Comparative Study of SRCNN and ESRGAN for Remote Sensing Image Super-Resolution on the Satellite Image Caption Generation Dataset

Harshdeep Singh[a]

[a]Sri Guru Granth Sahib World University, Fatehgarh Sahib, India

ABSTRACT

Super-resolution (SR) is crucial in remote sensing, where satellite image quality impacts land-use analysis, object detection, and caption generation. This paper compares the Super-Resolution Convolutional Neural Network (SRCNN) and Enhanced Super-Resolution Generative Adversarial Network (ESRGAN) on the Kaggle *Satellite Image Caption Generation* dataset. Low-resolution inputs were generated via bicubic downsampling ($\times 2$ and $\times 4$). Reconstructions were evaluated against standardized 256×256 ground truth images using Structural Similarity Index Measure (SSIM) and Peak Signal-to-Noise Ratio (PSNR). Results show ESRGAN consistently outperforms SRCNN, especially in texture-rich scenes, by producing sharper details and higher perceptual fidelity. This study highlights the role of GAN-based SR models for enhancing satellite imagery quality.

Keywords: Super-resolution, SRCNN, ESRGAN, Satellite Imagery, SSIM, PSNR, Remote Sensing

1. INTRODUCTION

Satellite imagery plays a vital role in remote sensing tasks such as land-use classification, environmental monitoring, and language-based caption generation. However, sensor limitations, atmospheric interference, and storage constraints often lead to low-resolution (LR) imagery that lacks fine textural and structural details. Image super-resolution (SR) aims to reconstruct high-resolution (HR) images from LR inputs, thereby enhancing interpretability and downstream task performance.

Traditional methods, such as nearest-neighbor, bilinear, and bicubic interpolation, are fast but cannot reconstruct missing high-frequency content, leading to blurred outputs and smoothed edges. Deep learning-based SR has changed this landscape: CNN models like SRCNN[1] demonstrated the feasibility of learning an LR-to-HR mapping directly from data, while GAN-based methods like SRGAN[2] and ESRGAN[3] enhanced perceptual realism via adversarial and perceptual losses.

This work conducts a controlled comparison of SRCNN and ESRGAN using a standardized SR evaluation protocol on the Kaggle *Satellite Image Caption Generation* dataset.[4] We evaluate both $\times 2$ and $\times 4$ upscaling using SSIM (primary) and PSNR (secondary), with careful attention to color handling, normalization, and image pairing. Quantitative findings are complemented by qualitative side-by-side panels and patch zooms.

1.1 Importance of SR in Remote Sensing

High-quality SR benefits:

- **Urban Planning:** Clearer road networks, roofs, and boundaries enable improved mapping and infrastructure planning.
- **Agriculture:** Better crop texture and field boundaries aid monitoring, irrigation optimization, and yield estimation.
- **Disaster Response:** Faster and more accurate damage assessment depends on recognizable object-level details.
- **Environmental Monitoring:** Tracking deforestation, coastline changes, and water bodies requires spatial detail beyond raw LR.

Further author information: (Send correspondence to H.S.)
H.S.: E-mail: harsh0129h@gmail.com

1.2 Limitations of Interpolation-Based Upscaling

Interpolation methods do not invent new information and cannot reconstruct lost high frequencies. Outputs tend to be smooth with poor edge definition, limiting performance in tasks requiring crisp boundaries and textural cues.

1.3 Advances in Deep Learning for SR

SRCNN[1] introduced an end-to-end CNN with three convolutional stages (patch embedding, non-linear mapping, reconstruction). **ESRGAN**[3] uses RRDBNet, a deeper residual-in-residual dense block network, with relativistic GAN loss and perceptual loss to reconstruct sharp, realistic textures. GAN-based SR is known to provide high perceptual quality that aligns better with human vision for complex textures.

2. DATASET

We use the Kaggle *Satellite Image Caption Generation* dataset,[4] which provides a diverse collection of RGB satellite images spanning urban, vegetation, water bodies, deserts, and farmland. Although the dataset was originally curated for image captioning, its variety and visual complexity make it apt for SR assessment.

