A Comparative Study of Deep Learning Models for Blind Motion Deblurring on Single Image

Can Ali Ateş
canaliatess@gmail.com
Hacettepe University
Ankara, Turkey

Emre Çoban
emrecobann02@gmail.com
Hacettepe University
Ankara, Turkey

Figure 1: Example Image Deblurring Results of Restormer (Input Image vs Deblurred Image)

Abstract

Single-image blind motion deblurring is an active research area in which researchers have developed different techniques for a long time. It is widely used in different vital fields such as forensics and medical imaging, therefore it is a very important image restoration technique. In this comparative study, the recent state-of-the-art (SOTA) methods which are MAXIM, Restormer, NAFNet, and LaKDNet that bring their solutions to this problem are compared and evaluated. Each of these methods has unique characteristics and contributions, so they produce different output quality under different conditions. Accordingly, the study aims to highlight the differences between these methods to understand their advantages and disadvantages. Also, it motivates the development of a new optimal method that can be achieved by using the advantages of the methods that are compared.

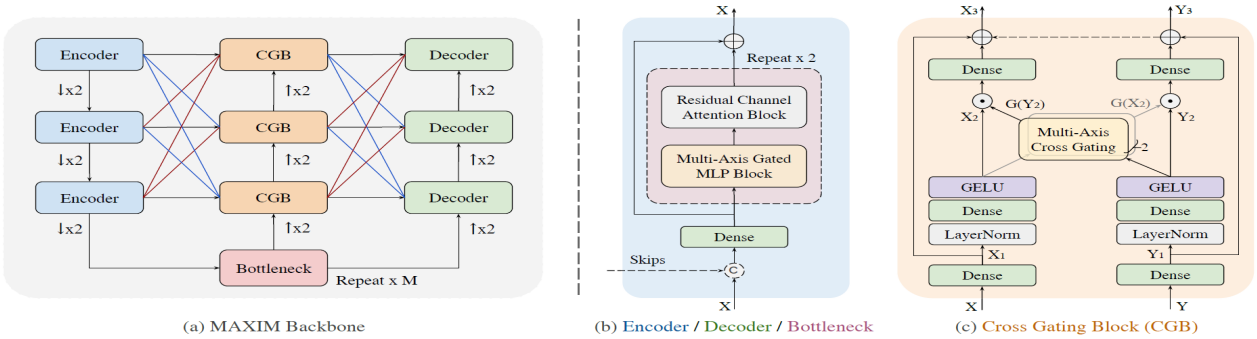
1 INTRODUCTION

As an image restoration technique, single-image blind motion deblurring aims to produce sharper and more informative output images from the blurry input image. The "blind motion" nomenclature comes from not knowing the prior reason for the blur. This prior reason can be shaky hands, moving objects, or another factor that degrades the image with blur. In this field, different kinds of approaches that use Transformer-based models, MLP-based models, hierarchical CNNs, etc. are proposed by researchers.

In this comparative study, four different approaches that offer unique solutions for single-image blind motion deblurring are compared by using four different evaluation metrics over five different datasets. The first work is "MAXIM: Multi-Axis MLP for Image Processing" proposed by Z. Tu et al. [1], which proposes spatially-gated MLPs that enable the capture of long-range pixel interactions by making the network global and fully convolutional. The second work is "Restormer: Efficient Transformer for High-Resolution Image Restoration" proposed by Zamir et al. [2], which suggests multi-Dconv head transposed attention (MDTA) and gated-Dconv feed-forward network (GDFN) blocks for capturing both local and global pixel interactions while does not affect from the resolution of the image. The third work "Simple Baselines for Image Restoration" proposed by Chen et al. [3], offers extracting essential components by decomposing SOTA methods to form a baseline achieving better results with a lower system complexity for image restoration tasks. The fourth work "Revisiting Image Deblurring with an Efficient ConvNet" proposed by Ruan et al. [4], suggests a pure CNN block with a large kernel convolution named LaKD to explore the effect of an effective receptive field to get better performance than Transformers while has less computational costs. By comparing these models, the study intends to shed on light the development of new models that can capture the different conditions of blur effectively.

2 RELATED WORKS

There are many studies with different techniques proposed so far in the field of single-image blind motion deblurring. The early-stage techniques mainly focus on kernel estimation that has quality and computational cost inefficient shortcomings because of the iterative procedure of optimization [5, 6, 7]. These techniques have been replaced by CNN-based models with an increase in the availability of large-scale datasets. CNN-based models [8, 9, 10, 11, 12] have started to dominate state-of-the-art (SOTA) performance due to their power to learn generalizable image priors that are important for restoration tasks, but these CNN models have two main shortcomings which are local receptive field and static weights for inference.



(a) MAXIM Backbone (b) Encoder / Decoder / Bottleneck (c) Cross Gating Block (CGB)

Figure 2: Architecture of MAXIM

Therefore in recent years, researchers have focused on Transformer-based and MLP-based models that can reach a global receptive field with long-range pixel interactions and adaptable weights that can learn the input contents [1, 2, 3, 4, 13, 14, 15]. The MIMO-UNet+ [8] and MRPNet [11] were the SOTA models until the methods that were analyzed and compared in this study. The MIMO-UNet+ [8] mainly proposes a single encoder is employed for multi-scale input images, multiple deblurred images at different scales are produced by a single decoder, and asymmetric feature fusion is introduced for efficient multi-scale feature merging. On the other hand, the MRPNet [11] study focuses on a multi-stage approach that inspires the MAXIM [1] and it applies a progressive learning strategy that inspires the Restormer [2] further. The MRPNet integrates contextualized features with a high-resolution branch to preserve local information to learn restoration functions for degraded inputs. Additionally, it incorporates per-pixel adaptive design and facilitates information exchange between stages through sequential and lateral connections.

