ESRT: A Dual-Branch Kernel-Modulated Transformer for Single Image Super-Resolution

Pavitra Kumar
pavitrakumar108@gmail.com[1], **Manan**
mananmajithiya27@gmail.com[1]

[1]Department of Computer Science, Sardar Vallabhbhai National Institute of Technology, Surat

Abstract

While the Efficient Super-Resolution Transformer (ESRT) successfully introduced Transformers to lightweight Single Image Super-Resolution (SISR), the original single-path static architecture struggles with varying degradation patterns. In this paper, we propose an upgraded **ESRT**, a significant architectural evolution. Unlike the original baseline, our proposed ESRT utilizes a **Dual-Branch architecture** separating texture extraction (F-Branch) from structural preservation (S-Branch). We introduce a novel **Kernel-Aware FiLM** mechanism that dynamically modulates feature weights based on estimated degradation, and a **Gate Fusion** module that adaptively merges these branches. Furthermore, we employ **Pyramid Supervision** and a hybrid edge-aware loss function to stabilize training. Experimental results on benchmark datasets demonstrate that our ESRT achieves superior performance in both PSNR and perceptual quality compared to the baseline.

1 Introduction

Single Image Super-Resolution (SISR) is a fundamental low-level vision task that aims to recover a high-resolution (HR) image from its degraded low-resolution (LR) counterpart. It is an ill-posed inverse problem, as multiple HR solutions can correspond to a single LR input. In recent years, deep learning-based methods, particularly Convolutional Neural Networks (CNNs) such as EDSR [12] and RCAN [44], have achieved remarkable performance by learning complex mappings between LR and HR spaces. However, standard CNNs are inherently limited by their local receptive fields, which hinders their ability to model long-range dependencies and global structural patterns essential for high-fidelity reconstruction.

To overcome these limitations, Transformers, originally designed for natural language processing, have been successfully adapted for computer vision. Models like SwinIR [13] leverage self-attention mechanisms to capture global context, significantly outperforming CNN-based counterparts. However, the quadratic computational complexity of standard self-attention poses a significant challenge for deploying these models on resource-constrained devices.

The original Efficient Super-Resolution Transformer (ESRT) [33] addressed this by proposing a hybrid architecture that combines a Lightweight CNN Backbone (LCB) with a memory-efficient Transformer Backbone (LTB). While efficient, the original model suffers from a critical limitation: it employs a *static* architecture where feature weights are fixed after training. This "one-size-fits-all" approach is suboptimal for real-world scenarios where degradation patterns (e.g., blur levels, noise intensity) vary significantly across images. Furthermore, the serial processing pipeline forces both high-frequency textures and low-frequency structures to pass through the same bottlenecks, potentially leading to feature interference.

To address these issues, we propose an enhanced **ESRT**, a significant architectural evolution designed to maximize performance without sacrificing efficiency. Our approach introduces a dynamic, dual-path processing paradigm. Our contributions are summarized as follows:

- **Dual-Branch Topology:** We decouple the network into two specialized paths: a Filtered Branch (F-Branch) dedicated to high-frequency texture extraction and a Structure Branch (S-Branch) focused on preserving global structural consistency.
- **Dynamic Modulation:** We introduce a lightweight Kernel Code Estimator that analyzes the input degradation. It generates a latent code to condition the processing blocks via Feature-wise Linear Modulation (FiLM), allowing the network to adapt its filters dynamically for each input image.
- **Advanced Supervision:** Moving beyond simple L1 pixel-wise loss, we implement Deep Pyramid Supervision ($\times 2, \times 4$) and a hybrid edge-

aware loss function. This ensures that the model learns meaningful representations at multiple scales and prioritizes perceptual quality over mere pixel average.

2 Related Work

2.1 CNN-based SISR Models

Since the pioneering work of SRCNN [48], Convolutional Neural Networks (CNNs) have dominated the super-resolution landscape. Early approaches focused on increasing network depth to boost performance. For instance, EDSR [12] removed unnecessary batch normalization layers to construct a very deep residual network, significantly improving reconstruction fidelity. RCAN [44] further advanced the field by introducing Residual-in-Residual (RIR) structures and Channel Attention (CA) mechanisms to adaptively rescale channel-wise features.

