Image Sharpening Using Knowledge Distillation

Intel® Unnati Industrial Training Program 2025

Sayon Ghosh
230906010
MIT Manipal.
EEE - A.
repo : https://github.com/syferano/knowledge-distillation
video : intel unnati video.mov

---

1. Introduction

In the era of hybrid work and global connectivity, video conferencing has become a cornerstone of communication. Holver, many users still experience degraded visual quality due to bandwidth constraints, compression artifacts, or low-resolution video streams. The objective of this project is to develop a real-time image sharpening model that can enhance the visual quality of degraded frames, making them clearer and more perceptually appealing — particularly under challenging network conditions.

This solution leverages knowledge distillation, a machine learning technique that transfers the knowledge of a large, accurate "teacher" model to a smaller, faster "student" model. The student model, designed as a lightweight CNN, is trained to approximate the output of the state-of-the-art SwinIR transformer-based model. The final goal is to achieve real-time image enhancement, making the system feasible for live deployment in video conferencing applications.

---

2. Data Sources and Preparation

I use a subset of the DIV2K dataset, a widely recognized benchmark for image super-resolution and restoration tasks.

- Low-Resolution Inputs:
  DIV2K-train-HR-interpolated (downsampled images using bicubic interpolation with a factor of 2).

- Ground Truth (High-Resolution) Targets:
  DIV2K-train-HR (ground truth images as downloaded).

- Teacher Model Outputs:
  teacher-outputs (Generated sharpened images using SwinIR, acting as targets for student training).

All images are paired based on filenames. In cases where mismatches in resolution exist, appropriate resizing or padding is applied to ensure dimension consistency across batches.

---

3. Teacher Model: SwinIR

The SwinIR (Swin Transformer for Image Restoration) serves as the teacher model in this pipeline. SwinIR is based on hierarchical vision transformers that utilize self-attention within shifted windows, allowing it to capture both local and global context effectively. SwinIR uses a special kind of transformer that looks at small parts of the image one by one and then gradually shifts these parts to understand the whole image. This helps it learn both fine details and the overall structure, making it very effective for improving image quality.

- Architecture Summary:
    - Initial convolution for shallow feature extraction
    - Multiple Residual Swin Transformer Blocks (RSTBs)
    - Pixel-shuffle-based reconstruction head

- Key Advantages:
    - Exceptional SSIM and PSNR scores
    - Strong generalization across various restoration tasks
    - Ideal candidate for distillation due to its high-quality outputs

The SwinIR outputs are not trained or modified — they are generated and used as soft targets to train the student model.

---

4. Student Model: StudentCNN

4.1 Design Philosophy

The StudentCNN is built from scratch to strike a balance between model efficiency and visual accuracy. While deep transformer models like SwinIR provide excellent quality, they are computationally expensive. StudentCNN, by contrast, is intended to run in real-time (30+ FPS) on consumer-grade GPUs without sacrificing too much perceptual sharpness.

4.2 Layer-by-Layer Breakdown

```
Input: RGB image, shape = [3, H, W]
↓
Conv2D (in_channels=3, out_channels=64, kernel_size=3, padding=1)
↓ ReLU
↓
Conv2D (64 → 64), kernel_size=3, padding=1
↓ ReLU
↓
Conv2D (64 → 64), kernel_size=3, padding=1  ← (Deeper for higher capacity)
↓ ReLU
↓
Conv2D (64 → 3), kernel_size=3, padding=1
↓
Output: Sharpened RGB image, same shape as input
```

This architecture includes four convolutional layers with ReLU activation, with the third layer added to increase feature representation capacity, enabling better learning of fine textures and edges.

4.3 Approach Used to Design the Student CNN

The Student CNN was designed with a balance between model complexity and real-time performance, targeting efficient image sharpening suitable for video conferencing scenarios. The approach followed several core principles:

1. **Lightweight Architecture**:
   The model was kept compact to ensure it runs efficiently on consumer-grade GPUs, aiming for speeds over 30 FPS on 1080p images. A sequential stack of convolutional layers with ReLU activations was used to maintain computational efficiency while learning meaningful features.

