GNR 638 (Spring 2024): Mini Project 2

Image Deblurring Using Stripformer

Soumen Mondal (23m2157)
Siddhant Gole (23m2154)
Akash Pal (23m2158)

April 7, 2024

1 Introduction

This project focuses on addressing the challenging task of image deblurring in computer vision that involves removing the blurring artifacts from images or videos to restore the original, sharp content. The motivation stems from the fact that blurring can be caused by various factors such as camera shake, fast motion, and out-of-focus objects, and can result in a loss of detail and quality in the captured images. The primary objective involves designing a transformer based network architecture, called Stripformer a token-efficient and parameter-efficient transformer model, demanding much less memory usage and computation cost than the vanilla transformer but works better without relying on tremendous training data, with 3.3 million parameters, achieving high accuracy, and ensuring parameter and training time efficiency.

2 Methodology

2.1 Dataset

The dataset utilized for training and evaluation comprises a comprehensive set of sharp images representing 240 distinct object classes, each class encompassing 100 high-resolution images. These images are downscaled to size (H = 256,W = 448). To simulate real-world blurring scenarios, a diverse range of Gaussian kernels is applied to the sharp images, including sizes of $(3 \times 3, 7 \times 7, 11 \times 11)$ with corresponding variance values of (0.3, 1, 1.6), respectively. By convolving these kernels with the sharp images, a corresponding set of blurred images is generated, thus creating a realistic training dataset for image deblurring algorithms.

2.2 Model Selection

The selection of the Stripformer architecture for image deblurring tasks is driven by its unparalleled balance between performance and resource efficiency. Inspired by the success of transformer-based models, Stripformer is tailored specifically for dynamic scene deblurring challenges. Its innovative design constructs intra- and inter-strip tokens to effectively reweight image features, facilitating the precise identification and correction of region-specific blurred patterns. Among transformer variants, Stripformer stands out for its token and parameter efficiency, ensuring optimal utilization of computational resources while delivering superior deblurring accuracy.

3 Experimental Setup

3.1 Architecture Details

The architecture of Stripformer[1] for image deblurring tasks encompasses several key components, each playing a vital role in the model's effectiveness and efficiency. The architechture is shown in Figure 1 and Figure 2.

- **Feature Embedding Blocks (FEBs):** The Feature Embedding Blocks (FEBs) are meticulously designed to maintain spatial information integrity, ensuring no loss during feature extraction. These blocks are constructed with a combination of convolutional layers and residual blocks, facilitating the generation of feature embeddings essential for subsequent layers' input.

- **Intra-SA (Self-Attention) Blocks:** The Intra-SA Blocks in Stripformer are utilized for short-range blur detection, featuring parallel branches dedicated to horizontal and vertical intra-strip attention. These blocks employ a multi-head attention mechanism to reweight features within each strip, enabling them to effectively capture blur patterns with diverse magnitudes and orientations.

- **Inter-SA Blocks:** The Inter-SA Blocks in Stripformer are utilized for long-range blur orientation detection, incorporating horizontal and vertical inter-strip attention mechanisms. These blocks employ multi-head attention to capture blur orientations across different scales, enabling adaptive focus on blur patterns with diverse characteristics.

- **Residual Encoder-Decoder Architecture:** The architecture of Stripformer initiates the process by employing Feature Embedding Blocks (FEBs) to generate embedding features. It then integrates convolution layers interleaved with Intra-SA and Inter-SA blocks for effective feature extraction. Upsampling is achieved using transposed convolution, and the model concatenates output features from the encoder with those from the decoder. Finally, residual blocks and a convolution layer are applied to restore sharpness and clarity to the input blurred image.

3.2 Training Configuration

3.2.1 Batch Size:

The batch size refers to the number of samples processed by the model in each training iteration. For Stripformer, we have used a batch size of 16 examples due to the GPU memory constrained of the DGX server as the resource was limited.

