Super-Resolution Techniques for Fluid Dynamics Data

***Abstract*—This paper explores Super-Resolution (SR) techniques for enhancing data resolution for fluid dynamics application. We focus on the Navier-Stokes Kraichnan turbulence problem and apply deep learning approaches to upscale image resolution while preserving physical accuracy. Our exploratory analysis includes Proper Orthogonal Decomposition (POD), turbulent intensity visualization, and energy spectrum analysis. We implement SwinIR- which is a transformer-based method as our baseline and use operator-learning based architectures like Fourier Neural Operator (FNO)and Deep Operator Network (DeepONet) for super-resolution tasks.**

***Index Terms*—Super-Resolution, Navier-Stokes Kraichnan Turbulence, Swin Transformer, FNO, DeepONet**

I. Introduction

A. Overview

Super-Resolution (SR) techniques have emerged as powerful tools for enhancing data resolution, improving the overall quality and fidelity of data representation, and retrieving fine-scale structures. These techniques find application in diverse fields such as image restoration and enhancement [1]. It is computationally costly to obtain the high resolution image by solving the underlying PDEs. Recent advances in machine learning architecture has allowed the researchers to successfully upscale the image resolution in domains like biology, facial recognition, medicine, etc. However, the challenges in the physical domain lie in the fact that upscaling images need to match the physical phenomenon that results in the images.

B. Objective

We aim to explore the problem in super-resolution by upscaling the low resolution images that can be obtained with low computational cost, guided by the physical laws. The dataset we are working with is vortical flow data based on Navier-Stokes Kraichnan turbulence. And the main objective with this project is to help researchers and scientists fed up with the low resolution of images by developing a framework that allows us to upscale these images to a higher resolution.

II. Technical Approach

A. Baseline Model

SwinIR [2] is actually a hybrid model with two CNN modules (shallow feature extraction and high-quality image reconstruction) at the two ends, and specially a Swin Transformer- based module (deep feature extraction) as the crucial component of the method.

The Swin Transformer [3] is a hierarchical vision transformer architecture that uses a ”shifted window” mechanism to efficiently process images, particularly high-resolution ones, by computing self-attention only within local windows and allowing for cross-window connections, thus reducing computational complexity.

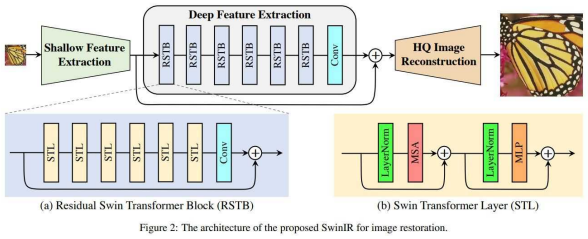


Figure 2: The architecture of the proposed SwinIR for image restoration.

Fig. 1. SwinIR architecture.

One of the major reasons for selecting swinIR unlike traditional CNNs is that SwinIR leverages Swin Transformer, which efficiently models long-range dependencies while maintaining computational efficiency.

The workflow consists of CNN in the first layer (a 3×3 conv) for initial feature extraction. Then, features are processed through multiple RSTB blocks. Each RSTB contains:

- WindowAttention (shifted or non-shifted). : self-attention with 4 stages each with a depth of 6.
- MLP for feature transformation.
- Residual connections (convolutional)

We use a patch size of 64 which means the image is divided into 64x64 size of non-overlapping tokens. We also try various windows size of 4 and 8 and observe the differences.

We use **pixel shuffle** [4] as an upsampler to increase resolution (alternative Nearest Neighbor + CNN).

B. Neural Operator Models

Operator learning is a type of machine learning that learns mappings between functions, not just input-output pairs. This is especially useful in physics informed problems like fluid

dynamics, where inputs and outputs are fields or solutions over space/time.