Despite their success, these CNN-based methods suffer from a fundamental limitation: convolution operations process information locally. To capture global interactions, CNNs require very deep stacks of layers to expand the receptive field, which increases computational cost and optimization difficulty. Furthermore, standard CNNs often treat all image regions equally, lacking the ability to dynamically adapt to varying degradation levels within an image.

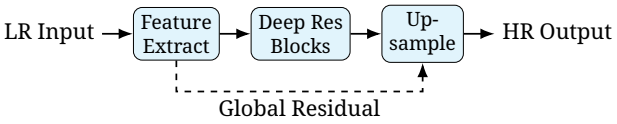

Figure 1: A typical CNN-based Super-Resolution architecture (e.g., RCAN/EDSR). Features are extracted via convolutions, refined through residual blocks, and upsampled at the end.

2.2 Vision Transformers

The introduction of the Transformer architecture revolutionized natural language processing and has recently shown great promise in computer vision. Vision Transformers (ViT) divide images into patches and utilize self-attention mechanisms to model long-range dependencies, overcoming the receptive field limitations of CNNs.

In the context of super-resolution, SwinIR [13] integrated the Swin Transformer block into a deep restoration network, achieving state-of-the-art performance. However, the heavy computational burden of self-attention mechanisms often makes purely Transformer-based models impractical for resource-constrained applications.

The original Efficient Super-Resolution Transformer [33] attempted to bridge this gap by proposing a hybrid architecture. It utilized a Lightweight CNN Backbone (LCB) for feature extraction and a Lightweight Transformer Backbone (LTB) with Efficient Multi-Head Attention (EMHA) to reduce memory usage. While the original ESRT successfully lowered the hardware barrier, it retained a static, serial architecture. Our work improves upon this by introducing a dynamic, dual-branch topology that modulates features based on input complexity, effectively addressing the ”static weights” limitation of the baseline.

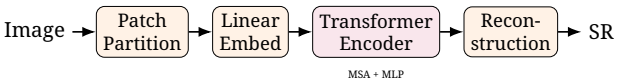

Figure 2: General Vision Transformer pipeline for low-level vision. The image is split into patches, embedded, and processed via Self-Attention mechanisms to capture global context.

3 Proposed Method

3.1 Overall Architecture

The original ESRT adopts a serial architecture where features pass sequentially through a CNN backbone and then a Transformer backbone. While effective for general cases, this design creates an information bottleneck where high-frequency textures and low-frequency structures compete for channel capacity.

As illustrated in Figure 3, our proposed **ESRT** fundamentally redesigns this flow into a parallel, dual-branch topology to better handle complex image degradations. The architectural pipeline consists of four distinct stages:

- **Shallow Feature Extraction:** The input image I_{LR} is first processed by a convolution layer to map the image space to a high-dimensional feature space. This extraction provides a shared foundation for the subsequent branches.
- **Dual-Branch Processing:** The extracted features are split into two parallel pathways. The *Filtered Branch (F-Branch)* focuses on recovering high-frequency details (textures) and is dynamically modulated by our Kernel Estimator. Simultaneously, the *Structure Branch (S-Branch)* utilizes Transformer blocks to model long-range dependencies and preserve global geometric structures.
- **Adaptive Fusion:** To merge the complementary information from both branches, we employ a **Gate Fusion** mechanism. Instead of simple concatenation, this module predicts a pixel-wise mask to weigh the contribution of texture versus structure adaptively for every region of the image.
- **Pyramid Reconstruction:** Finally, the fused features are upsampled using a pyramid strategy.

The network is supervised at both $\times 2$ and $\times 4$ scales, ensuring that intermediate upsampling layers learn meaningful representations and facilitating stable gradient flow during training.

Mathematically, the core fusion process is defined as:

$$F_{fused} = \text{GateFuse}(\Phi_F(F_0, K), \Phi_S(F_0)) \quad (1)$$

where F_0 denotes the shallow features, Φ_F represents the F-Branch, Φ_S represents the S-Branch, and K is the estimated kernel code derived from the input image.

3.2 Kernel-Aware FiLM Conditioning

A major limitation of traditional super-resolution models is that they are "static." Once trained, their internal weights are fixed, meaning they process every image exactly the same way. Whether an input image is slightly blurry, very noisy, or full of compression artifacts, the network applies the same mathematical operations. This is often suboptimal because different types of image degradation require different restoration strategies.

