DEPARTMENT OF ELECTRICAL
ENGINEERING
INDIAN INSTITUTE OF
TECHNOLOGY
MADRAS
CHENNAI - 600036

DEEP UNSUPERVISED SINGLE IMAGE DEBLURRING

A Project Report

Submitted by

LOKESH KUMAR T

In the partial fulfilment of requirements

For the award of the

DUAL DEGREE (B.Tech. and M.Tech.)

June 2021

QUOTATIONS

कर्मण्येवाधिकारस्ते मा फलेषु कदाचन |
मा कर्मफलहेतुर्भूर्मा ते सङ्गोऽस्त्वकर्मणि ||

— श्रीमद् भगवद् गीता।

karmaṇy-evādhikāras te mā phaleṣhu kadāchana
mā karma-phala-hetur bhūr mā te saṅgo 'stvakarmaṇi

— Srimad Bhagavad Gītā

Translation: Your right is for action alone, never for the results. Do not become the agent of the results of action. May you not have any inclination for inaction.

DEDICATION

to Amma, Appa, Teachers & Almighty

CERTIFICATE

This is to undertake that the Thesis titled **DEEP UNSUPERVISED SINGLE IMAGE DEBLURRING**, submitted by me to the Indian Institute of Technology Madras, for the award of **Bachelor of Technology and Master of Technology**, is a bona fide record of the research work done by me under the supervision of Dr. Rajagopalan AN. The contents of this project report, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Place: Chennai 600 036

Date: 15 July 2021

Lokesh Kumar T

Prof. Rajagopalan A N
Research Guide
Professor
Dept. of Electrical Engineering
IIT-Madras, 600 036

ACKNOWLEDGEMENTS

I would like to thank **Prof. Rajagopalan AN** for allowing me to explore my area of interest. I learnt a lot from his way of approaching a problem, and from his thought-provoking questions in our meetings.

I'm thankful to **Praveen Kandula** with whom I collaborated on this project. He provided me with a lot of insights that helped in this project and also in my understanding of the field. I would like to acknowledge the kind extension of the Samsung - IITM Pravartak Fellowship for the project.

Last but not the least, I would like to thank my mother **Karpagam T**, my father **Thirunavukkarasu P**, my litte brother **Sailesh Kumar T** for the endless support and encouragement during all these years and for allowing me to pursue my passion. I'm forever grateful to them.

ABSTRACT

It has been established in the literature that scale recurrent approaches, which are approaches that progressively restore images from lower resolutions can be successfully employed. Different attention schemes have been proposed which gives the network ability to focus on certain aspects of the task and improve its performance. Many supervised methods are present for image restoration tasks, but the main disadvantage of them is the demand for paired datasets which are cumbersome to obtain. Moreover, the strong supervision of such networks bias them towards specific deformations in the training dataset and when exposed to new deformations during inference it entails sub-optimal performance.

To address the issues stated above, we propose unsupervised domain-specific deblurring using a scale-adaptive attention module (SAAM). As the network is unsupervised, it does not require paired blurred-sharp images for training. Our network is guided by adversarial loss, thus making our network suitable for a distribution of blur functions. Given a blurred image input, different resolutions of the same image are used in our model during training and SAAM allows an effective flow of feature information from different resolution layers seamlessly. Ablation studies show that our coarse-to-fine mechanism outperforms end-to-end unsupervised models and SAAM is a better attention scheme than other proposed attention models in the literature. Quantitative and Qualitative comparisons show that our method performs existing unsupervised methods.

We also analyse the frequency perspective of attention and propose a learning-based frequency attention mechanism, Fourier Attention, where we aim to focus on important frequencies and reject trivial frequencies. This proposed method of attending in the frequency domain is a task agnostic module, which can be used in any task and any part of the network.

Contents

List of Tables

List of Figures

ABBREVIATIONS

CNN	Convolutional Neural Network
ConvLSTM	Convolutional Long Short Term Memory (cells)
DFT	Discrete Fourier Transform
FFT	Fourier Transform
GAN	Generative Adversarial Network
IFFT	Inverse Fast Fourier Transform
IITM	Indian Institute of Technology Madras
KL	Kullback–Leibler
LSTM	Long Short Term Memory (cells)
LTI	Linear Time Invariant
MAP	Maximum A Posteriori
MDF	Motion Density Function
MRF	Markov Random Fields
MSE	Mean Squared Error
PSF	Point Spread Function
PSNR	Peak Signal to Noise Ratio
RNN	Recurrent Neural Networks
SAAM	Scale Adaptive Attention Module
SSIM	Structural Similarity Index
UFAN	Unsupervised Fourier Attention Network
USAAD	Unsupervised Scale Adaptive Attention Network

NOTATION

Ψ	Image Prior Function
β	Flight path in degrees
Ψ_p	Pixel Patch with p as the center pixel
I_M^b	Input image downsampled to $M \times M$ dimension
$G_{B \to S}$	Generator network which transfers blur images to sharp images
$G_{B \to S}^E$	Encoder network in $G_{B \to S}$
$G_{B \to S}^D$	Decoder network in $G_{B \to S}$
$G_{S \to B}$	Generator network which transfers sharp images to blur images
P_a, g_d	Average Pooling Layer
$\mathcal{F}$	Fourier Transform
$\otimes$	Convolution Operator
∇	Gradient Operator
$\|.\|$	Absolute Magnitude Operator
$\phi_l(.)$	$l-$th layer of pretrained VGG-19 network
$\Phi_u(.)$	Fourier Phase response
ψ	Number of estimated filter samples
τ_m	Magnitude Filter CNN
τ_ϕ	Phase Filter CNN
F_u	FFT of the input feature map
F_f	FFT of the estimated filter
$FFT(.)$	Fast Fourier Transform
$iFFT(.)$	Inverse Fourier Transform
$Spline(.)$	Cubic Spline Interpolation
m_s^I	Interpolated Filter Magnitude Spectrum
ϕ_s^I	Interpolated Filter Phase Spectrum
θ_{UFAN}	Trainable parameters in Unsupervised Fourier Attention Network
θ_{USAAD}	Trainable parameters in Unsupervised Scale Adaptive Attention Network

Chapter 1

INTRODUCTION

When an object of interest has a non-trivial relative motion relative to the camera, we get an undesired phenomenon known as a blur. Though blur can be used for aesthetic purposes, it affects the performance of several downstream computer vision tasks like face recognition (Lu *et al.*, 2019*a*), object recognition (Kupyn *et al.*, 2018) and classification (Pei *et al.*, 2018). Given a blurred image, the deblurring task aims to recover the underlying latent sharp image. The significant progress in single image blind deblurring can be attributed to the advancement of inference algorithms, various natural image priors, availability of more general blur.

There are presently two methods to approach the problem, namely the conventional traditional methods and the recent deep learning models. In the domain of conventional methods, we estimate either the underlying camera motion or the blur kernel using an optimization framework. As this estimation problem is ill-posed, different methods use different assumptions and informative priors (Fang *et al.*, 2020; Xu *et al.*, 2013; Yan *et al.*, 2017; Vasu and Rajagopalan, 2017) on the image model and the nature of the blur kernel. These methods do not have any domain-specific conditions (specific for face images and text images) incorporated in them which can be exploited to boost the performance. Different priors have been proposed to handle domain-specific blur (Pan *et al.*, 2014*b*,*a*). However, the heavy dependence on the prior selection and their stoppage points during optimization is a limitation.

Deep learning has tremendously helped the field of deblurring. Convolutional neural networks (CNN) based supervised methods (Kupyn *et al.*, 2018,?; Nah *et al.*, 2017; Shen *et al.*, 2018; Simonyan and Zisserman, 2014; Purohit and Rajagopalan, 2020; Suin *et al.*, 2021; Nimisha *et al.*, 2017; Vasu *et al.*, 2018) were proposed for the task of deblurring. These algorithms forgo the need to define any priors due

to implicit learning of weight parameters during training. The main limitation of these methods is the demand for large amounts of paired training data which is complicated to obtain. Additionally, due to the strong supervision of loss functions during training, these networks incorporate dataset-specific biases which yield sub-optimal performances during deployment.

Unsupervised deblurring was proposed recently to relax the necessity of paired training data. (Madam Nimisha *et al.*, 2018) used generative adversarial networks (GAN) to transfer images from blur domain to sharp domain. An additional re-blurring network and gradient loss was used to maintain fidelity. (Lu *et al.*, 2019*b*) proposed an unsupervised network where blur can be disentangled into an encoder network using KL divergence loss. Methods consider deblurring as an end-to-end problem where GAN loss is used for training at a single scale. As a result, these methods give a suboptimal performance while handling coarse as well as fine details.

We address the above challenges by using a multi-scale architecture with a scale-adaptive attention module (SAAM). Several multi-scale supervised deblurring algorithms have been proposed in the past that use coarse-to-fine mechanism take advantage of processing different scales. These multi-scale methods use supervision loss to guarantee stability during training. In this thesis, we propose a multi-scale network for deblurring in an unsupervised setting. Training instability in GANs is well-studied in literature, and several solutions were proposed (Radford *et al.*, 2015). In this approach, instead of cascading the multi-resolution features, we use SAAM to attend to feature maps of lower scales as a function of the present scale. There are many advantages of such a procedure. Firstly, the hidden state uses information from different scales due to shared parameters. Secondly, the multi-scale approach reduces the training instability problems such as mode collapse and unwanted artefacts in the final image. Lastly, the SAAM module helps select relevant information from the lower scales, further improving the deblurring quality.

Different ablation studies show that the coarse-to-fine mechanism using SAAM gives better deblurring results than end-to-end counterparts devoid of recurrent

connections.

Our contributions in this area are summarized below:

We propose an unsupervised deblurring network with multi-scale architecture and a scale-dependent attention module. Different ablation studies show that scale recurrent networks give superior performance compared to end-to-end methods in an unsupervised setting.

We further show that SAAM facilitates better information flow across different scales, in contrast, to directly cascading or adding feature maps. We further show the efficacy of using SAAM over different attention modules.

We provide extensive comparisons on supervised and unsupervised methods and show that our method performs favourably against supervised and outperforms unsupervised methods qualitatively and quantitatively (on no-reference metrics) when tested on different datasets.

Chapter 2

CONVENTIONAL DEBLURRING METHODS

In this chapter, we will visit various works which fall in the traditional or conventional methods of deblurring. These methods use mathematical models to model the blur phenomenon and attempt to formulate it as an optimization problem. Depending on the nature of the blur, the models differ so as to best perform given a situation.

It's well known that the result of averaging intensity values on a frame due to the relative motion between the camera and the scene due to the exposure time is a significant cause for motion blur. Motion blur for all practical purposes is a nuisance, with an exception in the areas of image forensics (Zhang *et al.*, 2019*b*), depth reconstruction (Hu *et al.*, 2014) and to increase the aesthetic appeal of the images. The research in this area began with non-blind deblurring where significant restrictions were imposed on motion blur kernel (Point spread function (PSF)) such as assuming the camera motion to be uniform etc. Research then naturally shifted to PSF estimation to account for arbitrarily shaped blur kernels resulting from real-life hand-held camera movements and object movements in scenes. This marked the beginning of blind deconvolution algorithms where the PSF, as well as the underlying latent image, has to be estimated.

Efforts in solving the motion blur problem from the image acquisition standpoint were also undertaken. Traditional deblurring has are agnostic to image acquisition and only treat it as a post-processing problem. Recent advances in computational photography have helped in developing sensors with integrated motion deblurring characteristics. From simple inertial sensor data processing to building hybrid system architectures consisting of multiple cameras with different characteristics to suitably tailor the PSF have been tried out.

With this in mind, we explore different classes of conventional mathematical models for image deblurring in the following sections.

2.1 Uniform Single Image Motion Deblurring

Works that fall under this class generally assume that the motion blur kernel is shift-invariant. This assumption reduces the problem to that of image deconvolution. In non-blind deconvolution, the motion blur kernel is computed separately, and the task is to estimate the unblurred latent image. The main challenge in these algorithms is the appearance of ringing artefacts near strong edges, increased noise and compute time. Blind deconvolution is a more difficult problem as both the blur kernel and the latent images are treated as unknowns. Both blind and non-blind deconvolution find excessive use in various fields such as image processing, computer vision, medical and astronomical imaging and digital communication.

2.1.1 Non-blind deconvolution

In accordance with our assumption, the blur observed is a linearly filtered version of the latent unblurred image, which can be represented as,

$$b = I \otimes f \tag{2.1}$$

where b, I, f are the blurred image, latent unblurred image and the PSF respectively. In the frequency domain,

$$\mathcal{F}(b) = \mathcal{F}(I) \cdot \mathcal{F}(f) \tag{2.2}$$

where $\mathcal{F}$ is the Fourier transform.

If $\mathcal{F}(f)$ has numerically favourable values to take element wise inverse, and the blurred image is noise free then using the properties of an LTI system, we can solve for I using the follwoing equation,

$$\mathcal{F}(I) = \mathcal{F}(b)/\mathcal{F}(f) \tag{2.3}$$

This in theory seems to work, but practical constraints make this method unfavourable. The inverse of f may not exist if some entries in f are zero or close to zero. The motion PSFs caused by object or camera motion are typically band-

limited in nature and therefore they have very small values at band ends. Lastly, the noise-free image assumption is infeasible as there are many near-inevitable and sometimes inevitable sources of error such as image noise, quantization error, colour saturation and non-linear camera response function. This violation can be modelled in a more flexible form as,

$$b = I \otimes f + n \tag{2.4}$$

where n denotes error in the blurred image. Many advanced non-blind deconvolution methods are Wiener Deconvolution, Least Square Filtering, Richardson-Lucy method and recursive Kalman Filtering.

In the bird's eye view, many algirthms minimize energy consisting of two terms, the *data* term E_{data} (corresponding to the *likelihood* in probability) and *regularization* (also known as *prior*) E_{prior}. E_{data} measures the difference between the convolved image and the blur image which can be written as,

$$E_{data} = \Phi\left(I \otimes f - b\right) \tag{2.5}$$

where Φ is any function holding the notion of distance. The widely used distance function is the $L2$-norm of all elements. Its exactly taking the likelihood of a Gaussian distribution. E_{prior} is denoted as a function $\Psi(I)$ which has different specifications for different algorithms which can yield different results. The estimation of latent unblurred image I can be formulated as an optimization problem which is expressed as,

$$\min_{I} ||I \otimes f - b||^2 + \lambda\Psi(I) \tag{2.6}$$

where λ is a scalar weight determining the influence of the prior (regularization term) on the solution.

Early approaches use squared regularization constraints. Two forms are $\Psi(I) = ||I||^2$ and $\Psi(I) = ||\nabla I||^2$ where ∇ is the gradient operator. This regularizer enforces smoothness on image intensity and gradient values and is referred to as Tikhonov and Gaussian regularizers. Substituting them in the Eq. (2.6) gives

$$\min_I ||I \otimes f - b||^2 + \lambda ||I||^2 \tag{2.7}$$

and

$$\min_I ||I \otimes f - b||^2 + \lambda ||\nabla I||^2 \tag{2.8}$$

for optimization. As the regularization is simple, this leads to a simple solution quite similar to that of the inverse filter. In fact the closed form solution for Eq. (2.7) can be obtained and is,

$$\nu(I) = \frac{F^T}{F^T F + \lambda \Lambda} \nu(b) \tag{2.9}$$

where $\nu(.)$ is a vectorizing operator, F is the sparse convolution matrix generated from f and Λ is the identity matrix of the same dimension as $F^T F$. Note that regularization induces a bias and is known as an error of deconvolution. Moreover, the noise, if present in the image tends to lose its structural properties after this process.

Recent works such as (Chan and Wong, 1998) used a total variation regularizer known as Laplacian prior,

$$\Psi(I) = ||\nabla I||_1 \tag{2.10}$$

where ∇ denotes the first-order derivative operator. (Shan *et al.*, 2008) used a custom natural prior for the latent image by concatenating two piecewise continuous convex functions,

$$\Psi(I) = \begin{cases} a|\nabla I| & \text{if } |\nabla I| \leq \zeta \\ b|\nabla I|^2 + c & \text{if } |\nabla I| > \zeta \end{cases} \tag{2.11}$$

(Levin *et al.*, 2007) suggested a hyper-Laplacian prior which can be expressed as,

$$\Psi(I) = ||\nabla I||^\alpha \tag{2.12}$$

where $\alpha < 1$ representing a norm corresponding to a sparser distribution.

(Yang *et al.*, 2009) and (Xu and Jia, 2010) suppressed noise by using Laplacian data term which can be written as,

$$\min_{I} ||I \otimes f - b||_1 + ||\nabla I||_1 \tag{2.13}$$

This likelihood can suppress strong Gaussian and impulse image noise.

2.1.2 Blind deconvolution

Blind deconvolution requires estimation of both f and I. (Ayers and Dainty, 1988) iterated between updating blur PSF and the latent unblurred image. (Fish *et al.*, 1995) solved blind deconvolution in a maximum likelihood format using Richardson-Lucy iteration and (Chan and Wong, 1998) applied total variation regularizers to both PSF and the latent image.

The main challenge in these algorithms is that the solution space is high dimensional. This requires a more meaningful constraints for the optimization to converge to a good optima. Moders objective functions can be expressed as,

$$\min_{I,f} \Phi(I \otimes f - b) + \lambda_1 \Psi(I) + \lambda_2 \gamma(f) \tag{2.14}$$

where λ_1, λ_2 are two weights, Φ, Ψ and γ are different functions to constrain noise, latent image and PSF respectively. Similar to the non-blind convolution case, L_2-norm, L_1-norm , hyper-Laplacian prior can be used. A notable observation is that the above objective function corresponds to the posterior probability,

$$p(I, f|b) \alpha\ p(b|I, f)p(I)p(f) \tag{2.15}$$

$$\alpha\ \exp\left(-\Phi(I \otimes f - b)\right) \cdot \exp(-\lambda_1 \Psi(I)) \cdot \exp(-\lambda_2 \gamma(f)) \tag{2.16}$$

we can also theoretically estiamte PSF by maximizing the marginalized posterior probability function. Due to the intractability of the posterior normalization,(Fergus *et al.*, 2006) approximated the posterior distribution using parametric

factorization,

$$p(I, f|b) \approx q(f, I) = q(f)q(I) \tag{2.17}$$

$$= \Pi_i q(f_i)\Pi_j q(I_j) \tag{2.18}$$

where f and I are assumed to be independent, even pixels within f and I are considered independent as well giving us the equation.

Alternative energy minimization has achieved great success in uniform blind deconvolution. Works like (Cho and Lee, 2009; Xu *et al.*, 2013; Xu and Jia, 2010) which use this method when written in C++ takes around seconds to process an 800 x 800 image. The main idea in these methods is to make the solver avoid trivial solutions by generating an intermediate sharp edge representation. This is based on the observation that a sharp edge will have its boundary blended into the background due to motion blur

As a derivative of this method, we have edge recovery methods that predict edges from the blurred image to guide PSF estimation. These methods can be explicit or implicit in nature. (Shan *et al.*, 2008) iterates between PSF estimation and latent image discovery by minimizing two equations alternatively,

$$\min_f ||If - b||^2 + \lambda_2||f||_1 \tag{2.19}$$

and

$$\min_I ||If - b||^2 + \lambda_1||\nabla I||_1 \tag{2.20}$$

This iteration continues till convergence is achieved. By tailoring the hyper-parameters we can mitigate the problem of ring artefacts, maintain strong edges and improve the quality of the final image.

An algorithm similar to (Shan *et al.*, 2008) was later proposed by (Krishnan *et al.*, 2011) which uses the idea of normalized L_1 regularization term on image gradients. By the design of the regularization function, trivial image solutions

aren't favoured by the optimization. This algorithm iteratively solves,

$$\min_{\nabla I} ||\nabla I \otimes f - \nabla b||^2 + \lambda_3 \frac{||\nabla I||_1}{||\nabla I||} \tag{2.21}$$

and

$$\min_{f} ||\nabla I \otimes f - \nabla b||^2 + \lambda_4 ||f||_1 \tag{2.22}$$

2.2 Spatially Varying Motion Deblurring

The blur induced on the image due to general camera shake can violate the assumptions of space-invariance of the blur kernel. This gives rise to a phenomenon known as the space-variant blur.

2.2.1 A Unified Camera Shake Model

Let i be the latent image of the scene and b be the recorded blurred image. This can be mathematically formulated as,

$$b = k \otimes i + n \tag{2.23}$$

where $n \in \mathcal{N}(0, \sigma^2)$. This model doesnt account for depth dependent and illumination dependent blur in a general case. The convolution model can be rewritten as,

$$B = \mathcal{K}I + N \tag{2.24}$$

where I, B, N denote the column vector forms of i, b, n respectively. $\mathcal{K}$ is an image resampling matrix that applies the convolution, with each row of $\mathcal{K}$ being the blur kernel placed at each pixel location and unravelled into a row vector. This definition can incorporate spacially variant blur as in this kind of a blur, each row of $\mathcal{K}$ will be a shifted version of each other. As is known that the camera can have six degrees of freedom, three translations and three rotations. Therefore, any camera motion can be represented as a 1D continuous path in six-dimensional

space, which is also called as *camera pose space*. This path is discretized for analysis, and it's taken that the camera spends a fraction of its exposure time at each discretized pose steps and this proportion is called the *density* of that pose. These densities together form the *motion density function* (MDF) form which the blur kernel can be estimated. The MDF for all the camera poses form a column vector (A) over positions of the camera in the camera pose space.