2.1 Preprocessing and Standardization

To ensure consistent evaluation:

- **HR Ground Truth:** All selected images were standardized to 256×256 resolution and converted to `PNG` to avoid JPEG compression artifacts.
- **LR Inputs:** LR images were generated via bicubic downsampling (OpenCV) at ×2 (128×128) and ×4 (64×64).
- **Naming and Pairing:** Filenames were normalized so that HR and predictions align by stem name. Evaluation scripts strip suffixes like `_x2`, `_x4`, `_rlt`, and method tags for robust pairing.

2.2 Directory Structure

- `data/processed/HR/` – Ground truth 256×256 PNGs.
- `data/processed/LR_x2/`, `LR_x4/` – Bicubic LR inputs.
- `results/SRCNN_x2/`, `results/SRCNN_x4/`, `results/ESRGAN_x2/`, `results/ESRGAN_x4/`.
- `results/visuals/` – Generated plots and panels.
- `eval_results.csv` – Per-image metrics (SSIM/PSNR).

2.3 Sanity Checks

Before metric computation, we validate:

1. Each prediction has an HR match by stem name.
2. Each prediction is RGB; channel ordering is consistent.
3. Shapes match HR (256×256); otherwise, we resize with bicubic for metric stability.

3. METHODOLOGY

3.1 Models

SRCNN:[1] A 3-layer CNN with patch embedding, non-linear mapping (ReLU), and reconstruction. Used as a strong, simple CNN baseline; implemented in PyTorch with pretrained weights.

ESRGAN:[3]

- **Backbone:** RRDBNet (Residual-in-Residual Dense Blocks) to enable deep feature reuse and stable training.
- **Losses:** Relativistic GAN (RaGAN) and VGG-based perceptual loss to sharpen textures and improve realism.
- **Weights:** Pretrained RRDB_ESRGAN_x4 weights (official implementation).

3.2 Inference Setup

- **SRCNN:** Upscales LR inputs and outputs are aligned to 256×256 during evaluation.
- **ESRGAN:** Directly upsamples LR to HR (e.g., 64→256 at ×4).
- All outputs saved to `results/{METHOD}_{scale}` with consistent stems.

3.3 Evaluation Protocol

We compute SSIM and PSNR for each prediction against its HR counterpart:

- **SSIM:** Computed in RGB, with `channel_axis=2` and `data_range=1.0` on normalized floats.[5,6]
- **PSNR:** Computed from MSE with `data_range=1.0`.[5,7]
- **Aggregation:** Per-image CSV; grouped means by class/method/scale; overall means for headline results.

3.4 Qualitative Assessment

We include:

- **Side-by-side panels:** LR → SRCNN → ESRGAN → HR.
- **Patch zooms:** Crops over edges/texture regions (roofs, roads, vegetation).
- **Difference maps:** |HR − Prediction| to highlight residual errors and structural mismatches.

3.5 Hardware and Runtime

Experiments run on a GPU-enabled environment (e.g., NVIDIA RTX-class GPU). Inference is real-time for SRCNN and near-real-time for ESRGAN given 256×256 HR targets. Evaluation is implemented in Python with OpenCV, NumPy, scikit-image, and pandas.

3.6 Reproducibility

The full pipeline (data prep, model inference, metrics, and plots) is scripted in the repository.[8] The evaluation script ensures deterministic pairing and uniform preprocessing.

Table 1. Overall Mean SSIM and PSNR (higher is better).

Scale	Method	SSIM	PSNR (dB)
×2	SRCNN	0.693	25.03
×2	ESRGAN	0.837	29.92
×4	SRCNN	0.409	20.95
×4	ESRGAN	0.644	25.91

4. RESULTS & DISCUSSION

4.1 Quantitative Results

Table 1 summarizes the overall means across all evaluated images. Values are computed from the exported CSV and align with the standardized protocol.