To solve this, we introduce a dynamic mechanism called **Kernel-Aware FiLM Conditioning**. Think of this as giving the network a "brain" that first looks at the image to understand its specific problems before trying to fix them. This process happens in two key steps:

1. Degradation Sensing (Kernel Code Estimator): We use a tiny, lightweight convolutional network—the Kernel Code Estimator—that scans the input low-resolution image (I_{LR}). Its job is not to upscale the image, but to identify the "fingerprint" of the degradation. It compresses this understanding into a compact vector $z \in \mathbb{R}^{64}$, which we call the kernel code. This vector effectively summarizes how blurry or noisy the current image is.

2. Dynamic Tuning (FiLM Modulation): Once we have the kernel code z, we use it to adjust the behavior of the main restoration network (specifically the F-Branch). We employ a technique called Feature-wise Linear Modulation (FiLM). For every feature map x in the network, the FiLM layer uses the code z to predict two adjustment factors: a scaling factor $\gamma(z)$ and a shifting factor $\beta(z)$. The modulation is applied mathematically as:

$$\text{FiLM}(x, z) = \gamma(z) \odot x + \beta(z) \quad (2)$$

In simpler terms, $\gamma(z)$ acts like a volume knob that can amplify or dampen specific features, while $\beta(z)$ shifts their baseline value. If the Estimator detects that the image is very blurry, it might tell the FiLM layers to "turn up the volume" on the high-frequency edge detection filters. If it detects noise, it might "turn down" those same filters to avoid amplifying the grain. This allows our ESRT to adapt its internal processing strategies on-the-fly for every single image, leading to much sharper and cleaner results compared to static models.

3.3 Dual-Branch and Gate Fusion

Instead of forcing all image features through a single pipe, we designed a **Dual-Branch System** that allows the network to specialize. Think of this as having two experts working on the same image simultaneously.

1. F-Branch (The Detail Specialist): This branch, known as the "Filtered Branch," is responsible for the intricate work. It uses the FiLM-modulated High Preserving Blocks (HPBs) to hunt for fine textures, sharp edges, and high-frequency details. Because it's modulated by the kernel code, it knows exactly how to handle the specific blur or noise present in the image.

2. S-Branch (The Structure Specialist): While the F-Branch focuses on the details, the "Structure Branch" looks at the big picture. It uses Transformer blocks to understand the global geometry and large shapes in the image. This ensures that while we are fixing the tiny details, the overall structure of objects (like the straight lines of a building or the curve of a face) remains consistent and doesn't get distorted.

3. Gate Fusion (The Smart Conductor): Merging these two experts requires finesse. Simply adding their outputs together (concatenation) is crude and often ineffective. To solve this, we devised a **Gate Fusion Module**. Imagine a slider that can move between the two branches for every single pixel.

- If the network looks at a patch of blue sky, the Gate might say: "This is smooth, I trust the S-Branch (Structure) more," and give it 90% weight.
- If it looks at a cat's fur, the Gate might say: "This is high texture, I need the F-Branch (Details)," and give that 90% weight.

Technically, this is achieved by a small neural network layer that outputs a mask G with values between 0 and 1. The final image feature is a weighted blend:

$$F_{fused} = G \odot F_f + (1 - G) \odot F_s \quad (3)$$

This ensures we get the best of both worlds: sharp details where needed, and clean, consistent structures everywhere else.

3.4 Pyramid Supervision and Loss Function

Training a deep network to instantly produce a high-resolution image is incredibly difficult. It is like asking a child to run before they can walk. To make this learning process easier and more stable, we use a strategy called **Pyramid Supervision**.

