UNIVERSITY OF SCIENCE AND TECHNOLOGY OF HANOI

DEPARTMENT OF INFORMATION AND COMMUNICATION TECHNOLOGY

COMPUTER VISION

Image Deblurring Using Generative Adversarial Network and Autoencoder

Group Members

Giap Do Anh Minh	22BI13282
Do Minh Quang	22BI13379
Le Anh Quang	22BI13380
Ho Thanh Thuy Tien	22BI13419
Vu Ha Vy	22BI13485

May, 2025

Table of Contents

List of Abbreviations

Symbol	Meaning
GAN	Generative Adversarial Networks
PSNR	Peak Signal-to-Noise Ratio
SSIM	Structural Similarity Index Measure
GPU	Graphics Processing Unit
ReLU	Rectified Linear Unit
RGB	Red, Green, Blue
ANN	Artificial Neural Network
MSE	Mean Squared Error
VGG	Visual Geometry Group

Abstract

Image deblurring is a classic problem in low-level computer vision with the aim of recovering a clear image from a blurry input. However, traditional methods do not perform well on complex and diverse blur patterns. As a result, a large number of deblurring approaches have been proposed to improve performance by using deep-learning techniques. This study will focus on two different deep-learning methods, namely Generative Adversarial Networks (GAN) [1] and Autoencoders [2], in addressing the image deblurring task. Our approach involves implementing and training both architectures on the GoPro [3] dataset to evaluate their ability to reconstruct image clarity. Additionally, the two models were evaluated using standard image quality metrics, including PSNR [4] and SSIM [5], and were compared to other baseline models trained on the same dataset. The results reveal the distinct advantages and limitations of each model. Hence, we can gain deeper insight into their practical applicability for real-world image deblurring scenarios.

Chapter 1

Introduction

1.1 Context and Motivation

Image deblurring plays a crucial role in low-level computer vision, especially in today's digital age, where cameras are integral components of most electronic devices. As reliance on visual data continues to grow in fields such as photography, surveillance, autonomous driving, and medical imaging, the demand for high-quality, sharp images has become more critical than ever. In this context, image deblurring aims to restore a clear image from a blurred input, where the blur may result from various factors, including camera shake, lack of focus, or fast-moving objects. These challenges significantly affect the performance of many vision-based applications. Some examples of blurred images are illustrated in Figure 1.1

(a) Camera shake blur (b) Defocus blur (c) Fast-moving object blur

Figure 1.1: Examples of different blurry images. Causes for blur include (a) camera shake, (b) defocus scene, and (c) moving objects.

Traditional approaches to image deblurring often rely on convolution with blur kernels and blind deconvolution techniques that incorporate hand-crafted priors and optimization algorithms to recover a sharp image from a blurry input [6]. Eventhough these non-deep learning methods have shown good performance in specific scenarios, they struggle to

generalize to more complex and realistic cases. In particular, they often fail to handle common but challenging situations, such as strong motion blur.

Recent advances in deep learning have revolutionized the field of computer vision, leading to major breakthroughs in areas such as image classification [7][8], segmentation, and object detection [9][10][11]. Image deblurring has also benefited from this progress, with numerous deep learning-based methods proposed for both single-image and video deblurring tasks. Among these, architectures such as GAN and Autoencoders have emerged as powerful alternatives to traditional methods. Their ability to learn data-driven representations enables more effective reconstruction of clear images, even under complex and dynamic blur conditions. In this paper, we explore and compare the capabilities of GAN and Autoencoders in tackling the image deblurring problem, using the GoPro dataset as a benchmark. By examining how each model handles different types of blur, we aim to gain deeper insight into their respective strengths and limitations, and evaluate their potential for practical, real-world applications where image clarity is crucial.

1.2 Objectives

This project's objective involve analyzing and comparing two prominent deep learning approaches: GANs and enhanced Autoencoders—for the task of single-image motion deblurring. The main objectives are to design and implement deblurring models based on GAN and Autoencoder architectures, and to train them on the high-quality GoPro dataset, which provides a realistic simulation of motion blur.

Furthermore, we aim to evaluate and compare the performance of these models using standard image quality metrics such as PSNR and SSIM, along with qualitative visual inspection. Through this analysis, we seek to identify the architectural strengths and potential limitations of each approach, offering insights into their suitability for practical image restoration tasks.