The blurred image B is an integration over the images seen by the camera at all these discrete poses in the path. Therefore the unified camera shake image generation model becomes,

$$B = \sum_j a_j(K_j I) + N \tag{2.25}$$

where K_j is a warping transformation form I (unblurred latent image seen in original camera pose) to the image seen in pose j, and a_j is the MSF at pose j. N is the Gaussian noise. Given a 6D pose of the camera at pose j, the homography that warps the scene at depth d is P_j

$$P_j = C\left(R_j + \frac{1}{d}t_j[0\ 0\ 1]\right)C^{-1} \tag{2.26}$$

where R_j and t_j are the rotation and translation matrices for pose j, and C is the matrix of camera intrinsics. Assuming that the depth d is known, K_j is a resampling matrix where each row contains the weights used to compute the values of the pixels in the warps by applying inverse homography. From Eq. (2.24) and (2.25), we can write the blur matrix $\mathcal{K}$ as

$$\mathcal{K} = \sum_j a_j K_j \tag{2.27}$$

Once the values of $\mathcal{K}$ are known, the image can be deblurred using non blind deconvolution methods. We have discussed methods for non blind convolution, a more popular method is *maximum a posteriori* (MAP) technique. In this method of bayesian estimation, we calculate the posterior distribution and maximize it to obtain the best parameters. The objective of the optimization is,

$$P(I|B,A) = P(B|I)\frac{P(I)}{P(B)} \tag{2.28}$$

$$\text{argmax}_I P(I|B) = \text{argmin}_I \left[L(B|I) + L(I)\right] \tag{2.29}$$

Similar to our definition of likelihood term (E_{data}), we can define a data negative log-likelihood as

$$L(B|I) = \frac{||B - \mathcal{K}I||^2}{\sigma^2} \tag{2.30}$$

With the overview of the method, we are ready to understand the estimation procedure in the area of single image deblurring using motion density functions. Eq. (2.25) relates the MDF to the latent image and the blurred image. This is posed as a Bayesian estimation problem to estimate MDF and the latent unblurred image, given the observation and the priors on the image and MDF. Using the MAP estimate Eq. (2.29) we formulate the problem as,

$$E = \left|\left|\left[\sum_j a_j K_j\right] I - B\right|\right|^2 + prior(A) + prior(I) \tag{2.31}$$

$$prior(A) = \lambda_1 ||A||^\gamma + \lambda_2 ||\nabla A||^2 \tag{2.32}$$

$$prior(I) = \phi(|\partial_x I|) + \phi(|\partial_y I|) \tag{2.33}$$

Note that we have sparsity prior on the MDF values, and a smoothness prior which incorporates the concept of MDF representing a path.

The proposed optimization in Eq. (2.31) is non-linear in I, A. There are several methods to solve the above optimization problem such as alternating coordinate descent and expectation-maximization procedure. The initial estimate can be obtained by selecting uniformly distributed patches on the blurred image, which can be independently deblurred using blind deconvolution procedure as suggested by (Shan *et al.*, 2008). (Joshi *et al.*, 2008) filtered out patches having a low average

value of Harris corner metric as kernel estimation requires good distribution of edge orientations.

2.2.2 Image Deblurring using Inertial Measurement Sensors

In this class of methods, both hardware and software approach for estimating the spatially varying blur and uses hardware equipment that is attached to any camera. This equipment uses inexpensive gyroscope and accelerometers to measure a camera's acceleration and angular velocity during an exposure. The potential problem with this approach is the phenomenon known as 'drift'. This phenomenon is due to the integration of noisy measurements tracked over time.

Due to the excessive noise in the measurement, we instead use both inertial data and recorded burry image together with an image prior in the "aided blind deconvolution" method that computes camera-induced blur and the latent image using energy minimization framework.

Accelerometers measure the total acceleration at a given point along an axis, while the gyroscopes measure the angular velocity at a given point around an axis. As the camera is a rigid body with a three-axis accelerometer and a three-axis gyroscope, we can measure accelerations and angular velocities.

$$\omega_t^t = R_t \omega_t \tag{2.34}$$

$$b_t^t = R_t(a_t + g + (\omega_t \times (\omega_t \times r)) + \alpha_r \times r)) \tag{2.35}$$

ω_t^t, b_t^t is the angular velocity and the accelerometer reading at time t in the coordinate frame at time t. R_t is the rotation from the initial coordinate frame to the frame at time t. $\theta_t, \omega_t, \alpha_t$ denote the angular position, angular velocity and angular acceleration at time t in the initial frame. g is the gravitational acceleration in the camera's initial frame of reference. Then standard processing on the data using computational kinematics we can recover the necessary motion to aid the image deblurring exercise.

These measurements cannot be directly used for analysis as integrating these signals with small noise can result in a temporally growing deviation from the computed motion from the true motion. In order to compensate for the effect of drift by assuming it is linear in time. Using this information, we can reduce the number of unknowns to solve, thereby improving the optimization and its performance. Specifically, the work defines a function g that given a potential endpoint (u, v) computes the camera's translational path that best describes the observed acceleration (in the least-squares sense)

$$g(a, u, v) = \text{argmin}_x \sum_{t=0}^{T} \left(\frac{d^2 x_t}{dt^2} - a_t \right)^2 + (\theta_{x,T} - u)^2 + (\theta_{y,T} - v)^2 \tag{2.36}$$

To maintain simplicity, lets define function ρ that forms the blur sampling matrix from the camera intrinsics, extrinsics and scene depth using the rigid-body dynamics and temporal integration,

$$A(d) = \rho(\theta, x, d, K) \tag{2.37}$$

Therefore, the drift compensated blur matrix and deconvolution equations are

$$A(d, u, v) = \rho(\omega, g(a, u, v), d, K) \tag{2.38}$$

$$I = \text{argmin}_{I,d,u,v} \left[||B - A(d, u, v)I||^2/\sigma^2 + \lambda ||\nabla I||^{0.8} \right] \tag{2.39}$$

Then the optimization searches over the space of (u, v) to find the (u, v) that results in the image I that has the maximum likelihood given the observation and the image prior which can be obtained via energy minimization using the Nelder-Mead simplex method.

2.3 Honorable Mentions

Deblurring attempts have been made by altering the image acquisition method by incorporating hardware assistance. This new hybrid-imaging system can combine a high-resolution camera with an auxiliary low-resolution camera to effect deblurring. The secondary camera is used to obtain information about the spatially invariant or variant discrete parametric 2D motion field. This flow field is then used to derive PSF and then to finally obtain the non-blurred image.

Some works depend on a compact global parameterization of camera shake blur, based on the 3D rotation of the camera during exposure. A model-based three-parameter homographies is used to connect the camera motion to image motion and this formulation can be viewed as a generalization of the standard, spatially invariant convolutional model of image blur. Several algorithms have been proposed to reduce the computational complexity of these algorithms.

Some methods propose a semi-blind implementation of image deblurring on a smartphone device. It leverages the accuracy of inertial measurements on modern smartphones to make an accurate estimation of camera motion trajectory and which consequently helps in estimating PSF. These methods can be used to handle both image blur problems and rolling shutter issues. These methods run quite fast enough to be acceptable to end-users.

Works that use more than one sensor have also been gaining traction among the computer vision community. These methods can work in low-light conditions, can combine the advantages of high speed and high resolution for reducing motion blur. A hybrid sensor configuration for an extension to low-light imaging conditions is also discussed in this area.

In some areas, researchers have modified the imaging process to avoid the loss of high-frequency information during capture time using coded exposure photography. Projective motion path blur model which in comparison to conventional methods based on space-invariant blur kernels is more effective at modelling spatially-varying motion blur. Some methods operate in the irradiance domain to estimate the high dynamic range irradiance of a static scene from a set of blurred

and differently exposed observations captured with a handheld camera. The two-step procedure is to derive the camera motion and then estimate the latent scene irradiation.

2.4 Acknowledgement

Portions of text in this chapter are based on the (Rajagopalan and Chellappa, 2014). I would like to thank the authors and the contributors.

Chapter 3

LEARNING METHODS IN DEBLURRING

In this chapter, we will go over the different methods presented by the deep learning community in an attempt to solve the single image blind deblurring problem. We will inspect different deep learning solutions and architectures proposed and their performance against standard metrics. Introductory works in deep learning estimate some features of the blur kernel and then moved to estimate the blur kernel, which enters the realm of non-blind deblurring. The most recent methods which give the state of the art performance are what we call the end to end methods where the networks estimate the latent unblurred image directly and blur kernel isn't explicitly estimated. Some parts of this chapter follow the explanation given in the survey conducted by (Sahu *et al.*, 2019).

3.1 Introduction

Deep learning based image deblurring can be classified broadly into areas where the blur kernel is estimated from the given blurred image using Fourier transform (Chakrabarti, 2016) or motion flow (Gong *et al.*, 2017; Sun *et al.*, 2015) which will then be used to sharpen the image and end to end methods where the network doesn't explicitly estimate the blur kernel, but estimate the latent blurred image. Some of the methods rely on generative models (Kupyn *et al.*, 2018; Nah *et al.*, 2017; Ramakrishnan *et al.*, 2017) which are trained in an adversarial method.

We will also briefly look into the architecture proposed by other works and understand the specific details in them. The main advantage of deep learning methods is that despite their computational demand and time taken to train are relatively large when compared to that of statistical and conventional methods, their inference time can be accelerated and in general, it's much faster than their counterparts. Needless to mention, they have a better ranking on benchmarking

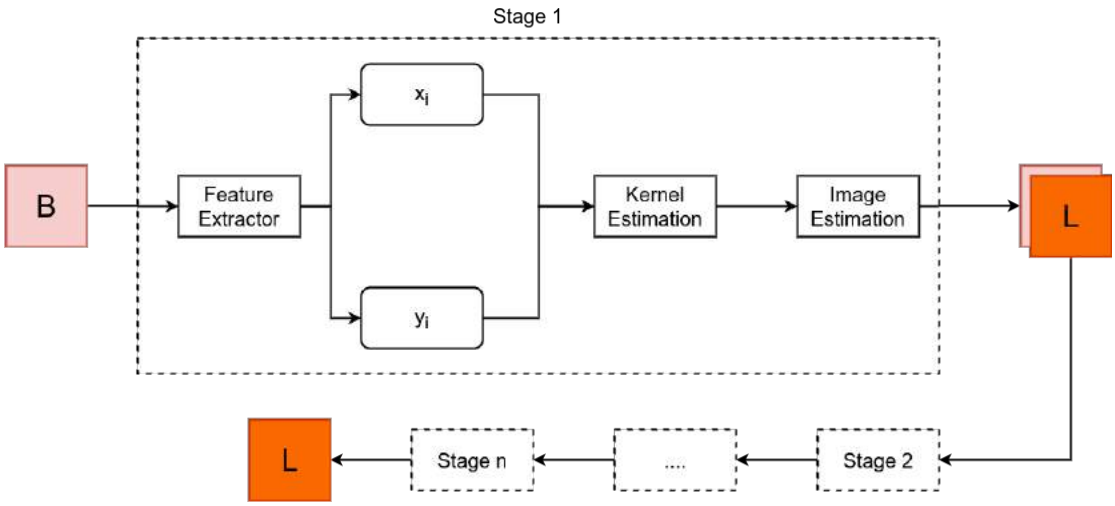


Fig. 3.1: The multi-stage architecture proposed by (Schuler *et al.*, 2015) Note that the first stage takes only blurred image as input and the subsequent stages taken a concatenation of both the refined image in the previous stage and the blurred image.

metrics (PSNR and SSIM).

3.2 Estimation of Blur Kernel

Deblurring requires global information from different parts of the image. In order to have connectivity across all the pixels in the image, we need to maintain a large number of parameters to an extent where the optimization becomes inefficient. So (Schuler *et al.*, 2015) proposed a method of using CNNs to extract features locally and then use those features in a multi-scale multi-stage architecture to estimate the latent image. There are three modules namely feature extraction, kernel estimation and latent image estimation modules. In the first stage, the blurry image is given as input and using the CNN extracted features the kernel is estimated which is used to estimate the latent image. This forms the first stage. From the second stage, both the estimated latent image from the previous stage and the blurred image is concatenated and passed through the three modules similar to the first stage for iterative refinement. The architecture diagram is shown in Fig. 3.1.

The paper used *tanh* activation on the extracted features f_j from the CNNs

to induce non-linearity. These hidden features are linearly recombined using coefficients α_{ij} and β_{ij} to form the hidden images x_i and y_i for stage i used for kernel estimation,

$$x_i = \sum_j \alpha_{ij} \tanh(f_j * y) \tag{3.1}$$

$$y_i = \sum_j \beta_{ij} \tanh(f_j * y) \tag{3.2}$$

where y is the blurred image B for the first stage or concatenation of B and the predicted sharper image L for later stages.

Given x_i and y_i, the kernel estimation module estimates the kernel K by minimizing,

$$\sum_i ||K * x_i - y_i||^2 + \beta_k ||K||^2 \tag{3.3}$$

Given K, we can find the latent image by minimizing,

$$\sum_i ||K * L - B||^2 + \beta_x ||L||^2 \tag{3.4}$$

where L is the latent image, β_x, β_k are regularization weights.

Another subclass within this class of methods are the methods that employ Fourier transform. Given a blurry image $B[n]$ where $n \in \mathbb{Z}^2$ are the indexes of the pixels. The task is to find the latent sharp image $L[n]$ such that it resembles the sharp image $I[n]$ closely,

$$B[n] = (I * K)[n] + N[n] \tag{3.5}$$

where $K[n]$ is the blur kernel such that $K[n] \geq 0$ (positivity constraint), $\sum_n K[n] = 1$ (unit sum constraint) and $N[n]$ the noise.

In the method given in (Chakrabarti, 2016), a blurry image $B[n]$ is divided into several overlapping patches. The surrounding pixels of the blurry patch $B_p = \{B[n] : n \in p\}$ is considered while computing the Fourier coefficients for better

results. Let the patch with the neighbouring pixels is called $B_{p^+} = \{B[n] : n \in p^+\}$ where $p \subset p^+$.

They use neural network to predict the Fourier coefficients of the deconvolution filter $\mathbf{G}_{p^+}[z]$ for the blurry patch B_{p^+}, where z is the two dimensional spatial frequencies in DFT. Then the filter is applied to the DFT of $\mathbf{B}_{p^+}$ to obtain the latent sharp image $\mathbf{L}_{p^+}[z]$,

$$\mathbf{L}_{p^+} = \mathbf{B}_{p^+}[z] \times \mathbf{G}_{p^+}[z] \tag{3.6}$$

Upon computing $\mathbf{L}_{p^+}$, we can use inverse discrete Fourier transform to get the latent image patch L_{p^+} from which L_p can be extracted.

To generate the coefficients of the filter the neural network uses a multi-resolution decomposition strategy, where the initial layers of the neural network are connected to only the adjacent bands of frequencies. The image is sampled into various patches which are then used to sample a higher frequency band using DFT. The loss function of this network is,

$$L = \frac{1}{|p|} \sum_{n \in p} (L_p[n] - I_p[n])^2 \tag{3.7}$$

It's assumed that the entire image is blurred by a single motion kernel K_λ which is obtained from the different kernels (from different patches) using the following,

$$K_\lambda = \text{argmin} \sum_i ||(K * (f_i * L_N)) - (f_i * B)||^2 + \lambda \sum_n |K[n]| \tag{3.8}$$

where f_i are the different derivative filters. After the estimation of K_λ, this becomes akin to that of a non-blind deblurring problem and deconvolution is used for final prediction.

Some methods use a motion vector for each patch of the input blurred image. Similar to that of (Chakrabarti, 2016), the method proposed by (Sun *et al.*, 2015) divides the image into overlapping patches and for each of them, a probability distribution of motion kernels is computed. The architecture diagram is shown in

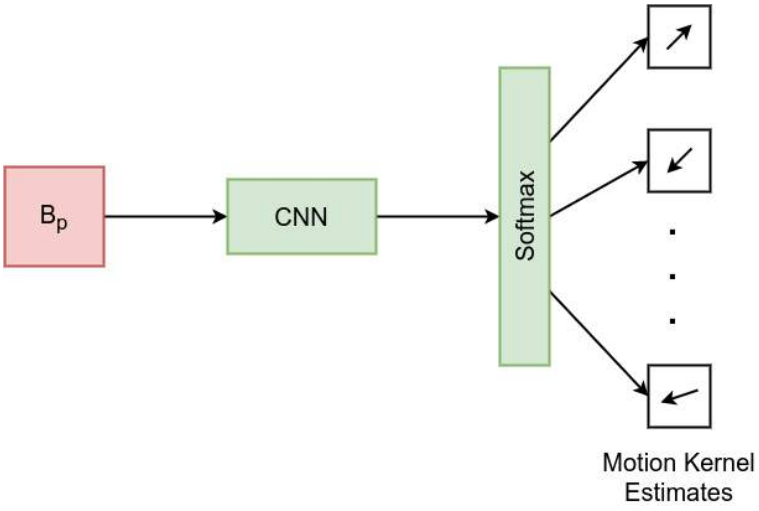


Fig. 3.2: The architecture diagram of the method proposed by (Sun *et al.*, 2015)

Fig. 3.2. Given a patch Ψ_p, centred at the pixel p, the network finds a probability distribution,

$$P(m = (l, o)|\Psi_p) \tag{3.9}$$

where $m = (l, o)$ is the motion kernel with length l and orientation o. Here $l \in S^l$ and $o \in S^o$ both S^l and S^o are discretized sets of length and orientation. This discretization results in artifacts which can be tackled by rotating the image and its motion kernel to get the new data entry. As they treat this like a multi class classification problem, the prediction is given as,

$$p(m = (l, o)|\Psi) = \frac{exp(z_i)}{\sum_{k=1}^{n} exp(z_k)} \tag{3.10}$$

where z is the output of the final fully connected layer and $n = |S^l| \times |S^o|$ i.e n is the total number of motion kernels. The loss function used to train is the loss function used in classification, cross-entropy. To compute the confidence of motion kernel from overlapping patches they use,

$$C(m_p = (l, o)) = \frac{1}{Z} \sum_{q:p \in \Psi_p} G_\sigma(||x_p - x_q||^2) P(m = (l, o)|\Psi_q) \tag{3.11}$$

where q is the center pixel of patch Ψ_q such that $p \in \Psi_q$. The weight G_σ is the Gaussian function which weights pixels closer to the centre more than pixels lying

in the outer perimeter. Z is the posterior normalization constant.

Post estimation of the motion kernels for all the patches, a Markov Random Function (MRF) is used to merge them all together and smoothen the transition of motion kernels. The dense motion field is generated by minimizing the energy function,

$$\sum_{p\in\Omega}[-C(m_p = (l_p, o_p)) + \sum_{q\in N(p)} \lambda[(u_p - u_q)^2 + (v_p - v_q)^2)]] \tag{3.12}$$

where Ω is a image region and u_p, u_q, v_p, v_q are defined as $u_i = l_i cos(o_i), v_i = l_i sin(o_i)$ for $i = p, q$. $N(p)$ is the neighborhood of p. After predicting the motion field, they deconvolve the blurred image and obtain the prediction of the deblurred image.

The previous approach Sun *et al.* (2015) predicts the motion flow for each patch, but methods (Gong *et al.*, 2017) have been proposed which use a CNN to estimate the pixel-wise dense motion flow for the entire image. The assumption of uniform homogeneous motion kernel assumption of (Sun *et al.*, 2015) is relaxed and this end to end method of estimating motion flow can handle real-life situations more effectively. No post-processing like MRF is required in this case. The architecture diagram of (Gong *et al.*, 2017) is shown in Fig. 3.3.

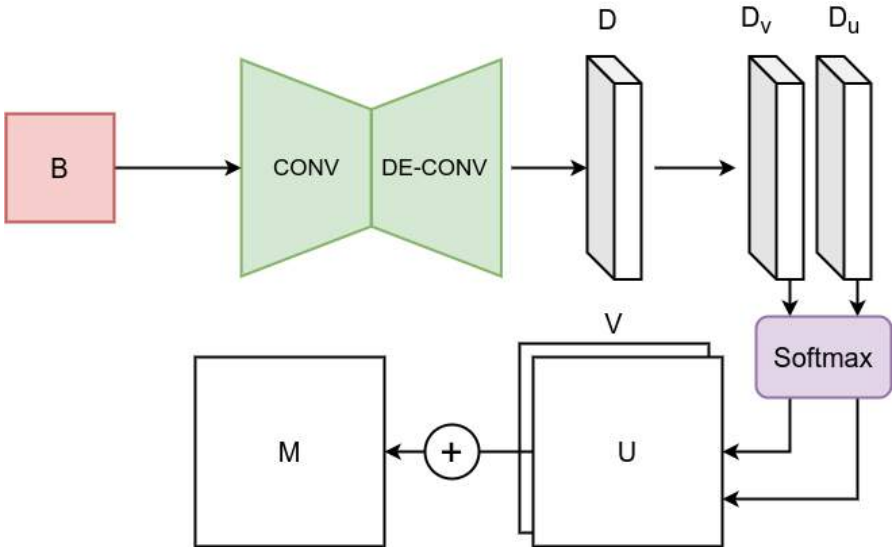


Fig. 3.3: The architecture diagram of the method proposed by (Gong *et al.*, 2017)

The network that estimates the motion field M is represented as f, if the

blurred image is B, then

$$f(B) = M \tag{3.13}$$

where the motion field can be represented as,

$$M = (U, V) \tag{3.14}$$

where U, V are horizontal and vertical motion maps respectively. Motion vectors are discretized and let $\mathbb{D}_u$ and $\mathbb{D}_v$ be the set that denotes the discretization. If the image of dimension $P \times Q$ is sent as input to the network, outputs of the network is of size $P \times Q \times D$ where $D = |\mathbb{D}_u| + |\mathbb{D}_v|$. The feature map is divided into $P \times Q \times |\mathbb{D}_u^+|$ and $P \times Q \times |\mathbb{D}_v|$ which are passed through the softmax layer to get the probabilities of the motion fields. Once the motion fields are obtained then the deblurring problem reduces to the deconvolution problem to obtain the sharp image.