ESRGAN consistently outperforms SRCNN across both scales, with larger margins at ×4. The SSIM gains reflect ESRGAN's ability to reconstruct fine structure and sharp textures, which is consistent with GAN-based perceptual optimization.

4.2 Per-Class Analysis

Texture-rich categories (e.g., urban/residential, forests, farmland) show stronger ESRGAN advantages due to complex edge/texture reconstruction. Smoother categories (e.g., deserts, bareland) show smaller margins and, in rare cases, similar PSNR between methods.

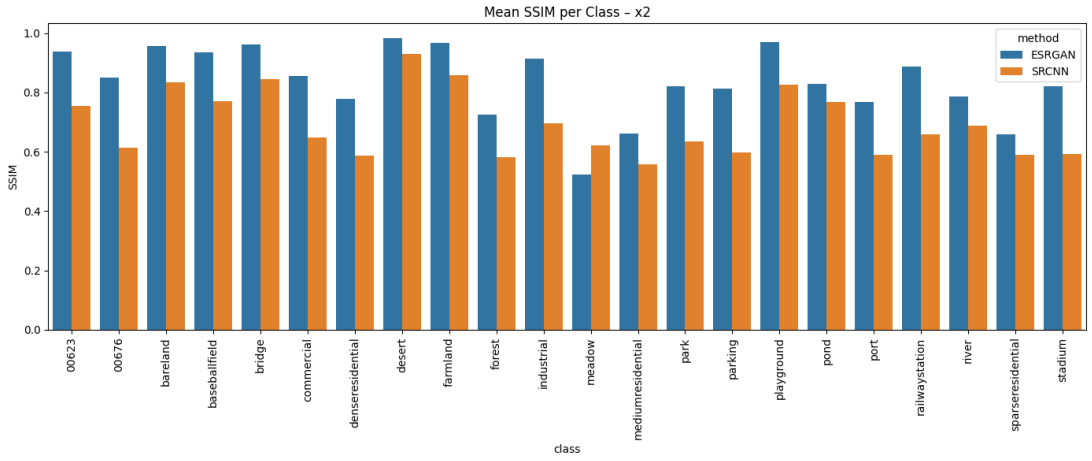

Figure 1. Mean SSIM per class at ×2 for SRCNN vs ESRGAN.

4.3 Delta SSIM Distribution

The histogram of ΔSSIM = SSIM(ESRGAN)−SSIM(SRCNN) shows a right-skewed distribution, indicating consistent improvements by ESRGAN across the majority of samples.

4.4 Qualitative Comparisons

Figure 4 shows a representative comparison panel. ESRGAN produces crisper edges and well-defined small structures (e.g., rooftops, roads), while SRCNN tends to oversmooth textures. Additional panels and zoomed patches are provided in the repository.

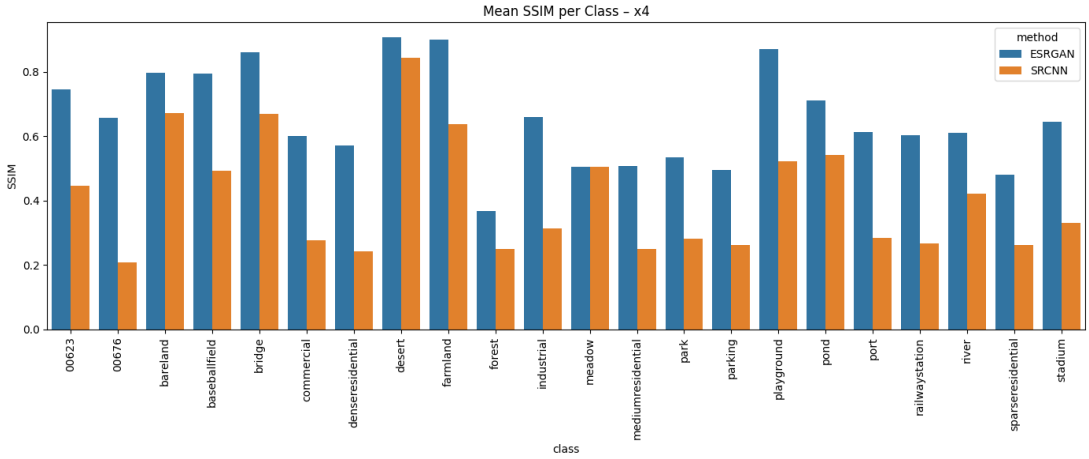

Figure 2. Mean SSIM per class at ×4 for SRCNN vs ESRGAN.