Also, the studies published that had compared the single-image blind motion deblurring models up to the present. El-Henawy, et al. [16] present a comprehensive study on image deblurring techniques, encompassing various blur types and evaluation metrics, focusing on blind deconvolution methods. Lai, Wei-Sheng, et al. [17] investigate the performance gap of single-image blind deblurring algorithms between synthetic and real-world blurred images, employing user studies to assess algorithm effectiveness. Zhang, K., Ren, W., Luo, W., et al. [18] offer an extensive survey of deep learning-based image deblurring techniques, categorizing methods based on architecture and application domains while discussing challenges and future directions.

3 METHODS

3.1 MAXIM: Multi-Axis MLP for Image Processing

MAXIM is published to overcome the "limited receptive field" and "locality" problems of the local-attention block that is used to avoid patch boundary artifacts of the self-attention block in Transformers. Z. Tu et al. [1] proposed "Multi-Axis Gated Block (MAB)" and "Cross-Gating Block (CGB)" that contain MLP architecture with spatial gates for the MAXIM backbone.

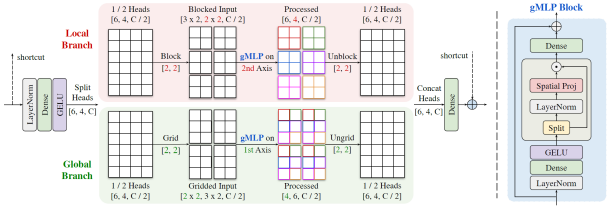


Figure 3: Architecture of MAB

The MAXIM backbone (Figure 2.a) can mix local and global features efficiently without affecting the scale of these features. This mixing operation is done with the MAB shown in Figure 3. The MAB uses multi-axis attention with a specific splitting strategy. In this splitting strategy, half of the channels are processed in the local branch while the other half is processed in the global branch. The main difference between these branches is the processing format of channels. In the local branch, the process is carried out as blocks while in the global branch, it is carried out as grids. During these processes, the gMLP block applies parallel for different order single-axis for each branch to make the MAB fully convolutional.

$$\Omega(\text{MAB}) = \underbrace{d^2HWC}_{\text{Global gMLP}} + \underbrace{b^2HWC}_{\text{Local gMLP}} + \underbrace{10HWC^2}_{\text{Dense layers}}, \quad (1)$$

Figure 2.b demonstrates that MAB stacks with a Residual Channel Attention Block (RCAB) and then inserts into each encoder, decoder, and bottleneck for the MAXIM Backbone. Also, the computational complexity of the MAB is linear as shown in equation (1).

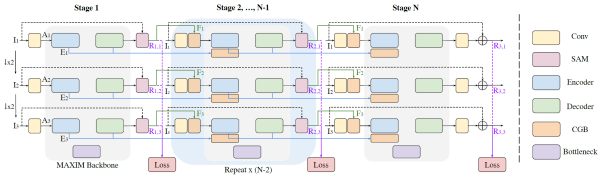


Figure 4: Multi-Stage Multi-Scale Framework with MAXIM

The MAXIM expands as a framework that operates in multiple stages and at multiple scales, as depicted in Figure 4. The CGB that

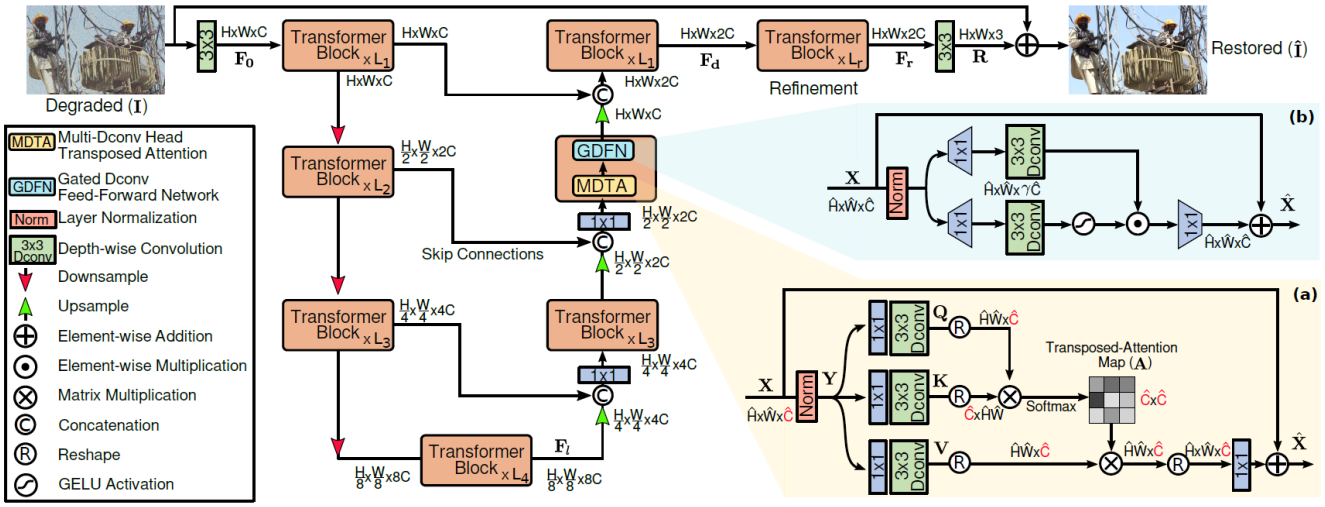


Figure 5: Architecture of Restormer

is demonstrated in Figure 2.c, applies multi-feature conditioning between these stages and it has a linear complexity that is bounded by the MAB's complexity.

$$\mathbf{X}_2 = \sigma(\mathbf{W}_1 \mathrm{LN}(\mathbf{X}_1)) \,, \quad \mathbf{Y}_2 = \sigma(\mathbf{W}_2 \mathrm{LN}(\mathbf{Y}_1)) \tag{2}$$

$$\hat{\mathbf{X}} = \mathbf{X}_2 \odot \mathrm{G}(\mathbf{Y}_2) \,, \quad \hat{\mathbf{Y}} = \mathbf{Y}_2 \odot \mathrm{G}(\mathbf{X}_2) \tag{3}$$

$$\mathrm{G}(\mathbf{x}) = \mathbf{W}_5([\mathbf{W}_3 \mathrm{Block}_b(\mathbf{z}_1), \mathbf{W}_4 \mathrm{Grid}_d(\mathbf{z}_2)]) \tag{4}$$

$$[\mathbf{z}_1, \mathbf{z}_2] = \mathbf{z} = \sigma(\mathbf{W}_6 \mathrm{LN}(\mathbf{x})), \tag{5}$$

$$\mathbf{X}_3 = \mathbf{X}_1 + \mathbf{W}_7 \hat{\mathbf{X}} \,, \quad \mathbf{Y}_3 = \mathbf{Y}_1 + \mathbf{W}_8 \hat{\mathbf{Y}}. \tag{6}$$