3.3 Supervised End to End Methods

Works in this area can be broadly divided into methods that fall under the adversarial category (which employ adversarial loss) and non-adversarial methods which are in general supervised.

3.3.1 Non - Adversarial Methods

Deblurring in general requires a large receptive field so that it can gather enough information to engage effectively. Primarily, for this reason, (Nah *et al.*, 2017) propose a multi-scale convolutional neural network that restores the sharp images in an end to end manner where the blur can be caused by many sources. They also present a loss function that mimics conventional coarse-to-fine methods. The multi-scaled approach increases the receptive field of the network but also makes the convergence much harder. They add residual connections to make the gradient flow much more profound. (Tao *et al.*, 2018) also explores the multi-scale strategy,

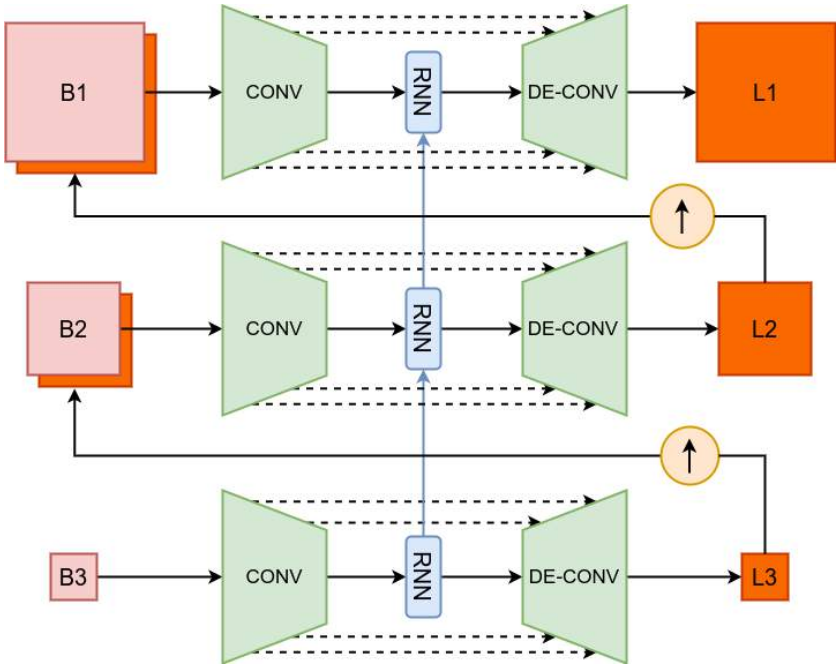


Fig. 3.4: Scale Recurrent Neural Network Architecture (Tao *et al.*, 2018)

but the proposed network has a lot less learnable parameters (which is convergence friendly when compared to the previous work) by using a small encoder-decoder type network with the recurrent module that also shares weights across resolution layers.

The scale recurrent network as shown in Fig. 3.4 consists of three parts the encoder (Net_E), recurrent layer (Net_R) and decoder (Net_D). The equations which represent the network functioning are,

$$f^i = Net_E(B^i, L^{i-1\uparrow}; \theta_E) \tag{3.15}$$

$$h^i, g^i = Net_R(h^{i-1\uparrow}, f^i; \theta_R) \tag{3.16}$$

$$L^i = Net_D(g^i; \theta_D) \tag{3.17}$$

where $\theta_E, \theta_R, \theta_D$ are the weights of their respective modules.

The encoder is a CNN with residual connections. For the first scale, the input is only the blurred image, but for the subsequent scales, the blurred image B^i is concatenated with the previous scale sharp image $L^{i-1,\uparrow}$. The feature is extracted from the encoder which we call f^i.

The module following the encoder module is the recurrent layer where Convolution LSTM (ConvLSTM) (Shi *et al.*, 2015) have been employed to give the best results. Ablation studies show that the addition of recurrent network across resolutions indeed contributes to improved performance. The inputs to the recurrent module are $h^{i-1\uparrow}$ the previous layer hidden state, and f^i the current layer encoder features. To mitigate the problem of gradient explosion, they use the gradient clipping method. The output of this module is the modified set of features g^i and the current resolution hidden state h^i.

The decoder module is again a small CNN with residual connections followed by a deconvolutional layer which increases the spatial dimension and the final estimate of the latent image is obtained for the current scale L^i. The combination of all these modules can be written as follows,

$$L^i, h^i = Net_{SR}(B^i, L^{i-1\uparrow}, h^{i-1,\uparrow}; \theta_{SR}) \tag{3.18}$$

where θ_{SR} is the weight shared across all scales. The loss function which guides the optimization is the Euclidean loss,

$$L = \sum_{i=1}^{n} \frac{\kappa_i}{N_i} ||L^i - I^i||_2^2 \tag{3.19}$$

where L^i and I^i are the latent restored image and the ground truth sharp image respectively. $\{\kappa_i\}$ are the weights assigned to the different resolutions with N_i being the normalization constant.

(Noroozi *et al.*, 2017) deblurs generic motion-blurred image by estimating the latent image directly through its three pyramid stages which allow removing blur gradually from a small amount in the lowest scale to the full amount in the input image scale. The three pyramid stages consist of several convolutional and deconvolutional layers which recreate the multiscale pyramid approach used in many other methods. The main idea of this paper is that the downsampled version of the image has a smaller blur compared to the full resolution image. Hence the network gradually mitigates the blur at that corresponding scale thereby turning the complex problem into manageable small units. Let N_1, N_2, N_3 refer to the

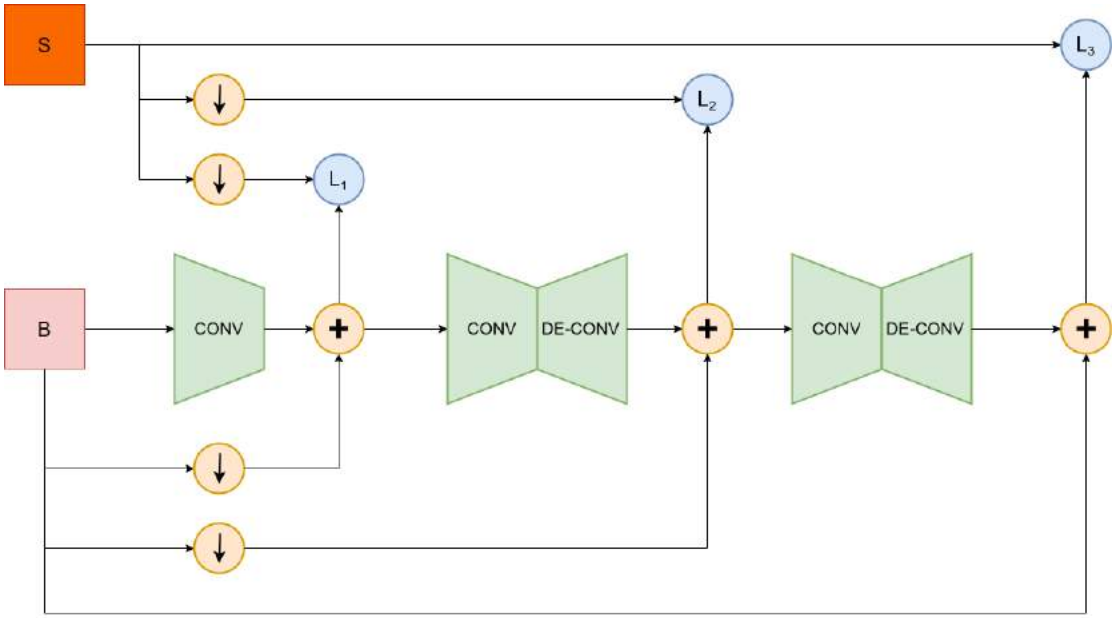


Fig. 3.5: Architecture Diagram of (Noroozi *et al.*, 2017). The three CNNs starting from the left denotes N_1, N_2, N_3 respectively.

pyramid units which is a combination of convolution and transpose convolution layers. Firstly the blurred image is given as input to N_1 which is a purely convolutional network and its output is concatenated with the downsampled version of the same blurred image which is sent to N_2, the second stage and then to the last stage N_3. The loss function is calculated as follows,

$$L_1 = \sum_{B,I} |N_1(B) + d_{1/4}(B) - d_{1/4}(I)|^2 \tag{3.20}$$

$$L_2 = \sum_{B,I} |N_2(N_1(B) + d_{1/4}(B)) + d_{1/2}(B) - d_{1/2}(I)|^2 \tag{3.21}$$

$$L_3 = \sum_{B,I} |N_3(N_2(N_1(B) + d_{1/4}(B)) + d_{1/2}(B)) + B - I|^2 \tag{3.22}$$

where d_x is a downsampling operator which reduces the dimension to x times the original dimension.

As we saw before the demand for a large receptive field for deburring increases the number of learnable parameters which makes the convergence more difficult. To address this challenge (Zhang *et al.*, 2018) proposed a network that is composed

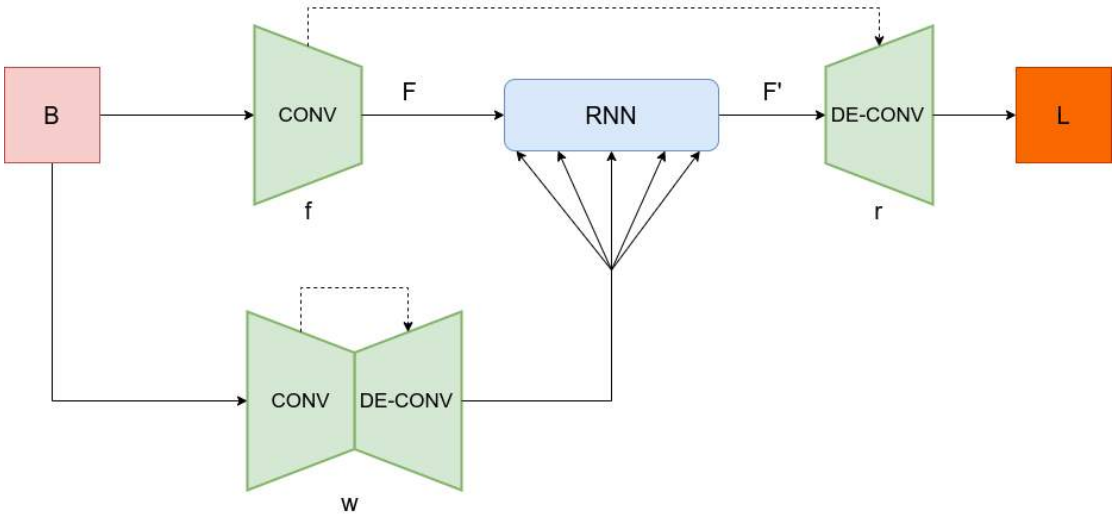


Fig. 3.6: Architecture Diagram of (Zhang *et al.*, 2018).

of three deep convolutional networks and a recurrent neural network. RNN is used as a deconvolution operator performed on feature maps extracted from the input image by one of the CNNs. Another CNN is used to learn the weights for RNNs in different spatial locations. Therefore, RNNs is spatially variant and could implicitly model the deblurring process with spatially variant kernels. The third CNN is used to reconstruct the final deblurred feature maps into the final restored image. The whole network is end to end trainable. The network architecture is shown in Fig. 3.6. By using this approach, they could achieve a large receptive field with a small network which increases both performance and speed by reducing computational complexity. As RNNs generate a receptive field in one direction, they use a convolutional layer after every RNN t fuse the receptive fields and obtain a two-dimensional structure. The architecture proposed by (Zhang *et al.*, 2018) can be summarized as,

$$F = f(B) \tag{3.23}$$

$$\theta = w(B) \tag{3.24}$$

$$F^{'} = rnn(F; \theta) \tag{3.25}$$

$$L = r(F^{'}) \tag{3.26}$$

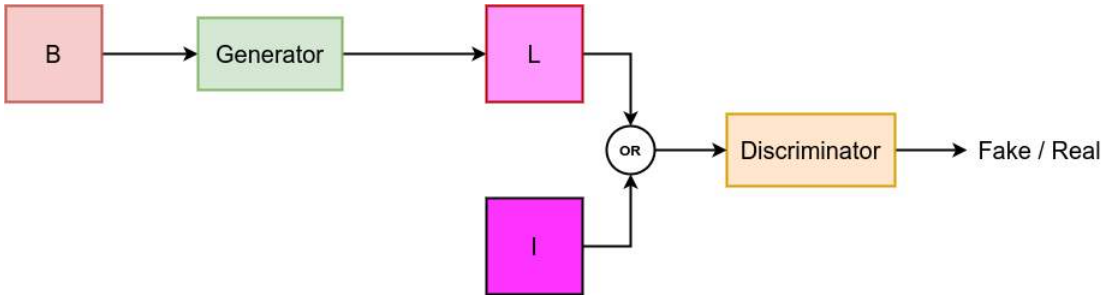


Fig. 3.7: Basic Structure of Generative Adversarial Network

where B is the blurry image, F is the extracted features, θ is the pixel-wise generated weights, $F^{'}$ are the modified features after passing through the RNN and L is the latent image.

3.3.2 Adversarial Methods

Blind deblurring can be solved in an end-to-end manner by using generative models like Generative Adversarial Networks (GANs) (Goodfellow *et al.*, 2014; Isola *et al.*, 2017; Arjovsky *et al.*, 2017). The architectural block diagram is shown in Fig. 3.7. The generative adversarial networks consist of two networks which we call generator and discriminator. The generator tries and generates data and the work of the discriminator is to examine whether the data is coming from the true real distribution. The generator aims to fool the discriminator into believing that its output is from the real distribution. These generator and discriminator are differentiable neural nets and can be trained via backpropagation. As the game proceeds, eventually, the generator will be forced to generate data from a distribution that is as close to the real world distribution as possible.

(Nah *et al.*, 2017) also uses the Multiscale convolutional network architecture that we saw in several other works. The architecture diagram is shown in Fig. 3.8. The network restores sharp images across scales in an end to end manner where the blur can be potentially caused by various sources. The lower resolution layers are used to obtain the global information in the image and the higher resolution layers are used to fine-tune the deblurring process. Residual connections used to ensure that all the parts of the network get ample gradient flow to train. MSE (Mean Squared Error) is used on each resolution and backpropagation is done.

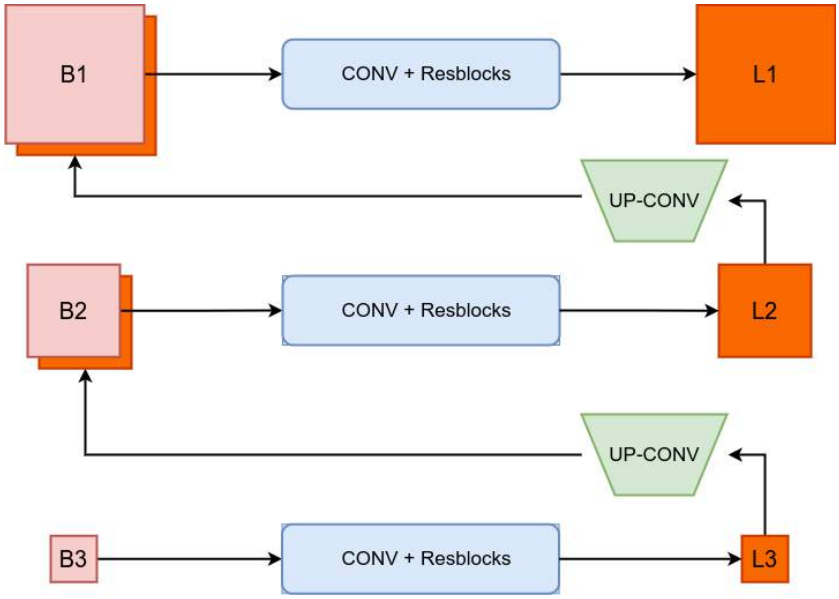


Fig. 3.8: Architecture Diagram proposed by (Nah *et al.*, 2017)

$$L_{content} = \frac{1}{2K} \sum_{k=1}^{K} \frac{1}{c_k h_k w_k} ||L_k - I_k||^2 \tag{3.27}$$

Here K is the total number of scales and c_k, h_k, w_k are the channels, height and width of the k^{th} scale while L_k, I_k are the latent and sharp images in the k^{th} resolution layer.

The output of the current scale is given as an input along with the blurred image to the resolution layer above it. Upconvolution (transposed convolution) layers are present in all the resolution layers except the highest resolution. The generated deblurred image of the last scale is given as input to the discriminator which tells whether the image is coming from the true sharp images distribution or from the multiscale generator. It's trained with the Discriminator loss function,

$$L_{adv} = \mathbb{E}_{S \sim p_{sharp}} log(D(S)) + \mathbb{E}_{B \sim p_{blurred}} [1 - log(G(B))] \tag{3.28}$$

where G, D are the generator and the discriminator respectively. Note that the images are generated across resolutions. The total loss function to train the network is given below,

$$L_{total} = L_{content} + \lambda L_{adv} \tag{3.29}$$

where λ is a weight constant. (Ramakrishnan *et al.*, 2017) uses a generator with a global skip connection in a way similar to (Huang *et al.*, 2017). There is no resolution reduction across the generator, the dimension of the feature maps are maintained throughout the generator. The benefits of DenseNet such as reduction in the vanishing gradient problem and stronger feature propagation help this small generator to perform better in this task of deblurring.

The generator is divided into three parts, namely head, dense field and tail. The generator head creates the activations which are then processed by the dense field which has several dense blocks with ReLU non-linear activation. The dense connection is achieved by feature concatenation of the previous layer features to the current layer. The output of the head is connected to the output of the tail via a global skip connection which enables rich gradient flow to the head.

Similar to (Nah *et al.*, 2017), the loss function here are divided into two parts, but with minor differences. MSE loss is taken from the features of VGG16 which is known as Perceptual Loss function (Johnson *et al.*, 2016).

$$L_{precep} = \frac{1}{W \times H} \sum_{x=1}^{W} \sum_{y=1}^{H} (\phi(I)_{x,y} - \phi(L)_{x,y}) \tag{3.30}$$

where ϕ denotes the function used to generate the features.

The adversarial loss used in this paper is conditional adversarial loss as the predicted image along with the corresponding blurred image is sent to the discriminator.

$$L_{adv_{con}} = -\mathbb{E}_{b \in B}[log(D(G(B))|B)] \tag{3.31}$$

where D is a discriminator. The total loss function used is, therefore,

$$L_{total} = L_{percep} + \lambda_1 L_{adv_{con}} + \lambda_2 L_{L_1} \tag{3.32}$$

where L_{L_1} is known as the L_1 loss, λ_1, λ_2 are the weights.

3.4 Unsupervised End to End Methods

Works in applying unsupervised methods in deblurring are relatively new and began with (Madam Nimisha *et al.*, 2018) and since then is a very active area of research. The method we propose also is an attempt to incorporate attentive scale recurrent training in the unsupervised domain.

Their work consists of a GAN which is used to learn a strong prior on the clean image domain using adversarial loss and maps the blurred image to its clean counterpart. There are three CNNs in this architecture, which are generator, discriminator and the re-blurring network. The generator takes as input a blurry image and transforms it to the estimate of a clean image. The discriminator's job is to find out whether the image sent to the discriminator is from the generator or from the true distribution. As the complete process is unsupervised, in order to maintain fidelity, we have a re-blur network that processes the generated restored image and reconstructs the blur. This image is expected to be identical to the input blurry image and therefore supervised losses can be applied. The architecture diagram along with the losses are shown in Fig. 3.9.

Apart from the contribution of unsupervised deblurring, they introduced a scale-space gradient module that effectively guided the training and stabilized the GAN training. The main idea is that the effect of blur reduces as the resolution of the image is decreased. This point is leveraged in the network and gradient losses are used in a multi-scale framework.

The adversarial loss was used to train the GAN setup.

$$L_{adv} = \min_{\theta} \frac{1}{N} \sum_{i} log(1 - D(G_\theta(B_i))) \tag{3.33}$$

where the unpaired blur and sharp domain datasets are $\{B_i\}, \{L_i\}$ and the trainable parameters is θ with G, D being generator and discriminator respectively.