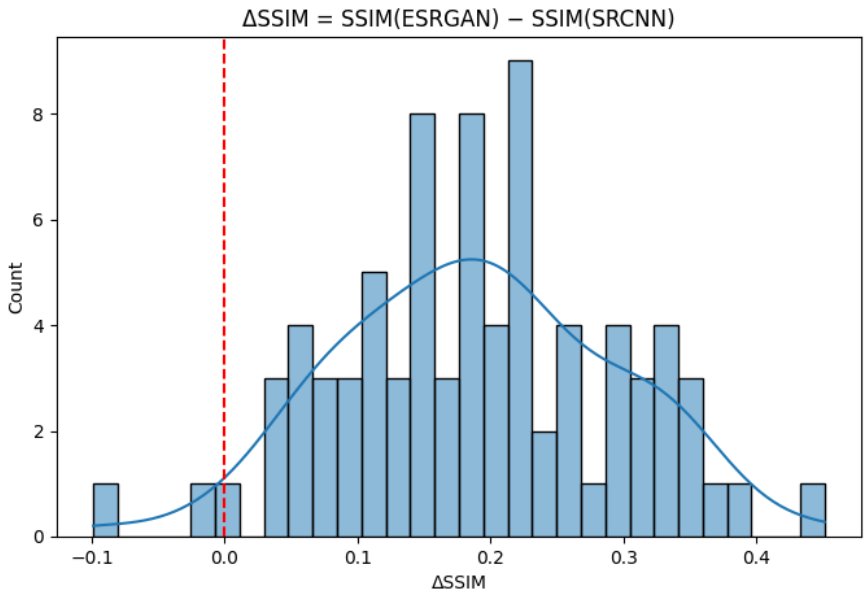

Figure 3. Distribution of ΔSSIM (ESRGAN minus SRCNN) across all images.

4.5 Discussion

SSIM vs PSNR: While PSNR improves with ESRGAN, SSIM captures perceptual quality differences more effectively, especially over textured regions. **Scale Sensitivity:** Both methods degrade at ×4, but ESRGAN remains more robust. **Use Cases:** ESRGAN is favorable for applications relying on human interpretability and texture fidelity; SRCNN may suffice where smoother outputs and lower compute cost are acceptable.

5. CONCLUSION

We presented a comparative study of SRCNN and ESRGAN for SR on the Kaggle *Satellite Image Caption Generation* dataset. Using a standardized evaluation protocol (bicubic degradation, RGB SSIM with channel-aware configuration, and PSNR on normalized floats), we showed that ESRGAN delivers superior structural and perceptual quality across ×2 and ×4 scales. Visual assessments corroborate quantitative gains.