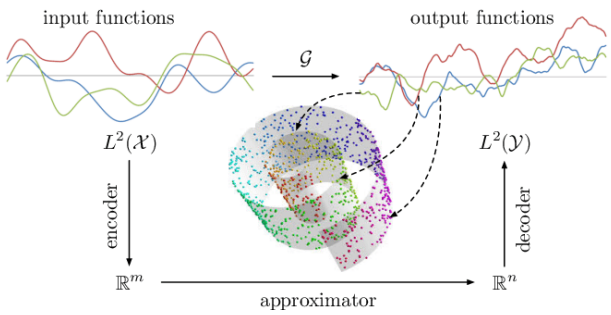


Fig. 2. Operator learning hypothesis. Operator G learns mapping between two function spaces.

1) DeepOnet: DeepONet [5] is a neural network designed to learn **operators**—meaning it does not just learn a simple input-output function, but rather a *function of functions*.

In fluid dynamics, instead of predicting a single velocity value from one input, DeepONet aims to predict an entire flow field given some initial or boundary condition. It is particularly suited for tasks governed by partial differential equations (PDEs), where the solution is itself a function over space and/or time.

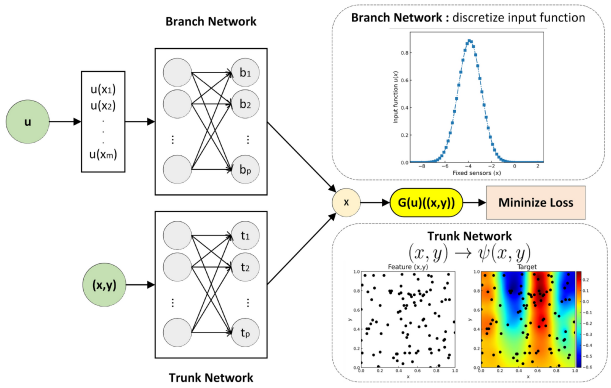


Fig. 3. DeepOnet architecture. In our implementation, the Branch net is a CNN based architecture whereas Trunk net is a MLP that is embedded with Fourier features.

DeepONet has two main components:

- **Branch Net**: Takes the input function (such as initial velocity field values or low-resolution data) and encodes the overall behavior or context into a latent feature vector.
- **Trunk Net**: Takes the spatial coordinates (e.g., (x, y)) at which the output is to be evaluated, and encodes information about the query location.

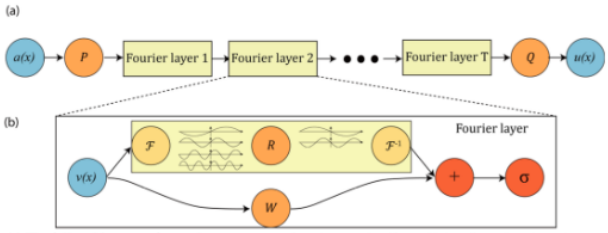
The final output is computed by taking the dot product of the outputs of the Branch and Trunk networks:

$$v(x) = \sum_{i=1}^{d} b_i \cdot t_i(x)$$

where b_i are features from the Branch Net and $t_i(x)$ are features from the Trunk Net evaluated at location x.

We also supplement our MLP in trunk net by Fourier Embeddings before passing through MLP. Turbulent flow problems are represented by PDEs, so the true solution requires information on large and small vortices. Large vortices contain bulk information,, whereas small vortices capture localizedd details. Therefore, Fourier features are used to capture higher frequencies as the MLP now easily detects local structures, because the input already consists of higher frequencies (small vortices).

2) FNO: FNO [6] is a neural network that learns mappings between functions (i.e., it learns operators), just like DeepONet—but with a key twist: It does most of its computation in the Fourier (frequency) domain, which is ideal for fluid flows that involve waves, turbulence, and multiscale structures.



(a) The full architecture of neural operator: start from input a. 1. Lift to a higher dimension channel space by a neural network P. 2. Apply four layers of integral operators and activation functions. 3. Project back to the target dimension by a neural network Q. Output u. **(b) Fourier layers**: Start from input v. On top: apply the Fourier transform $\mathcal{F}$; a linear transform R on the lower Fourier modes and filters out the higher modes; then apply the inverse Fourier transform $\mathcal{F}^{-1}$. On the bottom: apply a local linear transform W.