As shown in Figure 2.c, the CGB takes two inputs. These two inputs are projected into X_1 and Y_1 with Dense layers. First of all, X_1 and Y_1 are projected into X_2 and Y_2 with (2). Secondly, these X_2 and Y_2 are subjected to the Multi-Axes Cross Gating by using of (5) and (4) respectively that is explained in the MAB part and these cross-gated features are $\odot$ with original features with (3). Finally, the residual connected original inputs are added to these dot-produced features to get the final outputs with (6).

$$\mathcal{L} = \sum_{s=1}^{S} \sum_{n=1}^{N} [\, \mathcal{L}_{charbonnier}(\mathbf{R}_{s,n}, \mathbf{T}_n) + \lambda \mathcal{L}_{frequency}(\mathbf{R}_{s,n}, \mathbf{T}_n) \,], \tag{7}$$

The whole framework trains with an end-to-end loss that covers all stages and scales as illustrated in equation (7). The Supervised Attention Module (SAM) supports this framework throughout the stages for consecutively propagating attentive features.

3.2 Restormer: Efficient Transformer for High-Resolution Image Restoration

The quadratic computational complexity increase depending on the spatial dimensions is one of the biggest problems for the self-attention block that is proposed to solve the "limited receptive field" and "static weight" problems of CNN blocks. Therefore, Zamir et. al [2] proposed the Restormer which is an effective Transformer architecture that can capture long-range pixel dependencies without affecting the spatial dimensions. In the Restormer, Multi-Dconv Head Transposed Attention (MDTA) Block, Gated-Dconv Feed-Forward Network (GDFN) Block, and progressive-learning strategy are proposed for this purpose.

$$\begin{aligned} \hat{\mathbf{X}} &= W_p \,\mathrm{Attention}\left(\hat{\mathbf{Q}}, \hat{\mathbf{K}}, \hat{\mathbf{V}}\right) + \mathbf{X}, \\ \mathrm{Attention}\left(\hat{\mathbf{Q}}, \hat{\mathbf{K}}, \hat{\mathbf{V}}\right) &= \hat{\mathbf{V}} \cdot \mathrm{Softmax}\left(\hat{\mathbf{K}} \cdot \hat{\mathbf{Q}} / \alpha\right), \end{aligned} \tag{8}$$

As shown in Figure 5.a and equation (8), self-attention is applied by the MDTA block across feature dimensions (channel-wise) instead of spatial dimensions. The $O(W^2H^2)$ complexity of classical self-attention reduces into the linear by using this channel-wise self-attention approach.

$$\begin{aligned} \hat{\mathbf{X}} &= W_p^0 \,\mathrm{Gating}\,(\mathbf{X}) + \mathbf{X}, \\ \mathrm{Gating}(\mathbf{X}) &= \phi(W_d^1 W_p^1(\mathrm{LN}(\mathbf{X}))) \odot W_d^2 W_p^2(\mathrm{LN}(\mathbf{X})), \end{aligned} \tag{9}$$

The GDFN Block is depicted in Figure 5.b controls the output's quality by surpassing the less informative features while choosing the useful ones to pass further in the network. The surpassing mechanism originates from the gating connection between two projected linear layers, one of them activated with the GELU activation function. The gating formulation of this block is illustrated in the (9).

The local context mixing in Restormer can be achievable by the consecutive 1x1 Conv and 3x3 DConv units in the MDTA and GDFN blocks which are depicted in Figure 5. This consecutiveness provides pixel-wise aggregation of cross-channel contexts and channel-wise aggregation of local contexts. The power of convolution brings into the Restormer with this local context mixing operation. Also, during the covariance-based attention map computing for the MDTA block, this operation models the contextualized global relationships between pixels.

The Restormer model is trained with a progressive learning strategy is first mentioned in the MRPNet [11]. In this learning, the model starts to train with large batches with small patches for the early epochs and continues to train in small batches with large image patches for later epochs. The Restormer is more compatible with different resolutions and performs better performance with this training approach.

The Restormer's working strategy is demonstrated in Figure 5. The Restormer's pipeline starts with a convolution operation over the degraded input image (I) to get low-lever feature embeddings which are named as shallow features (F_0). These features run through a 4-level symmetric encoder-decoder to transform into deep features (F_d). In this 4-level, for the recovery process, the encoders are connected to decoders with skip connections. The encoder expands the channel capacity of high-resolution images by reducing the spatial size hierarchically to get low-resolution latent features (F_l) while the decoder recovers the high-resolution representations from these features consecutively. During the downsampling, pixel unshuffle is used while the pixel shuffle is applied for upsampling. In the refinement stage, the deep features (F_d) are further enriched (F_r) at high spatial resolution. In the final, to generate a residual image (R) which is added to the degraded image (I) to obtain a restored image ($\hat{I} = I + R$), the convolution is applied to (F_r).