With the re-blurring module, the generator is more constrained to estimate the ground truth image. Let the clean image from the generator be $L = G(B)$ which is passed again through the CNN module to obtain a reconstructed blurry image. The re-blurring loss can be expressed as,

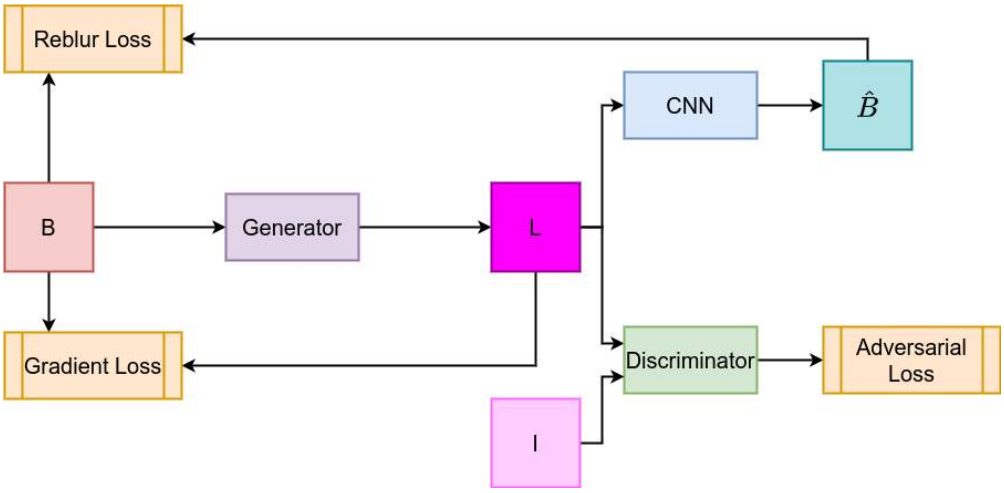


Fig. 3.9: Architecture Diagram proposed by (Madam Nimisha *et al.*, 2018)

$$L_{reblur}||B - \text{CNN}(L)||_2^2 \tag{3.34}$$

Along with the above losses, the authors use gradient loss at different scales which can be represented as,

$$L_{grad} = \sum_{s\in\{1,2,4,8,16\}} \lambda_s||\nabla B_{s\downarrow} - \nabla \hat{L}_{s\downarrow}| \tag{3.35}$$

Here ∇ denotes the gradient operator. In thier case, they use the Laplacian operator $\left(\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}\right)$ to calculate the gradients at different scales as shown in the summation (s). The weights increase with s and are set to be $[0.0001, 0.001, 0.01, 0.1, 1]$. Its demonstrated that addition of gradient loss removes the unwanted ringing artifacts in the final image and smoothens the result. The inclusion of the supporting loss functions (reblurring loss and gradient loss) makes output image comparable to the ground truth. Finally, the generator is trained with the combined loss function as shown below,

$$L_G = \gamma_{adv}L_{adv} + \gamma_{reblur}L_{reblur} + \gamma_{grad}L_{grad} \tag{3.36}$$

More recent work in this area of unsupervised blind deblurring is (Lu *et al.*,

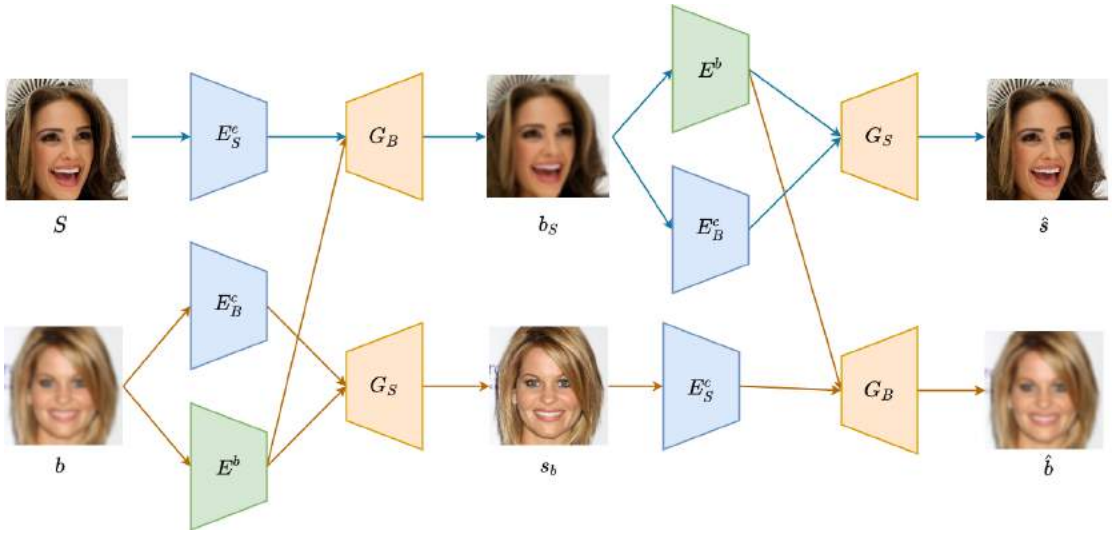


Fig. 3.10: Architecture Diagram proposed by (Lu *et al.*, 2019*b*)

2019*b*). They deal in domain-specific cases where they apply an unsupervised method based on disentangled representations. The disentanglement is achieved by splitting the content and the blur features in a blurred image using the content and the blur encoders. KL Divergence loss is used to regularize the blur attributes to minimize the leakage of content information in them. To maintain the fidelity of the generated images, they use re-blurring branch and cycle consistency loss.

The network proposed by them consists of four parts namely, 1) the content encoders E^c_B and E^c_S which denotes the blurred and sharp image generators, 2) blur encoder E^b; 3) blurred and sharp image generators G_B and G_S; 4) blurred and sharp image discriminators D_B and D_S. The architecture diagram is shown in Fig3.10. Given a blurry image b, and a sharp image s (note that they needn't be paired) the content encoders E^c_B, E^c_S extract the content information from their corresponding images and E^b estimates the blur information from the blurry image b. With the representations disentangled we can proceed with the generation of the sharp image. The generator G_S takes $E^c_B(b)$ and $E^b(b)$ as input and generates the sharp image s_b while G_B takes $E^c_S(s)$ and $E^b(b)$ to generate a blurred image b_s. Then the discriminators are employed to distinguish the samples and the entire architecture is trained in an end-to-end manner.

Let's discuss the different loss functions used by them to ensure proper convergence and performance. The authors use KL divergence loss to regularize the

extracted representations. As we know that sharp images do not have a blur component associated with them, E_S^c tends to be a good content extractor. The last layers of E_B^c and E_S^c share weights so as to guide E_B^c to effectively extract content information.

For E^b to encode only the blur information, the authors propose two methods. In first one, they feed $E^b(b)$ together with $E_S^c(b)$ into G_B to generate b_s. Since b_s is a blurred version of s, the content information of b will be present in a minimal fashion. This discourages $E^b(b)$ to encode content information of b. Second, a KL divergence loss is used to regularize the distribution of the representation. Let the blur features be $z_b = E^b(b)$ and its assumed to be close to the normal distribution $p(z) \sim N(0, 1)$. This is shown in (Bao *et al.*, 2018) to further suppress content information in the blur representation. The KL loss can be expressed as follows,

$$KL(q(z_b)||p(z)) = -\int q(z_b) log \frac{p(z)}{q(z_b)} dz \tag{3.37}$$

Minimizing the KL loss is equivalent to minimizing the following,

$$\mathcal{L}_{KL} = \frac{1}{2} \sum_{i=1}^{N} (\mu_i^2 + \sigma_i^2 - log(\sigma_i^2) - 1) \tag{3.38}$$

where μ, σ are the mean and standard deviation of z_b and N is the dimension of z_b. Here z_b is sampled as $z_b = \mu + z \odot \sigma$ where $p(z) \sim N(0, 1)$ and $\odot$ represents element-wise multiplication.

Adversarial losses are used to make the generated images look more realistic. The adversarial losses for the two discriminators D_S, D_B are shown below.

$$\mathcal{L}_{D_S} = \mathbb{E}_{s \sim p(s)} \left[log D_S(s) \right] + \mathbb{E}_{b \sim p(b)} \left[log(1 - D_S(G_S(E_B^c(b), z_b))) \right] \tag{3.39}$$

$$\mathcal{L}_{D_B} = \mathbb{E}_{b \sim p(b)} \left[log D_B(b) \right] + \mathbb{E}_{s \sim p(s)} \left[log(1 - D_B(G_B(E_S^c(s), z_b))) \right] \tag{3.40}$$

The adversarial loss ensures that the images are generated from a distribution

that is near the true distribution. In order for the images to be visually similar (maintain fidelity) as there is no pairwise supervision, we need cycle consistency loss. The loss ensures that the deblurred image s_b can be re-blurred to reconstruct the original blurred image and the b_s can be translated back to the original sharp domain.

$$s_b = G_S(E_B^c(b), E^b(b)), b_s = G_B(E_S^c(s), E^b(b)) \tag{3.41}$$

$$\hat{b} = G_B(E_S^c(s_b), E^b(b_s)), \hat{s} = G_S(E_B^c(b_s), E^b(b_s)) \tag{3.42}$$

The cycle consistency loss can be formulated as,

$$\mathcal{L}_{cc} = \mathbb{E}_{s\sim p(s)}||s - \hat{s}||_1 + \mathbb{E}_{b\sim p(b)}||b - \hat{b}||_1 \tag{3.43}$$

To further enhance the perceptual quality of the images, they use perceptual loss (Johnson *et al.*, 2016) between deblurred and the originally blurred image which is defined as

$$\mathcal{L}_p = ||\phi_l(s_b) - \phi_l(b)||_2^2 \tag{3.44}$$

where $\phi_l(x)$ is the features of the l-th layer of the pretrained VGG-19 network. The full objective function therefore is the weighted sum of the losses,

$$\mathcal{L} = \lambda_{adv}\mathcal{L}_{adv} + \lambda_{KL}\mathcal{L}_{KL} + \lambda_{cc}\mathcal{L}_{cc} + \lambda_p\mathcal{L}_p + \tag{3.45}$$

where $\mathcal{L}_{adv} = \mathcal{L}_{D_S} + \mathcal{L}_{D_B}$.

3.5 Performance Evaluation

There are several metrics that measure the similarity between the restored image and the ground truth image. Peak Signal to Noise Ratio (PSNR) can be thought of as a reciprocal of MSE which is,

$$PSNR = \frac{m^2}{MSE} \tag{3.46}$$

where m is the maximum possible intensity value since we are using an 8-bit integer to represent a pixel in the channel, $m = 255$.

SSIM estimates the structural similarity between two images and the computation is as follows,

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \tag{3.47}$$

where x, y are the windows of equal dimension for B, I respectively. μ_x, μ_y denotes mean of x, y σ_x, σ_y denotes the variances of x, y respectively. σ_{xy} is the covariance between x and y. c_1, c_2 are constants that are used to stabilize the division.

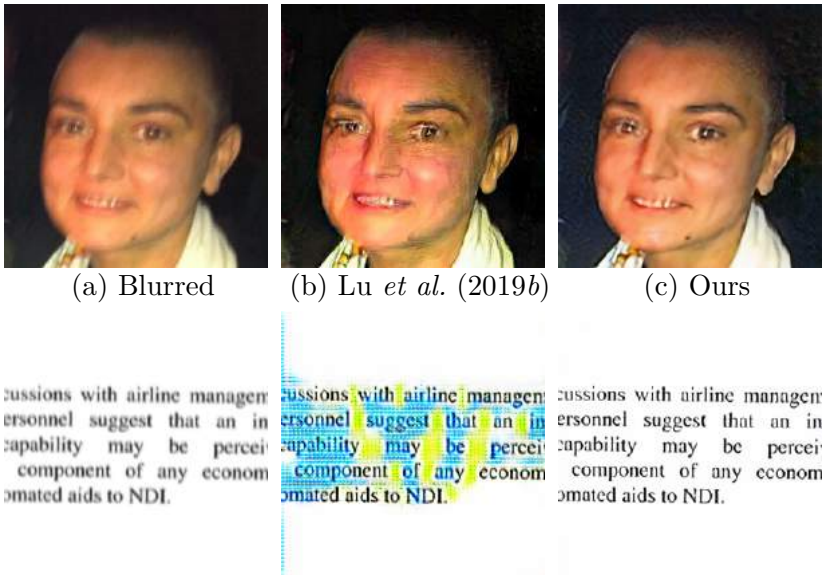
Chapter 4

UNSUPERVISED SCALE ADAPTIVE DEBLURRING

Deep learning has tremendously helped the field of deblurring. Convolutional neural networks (CNN) based supervised methods (Kupyn *et al.*, 2018,?; Nah *et al.*, 2017; Shen *et al.*, 2018; Simonyan and Zisserman, 2014; Purohit and Rajagopalan, 2020; Suin *et al.*, 2021; Nimisha *et al.*, 2017; Vasu *et al.*, 2018) were proposed for the task of deblurring. These algorithms forgo the need to define any priors due to implicit learning of weight parameters during training. The main limitation of these methods is the demand for large amounts of paired training data which is complicated to obtain. Additionally, due to the strong supervision of loss functions during training, these networks incorporate dataset-specific biases which yield sub-optimal performances during deployment.

Unsupervised deblurring was proposed recently to relax the necessity of paired training data. (Madam Nimisha *et al.*, 2018) used generative adversarial networks (GAN) to transfer images from blur domain to sharp domain. An additional re-blurring network and gradient loss was used to maintain fidelity. (Lu *et al.*, 2019*b*) proposed an unsupervised network where blur can be disentangled into an encoder network using KL divergence loss. Methods consider deblurring as an end-to-end problem where GAN loss is used for training at a single scale. As a result, these methods give a suboptimal performance while handling coarse as well as fine details.

We address the above challenges by using a multi-scale architecture with **Scale-Adaptive Attention Module (SAAM)**. Several multi-scale supervised deblurring algorithms have been proposed in the past that use a coarse-to-fine mechanism that takes advantage of processing different scales. These multi-scale methods use supervision loss to guarantee stability during training. In this thesis, we propose



(a) Blurred (b) Lu *et al.* (2019*b*) (c) Ours

(d) Blurred (e) Zhu *et al.* (2017) (f) Ours

Fig. 4.1: Comparison of deblurring results on real blurred images with prior unsupervised mathods. (a) Blurred image from (Lai *et al.*, 2016), (b) result using pretrained model of (Lu *et al.*, 2019*b*) and (c) Our result. (d) is the text image taken from (Hradiš *et al.*, 2015) and (e) is the result of (Zhu *et al.*, 2017) retrained on text datset (Hradiš *et al.*, 2015). (f) Our result.

a multi-scale network for deblurring in an unsupervised setting. Training instability in GANs is well-studied in literature, and several solutions were proposed (Radford *et al.*, 2015). In this approach, instead of cascading the multi-resolution features, we use SAAM to attend to feature maps of lower scales as a function of the present scale. There are many advantages of such a procedure. Firstly, hidden states use information from different scales due to shared parameters. Secondly, the multi-scale approach reduces the training instability problems such as mode collapse and unwanted artefacts in the final image. Lastly, the SAAM module helps select relevant information from the lower scales, further improving the deblurring quality.

Different ablation studies show that the coarse-to-fine mechanism using SAAM gives better deblurring results than end-to-end counterparts devoid of recurrent connections.

Our contributions are summarized below:

We propose an unsupervised deblurring network with multi-scale architecture

and a scale-dependent attention module. Different ablation studies show that scale recurrent networks give superior performance compared to end-to-end methods in an unsupervised setting.

We further show that SAAM facilitates better information flow across different scales, in contrast, to directly cascading or adding feature maps. We further show the efficacy of using SAAM over different attention modules.

We provide extensive comparisons on supervised and unsupervised methods and show that our method performs favourably against supervised and outperforms unsupervised methods qualitatively and quantitatively (on no-reference metrics) when tested on different datasets.

This is a collaborative work done by me and one of my labmates. For completeness, the entire work is mentioned in this report. My specific contributions are mentioned here. I participated in the discussion of multi-scale architecture and was involved in the Scale Adaptive Attention Module ideation. I helped in implementing the model architecture and the attention module in PyTorch. I worked extensively in enabling fast, effective and optimized execution (in both time and memory terms) of these programs. In the experiments section, my contribution lies in running the ablation studies and comparison tests, analysing the results and individual model performance.

4.1 PROPOSED METHOD

Our proposed network, unsupervised scale adaptive attention deblurring network (USAAD), is illustrated in Fig. 4.2, along with the scale-adaptive attention module (SAAM) in Fig. 4.3. Our network architecture is inspired by the recent success of scale recurrent structures in image restoration tasks. Given a blurred image I^b_M, three samples of input image are used for training i.e., I^b_M, $I^b_{M/2}$ and $I^b_{M/4}$ where I^b_p denotes input image downsampled to $p \times p$ dimension. The training mechanism of our algorithm has three steps for every input image. First, at the coarsest scale, generator $G_{B\to S}$ converts $I^b_{M/4}$ from blur to sharp domain using adversarial loss. $G_{B\to S}$ consists of three networks, a encoder network $G^E_{B\to S}$, followed by a series of

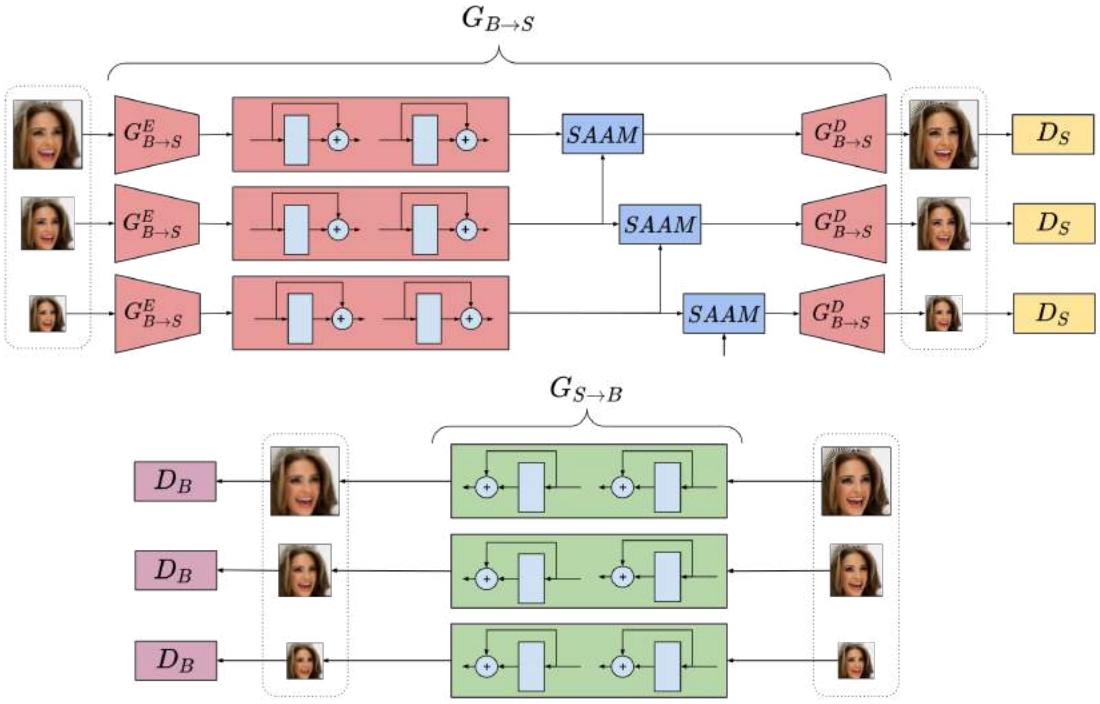


Fig. 4.2: Proposed unsupervised scale adaptive attention deblurring network (US-AAD).

nine residual blocks (He *et al.*, 2016) and decoder network $G^D_{B\to S}$. $G_{S\to B}$ blurs the generated sharp image which is then compared with the input image to maintain the fidelity of contents. The same procedure is followed in the next scale with $I^b_{M/2}$, except that the decoder, $G^D_{B\to S}$, takes the output of SAAM instead of the final residual block. SAAM helps the present scale to use important information from the previous scale to improve deblurring quality (see Sec. 3.1). The same procedure is repeated at the finest scale with I^b_M, and the estimated sharp image is the final restored output. The deblurring mechanism of our method can be represented as

$$\mathcal{I}^i, \mathcal{F}^i = Net_{USAAD}(\mathcal{I}^{i-1}, \mathcal{F}^{i-1}, \mathcal{B}^i; \theta_{USAAD}) \tag{4.1}$$

where i denotes the present scale and $i \in 1, 2, 3$. Inspired by (Nah *et al.*, 2017), we use three resolutions of the input image to train the network and $M = 256$ unless mentioned otherwise. $\mathcal{I}$, $\mathcal{F}$ and $\mathcal{B}$ denote estimated sharp image, output features of last residual block and input blurry image, respectively, and θ denotes learnable parameters of our network. The generator and discriminator networks in our architecture share the same parameters. Although supervised methods

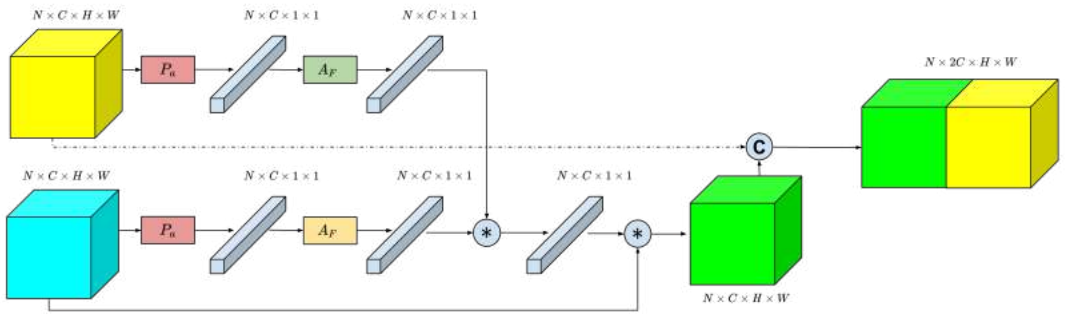


Fig. 4.3: Proposed scale adaptive attention module (SAAM). The cyan block ($N \times C \times H \times W$) is the input feature map from the lower resolution layer, while the yellow block ($N \times C \times H \times W$) is the feature map from the higher resolution layers. P_a refers to the average pooling layer, and A_F is a convolutional network block. Note that there are two independent A_F's each operating on the feature maps of its respective resolution layers (best viewed in colour).