Figure 2: **top:** The architecture of the neural operators; **bottom:** Fourier layer.

Fig. 4. FNO architecture.

The network takes a discretized input function, such as a coarse-resolution flow field. The network then applies the Fast Fourier Transform (FFT) to the input, moving it to the frequency domain. This allows the model to analyze the global structure of the input (e.g., large and small eddies, frequency components). The model then learns complex-valued weights to modulate these Fourier coefficients. It keeps only a few low-frequency modes (truncated) which make computation fast while focusing on the most important features. After that, the modified frequencies are transformed back to the spatial domain using an inverse FFT (IFFT). After returning to the spatial domain, the model applies non-linear activations (like ReLU) and stacks multiple layers.

III. Dataset

A. Dataset Description

We have generated a dataset by solving the Navier-Stokes Kraichnan turbulence problem numerically with accuracy of 4th order Runge-Kutta. The generated dataset is 2048*2048 pixels for each snapshots with a time-stepping of $1e-4$ secs for 1000 timesteps.

Fluids are characterized by their velocity field where each point in the fluid has a corresponding velocity vector. For a 2D flow field, the velocity vector at any point can be broken down into its components as follows:

- u represents the velocity component in the x-direction.
- v represents the velocity component in the y-direction.
- ω represents the vorticity component in the z-direction.

Therefore, the shape of our dataset is (1000, 3, 2048, 2048).

For our dataset pair, we need a low resolution paired with this high resolution dataset samples. For this we downsample using Nearest Neighbor or Bicubic Interpolation [7] with a scale of 4 and 8. Therefore if a patch of 16x16 is used as a low resolution we will get a corresponding high resolution as 64×64 (scale = 4) or 128×128 (scale = 8).

B. Data Analysis

We perform Proper Orthogonal Decomposition (POD) [8] on the flow field (u, v, ω) to extract dominant coherent structures (modes) and quantify their energy contributions.

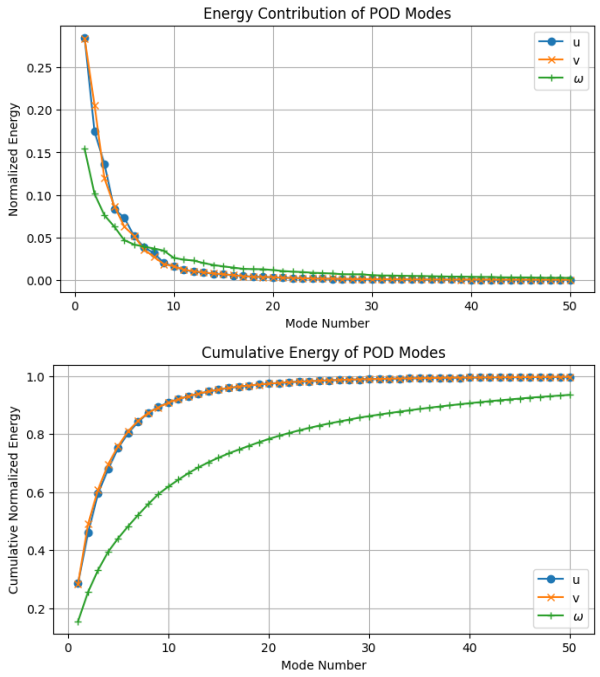


Fig. 5. Energy contribution of POD modes.

We observe from figure 5 that it has rapid drop-off after the first few modes. Most of the energy is concentrated in the first ˜10 modes. And we also discover that u dominates in this shear flows. Further, the plot of how much energy is captured by including N nodes. From the plot we find that, first 10 modes might capture greater than 90% energy, allowing reduced-order modeling.

We notice component differences:

- u (x-velocity) has higher energy than v (y-velocity).
- ω (vorticity) highlights rotational structures.