1.3 Expected Outcomes

This research is expected to achieve several outcomes. First of all, we expect to develop fully functional image deblurring systems based on both GAN and Autoencoder architectures. These systems will be evaluated through a combination of quantitative metrics, such as PSNR and SSIM, as well as qualitative visual assessments to demonstrate the effectiveness of each model.

Additionally, the project seeks to achieve strong performance in restoring sharp, high-quality images from blurred inputs, showcasing the practical potential of these deep learning approaches in real-world applications.

Chapter 2

Materials and Methods

2.1 Dataset

2.1.1 Data Description

In this study, we utilize the GoPro dataset, which was introduced by Nah et al. [3] in 2017. Since its release, the GoPro dataset has become a cornerstone in the development and evaluation of image deblurring techniques. Specifically, it was created using a GoPro Hero4 Black camera, which captured sequences of sharp images at 240 frames per second. To simulate realistic motion blur, blurry images of varying strengths were then generated by averaging a subset of consecutive frames. Finally, the dataset composed a total of 3,214 image pairs, each consisting of a blurry image and its corresponding sharp ground truth, with a resolution of 1280×720 pixels. Thanks to this structure, the dataset offers a rich source of high-resolution images captured under real-world conditions. Therefore, it is particularly valuable for training, and benchmarking deep learning models aimed at removing motion blur from images.

2.1.2 Data pre-processing

Before feeding the data into the models, all images were preprocessed to ensure consistency in resolution and format. In terms of the GAN model, the process begins by loading image pairs from the GoPro dataset, specifically matching blurred images with their corresponding sharp ground truths. After that, each image is resized to a fixed resolution 448×448 to ensure consistency across the dataset and to accommodate GPU memory limitations. After resizing, the images are converted into tensor format, making them compatible with PyTorch.

Similarly, in the autoencoder model, the image pairs are resized to a fixed resolution of

224 × 224 pixels. Following the resizing step, the pixel values are normalized, typically scaling them to the range between 0 and 1, to ensure the input data is appropriately prepared for training. This preprocessing workflow guarantees that both models receive consistently formatted input, enabling effective training and evaluation.

2.2 Model Architecture

2.2.1 Generative Adversarial Network

Our Generative Adversarial Network (GAN) model is composed of two main components: the Generator and the Discriminator, as illustrated in Figure 2.1. The input to the GAN model includes the sharp-blur image pairs from the dataset, the sharp image will go into the Discriminator, and the blur image will go into the Generator.

Generator Architecture

The Generator is in charge of learning the features from the input images and recreating realistic images. The Generator is designed as an encoder-decoder architecture with a bottleneck in between.

The encoder part consists of a series of convolutional layers, each followed by batch normalization and ReLU activation. As going deeper into the network, the higher-level features from the blurred input image will be extracted.

After the encoder, the bottleneck is composed of a series of Residual blocks. These residual blocks efficiently allow the features to propagate through the network. Each block consists of two convolutional layers, followed by batch normalization and ReLU activation. Importantly, the output of each block is added to its input through skip connections, which preserves the information and allows the model to learn complex features.

The decoder part of the Generator employs a series of transposed convolutions, which are used to upsample the feature maps back to the original spatial resolution of the input image. Each transposed convolution block is followed by batch normalization and ReLU

activation. The final convolutional layer in the decoder reduces the output to three channels, corresponding to the RGB color space of the sharp image. Finally, a Tanh activation function is used at the final layer to ensure the output pixel values are normalized within the range of (-1, 1).

