

April 2025

Real-Time Image Sharpening for Video Conferencing via Knowledge Distillation: Technical Synopsis

1. Executive Summary

Video conferencing often looks blurry when internet bandwidth is limited. This can make remote classes, online medical consultations, and working with others online difficult. This project aims to create a small and fast computer program (specifically, a convolutional neural network or CNN) that can make blurry video clearer in real-time. We'll use a technique called "knowledge distillation" to teach this small program to be good at sharpening 1080p videos.

Our main goals are to make the sharpening happen quickly enough for smooth video (at least 30 frames per second) on regular computers, make the video look significantly better (with a Structural Similarity Index Measure or SSIM above 0.90), and keep the program small (under 5 megabytes). Unlike some existing methods that either don't improve the video quality much or need very large programs, our approach uses a "teacher-student" setup to achieve clear video even when the internet connection isn't great.

- **Why this matters:** Clearer video in online applications like remote medical diagnoses, online lectures, and screen sharing.
- **What we're aiming for:** 1080p video at 30 frames per second or faster, an SSIM score above 0.90, and a program size under 5 MB.

2. Technical Framework

2.1 Designing the Program's Structure

- **The "Teacher" Program:** We'll start with well-established CNN architectures used for making low-resolution images look high-resolution (like SRResNet). This "teacher" will be trained to produce the best possible sharp images. We'll train it using special techniques that focus on both the overall image quality (PSNR) and how good it looks to the human eye. We'll use standard, high-quality image datasets (like DIV2K and Flickr2K) for this initial training.
- **The "Student" Program:** This will be a much smaller CNN designed to run quickly. We'll use efficient building blocks like depthwise separable convolutions and possibly lightweight attention mechanisms (which help the program focus on important parts of the image). To make it even smaller and faster, we'll also explore "quantization-aware training," which allows us to use more efficient integer-based calculations (INT8).

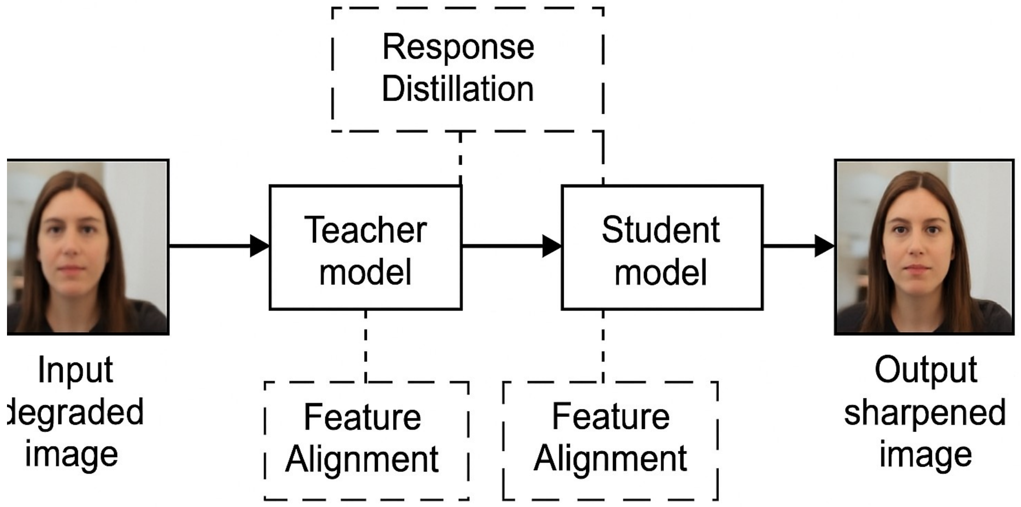

Figure 1. Teacher–Student distillation pipeline.

2.2 Teaching the Student: Knowledge Distillation

We'll use a step-by-step process to transfer the "knowledge" of the large, accurate teacher program to our small, efficient student program:

1. **Making Outputs Similar (Response Distillation):** We'll train the student to produce outputs that are very similar to the teacher's outputs when both are given blurry video as input. We'll use standard pixel-level comparison methods (like L1 or L2 loss) for this.
2. **Matching Internal Representations (Feature Alignment):** We'll also try to make the internal workings of the student program similar to the teacher program at certain key layers. This helps the student learn the important features that the teacher has learned. We'll use L2 loss to compare these internal representations.