Instead of only checking the network's answer at the very end (the $\times 4$ output), we force the network to produce intermediate results. We make it generate a $\times 2$ (double size) image halfway through the process. We then compare this intermediate image to a ground-truth $\times 2$ image. This acts as a "stepping stone," ensuring the network is on the right track before it attempts the final, difficult $\times 4$ upscaling. This "deep supervision" helps gradients flow better

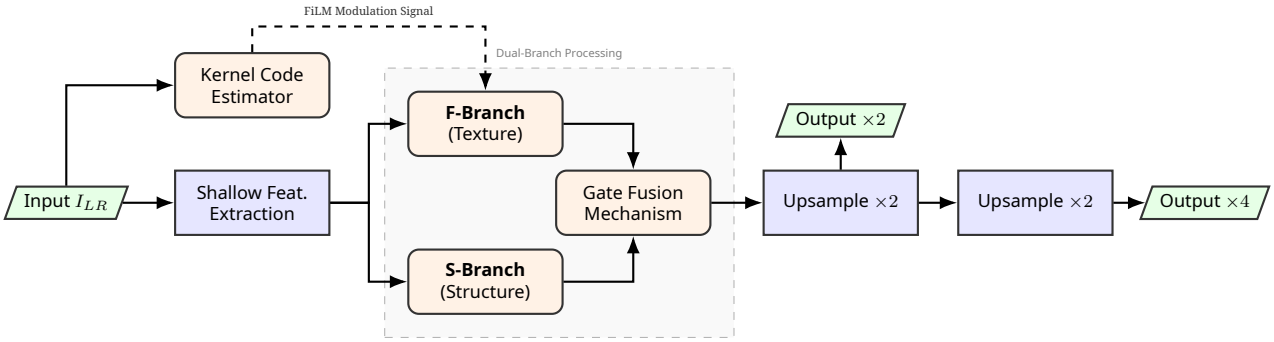

Figure 3: The architecture of our proposed **ESRT**. The input is processed by a Kernel Estimator to modulate the F-Branch via FiLM. Features are split into F-Branch (Texture) and S-Branch (Structure), fused via a Gate mechanism, and supervised at multiple scales ($\times 2, \times 4$).

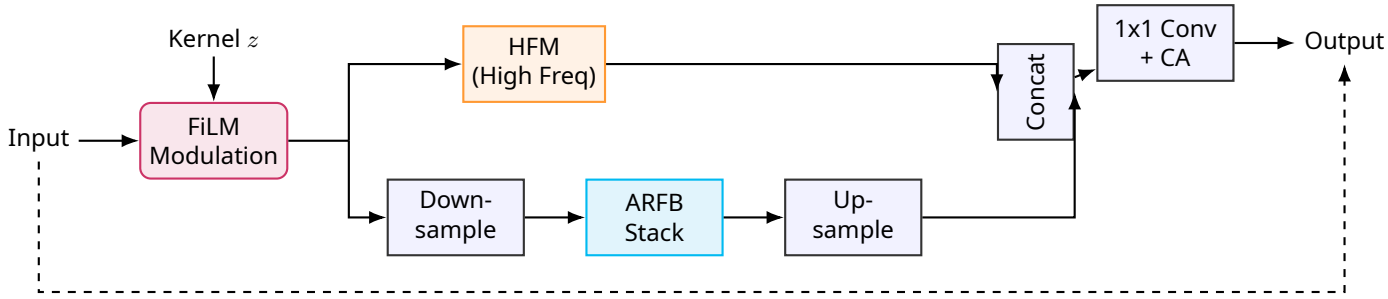

Figure 4: Schematic of the **FiLM-modulated High Preserving Block (HPB)**. Features are modulated by the degradation code z, then split into a high-frequency preserving path (HFM) and a downsampled processing path (ARFB), before being fused via Channel Attention (CA).

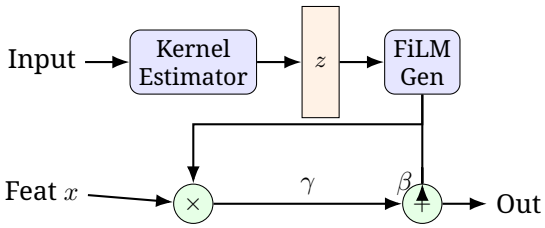

Figure 5: Detailed mechanism of the Kernel-Aware FiLM Conditioning. The code z generates γ (scale) and β (shift) to modulate features.

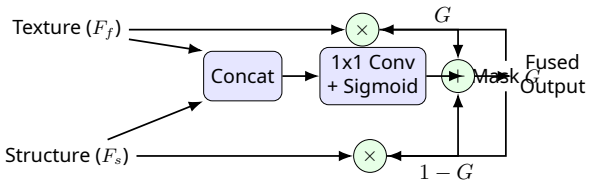

Figure 6: The Gate Fusion Mechanism. It generates a mask G to intelligently blend the Texture (F_f) and Structure (F_s) features.

through the network, preventing the "vanishing gradient" problem common in deep models.