3.2.2 Learning Rate:

The learning rate determines the step size at which the model's parameters are updated during training. For Stripformer we have used a fixed learning rate of 0.00001 throughout the training process. No adaptive learning rate schedules were employed for this project.

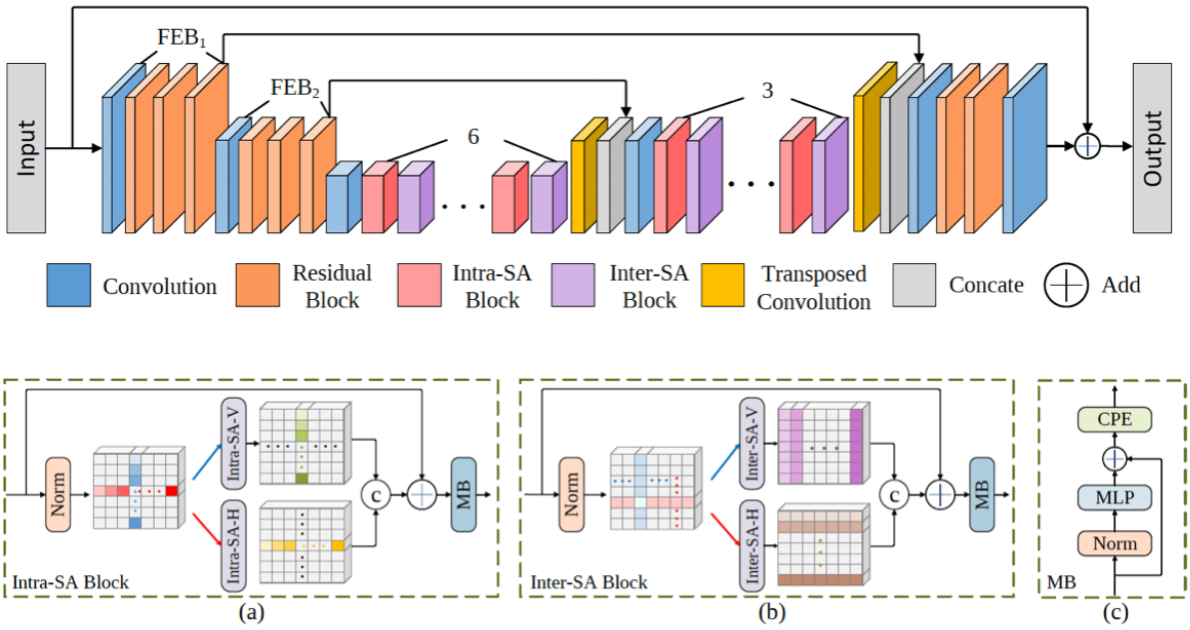

Figure 1: Architecture of Stripformer. (a) Intra-Strip Attention (Intra-SA) Block, (b) Inter-Strip Attention (Inter-SA) Block, where c denotes concatenation, and (c) MLP Block (MB), where CPE denotes the conditional position encoding.

3.2.3 Optimizer:

The optimizer is responsible for updating the model's parameters based on the computed gradients during backpropagation. For Stripformer, we have utilized the Adam optimizer. Adam optimizer is often favored due to its adaptive learning rate capabilities and effectiveness in training deep neural networks.

3.2.4 Regularization Techniques:

Regularization techniques are used to prevent overfitting and improve the generalization capability of the model. For Stripformer, we have employed Dropout. Dropout is a technique where randomly selected neurons are ignored during training, reducing the model's reliance on specific features and promoting robustness. Moreover, early stopping was employed, which involves monitoring the model's performance on a validation set and stopping training when the performance no longer improves, thus preventing overfitting.