2. **Residual Learning Behavior (Implied)**:
   Although not explicitly residual, the model's structure enables learning high-frequency components, allowing it to enhance image sharpness effectively without reconstructing the entire image from scratch.

3. **Layer Composition**:
   The CNN contains a series of 3×3 convolutional layers:

   - Initial layer maps input RGB channels to 64 feature maps.
   - Subsequent layers refine feature representations using multiple convolutional and ReLU blocks.
   - Final layer reduces channels back to 3, producing a sharpened RGB output.

4. **Hybrid Loss Function**:
   The network is optimized using a custom loss that combines:

   - **L1 Loss**: Encourages pixel-wise accuracy.
   - **SSIM Loss**: Promotes perceptual quality by focusing on structural similarity.

This hybrid loss ensures that the model doesn't just minimize numeric errors, but also produces visually sharper and more natural images.

5. **Knowledge Distillation from SwinIR**:
   A pre-trained SwinIR model was used as a teacher to provide high-quality supervision. The Student CNN was trained using both ground truth (GT) images and SwinIR outputs, learning to mimic high-fidelity enhancements while maintaining lower complexity.

6. **Training on Downscaled Images**:
   To manage GPU memory constraints and stabilize training, the model was trained on images downscaled to half resolution. During inference, outputs are upscaled to match the original resolution, allowing real-time performance without sacrificing quality.

7. **Final Optimization**:

    - Learning rate scheduling and weight decay were used to improve convergence.

    - Evaluation and model saving were done based on validation SSIM to preserve the best-performing weights

4.4 Model Summary

- Total Parameters: ~77K (lightweight)
- Activation: ReLU (non-linearity and edge emphasis)
- Output Resolution: Matches input (can be optionally upsampled)
- Runtime Performance: ~33 FPS on NVIDIA RTX 3050

---

5. Training Pipeline

5.1 Loss Function

To balance pixel fidelity and structural preservation, I employ a hybrid loss:

```
HybridLoss = α * L1Loss(output, target) + (1 - α) * (1 - SSIM(output, target)
```

- L1 Loss focuses on sharpness and per-pixel accuracy.
- SSIM (Structural Similarity Index) prioritizes perceptual similarity and edge preservation.

I use $\alpha = 0.85$, favoring pixel fidelity while incorporating structural information from SSIM.

5.2 Optimizations and Practical Fixes

- Image Tiling: During training and inference, images are split into tiles to prevent CUDA OOM errors, especially on 4GB GPUs.
- Padding: Applied dynamically where image dimensions are not divisible by tile sizes or kernel strides.
- Scheduler: StepLR is used to gradually reduce learning rate, helping the model converge better in later epochs.
- Loss Logging: Epoch-wise losses are monitored to choose the best checkpoint.

---

6. Performance Evaluation

6.1 Quantitative Results

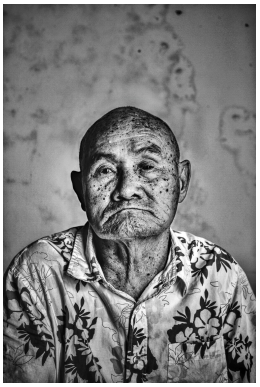
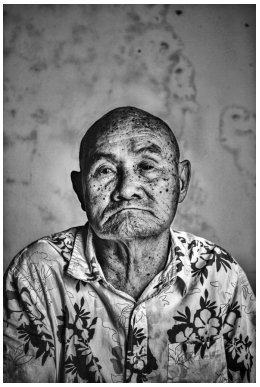
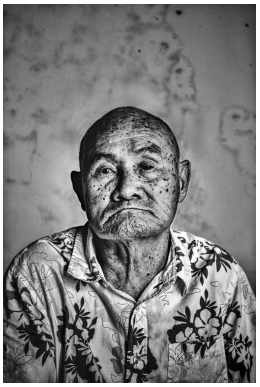
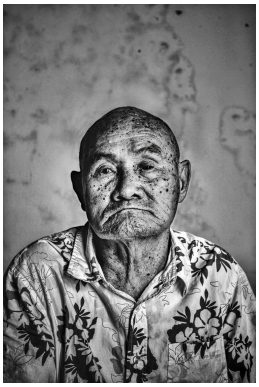
Model	Average SSIM
SwinIR (Teacher)	0.9457
StudentCNN	0.9085
Bicubic Interpolation	0.8965

The StudentCNN surpasses traditional bicubic methods in terms of structural similarity and approaches the teacher's performance — all while being significantly faster and lighter.