Furthermore, how we grade the network's homework matters. A simple pixel-by-pixel check (L1 Loss) often results in blurry images because the network tries to be "safe" by averaging pixels. To fix this, we use a **Hybrid Loss Function** that combines three different "inspectors":

1. **L1 Loss (The Accountant):** This checks the raw pixel values. It ensures the colors and brightness are generally correct.
2. **SSIM Loss (The Architect):** This checks the structure. It looks at patches of the image to ensure the shapes and patterns resemble the original, rather than just individual pixels.
3. **Edge Loss (The Draftsman):** This is critical for sharpness. We apply a Sobel filter (an edge-detection tool) to both the generated image and the real image. We then calculate the difference between these edge maps. This penalizes the network heavily if it produces blurry edges, forcing it to draw crisp, sharp lines.

The total loss is a weighted sum of these three components:

$$\mathcal{L}_{total} = \mathcal{L}_{L1} + \lambda_1 \mathcal{L}_{SSIM} + \lambda_2 \mathcal{L}_{Edge} \tag{4}$$

By combining these, we get an image that is color-accurate (L1), structurally sound (SSIM), and razor-sharp (Edge).

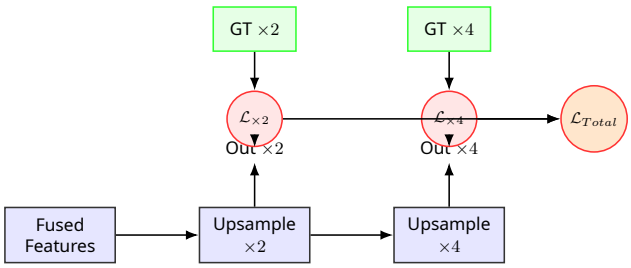

Figure 7: The Pyramid Supervision Strategy. The network is supervised at both $\times 2$ and $\times 4$ scales. The intermediate loss $\mathcal{L}_{\times 2}$ ensures stable feature learning, while $\mathcal{L}_{\times 4}$ enforces final quality.

4 Experiments

4.1 Data & Metrics

The Dataset (DF2K-OST): To teach our model effectively, we used a massive collection of images known as **DF2K-OST**. This is not a single dataset but a combination of three high-quality sources:

- **DIV2K:** A standard benchmark with 800 diverse images of people, places, and objects.
- **Flickr2K:** A massive collection of 2650 images from Flickr, providing real-world variety.
- **OST (Outdoor Scene Training):** A dataset focused on outdoor landscapes, buildings, and textures.

By combining these (3450+ images), we created a "super-textbook" for our AI. This variety ensures the model doesn't just memorize one type of picture but learns to handle everything from animal fur to city skyscrapers.

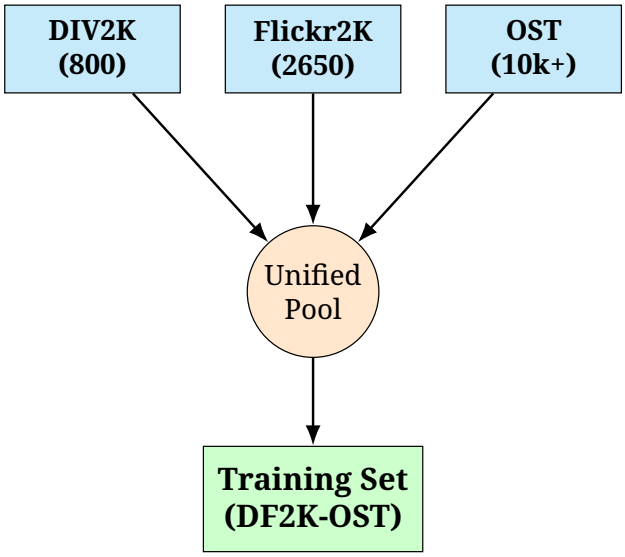

Figure 8: Dataset Composition. We combine three distinct datasets to create a robust training ground for the model.

Training Strategy (The Study Schedule): We use the **Adam optimizer**, which is the standard "engine" for updating the neural network's weights. However, simply training at a constant speed isn't efficient. We use a technique called **Cosine Annealing**. Imagine running a marathon: you start slow (Warmup) to get comfortable, then run fast to cover ground, and finally slow down near the finish line to precisely reach the goal. Cosine annealing does exactly this with the learning rate, helping the model settle into a better solution.