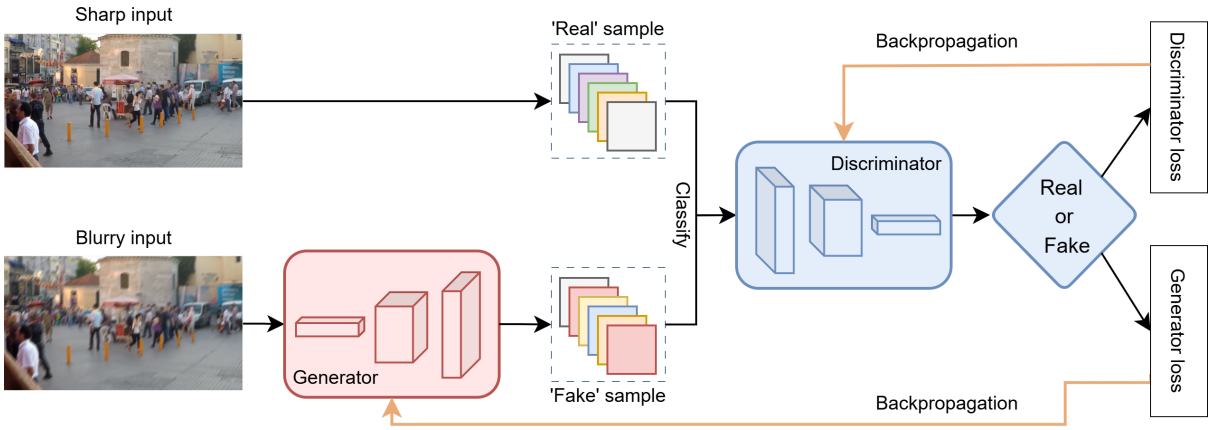

Figure 2.1: GAN model architecture.

Discriminator Architecture

The Discriminator is in charge of classifying the input images as either "real" (sharp) or "fake" (generated by the Generator). It also provides feedback to the Generator, thus encouraging it to generate more realistic output.

The Discriminator is composed of multiple convolution layers that progressively downsample the input image. Each convolution layer is followed by batch normalization and LeakyReLU activation. The final layer of the Discriminator reduces the output to a single channel, indicating the probability of the input image being real or fake. Finally, a Sigmoid activation is used in the final layer to produce a validity score between 0 and 1, where a value closer to 1 indicates that the Discriminator believes the image is real, and a value closer to 0 indicates it is fake.

2.2.2 Autoencoder

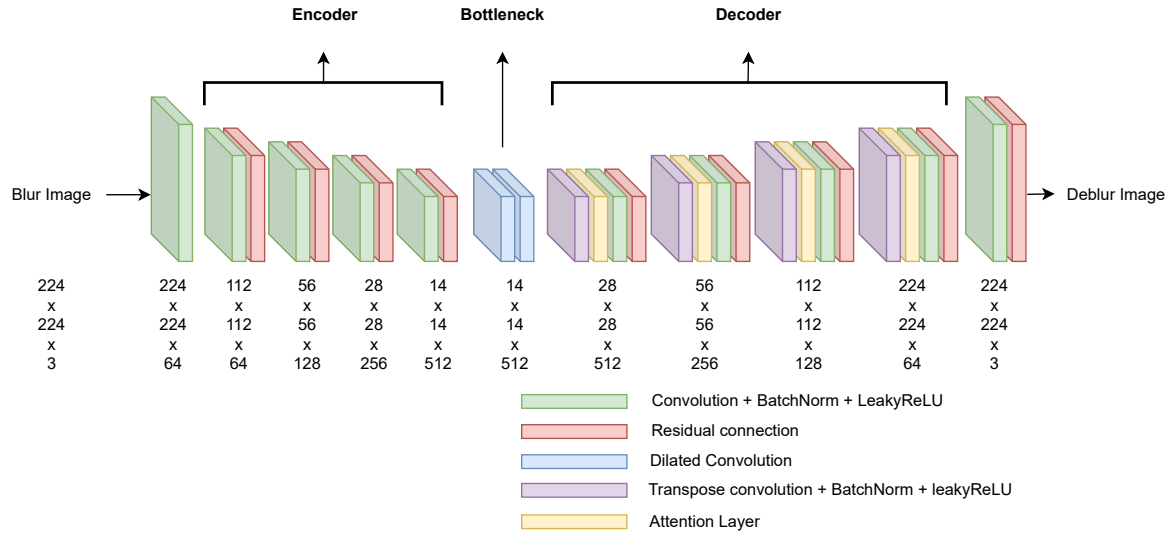

Figure 2.2: Architecture of enhanced autoencoder model

An autoencoder is a type of artificial neural network (ANN) used to learn the lower-dimensional feature representations of the input data in an unsupervised manner and this architecture is usually used for image reconstruction and image deblurring. In order to improve the performance of autoencoder for image deblurring, this article proposed an enhanced architecture of autoencoder by by incorporating several state-of-the-art deep learning techniques including residual connections, attention mechanisms, and dilated convolutions to effectively restore clarity to blurred images.