2.3 Building Our Training Data

- **Training Images:** We'll leverage standard image datasets (like BSD500) to provide a foundation of diverse image content. To tailor the training to video conferencing scenarios, we will also capture and generate a targeted set of video conference-style images, focusing on representing key variations such as faces, text, and screen-sharing content. Rather than aiming for a very large number of unique frames, we will prioritize maximizing the diversity of our dataset through careful selection and data augmentation techniques. Our combined training dataset will include images and video frames, with a goal of at least 2,000 unique high-resolution source images, which will be expanded through augmentation.

- **Simulating Blurry Video:** We'll create realistic blurry video by:
    - **Simulating Compression:** Writing code to downscale and then upscale images, mimicking the loss of detail that happens with low-bandwidth video.

- **Adding Realistic Artifacts:** We'll also add things like motion blur, JPEG compression artifacts, random noise, and simulate data loss during transmission to make our training data more diverse.

3. Performance Validation

3.1 Quantitative Measurements

- **SSIM (Structural Similarity Index Measure):** We aim for a score above **0.90**, which means the video should look close to the original high-quality version.
- **Frame Rate:** Our program should run at **30 FPS or better** on a normal computer.

3.2 How Good Does it Look? Qualitative Evaluation

- **User Study (Mean Opinion Score – MOS):** We'll have around 20-30 people compare the original blurry video with the sharpened video. They won't know which is which, and they'll rate the sharpness and overall visual quality on a scale of 1 to 5.
- **Measuring Edge Sharpness:** We'll also objectively measure how much sharper the edges in the images become using the ratio of gradient magnitudes. We're aiming for a noticeable improvement (e.g., a ratio of 1.5-2.0) compared to the original video (which would have a ratio of 1.0).

4. Implementation Workflow

To keep things organized, we've broken down the project into these four main phases:

- **Phase 1:** This is our initial setup phase. We'll start by doing a Literature Review to see what other researchers have done. Then, we'll focus on getting the Teacher model working perfectly (Teacher Model Setup/Fine-tuning) and designing/training the Student model (Student Architecture Design & Initial Training). We also need to get our training data ready (Dataset Curation & Augmentation Pipeline).
- **Phase 2:** In this phase, we'll actually implement the knowledge distillation – this is the core of the project! We'll be experimenting with the different distillation methods (Response, Feature, SIFT-KD Exploration) and trying to optimize the Student model (Model Pruning/Optimization Exploration).
- **Phase 3:** Here, we'll work on making the model super-efficient using INT8 (INT8 Quantization-Aware Training/Calibration) and test its speed and performance (Performance Benchmarking). We'll also look at how to make it work with things like ONNX and WebRTC (Deployment Testing).
- **Phase 4:** Finally, we'll get feedback from users with our MOS study (Final Evaluation), write down everything we learned (Documentation), and put it all together in a report (Report Writing).

5. What Makes This Project Unique

- **Small but Powerful:** We're keeping the model tiny but making it work really well.

- **Smart Training:** We're using advanced training tricks like knowledge distillation to transfer skills from a big model to a small one.
- **Real Use Cases:** We're testing with real-world video conferencing data—faces, text, shared screens—so the tool works where it's needed most.
- **Practical Output:** We'll make sure the program can be used in different formats (**ONNX/TFLite**), so it can run on desktops, browsers, and even phones.

6. Expected Outcomes & Practical Impact

- **Deliverables:** Well-documented Python code (using PyTorch or TensorFlow), the trained program (in both standard floating-point format and potentially a more efficient integer format), exported versions of the program that can be easily used in other applications (ONNX/TFLite), scripts for measuring performance, and a final project report.
- **Better Communication:** Sharpening video will improve the clarity of shared documents, presentations, and facial expressions, which is crucial for effective remote work and learning. We also expect to see improvements in how well text can be read in the video.
- **Ready for Use:** Providing the program in standard formats (ONNX/TFLite) will make it easier to integrate into various platforms (desktop applications, web browsers using ONNX.js, and potentially mobile apps).

References

- Ledig, C., Theis, L., Huszár, F., Caballero, J., Cunningham, A., Acosta, A., Aitken, A., Tejani, A., Totz, J., Wang, Z., & Shi, W. (2017). Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4681-4690).
- Hinton, G., Vinyals, O., & Dean, J. (2015). Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.
- Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andriyanov, M., Adam, H., & Verbeek, C. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*. (Note: This was also published at CVPR 2017)
- Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, *60*(2), 91-110.
- Martin, D., Fowlkes, C., Tal, D., & Malik, J. (2001). A database of human segmented natural images and its application to evaluating segmentation algorithms and methodologies. In *Proceedings of the eighth international conference on computer vision* (Vol. 2, pp. 416-423). IEEE.