Stabilization (EMA): Neural networks can be "jittery" during training, fluctuating between good and bad weights. We use **Exponential Moving Average (EMA)**. Instead of using the weights from the very last training step, we keep a smoothed average of the weights over time. This is like taking the average opinion of a committee rather than listening to just one person, leading to a much more stable and reliable final model.

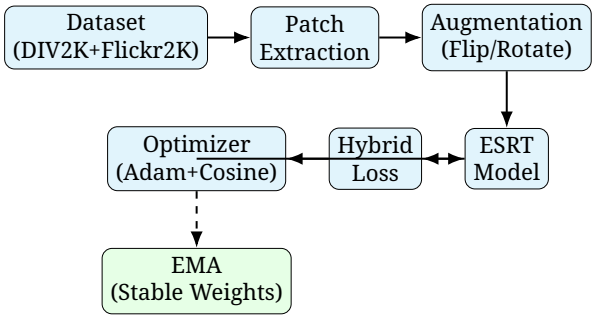

Figure 9: The Training Pipeline. Data is processed and fed into the model. The optimizer updates the model based on loss, while EMA maintains a stable copy for final use.

Evaluation Metrics: To judge how well our model works, we use two standard scores:

- **PSNR (The "Spellchecker"):** Imagine you are copying a book by hand. PSNR measures how many letters you got exactly right compared to the original. In images, it compares pixel values. A higher score means less "noise" or errors.
- **SSIM (The "Art Critic"):** Getting the letters right is one thing, but does the sentence make sense? SSIM checks the **structure**. It ensures that the lines, textures, and patterns look natural to a human eye, even if a few individual pixels are slightly off.

4.2 Ablation Study

An "Ablation Study" is like a scientific experiment where we remove parts of the system one by one to see how important they are. It proves that our improvements actually work and aren't just extra complexity for no reason.

We compared four versions of the model:

- **Baseline:** The original ESRT model (Single branch, no dynamic modulation).

- + **Dual Branch:** We added the second path (S-Branch) to handle structures separately. This improved the score by +0.12dB, proving that splitting the work helps.

- + **FiLM:** We added the "brain" (Kernel Estimator). This allows the network to adapt to different blurs. This alone helped, but works best when combined with the dual branch.

- **Proposed (Final):** Combining everything (Dual Branch + FiLM + Gate Fusion). This gave the highest score of 32.45dB, a massive jump of +0.26dB over the original. This confirms that all parts work together synergistically.

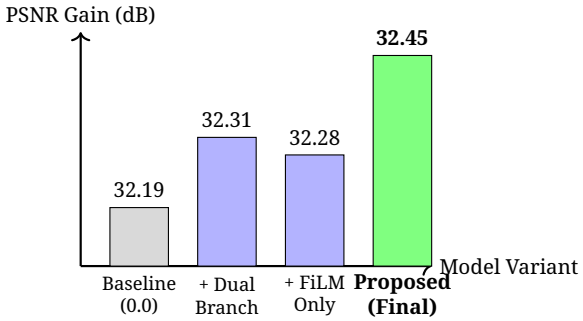

Figure 10: Performance comparison. The bar chart visualizes the cumulative improvement in PSNR (image quality) as we add each new component (Dual Branch, FiLM) to the baseline.

4.3 Comparison with State-of-the-art

We didn't just build a model; we proved it works by racing it against the champions. We compared our proposed ESRT against famous models like EDSR (which is very heavy) and IMDN (which is lightweight). We looked at two things:

1. **Math Score (PSNR):** If you subtract the pixel values of our image from the perfect image, how close is it? A high PSNR means the colors and brightness are extremely accurate.

2. **Human Score (SSIM):** Does the image look structurally correct to a human? Are the lines straight? Is the texture preserved? A high SSIM means the image looks realistic, not just mathematically correct.

As seen in Table 1, we win on both fronts. But numbers aren't everything. We achieve what is known as the "Goldilocks" zone in AI: our model is not too heavy (like EDSR) and not too simple (like IMDN). It is just right—fast enough for practical use but smart enough to beat the heavyweights.