3.3 NAFNet: Simple Baselines for Image Restoration

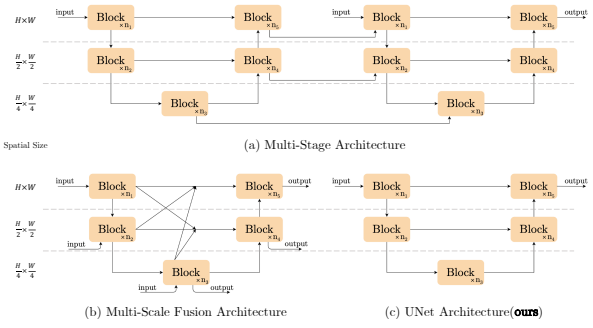


Figure 6: Architecture Comparison

The complexity of state-of-the-art methods has increased so much with all the advances happening in the single-image blind motion deblurring field and some of the researchers have started considering whether a simple, not too complex model can show results comparable to SOTA methods. Therefore Chen et al. [3] decided to research this topic. For this purpose, they divide complexity into two categories which are inter-block and intra-block complexity. The inter-block complexity refers to the general architectural complexity of the model, such as links between feature maps. In contrast, intra-block complexity refers to what the blocks of the model are made of, such as activation functions and attention layers.

By considering these complexities, Chen et al. [3] desire to seek whether a model with lower complexity that can achieve state-of-the-art performance. The first problem that needs to be dealt with is low inter-block complexity to build a model satisfying the low complexity. The authors decided to use a single-stage U-Net architecture that is depicted in Figure 6, just like most state-of-the-art approaches have done for lower inter-block complexity.

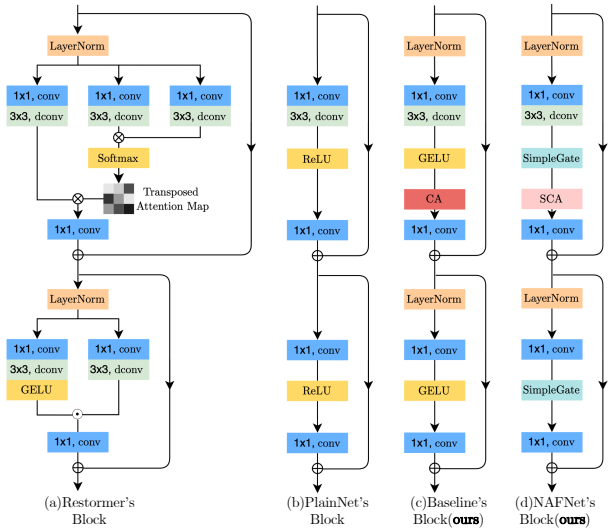


Figure 7: Intra-block structure comparison

The second problem to get a low complexity model is having lower intra-block complexity. To achieve this, the authors start with a plain block (Figure 7.b), containing only the most common components such as Convolution, ReLU, and shortcuts. They systematically add and replace state-of-the-art method elements to measure how much performance boost these components provide based on the experiments made.

Due to the problem of batch normalization not being stable during training, as this is one of the problems encountered nowadays across the state-of-the-art, the authors applied Layer Normalization to the plain block to help stabilize the training process. This adjustment can make training more efficient, even at higher learning rates. The majority of state-of-the-art works also use Layer Normalization. The activation function used in the plain block is ReLU, but in the baseline that goes beyond the plain block, the authors decided to use GELU, just like most state-of-the-art methods, since it proved in recent works providing better performance in most cases. Original self-attention has quadratic time complexity. Thus,

the authors decided to use a new kind of attention, "channel attention." This attention mechanism is based on spatial-wise attention, except it applies it to channels. This new attention mechanism helps reduce complexity while still capturing global dependencies.

With the new modifications mentioned above, the authors found a simple baseline shown in Figure 7.c network. This model was performing well but not so close to state-of-the-art methods, so the authors decided to see if they could improve the performance of the model way more while also maintaining a lower complexity, maybe even lower than the baseline provided.

To deal with this, they search for state-of-the-art methods, see what they applied in their models, compare them, and observe what key components make a difference. The first key thing they observed was that gated linear units (GLU) are used dominantly in most of them. The Gated Linear Units can be formulated as:

$$Gate(\mathbf{X}, f, g, \sigma) = f(\mathbf{X}) \odot \sigma(g(\mathbf{X})), \tag{10}$$

$\mathbf{X}$ denotes the feature map, f and g are linear transformers, σ is the sigmoid function, and $\odot$ means element-wise multiplication. Using the GLU improves the performance, but sadly, it also increases the complexity. To handle this, the authors investigate what GELU is.

$$GELU(x) = x\Phi(x), \tag{11}$$

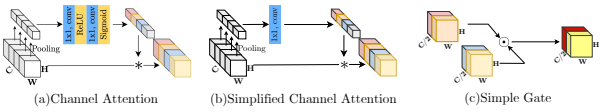
Φ denotes the cumulative distribution function of the standard normal distribution. GELU may be implemented by:

$$0.5x(1 + tanh[\sqrt{2/\pi}(x + 0.044715x^3)]). \tag{12}$$

Eqn. 10 and Eqn. 11 show that GELU is a particular case of GLU From these equations above, the authors derive the idea that GLU can be considered as an abstract generalization of activation functions, and by using GLU, they can replace nonlinear activation functions. Investigating it more, they observe that the GLU has nonlinearity and does not rely on σ. The equation $Gate(\mathbf{X}) = f(\mathbf{X}) \odot g(\mathbf{X})$ is nonlinear with or without the σ. Thus, they use a simple GLU named *SimpleGate*. Which can be formulated as

$$SimpleGate(\mathbf{X}, \mathbf{Y}) = \mathbf{X} \odot \mathbf{Y}, \tag{13}$$

where $\mathbf{X}$ and $\mathbf{Y}$ are feature maps.



(a)Channel Attention (b)Simplified Channel Attention (c)Simple Gate

Figure 8: Illustration of (a) Channel Attention(CA), (b) Simplified Channel Attention (SCA), and (c) Simple Gate (SG). $\odot$/$*$: element-wise/channel-wise multiplication

After this step, the authors decided to simplify the channel attention used in the baseline shown in Figure 8. Channel Attention (CA) equation can be written as:

$$CA(\mathbf{X}) = \mathbf{X} * \sigma(W_2 max(0, W_1 pool(\mathbf{X}))), \tag{14}$$

$\mathbf{X}$ means the feature map, *pool* is the global average pooling. σ is Sigmoid function, W_1, W_2 are fully-connected layers; and ReLU is between two fully-connected layers. $*$ is a channel-wise product operation.