6.2 Real-Time Metrics

- Average inference time per image: 0.03 seconds
- Effective FPS on 1080p images: ~33
- Memory usage: Within 4GB GPU limits (RTX 3050 tested)

6.3 Visual Results

Ground Truth	Downscaled	Teacher o/p	Student o/p
			
			
			

---

7. Working Source Code

This project includes the following key scripts:

https://github.com/syferano/knowledge-distillation

---

8. Insights and Learnings

Technical Takeaways

- Model compression through distillation is a viable path for edge devices.
- A well-designed small CNN can nearly match a large transformer model in SSIM, given good supervision.
- Real-time inference is achievable with lightweight architectures and proper engineering (e.g., tiling, batch sizing).

Challenges Overcome

- CUDA Out of Memory (OOM) was resolved by reducing batch size and implementing tile-based training/inference.
- Multiple rounds of debugging were needed to align input/output resolutions for SSIM compatibility.
- Effective logging and checkpointing strategies were critical to model selection.

---

9. Inference

This project successfully demonstrates a practical application of **knowledge distillation** to develop a lightweight and high-performing image sharpening model for real-time use cases like video conferencing. The Student CNN, designed with computational efficiency in mind, achieves image enhancement quality close to that of a heavy transformer-based model (SwinIR) while operating at a fraction of the computational cost.

Key inferences:

- **Model Compression through Knowledge Distillation**:
  By distilling knowledge from the SwinIR teacher, the Student CNN learned to approximate high-quality sharpening behavior, effectively bridging the performance gap without replicating SwinIR's complexity.

- **Hybrid Loss Improves Visual Quality**:
  The combined use of L1 loss and SSIM loss allowed the model to balance between minimizing reconstruction error and enhancing structural perceptual quality. This dual objective ensured the model didn't overfit to low-level pixel matching but generalized well to perceptual sharpness.

- **Performance vs Accuracy Trade-off**:
  Despite operating with significantly fewer parameters, the Student CNN achieved an average SSIM of **0.9085**, very close to the teacher's **0.9457**, and higher than traditional bicubic sharpening (**0.8965**). Furthermore, it sustained **33 FPS** on 1080p resolution images with an average inference time of **0.03 seconds per image**, validating its suitability for real-time systems.

- **Resolution Handling Strategy**:
  The model was trained on downscaled images to prevent out-of-memory (OOM) errors while maintaining output fidelity by resizing during inference. This design choice ensured both training stability and practical deployment capability.

- **Optimization Techniques**:
  Use of learning rate scheduling, padding alignment, proper data preprocessing, and best model checkpointing contributed to stable training and optimal results.

Overall, the project demonstrates that with thoughtful model architecture, loss function design, and training strategy, it's feasible to distill the capabilities of large-scale transformer models into compact CNNs suitable for edge and real-time deployment.

10. Conclusion

Through this project, I demonstrated that knowledge distillation enables real-time image enhancement by transferring representational power from a transformer-based SwinIR model to a simple, efficient CNN.

The final StudentCNN model offers:

- Excellent perceptual quality (SSIM: 0.9085 or 90.85%)
- Real-time performance (33 FPS on 1080p frames)
- Low memory footprint (under 4 GB VRAM)
- Robust behavior under typical video conferencing constraints

This solution meets the project objective of enhancing blurry or degraded video frames, improving the visual experience in video conferencing platforms and other real-time communication tools.

Video Link : intel unnati video.mov