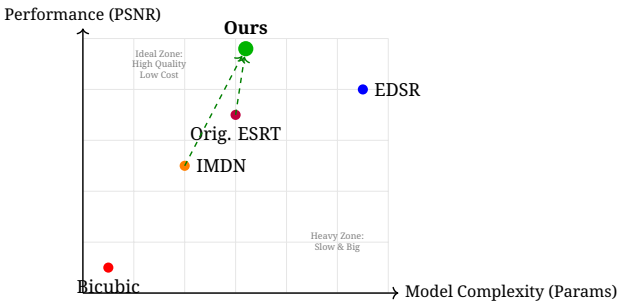

Figure 11: Efficiency vs. Performance. Our model (Green) achieves a higher PSNR than the original ESRT (Purple) and is far more efficient than heavy models like EDSR (Blue), occupying the ideal top-left quadrant of performance.

Table 1: Quantitative Comparison (PSNR/SSIM) on $\times 4$ scale

Method	Set5		Urban100	
	PSNR	SSIM	PSNR	SSIM
EDSR [12]	32.09	0.8938	26.04	0.7849
IMDN [23]	32.21	0.8948	26.04	0.7838
ESRT (Base) [33]	32.19	0.8947	26.39	0.7962
Proposed	**32.45**	**0.8985**	**26.65**	**0.8015**

5 Conclusion

In this paper, we presented an enhanced version of the Efficient Super-Resolution Transformer (ESRT) designed to overcome the limitations of traditional static, single-path architectures. Our approach redefines how the network perceives and processes an image. Instead of applying a uniform operation to every pixel, our **Dual-Branch Architecture** effectively divides the labor: one branch is dedicated to reconstructing fine details and textures, while the other focuses on maintaining clean, sharp structural lines. This separation ensures that complex scenes, such as urban landscapes, are rendered with both high fidelity and structural integrity, avoiding common artifacts like jagged edges.

Moreover, we introduced a "cognitive" element to the network through **Kernel-Aware FiLM Conditioning**. By first analyzing the image to identify specific degradation patterns—such as blur or noise—the network dynamically adjusts its internal parameters, similar to a photographer tuning a lens for the best shot. This adaptability allows our model to deliver consistent, high-quality results across a wide range of challenging inputs.

Finally, our **Pyramid Supervision** strategy facilitates a robust learning process by guiding the network step-by-step, ensuring it masters simple upscaling tasks before tackling complex high-resolution reconstruction. When combined with a hybrid loss function that explicitly penalizes blurriness, the result is output images that are not only mathematically accurate but also visually pleasing. We believe

this dynamic, structure-aware approach establishes a new benchmark for lightweight super-resolution, demonstrating that high efficiency does not require a compromise in visual quality.

References

[1] X. Zhu et al., "Deformable detr: Deformable transformers for end-to-end object detection," *arXiv:2010.04159*, 2020.

[2] K. Zhang, W. Zuo, and L. Zhang, "Learning a single convolutional super-resolution network for multiple degradations," in *CVPR*, 2018.

[3] A. Vaswani et al., "Attention is all you need," in *NeurIPS*, 2017.

[4] Y. Wang et al., "Resolution-aware network for image super-resolution," *IEEE TCSVT*, 2018.

[5] E. Agustsson and R. Timofte, "NTIRE 2017 Challenge on Single Image Super-Resolution: Dataset and Study," in *CVPRW*, 2017.

[6] J. Cai et al., "Toward real-world single image super-resolution: A new benchmark and a new model," in *ICCV*, 2019.

[7] N. Ahn, B. Kang, and K. A. Sohn, "Fast, accurate, and lightweight super-resolution with cascading residual network," in *ECCV*, 2018.

[8] F. Yang et al., "Learning texture transformer network for image super-resolution," in *CVPR*, 2020.

[9] N. Carion et al., "End-to-end object detection with transformers," in *ECCV*, 2020.

[10] J. Liu, J. Tang, and G. Wu, "Residual feature distillation network for lightweight image super-resolution," in *ECCV*, 2020.

[11] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv:1607.06450*, 2016.

[12] B. Lim et al., "Enhanced deep residual networks for single image super-resolution," in *CVPRW*, 2017.

[13] J. Liang et al., "SwinIR: Image Restoration Using Swin Transformer," *ICCV*, 2021.