The encoder implements progressive downsampling with increasing feature depth, creating a multi-scale representation that captures important blur patterns at various resolutions. Then, the bottleneck employs dilated convolutions that dramatically expand the receptive field allowing the network to understand wide contextual information essential for accurate blur estimation. The decoder blocks will perform upsampling with decreasing feature depth and then output reconstructed sharp image.

In our approach, residual connections and attention mechanism are employed in the encoder-decoder framework. This architecture contains dual implementation of residual learning: local residual within encoder, decoder stages and global residual connection from the encoder's input to the decoder's output. The local residual learning allow the

network to focus on learning only the residual component, the difference between blur and sharp image features at each stage. The global skip connection acts as a direct path for passing original content to output so that the network retain overall image structure, fine details and appearance consistency. Residual blocks form the backbone of both encoder and decoder paths, each residual block processes input tensors through two consecutive convolutional layers followed by batch normalization and LeakyReLU activation. The skip connection adds the input feature map to the output of these layers so it allows gradients to flow more easily through the network during backpropagation, addressing the vanishing gradient problem. In image deblurring, not all spatial features are equally important, blur can be concentrated in specific regions Therefore, attetion block will help model prioritize and ensuring that the decoder focuses on correcting those blurred areas while ignoring irrelevant or already sharp regions. The attention block works by comparing encoder and decoder features to identify which spatial regions are most relevant for reconstruction. It generates an attention map that highlights these important areas and suppresses less useful information.

2.3 Model Configuration and Training

The hyperparameters used during the training of GAN model are summarized in Table 2.2. The training of GAN involves training two components: a Generator and a Discriminator. The loss functions used in the training process were a combination of two components. First, the adversarial loss was computed using binary cross-entropy loss with logits. Second, the Generator's performance was also evaluated using a pixel-wise reconstruction loss, calculated as the mean absolute error between the generated and the ground-truth sharp images.

Due to the limited resources, the training process spanned for a total of 95 epochs with a learning rate of 2e-4 and a batch size of 16. The Adam optimizer was used to update the parameters of both the Generator and the Discriminator with the betas of 0.5 and 0.999, which are commonly used in GAN models.

Additionally, during each epoch, the Discriminator was updated first by computing its loss on both real and generated samples and performing a backward pass. Afterward,

the Generator was updated using the adversarial loss from the Discriminator's feedback and the pixel-wise loss. Gradually, the Generator improves its ability to generate realistic images and the Discriminator gets better at distinguishing between real and fake images.

Table 2.1: Hyperparameters for GAN model training.

Hyperparameters	**Value**
Epochs	95
Learning Rate	2e-4
Generator Optimizer	Adam
Discriminator Optimizer	Adam
Batch size	16
AdamW β	(0.5, 0.999)

The GAN model was conducted on Kaggle using 2 NVIDIA T4-16GB GPUs. We used distributed training to speed up the training process, therefore, the training lasted for nearly 11 hours 30 minutes.

For the enhanced autoencoder architecture, in order to guide the optimization process, the model uses a perceptual loss function that combine a standard pixel-wise MSE and a feature-based loss derived from a pre-trained VGG19 network features, with the perceptual component weighted by 0.001 to ensure that while the model learns to minimize low-level reconstruction errors, it also maintains high-level structural. A learning rate scheduler is employed to improve convergence and training stability, the scheduler will reduce the learning rate by a factor of 0.5 if the validation loss is not decrease for 10 consecutive epochs.

Table 2.2: Hyperparameters for enhanced autoencoder model training.

Hyperparameters	**Value**
Epochs	100
Learning Rate	2e-4
Optimizer	Adam
Batch size	16
AdamW β	0.5

2.4 Model Evaluation

To evaluate our model performance, we use two image deblurring metrics: Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index Measure (SSIM)

2.4.1 Peak Signal-to-Noise Ratio (PSNR)

The Peak Signal-to-Noise Ratio (PSNR) is an expression for the ratio between the maximum possible value (power) of a signal and the power of distorting noise that affects the quality of its representation. It is computed based on the Mean Squared Error (MSE) between the original image I and the deblurred image K, both of size $M \times N$. The formulas are as follows:

$$\text{MSE} = \frac{1}{M \cdot N} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} [I(i,j) - K(i,j)]^2 \tag{2.1}$$

$$\text{PSNR} = 10 \cdot \log_{10} \left(\frac{\text{MAX}_I^2}{\text{MSE}} \right) \tag{2.2}$$

Here, MAX_I represents the maximum possible pixel value (e.g., 255 for 8-bit grayscale images). Higher PSNR values indicate better image quality. However, PSNR is primarily sensitive to pixel-wise differences and may not fully capture perceptual quality.