(Zhang *et al.*, 2019*a*; Suin *et al.*, 2020) regress only for residual at each scale, our unsupervised network regress for sharper images to counter training instability due to GANs. Along with the sharing parameters of the network across resolutions, the information across different resolutions is passed effectively using SAAM to improve deblurring quality. The following subsections give a detailed discussion of the SAAM module followed by loss functions used in our model and network architecture.

4.1.1 Scale-adaptive attention module (SAAM)

The objective of SAAM is to use information from the previous scale to improve the deblurring quality at the present scale. A trivial way to achieve this is to directly concatenate or add features from the last residual blocks of both the scales and pass them to the decoder. However, not all the lower-scale features are equally important in improving the deblurring quality. Therefore, concatenating or adding the entire set of lower-scale features can result in sub-optimal performance due to irrelevant channels. Instead of considering each channel equally, SAAM uses both the lower and higher scale feature maps to selectively pay attention to more relevant channels in the lower-scale features. Similar to channel attention (Chen *et al.*, 2017), SAAM can be seen as a process of selecting relevant semantic

attributes.

SAAM takes two feature maps $U_{2X}, U_X \in \mathbb{R}^{N\times C\times H\times W}$, from the last residual block in $G_{B\to S}$ at the present scale and the immediate previous scale, respectively. Here N denotes the batch size, C is the total number of channels and H, W are the height and width of the feature map, respectively. Without loss of generality, we consider $N = 1$ and we represent both the input feature maps as $U = [u^1, u^2, ..., u^C]$, where $u^i \in \mathbb{R}^{H\times W}$ for $\in \{2X, X\}$. We apply mean pooling (P_a) for each channel and get channel vectors for both the feature maps as

$$u_{2X}^M = [\bar{u}_{2X}^1, \bar{u}_{2X}^2, ..., \bar{u}_{2X}^C] \qquad \in \mathbb{R}^C \tag{4.2}$$

$$u_X^M = [\bar{u}_X^1, \bar{u}_X^2, ..., \bar{u}_X^C] \qquad \in \mathbb{R}^C \tag{4.3}$$

where $\bar{u}^i$ is the mean of channel u^i features. The channel vectors u_{2X}^M, u_X^M are passed through convolutional network Φ_{2X} and Φ_X, respectively (denoted as A_F in Fig. 4.3), to obtain the learned scale attention representations v_{2X}, v_X where

$$v = \Phi(u_*^M) \quad \in \{2X, X\} \tag{4.4}$$

The effective channel attention vector $\beta \in \mathbb{R}^C$ is defined as a function of v_{2X} and v_X as follows,

$$\beta = \sigma\left(v_{2X} \times v_X\right) \in \mathbb{R}^C \tag{4.5}$$

where σ denotes the sigmoid function, and $\times$ refers to element wise multiplication. Sigmoid activation is used to normalize the attention weights between 0 and 1 to represent the channel importance. The multiplication of scale attention representations $(v's)$ ensures that the channel representations which are aligned get greater attention than misaligned channels.

Channel attention is applied on U_X by multiplying channel-wise the attention

coefficients β , which can be represented as $U_X^a \in \mathbb{R}^{N\times C\times H\times W}$,

$$U_X^a = \beta \odot U_X \tag{4.6}$$

where $\odot$ refers to channel-wise multiplication. The resultant lower scale feature map is concatenated with the higher resolution feature map U_{2X} along the channel dimension and passed through the decoder. This procedure ensures that lower scale feature information relevant for deblurring is effectively passed on to higher resolution layers.

4.1.2 Loss functions

Given a real blur image (I^b), the generator network $G_{B\to S}$ transfers the image from blur to sharp domain. The output $\hat{I}^s$ of decoder $G^D_{B\to S}$ is used by discriminator D_s to distinguish if the resultant image is sharp or not.

$$\hat{I}^s = G_{B\to S}(I^b)$$

The following loss function is used to optimize both generator $G_{B\to S}$ and discriminator D_s simultaneously

$$\mathbb{L}_{GAN}(G_{B\to S}, D_S) = \mathbb{E}_{I_s\sim p(I_s)}\Big[\log D_S(I_s)\Big] + \mathbb{E}_{I_b\sim p(I_b)}\Big[\log(1 - D_S(G_{B\to S}(I_b)))\Big] \tag{4.7}$$

where $\mathbb{E}$ is the error function, p denotes the data distribution, $I_b \sim p(I_b)$ and $I_s \sim p(I_s)$ denote images sampled from blur and sharp image distributions respectively.

Akin to Eq. 1, the output of decoder $G^D_{S\to B}$ is used by discriminator D_B to distinguish if the resultant image is blurred or not. The loss function used to optimize both generator $G_{S\to B}$ and discriminator D_B simultaneously is

$$\mathbb{L}_{GAN}(G_{S\to B}, D_B) = \mathbb{E}_{I_b\sim p(I_b)}\Big[\log D_B(I_b)\Big] + \mathbb{E}_{I_s\sim p(I_s)}\Big[\log(1 - D_B(G_{S\to B}(I_s)))\Big] \tag{4.8}$$

The above adversarial loss functions are sufficient to generate visually sharp images. However, the estimated sharp image's content need not exactly match that of the input image due to the unavailability of supervised pairs. Inspired by cycleGAN (Zhu *et al.*, 2017), we use cycle consistency loss, where the estimated sharp image is projected into blur domain using $G_{S\to B}$ and compared with the input blur image. The projected blur image can be represented as

$$\hat{I}^b = G_{S\to B}(I^s)$$

The cycle consistency loss function can be defined as

$$\mathbb{L}_{cyc\_b}(G_{B\to S}, G_{S\to B}) = \mathbb{E}_{I_b\sim p(I_b)}\Big[||G_{S\to B}(G_{B\to S}(I_b)) - I_b||_1\Big] \tag{4.9}$$

Similarly, the cycle consistency loss can be applied for the other domain by projecting the estimated blur image to the sharp domain using $G_{B\to S}$ and comparing with the real sharp image. The resultant loss function can be defined as

$$\mathbb{L}_{cyc\_s}(G_{S\to B}, G_{B\to S}) = \mathbb{E}_{I_s\sim p(I_s)}\Big[||G_{B\to S}(G_{S\to B}(I_s)) - I_s||_1\Big] \tag{4.10}$$

These loss functions are calculated at a single scale; however, since our network is trained for n scales, the total loss function can be written as

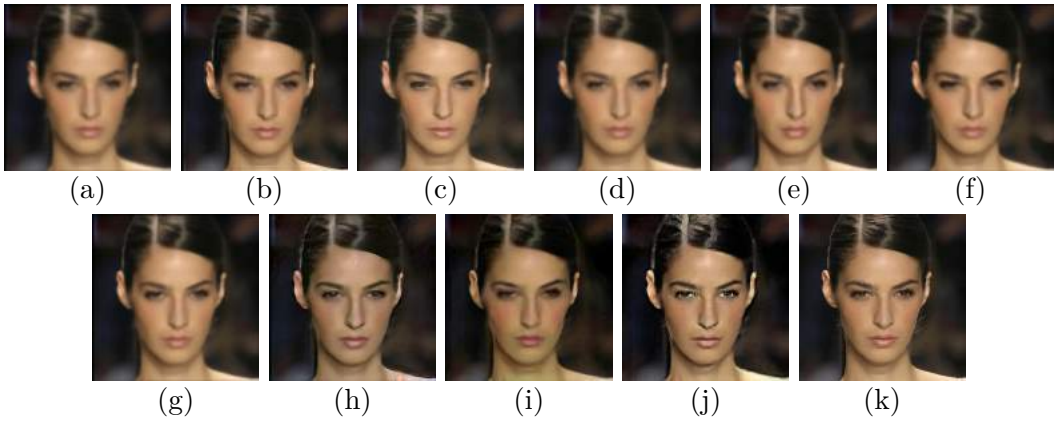
$$
\begin{aligned}
&\mathbb{L}_{Total}(G_{S\to B}, G_{B\to S}, D_S, D_B) \\
&= \sum_{i=1}^{n} \lambda_{adv}\mathbb{L}^{i}_{GAN}(G_{B\to S}, D_S) \\
&\quad + \lambda_{adv}\mathbb{L}^{i}_{GAN}(G_{S\to B}, D_B) \\
&\quad + \lambda_{cyc}\mathbb{L}^{i}_{cyc\_s}(G_{S\to B}, G_{B\to S}) \\
&\quad + \lambda_{cyc}\mathbb{L}^{i}_{cyc\_b}(G_{B\to S}, G_{S\to B})
\end{aligned}
\tag{4.11}
$$

where n is the number of scales the network is trained on. We used $n = 3$ for our model following (Nah *et al.*, 2017). Following (Zhu *et al.*, 2017), the weights for λ_{adv} and λ_{cyc} are set as 1 and 10 respectively. The whole network is trained in a min-max fashion as

$$
\arg \min_{G_{S\to B}, G_{B\to S}} \max_{D_B, D_B} \mathbb{L}_{Total}(G_{S\to B}, G_{B\to S}, D_S, D_B) \tag{4.12}
$$

4.1.3 Network architecture

The encoder network $G^{E}_{B\to S}$ in Fig. 4.2, consists of two convolutional layers with stride two, thus downsampling the input sample by a factor of four. A series of nine residual blocks follow the encoder network. At the coarsest level, the network cannot take features from the previous scale. However, the last residual block features are concatenated with the next level using the SAAM module. Finally, the concatenated features are passed through a decoder network, $G^{D}_{B\to S}$, a mirror representation of the encoder, but deconvolutional layers replace the convolutional layers. The decoder's output is passed through $G_{S\to B}$, which transfers the image from sharp to blur domain. $G_{S\to B}$ is a lightweight network with four convolutional layers using a filter size of 3 and maintaining the same spatial size using padding. Our reason for the simple architecture for $G_{S\to B}$ is to reduce the number of parameters and computational time. Also, the deblurring task is far more complicated than inducing blur into a sharp image. For discriminators D_S and D_B, we use PatchGAN (Isola *et al.*, 2017) to differentiate between real and



(a) (b) (c) (d) (e) (f)

(g) (h) (i) (j) (k)

Fig. 4.4: Visual comparisons with start of the art results on face test dataset (Lee *et al.*, 2020). (a) Blurred (b) Xu *et al.* (2013) (c) Kupyn *et al.* (2018) (d) Kupyn *et al.* (2019) (e) Zhang *et al.* (2019*a*) (f) Nah *et al.* (2017) (g) Suin *et al.* (2020) (h) Zhu *et al.* (2017) (i) Lu *et al.* (2019*b*) (j) Ours (k) Sharp

fake samples.

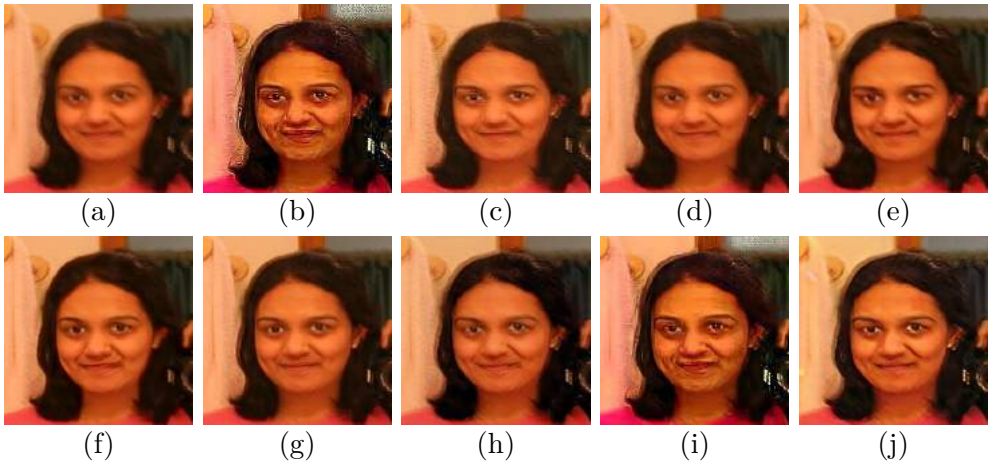
4.2 Experiments

This section is arranged as follows 1. Dataset creation and metrics used 2. Ablation studies, 3. Comparisons on the face and text test sets and 4. Visual comparisons on real face dataset.

4.2.1 Dataset and metrics:

CelebA dataset: We use the face dataset of (Lee *et al.*, 2020) to train our model. (Lee *et al.*, 2020) contains 30K face images and 700 images randomly selected and used as a test dataset for comparisons with state of the art methods. The remaining 29.3K images are grouped into two halves, and the blur model of (Kupyn *et al.*, 2018) is applied to one of the groups keeping the other intact. Thus unsupervised pairs of clean and blur face images are created for training.

Text dataset: We used the text dataset provided by (Hradis *et al.*, 2015) which contains a large collection of 66K blur text images generated using motion and defocus blur. The 66K images are grouped into two halves, with one group containing the sharp images, while the other contains only blur images. The



(a) (b) (c) (d) (e)

(f) (g) (h) (i) (j)

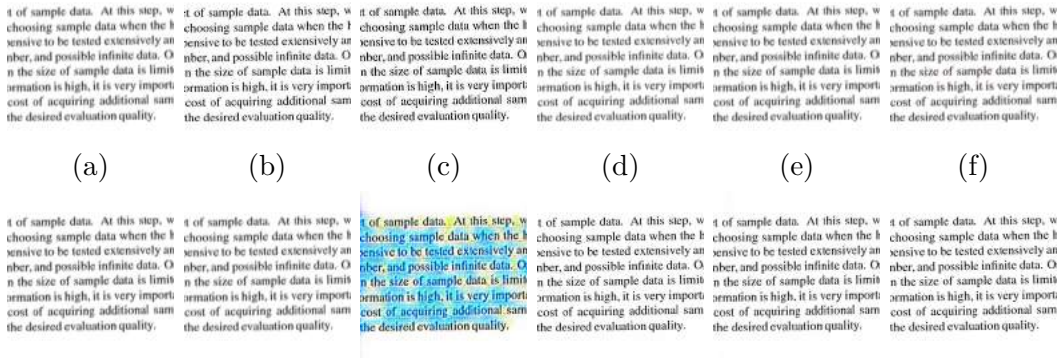
Fig. 4.5: Visual comparisons with start of the art results on real blurred face images of Lai *et al.* (2016). (a) blurred image (b) Xu *et al.* (2013) (c) Kupyn *et al.* (2018) (d) Kupyn *et al.* (2019) (e) Zhang *et al.* (2019*a*) (f) Nah *et al.* (2017) (g) Suin *et al.* (2020) (h) Zhu *et al.* (2017) (i) Lu *et al.* (2019*b*) (j) Ours

dataset is created such that there is no correspondence between the two groups. Since the images are already blurred, we did not apply any blur model, and the above dataset is used for training. We used a separate test dataset provided by (Hradis *et al.*, 2015) to compare with competing methods.

Table 4.1: Quantitative comparisons of different ablation studies of our model on the face dataset. Scales indicate the number of resolutions the network was trained on. $A.F$ and $C.F$ indicate that feature maps across the resolution are added and concatenated respectively, while $C.A$ and $S.A$ indicate channel Hu *et al.* (2018) and spatial attentionWoo *et al.* (2018) respectively.

Design	Scales	$A.F$	$C.F$	$C.A$	$S.A$	SAAM	brisque
Net1	1	✗	✗	✗	✗	✗	32.89
Net2	2	✗	✗	✗	✗	✗	31.29
Net3	3	✗	✗	✗	✗	✗	30.34
Net4	3	✓	✗	✗	✗	✗	33.53
Net5	3	✗	✓	✗	✗	✗	30.21
Net6	3	✗	✓	✓	✗	✗	29.52
Net7	3	✗	✓	✗	✓	✗	27.38
Net8	3	✗	✓	✗	✗	✓	**25.52**

We used PSNR, NIQE and BRISQUE to provide quantitative comparisons with state of the art results. While PSNR requires ground truth or reference image, NIQE and BRISQUE do not require any reference image and can be calculated



(a) (b) (c) (d) (e) (f)

(g) (h) (i) (j) (k) (l)

Fig. 4.6: Visual comparisons with start of the art results on text dataset (Hradiš *et al.*, 2015). (a) Blurred (b) Xu *et al.* (2013) (c) Pan *et al.* (2014*b*) (d) Kupyn *et al.* (2018) (e) Kupyn *et al.* (2019) (f) Zhang *et al.* (2019*a*) (g) Nah *et al.* (2017) (h) Suin *et al.* (2020) (i) Zhu *et al.* (2017) (j) Lu *et al.* (2019*b*) (k) Ours (l) Sharp

given a single image. A brief discussion of BRISQUE and NIQE is given below.

BRISQUE (Mittal *et al.*, 2012) stands for Blind/Referenceless Image Spatial Quality Evaluator. BRISQUE uses scene statistics instead of distortion stats to calculate the naturalness of the given image. The low computational capacity of BRISQUE makes it well-suited for real-world applications. A lower BRISQUE score on an image indicates good perceptual quality, and its values range between 1-100.

PIQE (Venkatanath *et al.*, 2015) stands for Perception-based Image Quality Evaluator. PIQE is a no-reference image metric that calculates the distortion present in the image based on block-level characteristics. PIQE estimates the quality of the image from perceptually significant portions rather than the whole image. Similar to BRISQUE, a lower score of PIQE indicates a better perceptual score, and its value ranges between 1-100.

4.2.2 Ablation studies

Since ours is the first take on using multi-scale architecture for image restoration in unsupervised settings, we first show that multi-scale helps to improve the deblurring quality and training stability of GANs compared to *end-to-end* methods (Table 4.1, row 2 to row 4). To further improve the deblurring quality, we pro-

pose to employ SAAM. We further show that information flow across different resolutions during training is better attended by the proposed SAAM block than standard attention modules Hu *et al.* (2018); Woo *et al.* (2018) used in literature. (Table 4.1, row 5 to row 9).

Net1 (Table 4.1, row 1 and Fig. 4.7 (b)): This network is the same as described in Section 3.3, except that it is trained for only a single resolution (256x256). From now, this network is defined as a base network. The loss used to train the network is Eq. (5) with $i \in 1$. Although the network deblurs the image, there are visible artefacts and several colour changes compared to the input image.

Net2 (Table 4.1, row 2 and Fig. 4.7 (c)): The base model is trained with two scales using loss function in Eq. (5) with $i \in 1, 2$, i.e. two resolutions of the input image are used to train the network. The artefacts are visibly reduced compared to Net1, although the deblurring quality remains the same.

Net3 (Table 4.1, row 3 and Fig. 4.7 (d)): Like Net2, the base network is trained with three scales using loss function in Eq. (5) with $i \in 1, 2, 3$, i.e. the network is trained with three resolutions of the input image. As can be seen, the estimated sharp images are free of artefacts. However, the deblurring quality has only minor improvements compared to Net1.

It can be inferred from the above ablation studies that multi-scale training helps to stabilize GAN training. However, for both Net2 and Net3, there are no intermediate connections across different scales while training. We used different attention mechanisms for relevant information flow across resolutions to improve deblurring quality. For all the models below, the network is trained for three scales.

Net4 (Table 4.1, row 4 and Fig. 4.7 (e)): Features maps of last residual blocks in the present and previous scale are added during training, i.e. instead of using SAAM in Fig. 4.3, feature maps are directly added and passed through decoder $G^D_{B \to S}$. Interestingly the deblurring quality is reduced compared to Net3. We reason that the dip in performance is due to the loss of information while adding feature maps across resolutions.

Net5 (Table 4.1, row 5 and Fig. 4.7 (f)): Inspired by Wang *et al.* (2018), fea-

(a) (b) (c) (d) (e) (f) (g) (h) (i) (j)

Fig. 4.7: Ablation study. (a) input blurry image and (j) is the sharp image. (b-i) are the resultant images of Net1-Net8. See section for detailed explanation section 4.2.2

ture maps across different scales are concatenated. As can be seen, the deblurring quality improves compared with Net3. Different cues from lower resolutions are aiding to improve deblurring quality.

Net6 (Table 4.1, row 6 and Fig. 4.7 (g)): Channel attention Hu *et al.* (2018) is applied on the feature maps from the last residual block in the previous scale and are concatenated with present scale ones. The deblurring quality of Net6 improves compared to Net5 due to the attention on feature maps of a lower scale.

Table 4.2: Quantitative comparisons with state of the art methods on the face and text dataset.