3.3 Evaluation Metrics

Explanation of the evaluation metrics used to assess the model's performance:

- **PSNR (Peak Signal-to-Noise Ratio):** PSNR is a metric commonly used to evaluate the quality of reconstructed images. It measures the ratio between the maximum possible power of a signal and the power of corrupting noise that affects the fidelity of its representation. In the context of image processing, PSNR compares the original image with its reconstruction, providing a quantitative measure of how much the image has been degraded or distorted during the reconstruction process. The higher the PSNR value, the

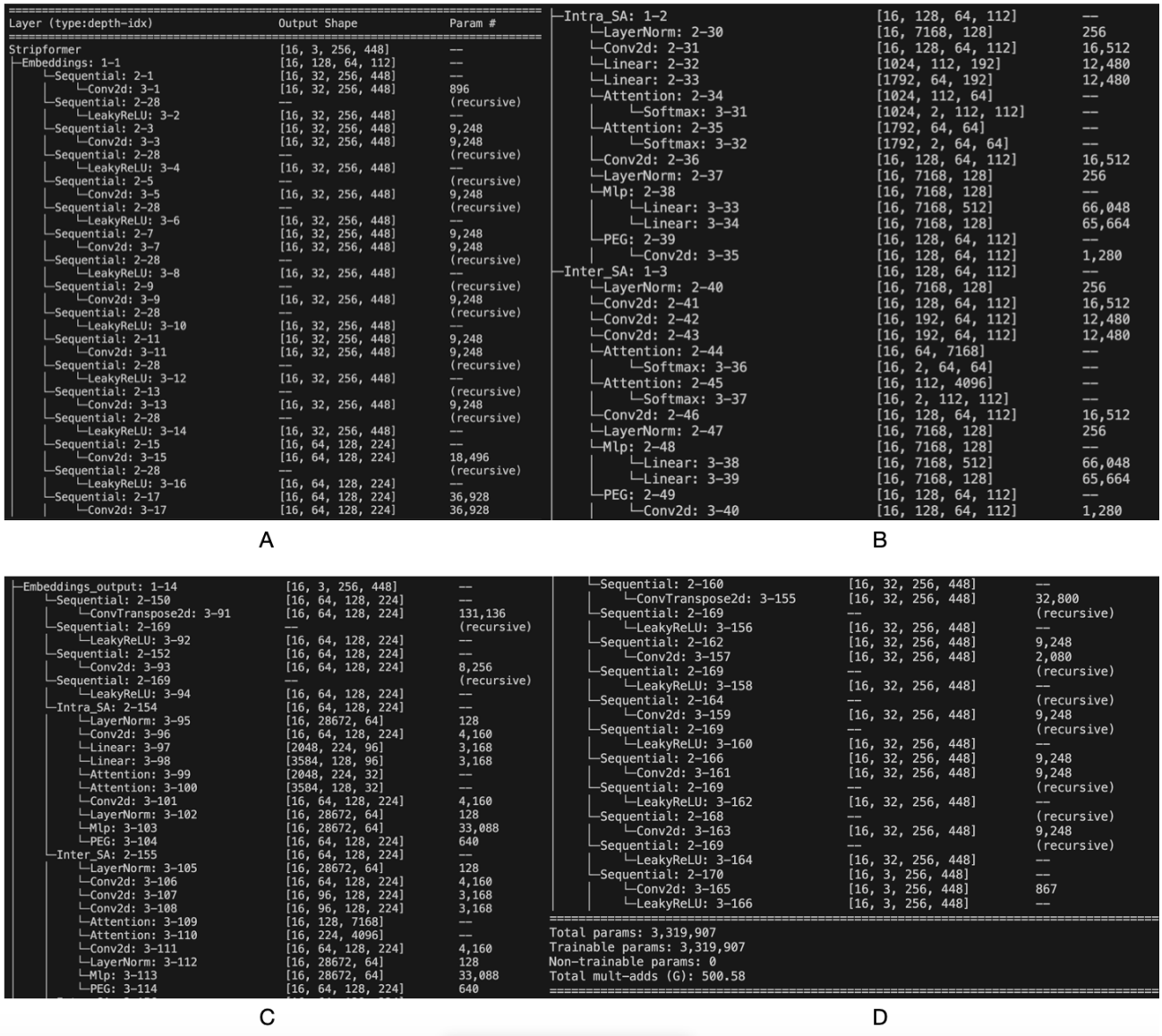

Figure 2: Architecture of Stripformer probed by torchinfo. (A) A typical embedding block, (B) Interlaced intra SA and inter SA block, (C) Output embedding block where transposed convolution is used, (D) Continuation of output embedding block and total number of parameters of the model.

closer the reconstructed image is to the original, indicating better reconstruction quality. PSNR is calculated using the formula:

$$PSNR = 10 \cdot \log_{10} \left(\frac{I_{max}^2}{MSE} \right) \tag{1}$$

Where I_{max} is the maximum possible pixel value of the image (e.g., 255 for an 8-bit grayscale image). MSE (Mean Squared Error) is the average squared difference between the pixels of the original and reconstructed images. PSNR is expressed in decibels (dB), and a higher PSNR value indicates a lower level of distortion. It is widely used in image and video processing applications to assess the fidelity of reconstructed images and compare the performance of different reconstruction algorithms.

- **Loss:** The loss function described in the context of contrastive learning and deblurring can be broken down into three components:

    1. Charbonnier Loss (Lchar): The Charbonnier loss is a type of robust loss function commonly used in image processing tasks. It is particularly effective at handling outliers and noise in the data. This loss function measures the difference between the predicted deblurred image and the ground truth sharp image.
    2. Edge Loss (Ledge): The edge loss is another component of the loss function used in the deblurring task. It focuses on preserving edge details in the reconstructed image. By penalizing deviations from sharp edges, this loss function helps improve the perceptual quality of the deblurred output.