Similar to the simplification of GELU to SimpleGate, we can see a similar pattern in the equations if we treat channel-attention as a function indicated as Ψ with input $\mathbf{X}$, Eqn. 14 might be written as:

$$CA(\mathbf{X}) = \mathbf{X} * \Psi(\mathbf{X}). \tag{15}$$

As we can see, the pattern between this equation and GLU equations. Just like done in the GELU case, this channel attention function can be considered as a subset of GLU, which later on can be simplified; thus, the authors simplify it and propose the Simplified Channel Attention.

$$SCA(\mathbf{X}) = \mathbf{X} * Wpool(\mathbf{X}). \tag{16}$$

At the end with all these simplification steps done on the baseline model, the authors reach a Nonlinear Activation Free Network (NAFNet) shown in Figure 7.d, which has a very simple architecture compared to recent state-of-the-art methods yet they claim it has comparable performance.

3.4 LaKDNet: Revisiting Image Deblurring with an Efficient ConvNet

LaKDNet was proposed on February 4, 2023, by Ruan et al. [4]. The authors stated in their paper that despite the recent state-of-the-art methods in image deblurring, such as Restormer and Uformer, using new transformer architecture, they chose to revisit the classic approach of deploying Convolutional Neural Networks. As a background of this, Vision Transformers [19] are good at deblurring tasks and they can handle long-range dependencies very well but this ability also brings a downside which is the quadratic time complexity due to the multi-head self-attention (MHSA). This high complexity makes transformers inefficient. There have been some attempts to reduce the time complexity of transformers. For example, one of the state-of-the-art methods like Restormer applies self-attention across feature dimensions instead of spatial dimensions. It reduces complexity, but it still has too much complexity when it is compared to what can be achieved by CNN.

In general, there were two opinions to merge the two approaches, namely Transformers and CNN, in a way that we could have inductive biases of the CNN models and the ability of transformers to capture global dependencies. One of the opinions was to bring inductive biases of Convolutional Neural Networks into Transformers, such as SwinTransformers [20], but the complexity is still high to be considered in tasks like image deblurring; the other opinion was to bring a larger effective receptive field (ERF) to CNN's with the usage of large kernels to capture global dependency just like transformers. This approach we mentioned is used in models such as Resnet [21] and Mobilenetv2 [22]. However, these methods mainly focus on problems that can work with low-resolution images, such as image classification, but in this field, we need to deal with higher-resolution images, which these models fail to achieve.

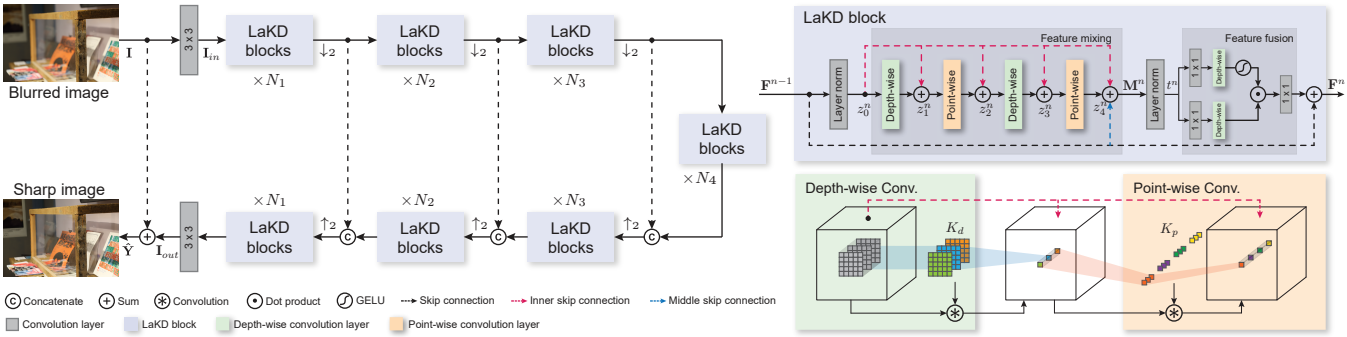


Figure 9: Architecture of LaKDNet and LaKD Block

In LaKDNet, there is a proposal for a new block, namely "LaKD block." this model is a unique CNN block with large kernel convolutions (depth-wise convolution and point-wise convolution) with the usage of large kernel sizes. This block aims to achieve a larger Effective Receptive Field (ERF), thus capturing dependencies similar to transformer-based methods in lower complexity and resulting in effective performance.

The overall architecture is a U-shape hierarchical network just like most of the state-of-the-art models such as Restormer and NAFNet; this U-shape architecture consists of 4 levels of symmetric encoder-decoder modules, with each module consisting of N LaKD block $N \in \{N_1, N_2, N_3, N_4\}$. Given an input image $I \in R^{H\times W\times 3}$ network first extracts low-level features $I_{in} \in R^{H\times W\times C}$ with C channels using a convolution layers. Then, these extracted features are forwarded through the encoder-decoder layers in order to complete the blur removal operation; after this step, there is a convolution layer at the end to get features $I_{out} \in R^{H\times W\times C}$ after there is pixel unshuffle/shuffle operation for downsampling and upsampling respectively. There is a shortcut in the architecture that is connecting I with I_{out} and finally getting the restored sharp image.

LaKD block, which was introduced by LaKDNet, has two stages. The first stage of this block is called the feature mixer, and the second stage is called the feature fusion stage. LaKD block achieves large ERF with the help of these modules. The feature mixer module consists of two depth-wise convolutions and two point-wise convolutions with larger kernel sizes (e.g., 9x9). At the beginning, Layer Normalization is applied to stabilize training. Also, there are skip connections between these layers. The feature mixer module allows the model to be used for distant spatial location mixtures, thus leading to high ERF. The second stage, feature fusion, also has depth-wise separable convolution layers separating two paths; there is a point-wise convolution before each depth-wise convolution in each path. At the end of one path, there is a GELU activation applied, and after this, they are concatenated.