Method	Face datset			Text dataset		
	brisque	*piqe*	PSNR	*brisque*	*piqe*	PSNR
Pan *et al.* (2014*b*)	✗	✗	✗	42.35	76.06	17.04
Xu *et al.* (2013)	36.82	55.41	18.07	45.15	77.87	15.30
Kupyn *et al.* (2018)	43.54	57.32	18.61	47.34	80.43	17.67
Kupyn *et al.* (2019)	44.36	57.78	**19.34**	46.58	80.76	17.90
Zhang *et al.* (2019*a*)	48.25	71.0	19.00	43.92	76.23	17.48
Nah *et al.* (2017)	47.88	77.73	18.62	46.69	81.33	17.84
Suin *et al.* (2020)	44.77	66.09	19.21	46.46	81.74	**18.97**
Zhu *et al.* (2017)	31.07	42.83	18.68	48.32	80.32	14.56
Lu *et al.* (2019*b*)	29.97	45.03	19.05	47.19	79.94	18.49
Ours	**25.52**	**35.93**	19.24	**39.64**	**74.05**	18.68

Net7: (Table 4.1, row 7 and Fig. 4.7 (h)): Similar to Net6, spatial attention Woo *et al.* (2018) is applied instead of channel attention on previous scale features and the resultant concatenated features are passed through the decoder. As observed, due to pixel-wise attention, the previous scale feature maps are better attended, further helping the deblurring quality.

Net8: (Table 4.1, row 8 and Fig. 4.7 (i)): For both Net6 and Net7, the

feature maps of the present scale do not play any role in attending to previous scale features. Motivated by this, we propose to use SAAM. SAAM attends to feature maps from the previous scale as a function of the present scale. As can be seen, Net8 gives improved deblurring performance compared to previous networks.

4.2.3 Competing methods

The results of our model are compared with conventional methods (Pan *et al.*, 2014*b*; Xu *et al.*, 2013), supervised methods (Zhang *et al.*, 2019*a*; Kupyn *et al.*, 2018, 2019; Nah *et al.*, 2017; Suin *et al.*, 2020) and unsupervised methods (Zhu *et al.*, 2017; Lu *et al.*, 2019*b*). Among conventional methods, (Pan *et al.*, 2014*b*) is a text deblurring method, while (Xu *et al.*, 2013) is a generic deblurring algorithm. In CNN based methods, (Lu *et al.*, 2019*b*; Zhu *et al.*, 2017) are domain-specific methods and (Suin *et al.*, 2020; Zhang *et al.*, 2019*a*; Kupyn *et al.*, 2019, 2018) are natural scene deblurring methods. For conventional methods, we ran the codes with default parameters provided by authors, while for CNN methods, we used the pretrained models provided by authors except for CycleGAN (Zhu *et al.*, 2017). We used the official code provided by authors to retrain the CycleGAN(Zhu *et al.*, 2017) on the face and text training datasets.

4.2.4 Comparisons

Test dataset results: Fig. 4.4 and Fig. 4.6 shows visual comparisons, while Table 4.2 illustrates quantitative comparisons with competing methods on the faces and text test set (described in Sec. 4.1). Our method outperforms conventional and unsupervised methods on all three metrics. Compared with supervised methods, our method performs comparably on the PSNR metric while giving superior performance on no-reference metrics. From Fig. 4.4 and 4.6, we can see that Xu *et al.* (2013) over blurs the image at specific regions and neglects the other portions, while the deblurring quality is poor in Kupyn *et al.* (2019, 2018). Among supervised methods, Zhang *et al.* (2019*a*); Suin *et al.* (2021); Nah *et al.* (2017) gives comparably good results due to recurrent structure but fails to deblur specific

portions. In unsupervised methods, CycleGANZhu *et al.* (2017) induces artifacts in the restored image (Fig. 4.6 (i)) while Lu *et al.* (2019*b*) fails to properly recover the latent image when encountered by complex blur (Fig. 4.4 (i).

Real dataset results: We cropped nine face images from the real world blurry images provided by Lai *et al.* (2016) and the corresponding visual comparisons are shown in Fig. 4.5. Consistent with test dataset results, Xu *et al.* (2013) tends to over blur some portions of the image while Kupyn *et al.* (2019, 2018) leave most of the portions to remain blurred. Zhang *et al.* (2019*a*); Suin *et al.* (2021); Nah *et al.* (2017) gives good results on the first image due to scale recurrent nature; however, some second image portions remain blurred. In unsupervised methods, Zhu *et al.* (2017) fails to recover the clean domain while Lu *et al.* (2019*b*) struggles to restore the clean image when a large amount of blur is present. Compared to the above methods, our methods give superior performance while handling blurred faces of the test dataset and real-world face images.

Chapter 5

FOURIER ATTENTION

5.1 Introduction

In this chapter, we propose the Fourier Attention module a simple and effective attention module for convolutional networks. It's a general-purpose module that can be present anywhere in the neural network preferably in the intermediate processing layers. Attention modules aim to increase the representation power of the network by enabling the network to focus on selective areas and ensure that information is best utilized to boost the performance of the network. Attention can be applied in various settings like in LSTM networks, CNN networks and various tasks such as image classification, image segmentation and language generation (no means an exhaustive list) have benefited from them. (Vaswani *et al.*, 2017) has introduced transformers that revolutionized Natural language processing, is build completely on an attention network and it has replaced the dominance of LSTM in this area. In Natural Language Generation (NLG) and NLP, transformers are the state of the art models today. The computer vision community have also used attention to boost performance for various tasks including image restoration.

Importance of attention is a well studied subject in deep learning (Mnih *et al.*, 2014; Ba *et al.*, 2014; Bahdanau *et al.*, 2014; Xu *et al.*, 2015; Jaderberg *et al.*, 2015). The main theme of attention modules is to streamline the information flow by enabling the network to highlight the relevant features and suppress the irrelevant and trivial features that have largely contributed to improved performance across various tasks as shown by many works. Fourier attention module unlike the other attention modules proposed in the literature enables the network to attend to specific frequency spectrum and reject trivial and redundant frequencies. Some recent works aim to attempt solve problems in this direction like (Xu *et al.*, 2020). We take FFT to obtain the frequency spectrum and learn a filter to selectively

amplify and suppress the frequency spectrum which can be trained in an end to end manner. Cubic spline interpolation is used to ensure that the filter behaves in a smooth manner and behaves as a regularizer. Though this attention module can be applied to any task, we apply it in the generic scene single image blind deblurring task to investigate its performance. Our main contributions are described as follows,

We propose a general-purpose task agnostic attention module, Fourier Attention module that can be used to increase the representation power of CNN networks and be applied to a large basket of tasks.

We analyse the effectiveness of the attention module through extensive ablation studies.

We investigate the performance of the proposed Fourier attention on blind single image deblurring problem with GoPro dataset (Nah *et al.*, 2017).

This is a collaborative work done by me and one of my labmates. For completeness, the entire work is mentioned in this report. My specific contributions are mentioned here. I participated in formulating the Fourier attention. I was actively engaged in drafting different key aspects of the Fourier attention pipeline such as radial allocation filter generation scheme, interpolation module, and reducing run-time. I ran experiments with different interpolation nodes and participated in analysing the performance of the unsupervised generic scene deblurring task.

5.2 Related Work

A lot of attention mechanisms have been proposed and in this section we will particularly look into channel attention (Hu *et al.*, 2018), spatial attention (Woo *et al.*, 2018), pixel attention (Zhao *et al.*, 2020) and Laplacian attention (Anwar and Barnes, 2020). Channel attention (Hu *et al.*, 2018) which they call as Squeeze and Excitation network address the fundamental architectural unit in CNNs, which is the channel relationship in the network. Each kernel convolves the input feature map and produces an output feature map. Multiple kernel outputs are concatenated and are sent down for further processing by the convolutional layers. In this

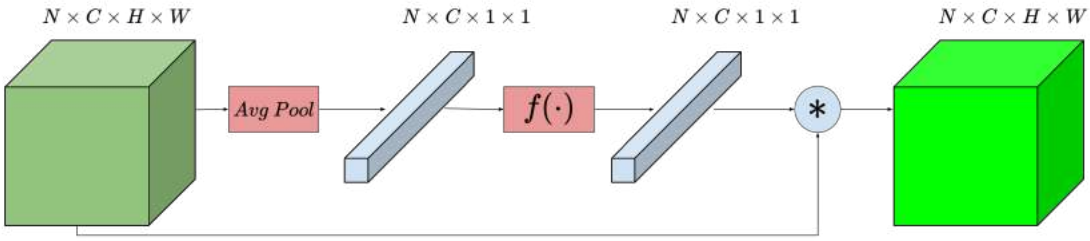


Fig. 5.1: Architecture Diagram of the Channel Attention. $f(\cdot)$ is the operation described in Eq. (5.2). (Woo *et al.*, 2018)

routine setup, there is an inherent assumption of channel independence as channels aren't weighted according to their importance. This work proposes an architectural unit which is termed as a "Squeeze-and-Excitation" block that re-calibrates channel-wise feature responses by explicitly modelling inter-dependencies between channels. The incoming feature map and a global average pooling are taken to obtain global receptive contextual information of each channel. This can be represented as,

$$z_c = P_a(u_c) = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} u_c(i, j) \tag{5.1}$$

After the information aggregation phase, the second operation aims to capture the channel-wise dependencies. The authors use a fully connected network to model the channel dependencies and it can be formulated as,

$$s = \sigma(W_2 \delta(W_1 z)) \tag{5.2}$$

where δ is the ReLU function, $W_1 \in \mathbb{R}^{\frac{C}{r} \times C}$ and $W_2 \in \mathbb{R}^{C \times \frac{C}{r}}$ where r is a hyperparameter which decides the complexity of the attention block. Excitation operation follows this where each channel is multiplied with the corresponding excitation obtained in Eq. (5.2). The final attended feature map is given by,

$$\bar{u}_c = s_c u_c \tag{5.3}$$

By multiplying s_c along the channels, the network can selectively focus on some kernels and learn to ignore some kernels feature maps.

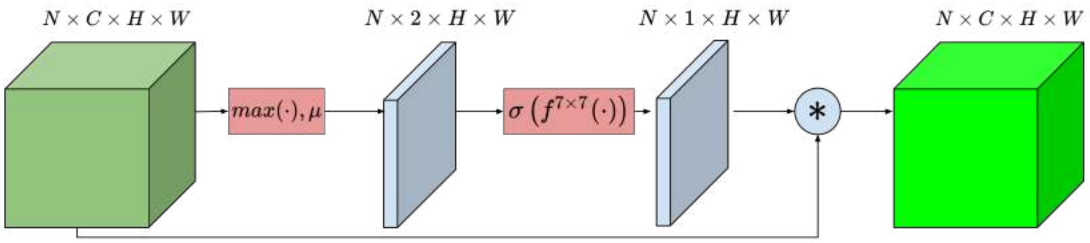


Fig. 5.2: Architecture Diagram of the Spatial Attention (Woo *et al.*, 2018)

Spatial attention (Woo *et al.*, 2018) aims to create a spatial attention map using the inter-spatial relationship of features. Schematic diagram of spatial attention is shown in Fig. 5.2. Different from channel attention, spatial attention focuses on the spatial location of the informative part. To compute spatial attention, we first compute the average pooling and max pooling outputs of the feature map along the channel dimension and concatenate them along the channel axis to obtain representation statistic of the spatial locations. On the concatenated feature output, we apply a convolutional layer to obtain the spatial attention map $M_s(F) \in \mathbb{R}^{H\times W}$ which encodes where to emphasis or suppress. This attention map is multiplied to input feature map to streamline the information flow.

The spatial attention can be expressed as,

$$M_s(F) = \sigma(f^{7\times7}([AvgPool(F), MaxPool(F)]) \tag{5.4}$$

$$= \sigma(f^{7\times7}([F^s_{avg}; F^s_{max}])) \tag{5.5}$$

where σ denotes the sigmoid function and $f^{7\times7}$ represents a convolutional operation with a filter size of 7×7.

(Zhao *et al.*, 2020) proposes efficient image super-resolution using pixel attention. Pixel attention is similar to channel and spatial attention in the formulation. Pixel attention produces a 3D attention map instead of a 1D (channel attention) or 2D (spatial attention) attention maps. This attention scheme enabled the performance in the case of image super-resolution to increase.

Pixel attention is a 1×1 convolutional layer with a sigmoid activation func-

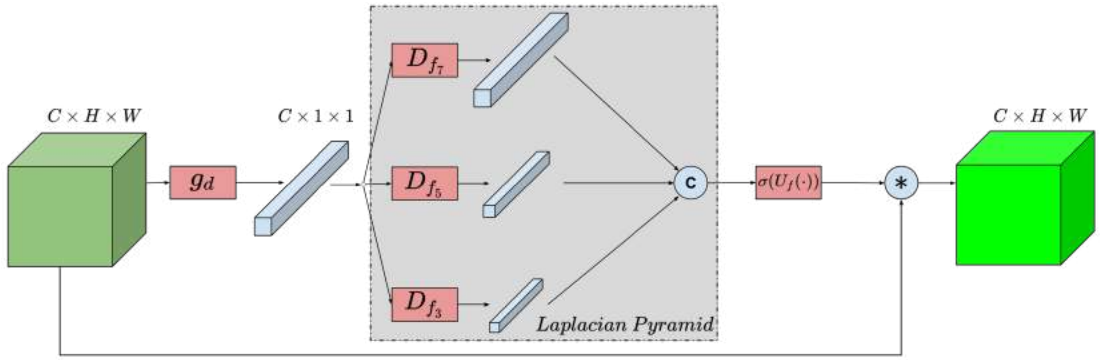


Fig. 5.3: Architecture Diagram of the Laplacian Attention (Anwar and Barnes, 2020)

tion to compute the attention coefficients. The point to note that is the dimension of the attention map is exactly equal to the input feature map. So, if $F \in \mathbb{C} \times \mathbb{H} \times \mathbb{W}$, the pixel attention operation can be represented as,

$$M_p(F) = f_{PA}(F) \cdot F \tag{5.6}$$

where $M_p(F)$ is the processed feature map and $f_{PA}(\cdot)$ is a 1×1 convolutional layer followed by a sigmoid function.

(Anwar and Barnes, 2020) introduced the Laplacian attention again in the context of image Super-resolution. Laplacian attention is pyramid-level attention to model the features non-linearly. The Laplacian attention weights the residual features at different sub-frequency bands. Schematic representation of Laplacian Attention is shown in Fig. 5.3. To produce attention differently at the Laplacian pyramids, the authors use a global descriptor to capture the statistics of the entire image. Laplacian attention weights the sub-band frequencies of high importance progressively to exploit the relationship between the features. The global feature descriptor used here is global average pooling.

$$g_d = \frac{1}{H \times W} \sum_{i=1^h} \sum_{j=1}^{w} f_c(i, j) \tag{5.7}$$

where $F_c(i, j)$ is the value at the position (i, j) in the feature map.

To capture the channel relationship, the authors use a gated appraoch. To implement gateing formally, they use a series of parallel feature reduction operators

(dilated convolution layers) and non linear activation functions (ReLu, Sigmoid). This can be expressed as,

$$r_3 = \tau(D_{f_6}(g_d)) \tag{5.8}$$

$$r_5 = \tau(D_{f_5}(g_d)) \tag{5.9}$$

$$r_7 = \tau(D_{f_7}(g_d)) \tag{5.10}$$

$$\tag{5.11}$$

where D denotes the dialated convolution operator, τ represents the ReLu function. These multi-level representations r_3, r_5, r_7 are concatenated to obtain the global descriptor denoted as g_p.

$$g_p = [r_3; r_5; r_7] \tag{5.12}$$

To regain the dimension lost due to dilated convolution operators, they use an upsampling operator U_f followed by sigmoid activation σ as shown.

$$L_p = \sigma(U_f(g_p)) \tag{5.13}$$

This learned statistic of Laplacian attention is utilized by rescaling the feature map.

$$\hat{f_c} = L_p \times f_c \tag{5.14}$$

5.3 Our Model

5.3.1 Network Architecture

The encoder network $G^E_{B\to S}$ in Fig. 5.4, consists of two convolutional layers with stride two, thus downsampling the input sample by a factor of four. A series of nine residual blocks follow the encoder network. In each residual block, we have a Fourier attention module. Finally, the concatenated features are passed

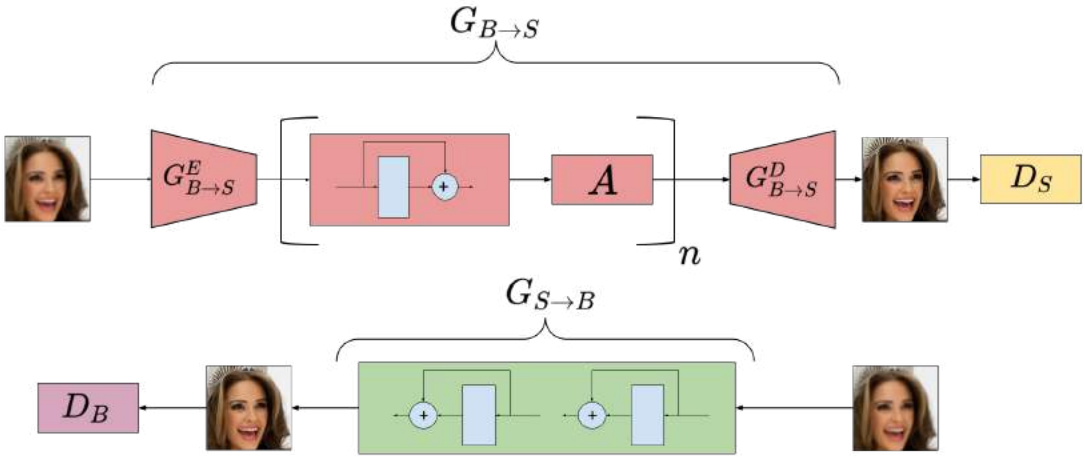


Fig. 5.4: Architecture diagram of the model for unsupervised deblurring. A denotes the Fourier attention module.

through a decoder network, $G^D_{B\to S}$, a mirror representation of the encoder, but deconvolutional layers replace the convolutional layers. The decoder's output is passed through $G_{S\to B}$, which transfers the image from sharp to blur domain. $G_{S\to B}$ is a lightweight network with four convolutional layers using a filter size of 3 and maintaining the same spatial size using padding. Our reason for the simple architecture for $G_{S\to B}$ is to reduce the number of parameters and computational time. Also, the deblurring task is far more complicated than inducing blur into a sharp image. For discriminators D_S and D_B, we use PatchGAN Isola *et al.* (2017) to differentiate between real and fake samples.

$$\mathcal{I} = Net_{UFAN}(\mathcal{B}; \theta_{UFAN}) \tag{5.15}$$

where $\mathcal{I}, \mathcal{B}$ and θ_{UFAN} represent the estimated latent sharp image, input blurry image and the learnable parameters in the network. In the next section, we will look in detail as to how the Fourier attention module architecture.

5.3.2 Fourier Attention

The objective of Fourier attention as stated before is to use information in the frequency spectrum of the feature maps to improve the deblurring quality of the estimated sharp image. This entire effort hinges on the premise that not all the frequencies are equally important and some need more focus than others. So as a

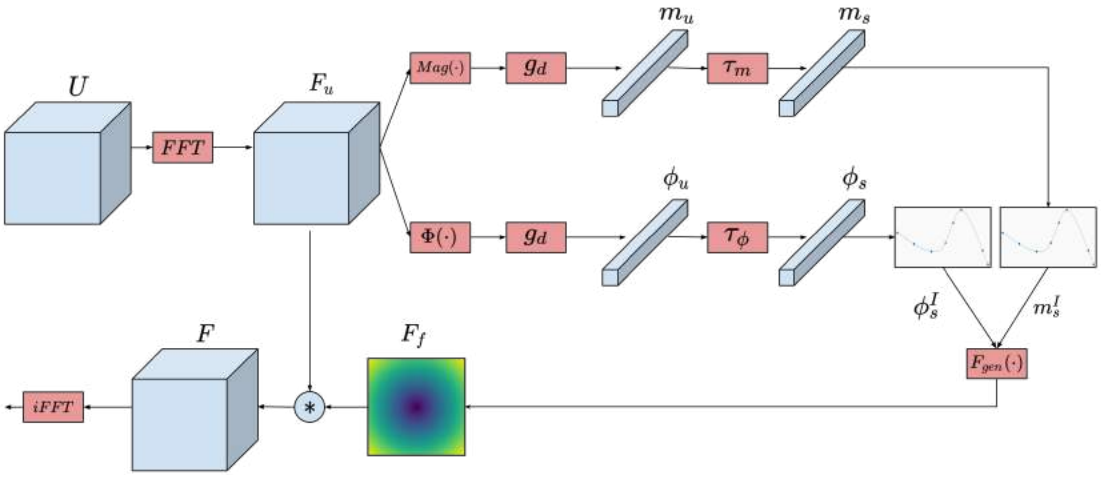


Fig. 5.5: Fourier attention architecture diagram. $F_{gen}(\cdot)$ represents the 2D filter generation stage of the module

first step, the Fourier attention module computes the 2D Fourier transform of the feature map. The architectural diagram of Fourier attention is shown in Fig. 5.5.