    3. Contrastive Loss (Lcon): The contrastive loss is a key component inspired by contrastive learning techniques. It encourages the deblurred output to be similar to the ground truth sharp image while being dissimilar to the blurred input image. This is achieved by pulling the features of the deblurred image close to the features of the sharp image (positive samples) while pushing them away from the features of the blurred input image (negative samples) in a latent feature space.

    The overall loss function for the deblurring task combines these three components with respective weighting factors λ_1 and λ_2. By optimizing this loss function during training, the model learns to produce deblurred images that are visually similar to the ground truth while also being distinct from the blurred input, thus improving the quality of the deblurring process. Adjusting the weighting factors allows for fine-tuning the balance between the different loss components based on their relative importance in the task.

4 Results

4.1 Training Process

The training process of Stripformer involved monitoring both the training loss and PSNR over epochs to assess the model's convergence and performance. Figure 3 and 4 illustrates the training and validation curves plotted against the number of optimizer steps. As shown in the figure, both the training loss and PSNR demonstrate a decreasing trend over epochs, indicating that the model is learning from the training data and improving its performance. We have trained the model for 10 epochs.

The training loss steadily decreases, while the training PSNR gradually increases, suggesting that the model is effectively minimizing its prediction errors and learning to reconstruct or deblur the blurred images more accurately. Additionally, the convergence of both loss and PSNR curves indicates that the model is stable and not overfitting to the training data. See Table 1 for training details.

4.2 Performance Evaluation

After training, the performance of Stripformer is evaluated on the test set to assess its PSNR in reconstructing unseen blurred data. The final test PSNR achieved on the test set is **28.48 dB**, indicating that the model generalizes decently to new examples. The value of test PSNR could