Feature M^n in nth LaKD block where F^{n-1} is the output of the feature fusion module in n-1th LaKD block and 1 < n <= N:

$$M^n = F^{n-1} + z_4^n \tag{17}$$

The intermediate feature z_k is calculated as:

$$z^N_{(k+1)} = z^N_0 + g(z^N_k), \quad g = \begin{cases} \text{depthwise}, & \text{if } k = 1,3 \\ \text{pointwise}, & \text{if } k = 2,4 \end{cases} \tag{18}$$

$$z_0^n = LN(F^{n-1}) \tag{19}$$

Where LN is layer normalization after these steps output feature F^n from feature fusion process is calculated as:

$$\mathbf{F}^n = \mathbf{F}^{n-1} + \mathrm{LN}\{\alpha[g(W_1(t^n))] \odot g(W_2(t^n))\}, \tag{20}$$

where $t^n = \mathrm{LN}(\mathbf{M}^n)$, W_1 and W_2 are two separate 1×1 convolution layers which are then combined with element-wise multiplication denoted as $\odot$ and followed by a GELU activation α. Here, g only applies depth-wise convolution with 3×3 kernel. The network has hierarchically distributed LaKD blocks that enable a large ERF and make a significant contribution to the restoration of fine details.

4 EXPERIMENTAL SETUP

4.1 DATASET

In this study, we are using four different popular datasets that are highly used for deblurring tasks. The GoPro [23] and the HIDE [24] datasets contain synthetic blurred and sharp image pairs with 1280x720 resolution. On the other hand, the RealBlur-R [25] and RealBlur-J [25] datasets contain real-world blurred and sharp image pairs with 680x773 resolution.

The models are trained with GoPro's training dataset which contains 2103 realistic blurry and sharp image pairs. For the testing stage, we used five different datasets. Firstly, we randomly collect 400 blurry and sharp image pairs from GoPro's test dataset, HIDE, RealBlur-R, and RealBlur-J datasets to evaluate the performances of models on each dataset. Finally, we created a custom dataset by randomly collecting 25 image pairs from each of these datasets to evaluate the performance of models in different conditions.

(a) Blurry Image

(b) Sharp Image

Figure 10: Example image pairs from GoPro Dataset

4.2 EVALUATION

In this study, Peak Signal-to-Noise Ratio (PSNR)[26], Structural Similarity Index (SSIM) [27], Learned Perceptual Image Patch Similarity (LPIPS) [28], and Deep Image Structure and Texture Similarity (DISTS) [29] are used to evaluate these methods. These metrics are generally used in image quality assessment tasks such as Image Deblurring, Super Pixel Resolution, Image Compression, and more. These metrics are applied to each dataset to determine the advantages and disadvantages of each method that is observed.

$$\text{PSNR} = 10 \cdot \log_{10}\left(\frac{\text{MAX}^2}{\text{MSE}}\right) \tag{21}$$

PSNR is a commonly used approach for evaluating the quality of reconstructed or processed signals, particularly in digital image processing. In our research, we will be using these metrics.PSNR quantifies an image's quality by calculating the ratio of peak signal intensity to destructive noise intensity, which influences representation fidelity. This metric is directly related to Mean Squared Error (MSE). The equation of this metric can be given in (21). Where *PSNR* is the Peak Signal-to-Noise Ratio, *MAX* is the highest value a pixel can get, and *MSE* is the classic Mean Squared Error between the ground truth and deblurred image in our case.

The Structural Similarity Index (SSIM) presented by Wang et al. [27] is a widely used metric like PSNR, in Image Deblurring tasks. What SSIM does is instead of calculating the numerical similarity of two images, it measures perceptual similarity. While the whole formula for this metric is complicated, the simple, straightforward equation that is used to measure between two windows x and y of standard size N × N can be given as equation (22).

$$\text{SSIM}(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \tag{22}$$

Where μ_x and μ_y are the means while σ_x^2 and σ_y^2 are the variances of the images. σ_{xy} is the covariance between images x and y. c_1 and c_2 are fixed values that stabilize the division with a weak denominator.

$$l(x,y) = \frac{2\mu_x\mu_y + c_1}{\mu_x^2 + \mu_y^2 + c_1} \tag{23}$$

$$c(x,y) = \frac{2\sigma_x\sigma_y + c_2}{\sigma_x^2 + \sigma_y^2 + c_2} \tag{24}$$

$$s(x,y) = \frac{\sigma_{xy} + c_3}{\sigma_x\sigma_y + c_3} \tag{25}$$

$$\text{SSIM}(x,y) = l(x,y)^\alpha \cdot c(x,y)^\beta \cdot s(x,y)^\gamma \tag{26}$$

There are three components to measure SSIM. Luminance(l), contrast (c), and structure (s). Each one of them was calculated. We define $c3 = c2 / 2$ and set α , β , γ to 1. SSIM is then a weighted combination of those components.

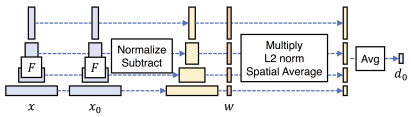


Figure 11: LPIPS Metric

LPIPS (Learned Perceptual Image Patch Similarity), e.g., Figure 11, can be considered a new metric relative to PSNR and SSIM. Even though PSNR and SSIM are still commonly used metrics in SOTA methods, they still have some problems, such as chrominance, intensities, etc. This advanced metric uses deep learning techniques to evaluate the similarity of two given images. We have used AlexNet's baseline LPIPS metric to evaluate the results we obtained.