All in all it suggests that in our dataset, a low-dimensional model could effectively capture the bulk flow dynamics. And we try different flavor of reconstruction using top 50 nodes:

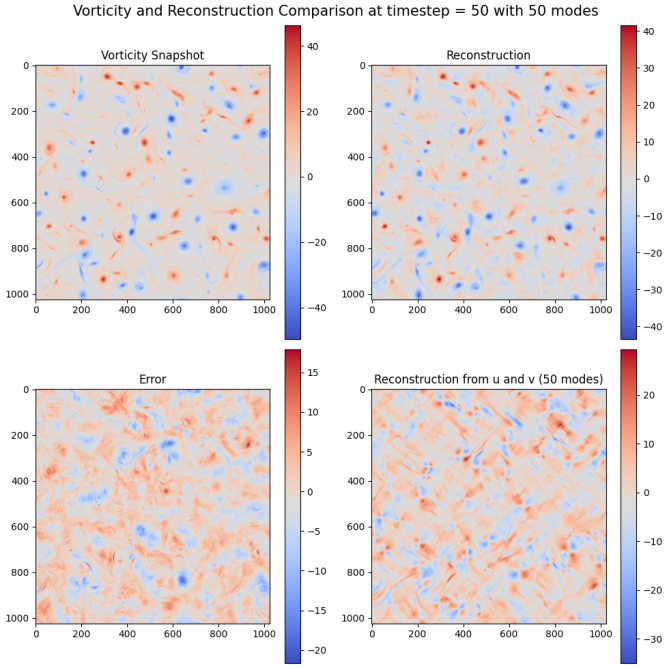


Fig. 6. Reconstruction with timesteps = 50.

IV. Results and Discussion

A. Experiment

For our super-resolution task, the high-resolution images were downscaled by a factor of 4 and 8 before training. The model was trained on a data set comprising 1000 samples, each with a shape of $(1000, 3, 1024, 1024)$, representing 3 channel fluid flow data. The images were divided into patches of size 64×64 or size 128×128 or 256×256, and training was conducted over 50 epochs or 100 epochs. The loss function used for optimization was one of the standard regression losses. L1, L2, MSE loss, and some physics Loss like Divergence and Vorticity Loss were implemented depending on the experimental setup.

During the evaluation, the same downscaling factor of 4 or 8 was applied to the high-resolution images. Bicubic Interpolation or Nearest Neighbor downsampler was used. The model was tested on a separate set of 100 unseen samples. To assess model performance, we used multiple metrics including SSIM (Structural Similarity Index Measure) [9], PSNR (Peak Signal-to-Noise Ratio), and L1 loss, ensuring a comprehensive evaluation of both perceptual quality and pixel-wise accuracy.

B. SwinIR

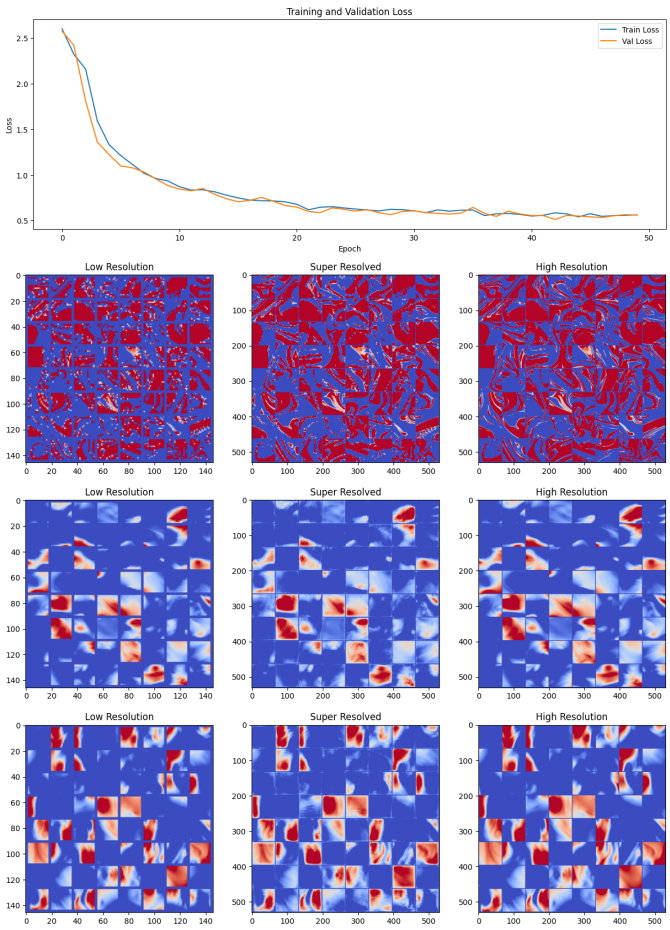


Fig. 7. SwinIR training and validation loss (top). Visualization of 64×64 (upscale factor = 4) patch of data trained in SwinIR (bottom)