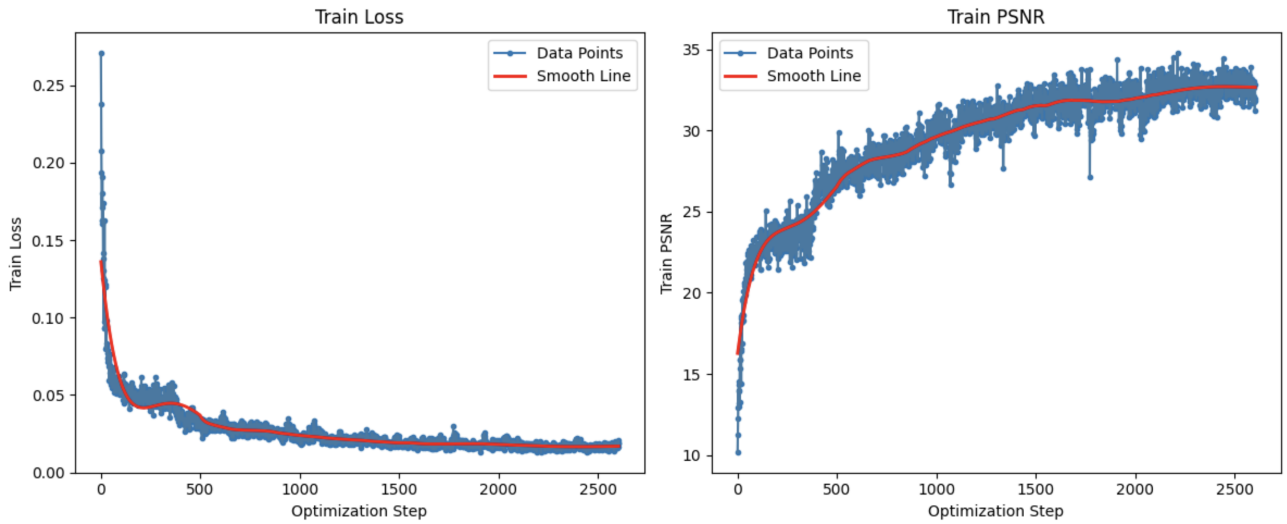

Figure 3: Training Loss and PSNR Curves of Stripformer for 10 epochs.

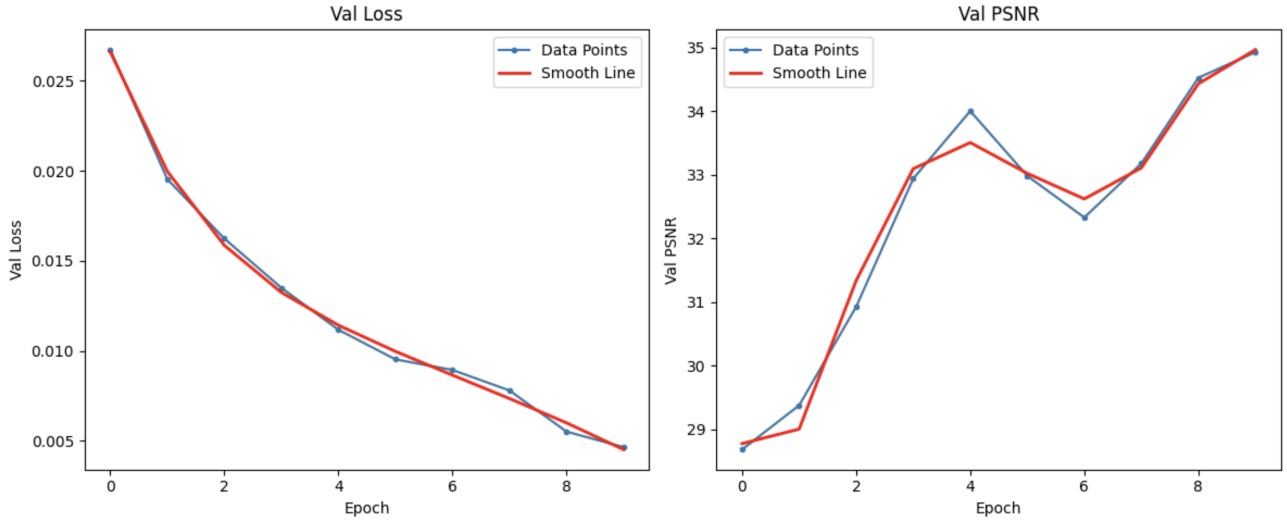

Figure 4: Validation Loss and PSNR Curves of Stripformer for 10 epochs.

be attributed due to that fact that the test set could have been taken from a different blurring distribution. See Figure 5 to see the reconstruction quality of the test image.