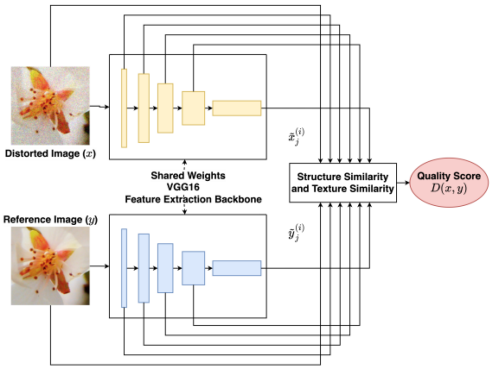


Figure 12: DISTS Metric

Deep Image Structure and Texture Similarity (DISTS) (e.g., Figure 12), just like LPIPS, is an advanced image quality metric that goes beyond PSNR and SSIM. It evaluates image quality by capturing textural and structural similarities similar to SSIM. It is built from an injective mapping function based on a VGG model. While metrics like LPIPS have problems with texture resampling, DISTS is way more robust in this problem. We also wanted to use this metric to get a better understanding of the experimental results.

Dataset	Metric	Models			
		MAXIM	**Restormer**	**NAFNet**	**LaKDNet**
Gopro	PSNR	32.6221	32.6832	**33.5394**	33.2130
	SSIM	0.9383	0.9377	**0.9463**	0.9432
	LPIPS	0.0842	0.0875	**0.0814**	0.0851
	DISTS	0.0751	0.0748	**0.0706**	0.0738
Hide	PSNR	**32.3655**	31.1139	31.2081	31.0420
	SSIM	**0.9272**	0.9156	0.9188	0.9167
	LPIPS	**0.1033**	0.1143	0.1064	0.1145
	DISTS	0.0716	0.0742	**0.0669**	0.0733
RealBlur-R	PSNR	32.6242	**32.8400**	32.5827	32.3265
	SSIM	0.9356	**0.9423**	0.9361	0.9360
	LPIPS	0.0675	**0.0575**	0.0613	0.0601
	DISTS	0.0879	0.0832	0.0839	**0.0831**
RealBlur-J	PSNR	26.2783	**26.3211**	25.9895	26.0923
	SSIM	0.8136	**0.8191**	0.7971	0.8111
	LPIPS	0.1537	0.1546	0.1686	**0.1504**
	DISTS	**0.1045**	0.1111	0.1185	0.1059
Mixed	PSNR	30.4837	30.4331	**30.5654**	30.3915
	SSIM	0.8950	**0.8960**	0.8935	0.8952
	LPIPS	0.1024	0.1039	0.1038	**0.1019**
	DISTS	0.0847	0.0862	0.0840	**0.0839**
Number of Parameters		22.2 M	26.13 M	67.1 M	**17.1 M**
MACs(G)		2314	1983	**890**	1125

Table 1: Qualitative comparison of the models on various datasets and metrics

5 EXPERIMENTAL RESULTS

As discussed in the experimental setup section, we have trained all models with the GoPro training dataset for a more stable and meaningful comparison. We tested each model on five datasets: two of them synthetically blurred ones (GoPro, HIDE), two of them real blurred images (RealBlur-R, RealBlur-J), and a mixed dataset we created. In table 1, the results of each model with respect to corresponding PSNR, SSIM, LPIPS, and DISTS metrics are shown.

Even though the NAFNet model has the highest number of parameters we observed in the training and inference phases, it is the most efficient model in terms of MACs(G) compared to other models due to its very simple architectural design. The LaKDNet model has the lowest parameters and is the least complicated model in terms of computational complexity after the NAFNet model. This lower complexity of NAFNet can be understood since, unlike recent trends, it does not use transformers-based architecture and returns to old CNN-based architecture. As we observed, the most computationally inefficient model was the Maxim model. Restormer has shown that it is less efficient than LaKDNet and NAFNet, thus leading us to consider a trade-off between performance and efficiency.

The NAFNet model, which we found to perform best on the GoPro dataset, has also proven effective in most datasets, giving results similar to the best model in each dataset. Considering the computational resources this model requires and the performance it provides, this model became one of the two favorite models with LaKDNet in our experiments with respect to the efficiency-performance trade-off. The Restormer model, on the other hand, can capture global dependencies much better than others due to its transformer-based architecture, which could explain the best results in the RealBlur-R and RealBlur-J datasets. We also observed that LaKDNet performs best on the mixed dataset, which includes a randomly selected subset of sharp and blurry image pairs from each dataset. In the example Figure 13, LaKDNet performs best in terms of restoring the numbers and the writing. We observed this model surpassing most models in restoring numerical examples and facial details, as the official LaKDNet paper said. In this example, we observed that LaKDNet achieves well on the textual and numerical data; however, we also observed that the model generally performs poorly in real blurred images, as seen in Table 1.

In the example Figure 13, at first, it seems the Maxim model has the best PSNR and SSIM values; however, as can be seen in the figures, the LaKDNet output shows writings and numbers more clearly. Also, LaKDNet output has the best LPIPS and DISTS value; thus, it shows a trade-off between metrics that measure numerical similarity and perceptual similarities like LPIPS and DISTS. We observed that perceptual similarity metrics are more reliable than

PSNR: **21.6237**
SSIM: **0.5901**
LPIPS: 0.2450
DISTS: 0.1774

PSNR: 21.0502
SSIM: 0.5330
LPIPS: 0.2783
DISTS: 0.2042

PSNR: 21.0752
SSIM: 0.5326
LPIPS: 0.2708
DISTS: 0.1985

PSNR: 20.8050
SSIM: 0.5384
LPIPS: **0.2239**
DISTS: **0.1582**

(a) Blurry image **(b) Maxim output** **(c) Restormer output** **(d) NAFNet output** **(e) LaKDNet output**

(f) Blurry cropped **(g) Maxim cropped** **(h) Restormer cropped** **(i) NAFNet cropped** **(j) LaKDNet cropped**

Figure 13: Comparison of motion deblurring models. The top row shows the full images with the cropped area highlighted in red. The bottom row shows the cropped regions. Metrics below each image indicate the performance of each model on PSNR, SSIM, LPIPS, and DISTS.

traditional metrics, which kind of measure which one looks pleasing to the human eye, as seen in this particular example.