2.4.2 Structural Similarity Index Measure (SSIM)

The Structural Similarity Index Measure (SSIM) assesses the similarity between two images by considering luminance, contrast, and structural components, making it more aligned with human visual perception. SSIM is computed over local patches of the images. For two patches x and y, the SSIM is defined as:

$$\text{SSIM}(x,y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \tag{2.3}$$

Where:

- μ_x, μ_y: Mean intensities of patches x and y.
- σ_x, σ_y: Standard deviations of patches x and y.
- σ_{xy}: Covariance between patches x and y.
- $C_1 = (K_1 L)^2$, $C_2 = (K_2 L)^2$: Constants to stabilize division, with L as the dynamic range (e.g., 255 for 8-bit images) and typical values $K_1 = 0.01$, $K_2 = 0.03$.

SSIM values range from 0 to 1, with 1 indicating identical images. Values above 0.8 typically reflect high perceptual similarity, while values below 0.5 indicate significant

degradation. SSIM is particularly valuable for evaluating deblurring tasks, as it accounts for structural integrity and perceptual quality.

Chapter 3

Result and Discussions

This chapter provides an analysis of our models' performance by evaluating the model's quantitative results. The evaluation was conducted using the metrics discussed in Section 2.4 and comparing the model's performance with the baseline performances. We will also discuss the challenges encountered during the experimentation and the model limitations.

3.1 Experimental Results

To effectively evaluate our models, we select some existing baselines. The first baseline is the model from Kupyn et.al. work using Conditional Adversarial Network [12]. The results are shown in the Table 3.1.

The second baseline is the results done by Nah. et.al [3]. They used Deep multi-scale convolutional neural network. The third baseline is the DeblurGAN model [13], and the final baseline is from Li et.al [14]. These results are also shown in the Table 3.1.

Additionally, our models' performances are also described in Table 3.1.

Table 3.1: Our models' performance compared with other baselines.

Model	**Parameters**	**PSNR**	**SSIM**
Kupyn et al [1]	-	28.7	0.958
Nah et al [3]	-	28.93	0.91
Ji et al [13]	-	31.15	0.96
Li et al [14]	11.4M	29.3	0.72
GAN (ours)	8.1M	27.24	0.8568
Autoencoder (ours)	10.3M	26.95	0.8392

Table 3.1 presents the evaluation of image deblurring performance among several established baselines and our two models: GAN and autoencoder. Among the methods, Ji et

al. with their DeblurGAN model achieved the highest performance with a PSNR of 31.15 and an SSIM of 0.96. All the baselines show very solid performance.

Whereas, our GAN model achieved a PSNR of 27.24 and SSIM of 0.8568, while the autoencoder recorded the PSNR of 26.95 and SSIM of 0.8392. Both models' SSIMs are higher than that of the model by Li et al., which indicates better preservation of structural information, but our PSNR scores are lower. Moreover, our GAN model demonstrated slightly higher results than our autoencoder. This is expected since the adversarial components of GAN encourage the Generator to generate sharper images. However, since our GAN model used only 6 residual blocks, compared to 12 blocks used in Ji et al.'s work and 9 blocks in Li et al.'s model, the results might not be as good as theirs.

One of the primary reasons for the performance gap of our models might be the simplicity of the architecture. Our GAN model utilizes a relatively lightweight encoder-residual-decoder structure without leveraging more advanced techniques such as attention mechanisms, multi-scale feature fusion, or Wasserstein GAN, which are used in the latest methods and are known to boost the performance in deblurring tasks. Another contributing factor is the loss function design. For keeping the simplicity and for learning, in this project, the Generator was trained with a combination of adversarial loss and pixel-wise L1 loss. However, relying solely on L1 loss can lead to overly smooth outputs and loss of fine details. Baseline models that incorporate perceptual loss or feature-space distances (e.g., VGG-based losses) are better equipped to preserve texture and structure, which directly improves SSIM. The lack of such perceptual constraints in our model likely limits the visual sharpness of generated outputs. Overall, our models achieve decent performances, but there needs to be more adjustments to enhance those models' performance further.