C. DeepOnet

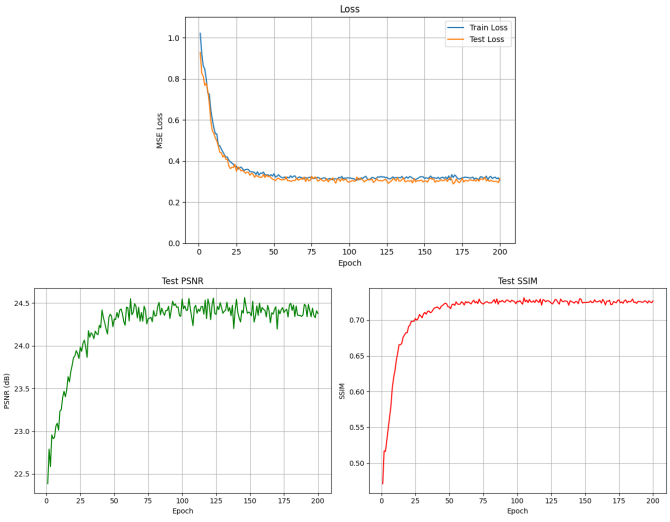


Fig. 8. DeepONet Training and Validation Loss curve (top) and SSIM and PSNR curve for Testing sample (bottom).

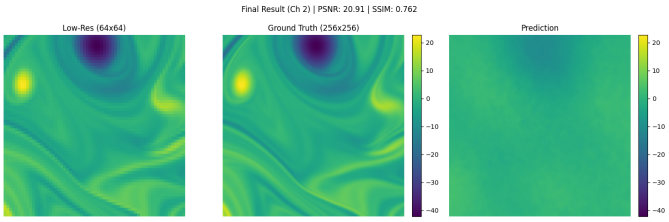


Fig. 9. Visualized Results for a random sample with 256x256 patch (upscale factor = 4).

D. FNO

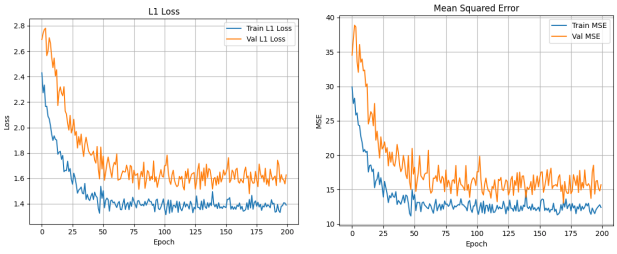


Fig. 10. FNO training and validation: L1, MSE Loss

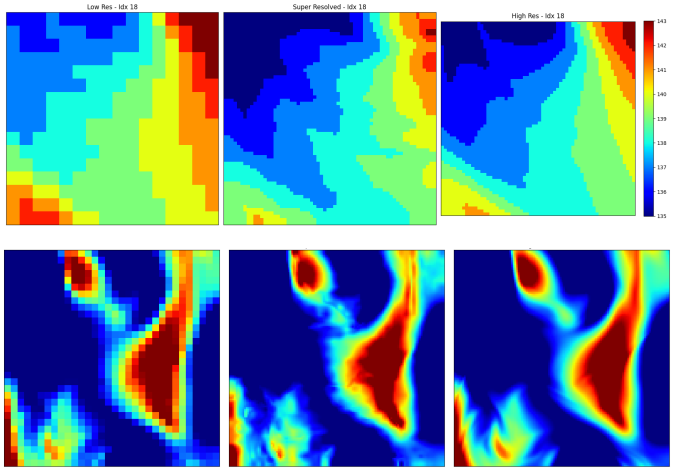


Fig. 11. Visualization of 64x64, 128x128 patch of data from different samples trained in FNO (upscale factor = 4).