6 CONCLUSION

In conclusion, four different state-of-the-art single-image blind motion deblurring models are compared using four evaluation metrics over five different datasets. Our experiments and observations demonstrated that each model has its own superiority over different datasets as expected. However, the NAFNet model comes to the forefront in terms of the model complexity, computation cost, and evaluation results. Also, it performed the best PSNR on the mixed dataset and also performed quite close to the best ones on the other metrics. This situation shows that the NAFNet can deal with different image conditions for deblurring so we can say that the NAFNet can capture different conditions better than others. Overall, by using this work and understanding the advantages of the models that are compared, the researchers can produce better methods that reach state-of-the-art performance and capture the different conditions for single-image blind motion deblurring.

For the future direction, the study suggests three additions. First of all, researchers should focus on understanding the strengths and weaknesses of current models such as MAXIM, Restormer, NAFNet, and LaKDNet to develop more efficient, high-quality output producers, and adaptable approaches for future directions. The second suggestion is about evaluation metrics. The new evaluation metrics should developed to capture perceptual image quality better, besides the commonly used PSNR, SSIM, LPIPS, and DISTS. The last and third recommendation is efforts should be made to improve and expand real-world datasets for use in both training and evaluation, ensuring that these models perform robustly in diverse and realistic scenarios because many existing models suffer from a lack of real-world data, which limits their practical applicability. By following these three directions, the field can advance towards more reliable and perceptually accurate image processing solutions.

7 REFERENCES

[1] Tu, Z., et al. (2022). MAXIM: Multi-Axis MLP for Image Processing. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).

[2] Zamir, S. W., Arora, A., Khan, S., Hayat, M., Khan, F. S., Yang, M. (2021). Restormer: Efficient Transformer for High-Resolution Image Restoration. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).

[3] Chen, L., Chu, X., Zhang, X., Sun, J. (2022). Simple baselines for image restoration. In Proceedings of the European Conference on Computer Vision (ECCV).

[4] Ruan, L., Bemana, M., Seidel, H., Myszkowski, K., Chen, B. (2023). Revisiting image deblurring with an efficient ConvNet.

[5] Cho, S., Lee, S. (2009). Fast motion deblurring. In ACM SIGGRAPH Asia 2009 papers (SIGGRAPH Asia '09) (pp. 1-8). Association for Computing Machinery.

[6] Shan, Q., Jia, J., Agarwala, A. (2008). High-quality motion deblurring from a single image. ACM Transactions on Graphics, 27(3), 1-10.

[7] Xu, L., Jia, J. (2010). Two-phase kernel estimation for robust motion deblurring. In European Conference on Computer Vision (pp. 157-170). Springer.

[8] Cho, S.-J., Ji, S.-W., Hong, J.-P., Jung, S.-W., Ko, S.-J. (2021). Rethinking coarse-to-fine approach in single image deblurring. In International Conference on Computer Vision (pp. 4641-4650).

[9] Kim, K., Lee, S., Cho, S. (2022). Mssnet: Multi-scale-stage network for single image deblurring.

[10] Nah, S., Kim, T. H., Lee, K. M. (2017). Deep multi-scale convolutional neural network for dynamic scene deblurring. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 3883-3891).

[11] Zamir, S. W., Arora, A., Khan, S., Hayat, M., Khan, F. S., Yang, M.-H., Shao, L. (2021). Multi-stage progressive image restoration. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 14821-14831).

[12] Zhang, H., Dai, Y., Li, H., Kołniak, P. (2019). Deep stacked hierarchical multi-patch network for image deblurring. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 5978-5986).

[13] Wang, Z., Cun, X., Bao, J., Zhou, W., Liu, J., Li, H. (2022). Uformer: A general U-shaped transformer for image restoration. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 17683-17693).

[14] Liang, J., Cao, J., Sun, G., Zhang, K., Van Gool, L., Timofte, R. (2021). Swinir: Image restoration using Swin Transformer. In International Conference on Computer Vision (pp. 1833-1844).

[15] Chen, H., Wang, Y., Guo, T., Xu, C., Deng, Y., Liu, Z., ... Gao, W. (2021). Pre-trained image processing transformer. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).

[16] El-Henawy, I. M., Amin, A., Ahmed, K., Adel, H. (2014). A comparative study on image deblurring techniques.

[17] Lai, W., Huang, J., Hu, Z., Ahuja, N., Yang, M. (2016). A comparative study for single image blind deblurring. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 1701-1709.

[18] Zhang, K., Ren, W., Luo, W., et al. (2022). Deep image deblurring: A survey. International Journal of Computer Vision, 2103–2130.

[19] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. (2021). An image is worth 16x16 words: Transformers for image recognition at scale. In Proceedings of the International Conference on Learning Representations.

[20] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. (2021). Swin transformer: Hierarchical vision transformer using shifted windows. In Proceedings of the International Conference on Computer Vision, 10012–10022.

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 770–778.

[22] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. (2018). Mobilenetv2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 4510–4520.

[23] Nah, S., Hyun, T., Kyoung, K., and Lee, M. (2016). Deep Multi-scale Convolutional Neural Network for Dynamic Scene Deblurring.

[24] Shen, Z., Wang, W., Lu, X., Shen, J., Ling, H., Xu, T., and Shao, L. (2019). Human-aware motion deblurring. In Proceedings of the International Conference on Computer Vision (pp. 5572–5581).

[25] Rim, J., Lee, H., Won, J., and Cho, S. (2020). Real-world blur dataset for learning and benchmarking deblurring algorithms. In European Conference on Computer Vision, pp. 184–201. Springer.

[26] Horé, A., Ziou, D. (2010). Image quality metrics: PSNR vs. SSIM. In 2010 20th International Conference on Pattern Recognition (pp. 2366-2369).

[27] Wang, Z., Bovik, A. C., Sheikh, H. R., and Simoncelli, E. P. (2004). Image quality assessment: from error visibility to structural similarity. IEEE transactions on image processing, 600-612.

[28] Zhang, R., Isola, P., Efros, A., Shechtman, E., Wang, O. (2018). The unreasonable effectiveness of deep features as a perceptual metric. In 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).

[29] Ding, K., Ma, K., Wang, S., Simoncelli, E. P. (2020). Image quality assessment: Unifying structure and texture similarity. arXiv.