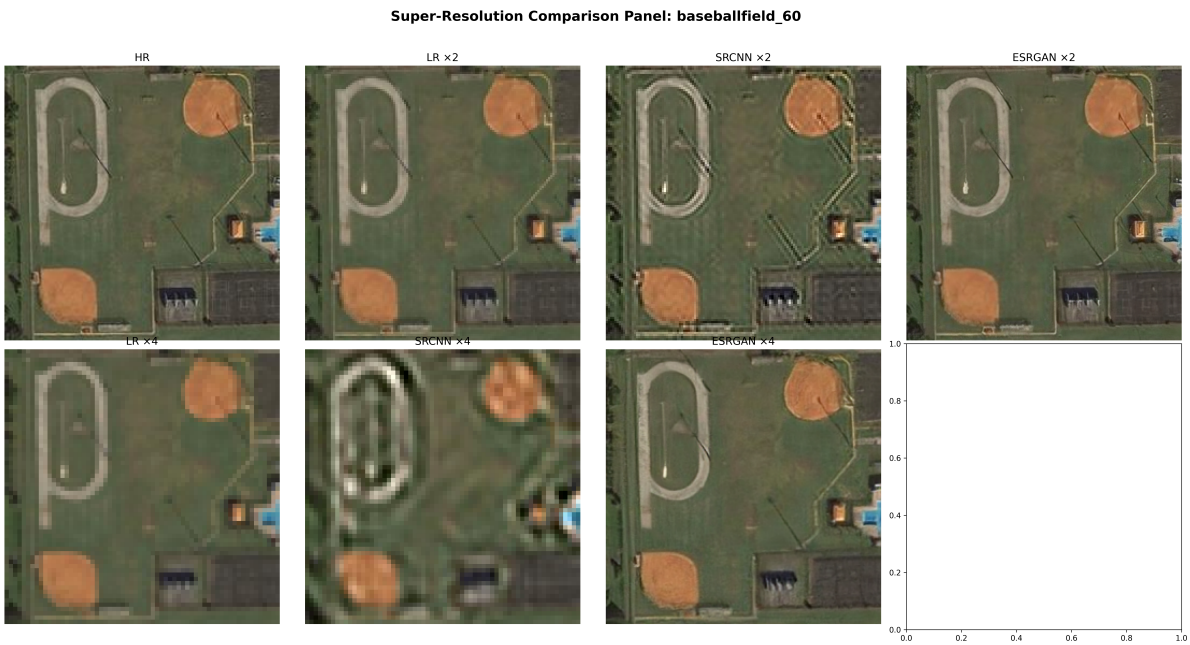

Figure 4. Representative side-by-side comparison (left to right): LR input, SRCNN, ESRGAN, HR ground truth.

Future work includes adding perceptual metrics (LPIPS, FID), exploring remote-sensing–specific SR models, and scaling to larger multi-sensor datasets. All scripts, metrics, and figures are available in the project repository[8] for full reproducibility.

REFERENCES

[1] C. Dong, C. C. Loy, K. He, and X. Tang, "Learning a deep convolutional network for image super-resolution," in *European Conference on Computer Vision (ECCV)*, pp. 184–199, Springer, 2014.

[2] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi, "Photo-realistic single image super-resolution using a generative adversarial network," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4681–4690, 2017.

[3] X. Wang, K. Yu, S. Wu, J. Gu, Y. Liu, C. Dong, Y. Qiao, and C. Change Loy, "Esrgan: Enhanced super-resolution generative adversarial networks," in *European Conference on Computer Vision (ECCV) Workshops*, 2018.

[4] T. Tillo, "Satellite image caption generation dataset." Kaggle, 2020.

[5] S. van der Walt, J. L. Schönberger, J. Nunez-Iglesias, F. Boulogne, J. D. Warner, N. Yager, E. Gouillart, and T. Yu, "scikit-image: image processing in python," *PeerJ* **2**, p. e453, 2014.

[6] GeeksforGeeks, "Image similarity — understanding and implementing methods," 2021.

[7] GeeksforGeeks, "Python — peak signal to noise ratio (psnr)," 2022.

[8] H. Singh, "Superresolutionproject." GitHub Repository, 2025.

APPENDIX A. REPRODUCIBILITY DETAILS

All preprocessing, inference, and evaluation code is available in our GitHub repository.[8] For clarity, we summarize the exact commands:

A.1 SRCNN Inference

```
python test_srcnn.py \
  --model models/srcnn.pth \
  --input data/processed/LR_x2 \
  --output results/SRCNN_x2
```

A.2 ESRGAN Inference

```
python test.py \
  --model models/RRDB_ESRGAN_x4.pth \
  --input data/processed/LR_x4 \
  --output results/ESRGAN_x4
```

A.3 Evaluation

```
python evaluate_models.py \
  --hr data/processed/HR \
  --pred results \
  --out eval_results.csv
```