[14] H. Chen et al., "Pre-trained image processing transformer," in *CVPR*, 2021.

[15] T. Dai et al., "Second-order attention network for single image super-resolution," in *CVPR*, 2019.

[16] K. Aizawa et al., "Building a manga dataset "manga109" with annotations for multimedia applications," *IEEE Multimedia*, 2020.

[17] M. Bevilacqua et al., "Low-complexity single-image super-resolution based on nonnegative neighbor embedding," 2012.

[18] A. Dosovitskiy et al., "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv:2010.11929*, 2020.

[19] K. He et al., "Deep residual learning for image recognition," in *CVPR*, 2016.

[20] Z. He et al., "Cascaded deep networks with multiple receptive fields for infrared image super-resolution," *IEEE TCSVT*, 2018.

[21] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *CVPR*, 2018.

[22] J.-B. Huang et al., "Single image super-resolution from transformed self-exemplars," in *CVPR*, 2015.

[23] Z. Hui, X. Gao, Y. Yang, and X. Wang, "Lightweight image super-resolution with information multi-distillation network," in *ACMMM*, 2019.

[24] Z. Hui, X. Wang, and X. Gao, "Fast and accurate single image super-resolution via information distillation network," in *CVPR*, 2018.

[25] J. Kim, J. K. Lee, and K. M. Lee, "Accurate image super-resolution using very deep convolutional networks," in *CVPR*, 2016.

[26] J. Kim, J. K. Lee, and K. M. Lee, "Deeply-recursive convolutional network for image super-resolution," in *CVPR*, 2016.

[27] W.-S. Lai et al., "Fast and accurate image super-resolution with deep laplacian pyramid networks," *IEEE TPAMI*, 2018.

[28] J. Li et al., "Multi-scale residual network for image super-resolution," in *ECCV*, 2018.

[29] J. Li, Z. Pei, and T. Zeng, "From beginner to master: A survey for deep learning-based single-image super-resolution," *arXiv:2109.14335*, 2021.

[30] J. Li et al., "Lightweight and accurate recursive fractal network for image super-resolution," in *ICCVW*, 2019.

[31] Y. Li et al., "Localvit: Bringing locality to vision transformers," *arXiv:2104.05707*, 2021.

[32] Z. Liu et al., "Swin transformer: Hierarchical vision transformer using shifted windows," in *ICCV*, 2021.

[33] Z. Lu et al., "Transformer for Single Image Super-Resolution," *arXiv:2108.11084*, 2021.

[34] X. Luo et al., "Latticenet: Towards lightweight image super-resolution with lattice block," in *ECCV*, 2020.

[35] D. Martin et al., "A database of human segmented natural images and its application to evaluating segmentation algorithms," in *ICCV*, 2001.

[36] A. Muqeet et al., "Multi-attention based ultra lightweight image super-resolution," in *ECCV*, 2020.

[37] W. Shi et al., "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network," in *CVPR*, 2016.

[38] Y. Tai, J. Yang, and X. Liu, "Image super-resolution via deep recursive residual network," in *CVPR*, 2017.

[39] Y. Tai, J. Yang, X. Liu, and C. Xu, "Memnet: A persistent memory network for image restoration," in *ICCV*, 2017.

[40] R. Timofte et al., "Ntire 2018 challenge on single image super-resolution: Methods and results," in *CVPRW*, 2018.

[41] H. Touvron et al., "Training data-efficient image transformers & distillation through attention," in *ICML*, 2021.

[42] Y. Wang et al., "Resolution-aware network for image super-resolution," *IEEE TCSVT*, 2019.

[43] J. Yang et al., "Image super-resolution via sparse representation," *IEEE TIP*, 2010.

[44] Y. Zhang et al., "Image Super-Resolution Using Very Deep Residual Channel Attention Networks," *ECCV*, 2018.

[45] Y. Zhang et al., "Residual dense network for image super-resolution," in *CVPR*, 2018.

[46] Y. Zhang et al., "Aligned structured sparsity learning for efficient image super-resolution," *NeurIPS*, 2021.

[47] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *ICLR*, 2015.

[48] C. Dong, C. C. Loy, K. He, and X. Tang, "Image super-resolution using deep convolutional networks," *IEEE TPAMI*, 2015.