E. Benchmarks

TABLE I
PERFORMANCE COMPARISON OF DIFFERENT MODELS

Metric	SwinIR	DeepONet	FNO
PSNR	36.2829	20.91	38.53
SSIM	0.9214	0.762	0.9576
Test loss	0.5471	0.322	0.501

V. Conclusions

A. Findings

Some of our findings that we gathered from the project are as follows:

- FNO showed the most potential during our testing which suggests it is suitable for the dataset of our choice with smooth gradient flows.
- Although FNO has better results, transformer-based architectures can still excel even in physics-rich domains, possibly due to their stronger capacity to model fine-grained textures and long-range dependencies.
- Throwing problem at the network doesn't work, especially for operator learning networks. So we need to properly fine tune those models to properly utilize them
- The general loss functions aren't the best measure for the training strategy in these physics based tasks.

B. Future Works

Some of the other tasks that need more attentions for future work are listed below:

- We have to try hybrid architecture to address the small scaled turbulence. We will try other specific hybrid model architecture like DeepOnet + FNO.
- We need physics-informed downsampler like the KS (Kolmogorov-Smirnov) filter to properly downsample for low resolution.
- The next crucial step will be to utilize physics-informed Loss Functions as L1 and L2 might not be able to capture fluid specific constraints.

VI. Work Division

Datasets - Anil Sapkota

Baseline Model - Kapildev Neupane, Aagat Pokhrel

DeepOnet Architecture - Anil Sapkota, Aagat Pokhrel

FNO - Chandra P. Raskoti, Kapildev Neupane

Report Writing - All

References

[1] Sung Cheol Park, Min Kyu Park, and Moon Gi Kang. "Super-resolution image reconstruction: a technical overview". In: *IEEE Signal Processing Magazine* 20.3 (2003), pp. 21–36. DOI: 10.1109/MSP.2003.1203207.

[2] Jingyun Liang et al. *SwinIR: Image Restoration Using Swin Transformer*. 2021. arXiv: 2108.10257 [eess.IV]. URL: https://arxiv.org/abs/2108.10257.

[3] Ze Liu et al. "Swin Transformer: Hierarchical Vision Transformer using Shifted Windows". In: *CoRR* abs/2103.14030 (2021). arXiv: 2103.14030. URL: https://arxiv.org/abs/2103.14030.

[4] Wenzhe Shi et al. "Real-Time Single Image and Video Super-Resolution Using an Efficient Sub-Pixel Convolutional Neural Network". In: *CoRR* abs/1609.05158 (2016). arXiv: 1609.05158. URL: http://arxiv.org/abs/1609.05158.

[5] Siyuan Yang. *Super Resolution Based on Deep Operator Networks*. 2024. arXiv: 2410.20706 [eess.IV]. URL: https://arxiv.org/abs/2410.20706.

[6] Zongyi Li et al. "Fourier Neural Operator for Parametric Partial Differential Equations". In: *CoRR* abs/2010.08895 (2020). arXiv: 2010.08895. URL: https://arxiv.org/abs/2010.08895.

[7] Donya Khaledyan et al. *Low-Cost Implementation of Bilinear and Bicubic Image Interpolation for Real-Time Image Super-Resolution*. 2020. arXiv: 2009.09622 [eess.IV]. URL: https://arxiv.org/abs/2009.09622.

[8] Anindya Chatterjee. "An introduction to the proper orthogonal decomposition". In: *Current Science* 78.7 (2000), pp. 808–817. ISSN: 00113891. URL: http://www.jstor.org/stable/24103957 (visited on 04/01/2025).

[9] Zhou Wang et al. "Image quality assessment: from error visibility to structural similarity". In: *IEEE Transactions on Image Processing* 13.4 (2004), pp. 600–612. DOI: 10.1109/TIP.2003.819861.