2D Discrete Fourier Transform

Let's take a feature block U, denoted as $f_u(x, y)$ with size $A \times B$, the DFT is computed according to the following expression,

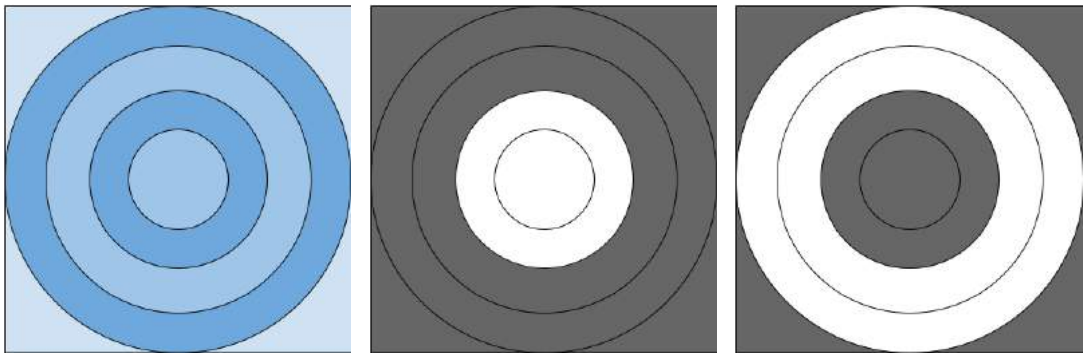
$$F_u(i,j) = \sum_{x=0}^{A-1} \sum_{y=0}^{B-1} f_i(x,y) exp\left(-j2\pi\left(\frac{ix}{A} + \frac{jy}{B}\right)\right) \tag{5.16}$$

Note that we are computing the DFT for each channel of dimension $H \times W$ for a feature map of dimension $u \in \mathbb{R}^{N \times C \times H \times W}$. Since DFT yields a matrix of complex numbers, we obtain the magnitude and phase of the spectrum according to the expression below,

$$M_u(i,j) = \sqrt{Real(F_u(i,j))^2 + Imag(F_u(i,j))^2} \tag{5.17}$$

$$\Phi_u(i,j) = tan^{-1}\left(\frac{Imag(F_u(i,j))}{Real(F_u(i,j))}\right) \tag{5.18}$$

In the DFT domain, the frequencies radially vary from low frequencies in the centre of the spectrum to high frequencies in the borders of the spectrum.



(a) DFT Frequency Bands (b) Low Pass Filter (c) Band Pass Filter

Fig. 5.6: The radial arrangement of frequencies in DFT is shown in Fig. 5.6 (a). The lower and of the frequency spectrum occupy the smaller radial bands, and as the radius increases, the frequency also increases. (b) shows a 2D low pass filter where only the inner radial bands are non zero and others are blocked. (c) is a band pass filter as only an intermediate band of frequencies are allowed. The black circular demarcations are shown for illustration purposes only.

Considering this arrangement, the filtering is done commonly in a radial way as shown in Fig. 5.6. The width of the frequency bands influences the number of filter weights that are needed to be estimated by the network. A detailed explanation of DFT is given in Appendix A.

Frequency Filter Weights Estimation

In this step, the filter weights will be learnt and the frequency-filter will be constructed. Let g_d represent the global average pooling layer. We subject both the magnitude and the phase of the frequency spectrum to a global average pooling layer to obtain global receptive contextual information of each channel in both the magnitude and phase spectrum. The global average pooling is defined in Eq. 5.7. This operation can be denoted as,

$$m_u = g_d(M_u) \tag{5.19}$$

$$\phi_u = g_d(\Phi_u) \tag{5.20}$$

where $m_u, \phi_u \in \mathbb{R}^{N \times C}$. These channel descriptors are subject independently to

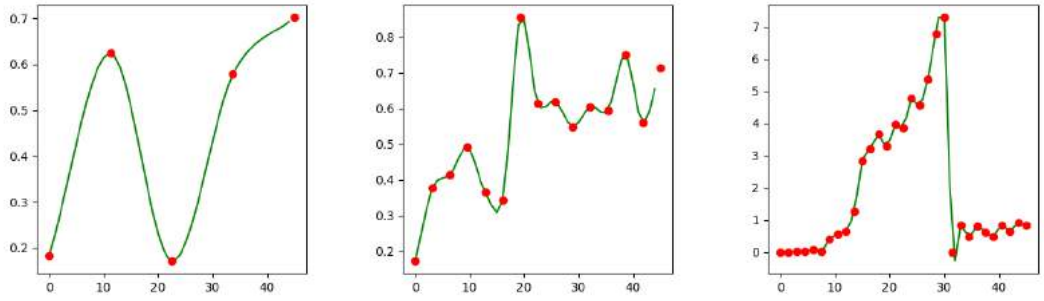
a series of 1×1 convolutions to obtain a vector of dimension magnitude and phase samples of the filter which will serve as the attention map in the frequency domain. Let the two convolutional networks processing the m_u and ϕ_u to obtain the final filter's magnitude and phase samples be denoted as τ_m, τ_ϕ. The filter we learn would enable the network to selectively focus on certain frequencies and reject the irrelevant frequencies. We use a learnable way to estimate the weights of the complex filter which will be used in the attention block. We sample a fixed number of points both in the magnitude and phase spectrum. Let the filter magnitude and phase be represented as $M_f, \Phi_f \in \mathbb{R}^{N \times H \times W}$. From the previous section, we know that DFT frequency arrangements are radial in nature. We discretize the filter by predicting the filter magnitude and phase along with the circular blocks in the frequency domain. As the magnitude and phase of the filter are constant radially, we predict the values of the magnitude and the phase of the filter at integer radii from the centre.

The dimensions of the filter depend on the input blurry image dimensions and this can vary during inference time. This necessarily means that the number of discretized magnitude and phase points of the filter to be estimated varies with the inference image dimensions. To mitigate this problem, we estimate a fixed number of samples ($\psi \in \mathbb{Z}^+$) of magnitudes and phase responses of the filter along with equidistant radial location from the networks τ_m, τ_ϕ. We later interpolate values for all integer radii from these samples. The exact radii where the samples are estimated is given below.

$$d = \left\lfloor 0.5\sqrt{H^2 + W^2} \right\rfloor \tag{5.21}$$

$$r_i = \left(\frac{d}{\psi - 1} \right) i \quad \forall i \in \{0, 1, \ldots, \psi - 1\} \tag{5.22}$$

where d is the integer part of the semi-diagonal distance of the input feature map u and the values $r_i \quad i \in \{0, 1, \ldots, \psi - 1\}$ denotes the radii where the convolutional networks τ_m, τ_ϕ estimates the magnitude and phase response of the final filter. We use cubic spline interpolation to interpolate the magnitude and



(a) $\psi = 5$ (b) $\psi = 15$ (c) $\psi = 31$

Fig. 5.7: Sample magnitude interpolated spectrum shown for different nodes. Note that the blue dots represent the samples from the CNN τ_m, and the green line is the cubic spline interpolation.

phase spectrum from the CNN generated samples. To formulate, let $m_s, \phi_s \in \mathbb{R}^{\psi}$ represent the samples estimated by CNN τ_m and τ_ϕ.

$$m_s = \tau_m(m_u) \tag{5.23}$$

$$\phi_s = \tau_m(\phi_u) \tag{5.24}$$

We interpolate the values for all integer radii from 0 to d (defined in Eq. (5.22)) so that we can cover till the corners of the filter in our radial filter generation scheme. The cubic spline interpolation operator can be expressed as,

$$m_s^I = \text{Spline}(r, m_s) \tag{5.25}$$

$$\phi_s^I = \text{Spline}(r, \phi_s) \tag{5.26}$$

where $m_s^I, \phi_s^I \in \mathbb{R}^d$ are the interpolated magnitude and phase spectrum and $r = [r_0, r_1, \ldots, r_{\psi-1}] \in \mathbb{R}^{\psi}$. In Fig. 5.10 we have shown some sample m_s^I obtained in our experiments.

Frequency Filter Generation

The next step is to generate the complex filter F_f from the interpolated magnitude and phase spectrum samples m_s^I and ϕ_s^I. We propose a radial grouping scheme in accordance with radial arrangement of frequency bands in DFT. There are no learnable parameters in this part of the module. From the interpolated 1D vectors we create a 2D filter matrix using which the filtering will be done on the feature maps. Let $m_f, \phi_f \in \mathbb{R}^{N\times H\times W}$ denote the magnitude and the phase of the filter F_f. We follow a radial filter generation scheme expressed below as,

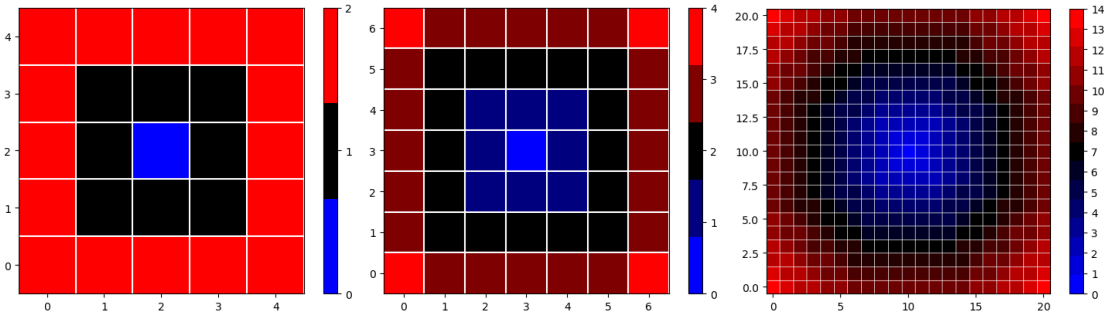
$$m_f(i,j) = m_s^I[rd(i,j)] \tag{5.27}$$

$$\phi_f(i,j) = \phi_s^I[rd(i,j)] \tag{5.28}$$

$[\cdot]$ operator here denotes the 1D array indexing as in many popular programming languages and $rd(i,j) \in \mathbb{Z}^+$ computes the integral part of the radial distance of the point (i,j) from the center of the filter $(H/2, W/2)$ which is,

$$rd(i,j) = \left\lfloor \sqrt{\left(i - \frac{H}{2}\right)^2 + \left(j - \frac{W}{2}\right)^2} \right\rfloor \in \mathbb{Z}^+ \tag{5.29}$$

In Fig. 5.8, we demonstrate this scheme by taking a representative vector of non negative consecutive integers $\zeta \in \mathbb{Z}^+, (\zeta = [0, 1, \ldots])$ and constructing the filter of different dimensions to illustrate scheme listed in Eq. 5.28. The circular contours aren't profound in Fig 5.8 (a) as the dimension of the filter is small (5×5). The arrangement takes a square form as shown. But as the filter dimension is increased the radial arrangement of entries in ζ is clearly visible as in Fig. 5.8 (c). In Fig. 5.8 (b) we can see the transition of square arrangement of ζ entries to a smoother radial arrangement which is more clearly visible as the filter size is increased. This arrangement is different from the Chebyshev distance grouping suggested by (Stuchi *et al.*, 2020). This radial grouping scheme proposed is approximated by (Stuchi *et al.*, 2020) for small dimension filters, but significant deviation can be reported at higher dimensions as in Fig . 5.8 (b), (c)



(a) 5×5 Filter (b) 7×7 Filter (c) 21×21 Filter

Fig. 5.8: Sample Filters generated via our scheme explained in Eq. (5.28). The values taken are shown in the colorbar adjacent to each image. Each unit in the color bar corresponds to filter parameter estimated from the spline interpolation on the CNN predictions in the Fourier attention modules. In (a) there are three distinct values present which correspond to three distinct weights (each for magnitude and phase spectrum) parameters which correspond to radius $r = 0, 1, 2$. The representative vector m_s^I is shown in the colorbar in these images.

The advantage in terms of reduction in the number of parameters to be estimated via this scheme is shown in Fig. 5.9.

Frequency Filtering

The complex FFT of the feature map F_u and the estimated complex filter F_f are multiplied following the rules of complex number multiplication. This ensures that the regions in the frequency spectrum that the network wants to attend to are indeed amplified and the trivial and irrelevant frequencies are suppressed. Note that the filter $F_f \in \mathbb{C}^{N \times H \times W}$ can be constructed from the 2D magnitude and the phase response calculated in the radial grouping scheme. This 2D filter is applied along the channels of the input feature map's Fourier transform which can be represented as,

$$F = F_u \odot F_f \tag{5.30}$$

where $\odot$ refers to the element-wise multiplication by broadcasting along the channel dimension. Then inverse FFT is taken so that we can obtain the representation of the processed representation in the feature domain.

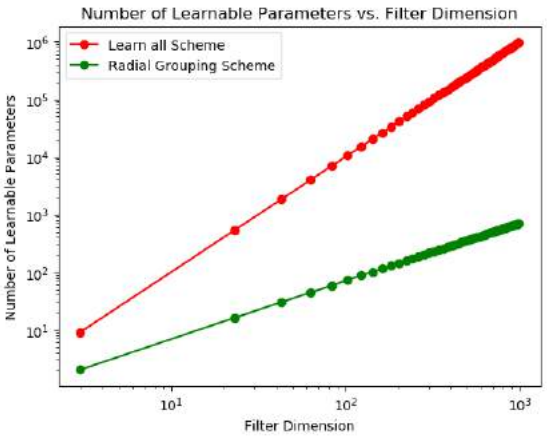


Fig. 5.9: Advantage of using this radial scheme than learning all the filter weights via the CNN. It's clear that in the log-log we are getting an order reduction in this scheme. For a $R \times R$ filter, the number of weights to be estimated from post interpolation is $\mathcal{O}(R)$ whereas, if the scheme is to learn all the weights in the filter (Learn-all scheme), then it's $\mathcal{O}(R^2)$.

$$u_p = iFFT(F) \tag{5.31}$$

where u_p is the processed output of the Fourier attention network.

5.3.3 Loss Functions

Given a real blur image (I^b), the generator network $G_{B\to S}$ transfers the image from blur to sharp domain. The output $\hat{I}^s$ of decoder $G^D_{B\to S}$ is used by discriminator D_s to distinguish if the resultant image is sharp or not.

$$\hat{I}^s = G_{B\to S}(I^b)$$

The following loss function is used to optimize both generator $G_{B\to S}$ and discriminator D_s simultaneously

$$\mathbb{L}_{GAN}(G_{B\to S}, D_S) = \mathbb{E}_{I_s\sim p(I_s)}\Big[\log D_S(I_s)\Big] + \mathbb{E}_{I_b\sim p(I_b)}\Big[\log(1 - D_S(G_{B\to S}(I_b)))\Big] \tag{5.32}$$

where $\mathbb{E}$ is the error function, p denotes the data distribution, $I_b \sim p(I_b)$ and $I_s \sim p(I_s)$ denote images sampled from blur and sharp image distributions respectively.

The output of decoder $G^D_{S\to B}$ is used by discriminator D_B to distinguish if the resultant image is blurred or not. The loss function used to optimize both generator $G_{S\to B}$ and discriminator D_B simultaneously is

$$\mathbb{L}_{GAN}(G_{S\to B}, D_B) = \mathbb{E}_{I_b\sim p(I_b)}\Big[\log D_B(I_b)\Big] + \mathbb{E}_{I_s\sim p(I_s)}\Big[\log(1 - D_B(G_{S\to B}(I_s)))\Big] \tag{5.33}$$

The above adversarial loss functions are sufficient to generate visually sharp images. However, the estimated sharp image's content need not exactly match that of the input image due to the unavailability of supervised pairs. Inspired by cycleGAN (Zhu *et al.*, 2017), we use cycle consistency loss, where the estimated sharp image is projected into blur domain using $G_{S\to B}$ and compared with the input blur image. The projected blur image can be represented as

$$\hat{I}^b = G_{S\to B}(I^s)$$

The cycle consistency loss function can be defined as

$$\mathbb{L}_{cyc\_b}(G_{B\to S}, G_{S\to B}) = \mathbb{E}_{I_b\sim p(I_b)}\Big[||G_{S\to B}(G_{B\to S}(I_b)) - I_b||_1\Big] \tag{5.34}$$

Similarly, the cycle consistency loss can be applied for the other domain by projecting the estimated blur image to the sharp domain using $G_{B\to S}$ and comparing with the real sharp image. The resultant loss function can be defined as

$$\mathbb{L}_{cyc\_s}(G_{S\to B}, G_{B\to S}) = \mathbb{E}_{I_s\sim p(I_s)}\Big[||G_{B\to S}(G_{S\to B}(I_s)) - I_s||_1\Big] \tag{5.35}$$

We optionally add the regularization loss on the magnitude spectrum of the frequency filters in the $G_{B\to S}$ Fourier attention blocks. Observed that the magnitude spectrum reaches large values in the course of optimization and therefore this loss ensures that the extreme values in the magnitude spectrum is discouraged.

(a) Blurry Image (b) Restored Image (c) Sharp Image

Fig. 5.10: Effect of ψ (number of interpolation nodes) is shown. In the first row $\psi = 5$, in the second row $\psi = 15$ and the last row has $\psi = 31$. $\psi = 15$ performs better when compared to other values.

Let m_f^i represent the 2D magnitude spectrum of the i^{th} Fourier attention block in $G_{B \to S}$. Then the regularization loss is represented as,

$$\mathbb{L}_{reg}(G_{B \to S}) = \sum_{i=1}^{n} ||m_f^i||_F^2 \tag{5.36}$$

where $||||_F$ represents the Frobenius norm of the matrix and n is the total number of Fourier blocks attention blocks in $G_{B \to S}$.

Total loss function can be written as

$$
\begin{aligned}
&\mathbb{L}_{Total}(G_{S\to B}, G_{B\to S}, D_S, D_B) \\
&= \lambda_{adv}\mathbb{L}_{GAN}(G_{B\to S}, D_S) \\
&+\lambda_{adv}\mathbb{L}_{GAN}(G_{S\to B}, D_B) \\
&+\lambda_{cyc}\mathbb{L}_{cyc\_s}(G_{S\to B}, G_{B\to S}) \\
&+\lambda_{cyc}\mathbb{L}_{cyc\_b}(G_{B\to S}, G_{S\to B}) \\
&+\lambda_{reg}\mathbb{L}_{reg}(G_{B\to S})
\end{aligned}
\tag{5.37}
$$

Following (Zhu *et al.*, 2017), the weights for λ_{adv} and λ_{cyc} are set as 1 and 10 respectively. The value of λ_{reg} is set to a small value to ensure sufficient space for the filter magnitude spectrum in optimization ($\lambda_{reg} = 0.0001$) The whole network is trained in a min-max fashion as

$$
\arg \min_{G_{S\to B}, G_{B\to S}} \max_{D_B, D_B} \mathbb{L}_{Total}(G_{S\to B}, G_{B\to S}, D_S, D_B) \tag{5.38}
$$

5.4 Experiments

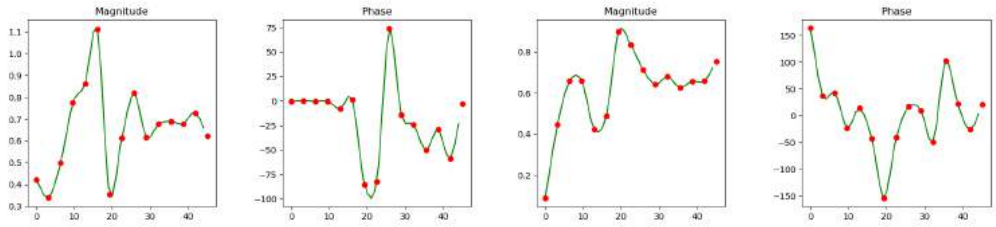
In this section, we will analyse the results and inferences of different experiments.

5.4.1 Datasets and Metrics

GoPro dataset: We use the dataset proposed in (Nah *et al.*, 2017) which used GOPRO4 Hero Black camera to generate the dataset. The dataset was shot at 240 fps and the blurred image was generated by averaging varying number (7-13) number of successive latent frames to produce blurs in different scales. The dataset is composed of 3214 pairs of blurry and sharp images at 1280x720 resolution.

5.4.2 Effect of the number of samples

The interpolation module acts as a regularizer which ensures that no all weights go to zero and stabilize the optimization of the frequency filter weights. The spline interpolation is added to impose a smoothness and continuity prior to the



(a) (b) (c) (d)

Fig. 5.11: (a),(c) are the regularized magnitude spectrum, and the corresponding phase spectrum is shown in (b),(d) respectively. Phase is shown in degrees (-180 to 180)

magnitude and the phase-frequency weights. Spline interpolation ensures that the network can run for arbitrarily shaped images by interpolating the values of the frequency spectrum.

Fourier attention CNN's τ_m semi-diagonally spaced radial values of the magnitude and the phase spectrum which are denoted as m_s respectively. We run experiments only with magnitude spectrum and muting the phase branch by assuming that the phase is 0, or in other words, the frequency-filter is completely real. From our experiments, we find that if the number of samples (ψ) are higher, then the spline interpolation cannot effectively exploit the correlation in the neighbourhood in both the phase and the magnitude spectrum. On the contrary, if the number of samples is very small, then the resultant frequency spectrum couldn't effectively attend to relevant frequencies, which caused a dip in performance. Therefore, there exists an optimal number of sample points, which balances the smoothness prior and also the representative capacity of the attention module. Refer to Table 5.1 for the results. In Fig. 5.10 some sample magnitude spectrum samples and the spline interpolation are plotted.

5.4.3 Effect of Phase on Fourier Attention

Phase plays an important role in FFT as it denotes the initial information of the sinusoid at the origin. When the phase is assumed to be uniformly zero, this assumption is equal to assuming that the filter is completely real. Therefore this assumption curtails the representation ability of the module. As expected, the

Experiment	PSNR	SSIM	*pique*	*brisque*	*niqe*
Base Model	23.34	0.78	31.71	27.25	2.92
$\psi = 5$ samples	22.96	0.74	31.86	23.57	2.83
$\psi = 15$ samples	23.64	0.79	37.50	25.00	3.19
$\psi = 31$ samples	22.57	0.74	23.72	18.84	3.19
Both Magnitude and Phase	26.13	0.85	49.75	33.69	3.66

Table 5.1: Quantitative Comparison of different experiments on the GoPro Dataset. The experiments with varying ψ was done by assuming the entire phase spectrum is zero. The final experiment (both magnitude and phase) was done with $\psi = 15$.

inclusion of phase (τ_ϕ) in the attention module improves the model performance. Its relative performance is shown in Fig. 5.12. For quantitative comparison, look in the last row of Table 5.1.