In terms of parameters, since we focused on implementing simple architectures, our models are more efficient compared to baselines. As shown in Table 3.1, our GAN model has only 8.1 million parameters, which are more efficient compared to similar models.

Figure 3.1: Deblurring performance of enhanced autoencoder architecture, from left to right: Input blur image, Ground truth sharp image, autoencoder output.

Figure 3.2: Deblurring performance of GAN architecture, from left to right: Input blur image, GAN output, Ground truth sharp image.

3.2 Challenges Encountered

One of the major challenges encountered during the development and training of our image deblurring models was the limitation of computational resources, which significantly impacted both model performance and the overall process. Fine-tuning large transformer-based models like GAN requires substantial GPU power, but due to limited

access to high-performance hardware, we had to rely on Kaggle's free version for training. As a result, we were constrained to using smaller batch sizes and fewer epochs to fit within the available GPU memory. These compromises not only extended the training time but also limited the model's potential performance.

3.3 Model Limitations

While the GAN and Autoencoder models showed promise in addressing the image deblurring problem, there are still some limitations to consider. Firstly, both GAN and Autoencoder models faced difficulty in restoring sharp image with fine details. This could be clearly seen in their performance, as the GAN achieved a PSNR of only 27.24, while Autoencoder recorded a PSNR of 26.95, both of which are relatively low considering the complexity of the image restoration task. These outcomes suggest that while the models can reduce overall blur, they struggle with reconstructing high-frequency components critical for image clarity.

Furthermore, the relatively simple architecture of the GAN, with a limited number of residual blocks and the lack of advanced regularization techniques restricted their ability to generalize complex blur patterns. This may explain why the model's overall performance still falls short of expectations.

Chapter 4

Conclusion and Future work

4.1 Conclusion

In this project, we have developed and evaluated our two deep learning models, a Generative Adversarial Network (GAN) and an enhanced Autoencoder architecture. The results demon- strate that our model can perform the image deblurring task. For GAN, PSNR metric is 27.24 and SSIM is 0.8568, while Autoencoder have PSNR with 26.95 and SSIM with 0.8392. Although our GAN performed marginally better in this particular implementation, we investigated more complex techniques for capturing blur features by integrating complicated elements like residual connections, attention mechanisms, and dilated convolutions into the Autoencoder. Furthermore, our analysis of these architectures and a comparison of their benchmarks can be a useful starting point for further research on improving deep learning-based deblurring methods.

4.2 Future work

Despite the promising results achieved by our GAN and Autoencoder models, several challenges need to be addressed in our future work. First, we plan to improve image deblurring performance by refining the models to enhance sharpness and recover finer details more effectively. And we aim to improve the models' ability to capture more nuanced textures and structures lost in the blur, thereby enhancing their overall learning process and pushing output quality towards state-of-the-art levels.

We also aim to build an application integrated with our models. This application will demonstrate our models' performance in real-world scenarios and showcase their potential real-world applications.

References

[1] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

[2] Dor Bank, Noam Koenigstein, and Raja Giryes. Autoencoders. *Machine learning for data science handbook: data mining and knowledge discovery handbook*, pages 353–374, 2023.

[3] Seungjun Nah, Tae Hyun Kim, and Kyoung Mu Lee. Deep multi-scale convolutional neural network for dynamic scene deblurring. In *CVPR*, July 2017.

[4] Ariel Hore and Djemel Ziou. Image quality metrics: Psnr vs. ssim. In *Proceedings of the IEEE International Conference on Pattern Recognition (ICPR)*, pages 2366–2369, 2010.

[5] Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

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

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

[8] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[9] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2961–2969, 2017.

[10] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE Con-*

ference on Computer Vision and Pattern Recognition (CVPR), pages 1125–1134, 2017.

[11] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 91–99, 2015.

[12] Orest Kupyn, Volodymyr Budzan, Mykola Mykhailych, Dmytro Mishkin, and Jiří Matas. Deblurgan: Blind motion deblurring using conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8183–8192, 2018.

[13] Wentao Ji, Xing Chen, and Yihong Li. Blind motion deblurring using improved deblurgan. *IET Image Processing*, 18(2):327–347, 2024.

[14] Zhengdong Li. Image deblurring using gan. *arXiv preprint arXiv:2312.09496*, 2023.