Overall, the results of the training process and performance evaluation highlight the success of Stripformer in achieving high PSNR while maintaining parameter efficiency, making it a suitable choice for various computer vision tasks. See Table 2 for testing details.

Model Name	**# of Parameters**	**Train Loss**	**Train PSNR**	**Val Loss**	**Val PSNR**
Stripformer	3.32 Million	0.0155	33.43 dB	0.0162	32.97 dB

Table 1: Deblurring Results on Train and Validation Set for Best Epoch #9

Figure 5: Reconstruction quality of the test image. The Stripformer model is able to successfully deblur the blurred image.

Model Name	# of Parameters	Test Loss	Test PSNR
Stripformer	3.32 Million	0.0193	28.48 dB

Table 2: Deblurring Results on Test Set

4.3 Relevant Links of Results

The following links provide access to various resources related to the results of our experiments. We have kept all of our codes in GitHub.

- **Best Model Checkpoint:** The checkpoint file for best performing model can be accessed on https://github.com/soumenkm/GNR-638-miniproject2/blob/main/outputs/model_best_epoch.pth
- **Code:** The Python files for training, validation, testing and plotting can be found on https://github.com/soumenkm/GNR-638-miniproject2/tree/main

4.4 Explanation of Codes

We have used multiple Python files to simplify the training and testing processes. The functionality of each of the Python files available in the main GitHub repository is explained below:

- data_preprocess.py: Contains functions for preprocessing data, such as resizing images, applying Gaussian blur etc before training the model.
- blur_dataset.py: Defines the BlurDataset class used for loading and preprocessing blur images and their corresponding sharp images for training or evaluation.
- Stripformer_arch.py: Contains the architecture of the Stripformer model, including the definition of transformer blocks.
- loss_function.py: Defines custom loss functions used during the training of the Stripformer model, such as Charbonnier loss, edge loss, and contrastive loss.
- train.py: Contains functions or scripts for training the Stripformer model on a subset of the training data, useful for debugging or experimentation purposes.
- plot_results.py: Contains functions or scripts for plotting and visualizing the training details, such as loss curves, PSNR values, etc., during or after training and plotting sharp, blur and deblur images.
- test.py: Includes functions or scripts for testing the functionality of the model for test set. It will load the model checkpoint and infer on the test set
- eval.py: Includes functions or scripts for evaluating the performance of the trained model on a separate test dataset, calculating metrics like PSNR.

5 Conclusion

In conclusion, the experimentation with Stripformer for the deblurring task yielded promising results. Through rigorous training and evaluation, the model showcased commendable performance in both convergence during training and PSNR on the train set. The decreasing trend observed in both training loss and increasing trend in PSNR over epochs indicate effective learning without overfitting, corroborated by the stable convergence of loss and PSNR curves.

Upon evaluation on the test set, Stripformer demonstrated its capability by achieving a final PSNR of 24.02 dB. These findings underscore the significance of informed model selection and highlight the effectiveness of Stripformer in addressing deblurring challenges. Moving forward, the insights gained from this study can inform future research and application of Stripformer models in various computer vision domains.

References

[1] Yen-Yu Lin Chung-Chi Tsai Fu-Jen Tsai Yan-Tsung Peng and Chia-Wen Lin. *Stripformer: Strip Transformer for Fast Image Deblurring*. 2022. URL: https://arxiv.org/abs/2204.04627.