5.4.4 Effect of magnitude regularization

To constrain the optimization, we add a frequency magnitude regularization loss which penalizes large values in the magnitude spectrum as expressed in Eq. (5.36). This loss successfully ensured that the stability of the magnitude spectrum is guaranteed and the performance of the model is shown. The regularized spectrum is shown in Fig. 5.11. The last row in Fig. 5.12 displays the results of weight regularization. For quantitative comparison, look in the last row of Table 5.1.

(a) Blurry Image (b) Restored Image (c) Sharp Image

Fig. 5.12: Effect of Φ (phase) and magnitude regularization is shown. In the first row, the unit magnitude spectrum is assumed uniformly and the attention block can adjust only the phase spectrum. In the second row, both magnitude and phase spectrum are learnable and the last row has both magnitude and phase spectrum learnable optimized with the regularization loss.

Chapter 6

FUTURE WORK AND CONCLUSION

6.1 Future Work

The works such as (Karras *et al.*, 2017) emphasise the difficulty of generating high-resolution images using GANs. They proposed a progressive approach to generate high resolution (1024 x 1024) images and they show that their approach generates higher quality images with variation and good training stability. We can aim to extend our scale recurrent approach to higher resolution images and deblur them. This approach of using selective information from the lower scales can stabilize training and can also potentially ease the high-resolution image generation process.

As the model is domain-specific, we can incorporate selective attention models to make it suitable for generic scene deblurring problem. Extending to generic scene deblurring using GoPro dataset (Nah *et al.*, 2017).

Currently, the unsupervised scale adaptive attention deblurring network attends to the nearest lower resolution branch. We can investigate the effect of the addition of multiple connections to more scales and observe its effect on the performance.

Improving the computational speed of Fourier attention is left for future work. The cubic spline interpolation due to an expensive matrix inversion takes considerable computation time increasing the computational load of the attention module. The benefits of using optimized FFT is not realized due to the bottleneck caused by these operations. Making the module lightweight without considerable reduction in performance is best kept as future work in this area.

6.2 Conclusion

We proposed a multi-scale unsupervised network for deblurring domain-specific data. We used a coarse-to-fine approach to stabilize GAN training and a scale adaptive attention module (SAAM) to aid relevant information flow across scales. Ablation studies show the importance of using our multi-scale approach in conjunction with SAAM. Qualitative and quantitative comparisons show that our methods perform on par with supervised methods while outperforming conventional and unsupervised methods.

In the second part, we proposed Fourier attention to extend the representative ability of the network to focus on the relevant frequency spectrum. Experiments are run by modifying the generator $G_{B \to S}$ to gain robust performance. Though this is a general-purpose module, we expect it to be generally useful in image restoration tasks, where the frequency spectrum plays an important role. Improving the computational speed of Fourier attention is left for future work in this area.

LIST OF PUBLICATIONS

I. CONFERENCE PRESENTATION

1. Praveen Kandula, Lokesh Kumar T, Rajagopalan AN; "Unsupervised Domain-Specific Deblurring with Scale-Adaptive Attention"; *Manuscript under preparation*

Appendix A

FREQUENCY DOMAIN IMAGE PROCESSING

A.1 Discrete Fourier Transform

Any image $f(x,y) \in \mathbb{R}^{M \times N}$ can be represented in frequency domain $F(u,v)$ using two dimensional discrete Fourier transform (DFT) as ($j = \sqrt{-1}$),

$$F(u,v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y)\ exp\left(-j2\pi\left(\frac{ux}{M} + \frac{vy}{N}\right)\right) \tag{A.1}$$

The main theme behind Fourier transform is that any waveform can be represented as a sum of sines and cosines. This can be seen in Eq. (A.1) where the exponential term can be expressed in terms of sines and cosines in the variables i and j determining these frequencies.

The inverse discrete Fourier transform can be obtained by,

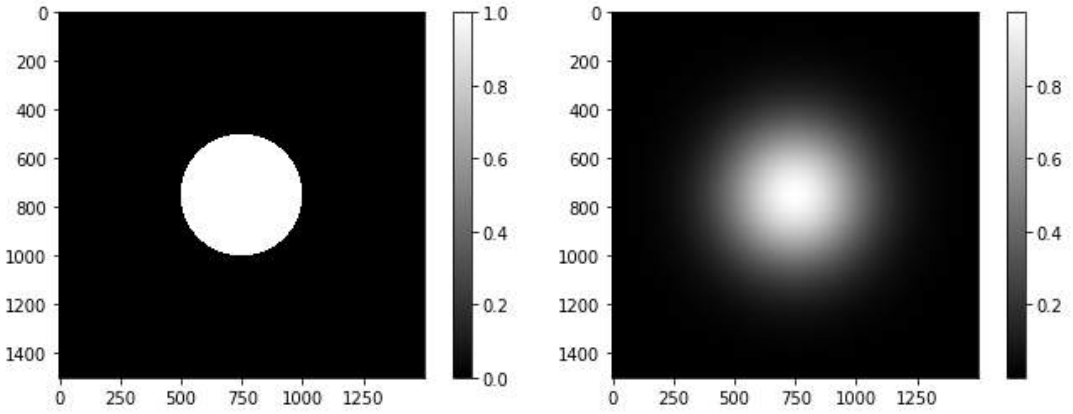
$$f(u,v) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} F(x,y)\ exp\left(j2\pi\left(\frac{ux}{M} + \frac{vy}{N}\right)\right) \tag{A.2}$$

It can be useful to note that $F(0,0)$, the value of the spectrum at the origin of the frequency domain is called the DC component which is equal to the average value of the image signal $f(x,y)$.

$$F(0,0) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) \tag{A.3}$$

A.2 Image Filtering

The convolution theorem reveals the relationship between spatial domain convolution and frequency domain multiplication operation which can be related as,



(a) Ideal Lowpass Filter (b) Gaussian Lowpass Filter

Fig. A.1: Frequency domain representations of lowpass filters

$$f(x,y) * h(x,y) \Leftrightarrow H(u,v)F(u,v) \tag{A.4}$$

and alternatively,

$$f(x,y)h(x,y) \Leftrightarrow H(u,v) * F(u,v) \tag{A.5}$$

where $*$ indicates the image convolution operator. This conveys that the multiplication of Fourier transforms corresponds to convolution in the spatial domain. Therefore, this theorem can be used to our advantage to apply some spatial filters.

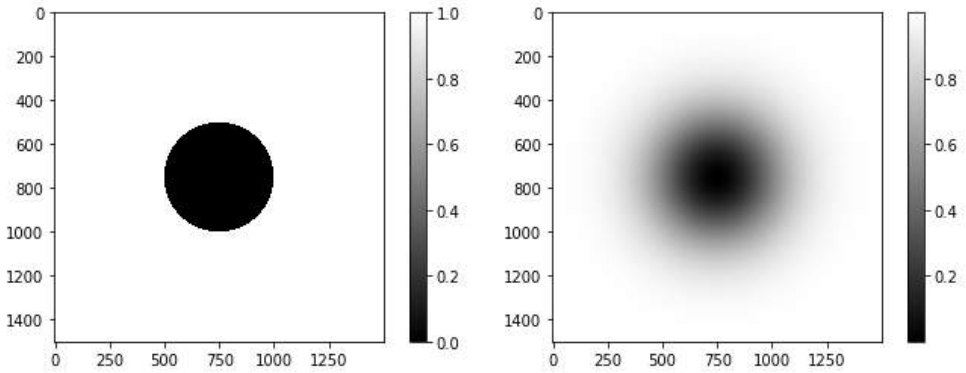
A.3 Frequency Domain Filters

The filters which can be created directly in the frequency domain are,

- Lowpass filters: Filters that cause blur on the image
- Highpass filters: Filters that sharpen the image (edge detectors)
- Notch Filters: Filters which are referred to as band-stop filters

A.3.1 Lowpass Filters

Lowpass filters are special filters that allow only low-frequency components of the image and attenuate the high-frequency spectrum in the Fourier domain. This act,



(a) Ideal Highpass Filter (b) Gaussian Highpass Filter

Fig. A.2: Frequency domain representations of highpass filters

therefore, results in the output image dominantly have low-frequency components. This causes a blurry effect on the resultant image.

If $D(u, v)$ denotes the distance from point (u, v) from the center of the filter, then an ideal lowpass filter in the frequency domain can be represented as,

$$H(u,v) = \begin{cases} 1 & if\ D(u,v) \leq D_0 \\ 0 & if\ D(u,v) > D_0 \end{cases} \tag{A.6}$$

where D_0 is a non-negative real number. Note that the DFT frequency bands are radial in nature and increase with the radius from the center. This is called ideal lowpass filter because the higher frequency radial bands are completely attenuated by this filter which therefore is a purely 'lowpass' filter. Fig. A.1 shows the 2D representation of lowpass filters.

Another lowpass filter is the Gaussian filter which can be represented as,

$$H(u,v) = exp\left(-\frac{D^2(u,v)}{2D_0^2}\right) \tag{A.7}$$

A.3.2 Highpass Filters

As opposed to lowpass filters, these filters attenuate low frequencies and allow higher frequencies to pass. This behaviour causes the resultant image to look like the edge map of the original input image. This is due to the attenuation of lower frequencies. The relation between lowpass and highpass filters can be expressed

as follows,

$$H_{hp}(u,v) = 1 - H_{lp}(u,v) \tag{A.8}$$

An ideal highpass filter does not allow any low frequency to leak through the filter operation. Some standard highpass filters are shown in Fig. A.2

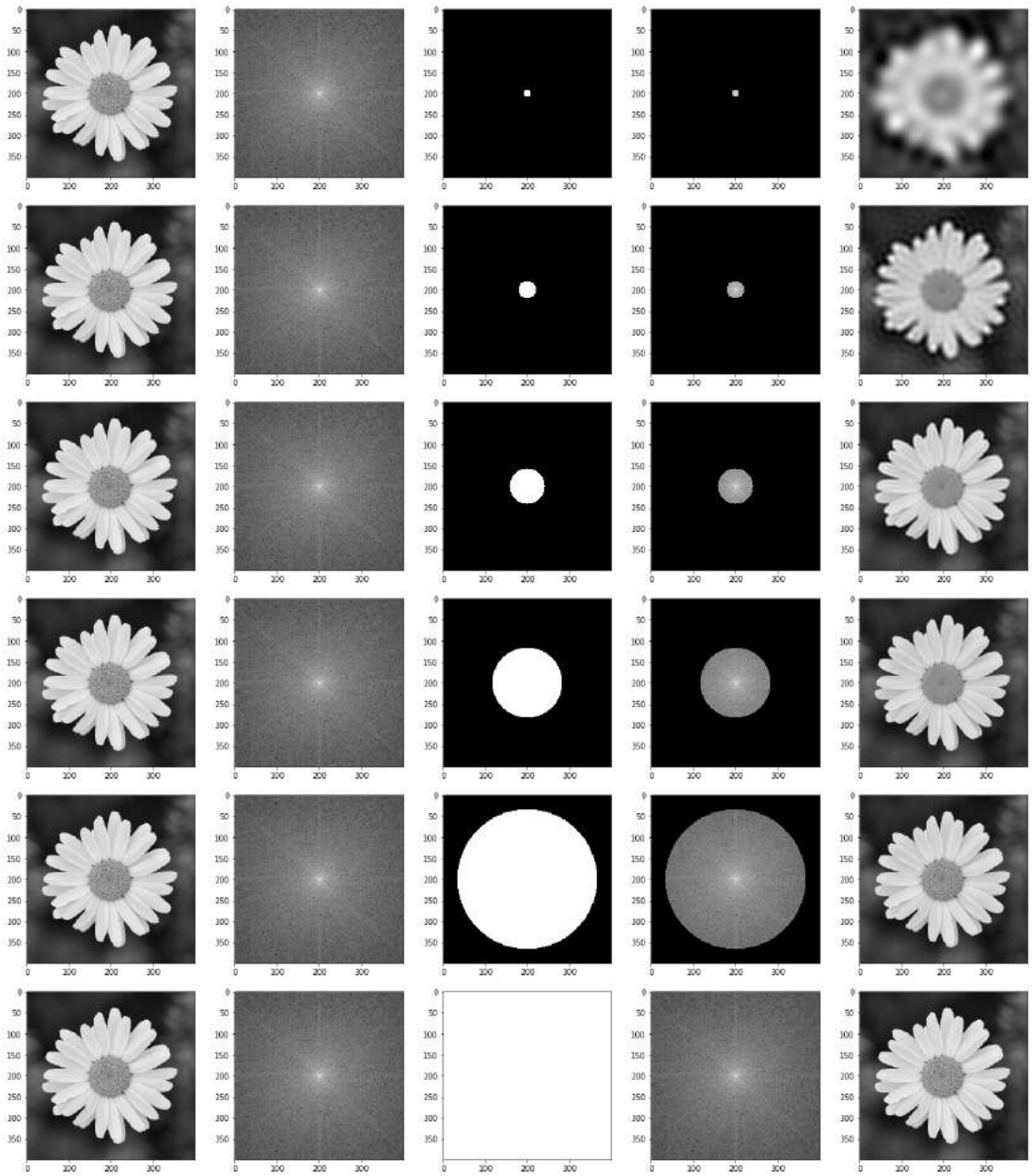


Fig. A.3: Ideal Lowpass filter acting on a flower image. (a) is the input image, (b) is the DFT of the image (a), (c) is the ideal lowpass filter, (d) is the output image's DFT computed as explained in Eq. (A.4), (e) is the resultant output image. In the first row, the lowpass filter is aggressive in the sense that most of the frequencies are blocked which caused a lot of noticeable blurs, but as the lowpass filter allows more and more frequencies, we see the blur decreasing (as higher frequencies also allowed). In the last few rows, no visible difference is seen between the input and the output images.

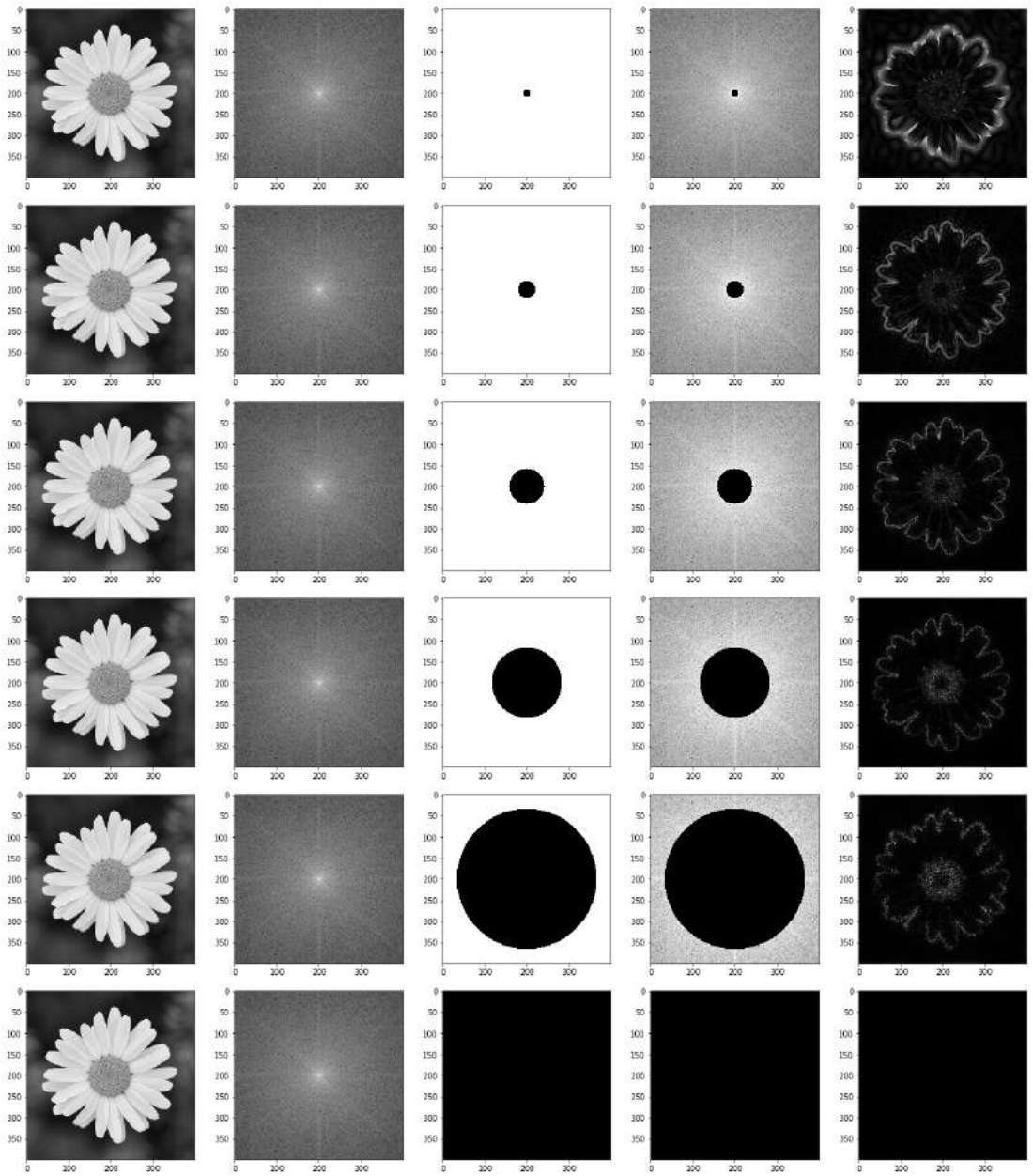


Fig. A.4: Ideal Highpass filter acting on a flower image. (a) is the input image, (b) is the DFT of the image (a), (c) is the ideal lowpass filter, (d) is the output image's DFT computed as explained in Eq. (A.4), (e) is the resultant output image. In the first row, the highpass filter is aggressively blocked low frequencies and allowed higher frequencies which caused a strong excitation of edges, but as the highpass filter blocks more and more frequencies (the increasing black circle in (c),(d)), we see the edge strength decreasing (as higher frequencies are blocked also causing loss of power). In the last row, the entire frequency spectrum is blocked which causes the power of the output to zero (no excitation in output image).

Bibliography

1. **Anwar, S.** and **N. Barnes** (2020). Densely residual laplacian super-resolution. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

2. **Arjovsky, M.**, **S. Chintala**, and **L. Bottou** (2017). Wasserstein generative adversarial networks. *In International conference on machine learning*. PMLR.

3. **Ayers, G.** and **J. C. Dainty** (1988). Iterative blind deconvolution method and its applications. *Optics letters*, **13**(7), 547–549.

4. **Ba, J.**, **V. Mnih**, and **K. Kavukcuoglu** (2014). Multiple object recognition with visual attention. *arXiv preprint arXiv:1412.7755*.

5. **Bahdanau, D.**, **K. Cho**, and **Y. Bengio** (2014). Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

6. **Bao, J.**, **D. Chen**, **F. Wen**, **H. Li**, and **G. Hua** (2018). Towards open-set identity preserving face synthesis. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

7. **Chakrabarti, A.** (2016). A neural approach to blind motion deblurring. *In European conference on computer vision*. Springer.

8. **Chan, T. F.** and **C.-K. Wong** (1998). Total variation blind deconvolution. *IEEE transactions on Image Processing*, **7**(3), 370–375.

9. **Chen, L.**, **H. Zhang**, **J. Xiao**, **L. Nie**, **J. Shao**, **W. Liu**, and **T.-S. Chua** (2017). Sca-cnn: Spatial and channel-wise attention in convolutional networks for image captioning. *In Proceedings of the IEEE conference on computer vision and pattern recognition*.

10. **Cho, S.** and **S. Lee** (2009). Fast motion deblurring. *In ACM SIGGRAPH Asia 2009 papers*, 1–8.

11. **Fang, X.**, **Q. Zhou**, **J. Shen**, **C. Jacquemin**, and **L. Shao** (2020). Text image deblurring using kernel sparsity prior. *IEEE Transactions on Cybernetics*, **50**(3), 997–1008, `doi:10.1109/TCYB.2018.2876511`.

12. **Fergus, R.**, **B. Singh**, **A. Hertzmann**, **S. T. Roweis**, and **W. T. Freeman** (2006). Removing camera shake from a single photograph. *In ACM SIGGRAPH 2006 Papers*, 787–794.

13. **Fish, D.**, **A. Brinicombe**, **E. Pike**, and **J. Walker** (1995). Blind deconvolution by means of the richardson–lucy algorithm. *JOSA A*, **12**(1), 58–65.

14. **Gong, D.**, **J. Yang**, **L. Liu**, **Y. Zhang**, **I. Reid**, **C. Shen**, **A. Van Den Hengel**, and **Q. Shi** (2017). From motion blur to motion flow: A deep learning solution for removing heterogeneous motion blur. *In Proceedings of the IEEE conference on computer vision and pattern recognition.*

15. **Goodfellow, I.**, **J. Pouget-Abadie**, **M. Mirza**, **B. Xu**, **D. Warde-Farley**, **S. Ozair**, **A. Courville**, and **Y. Bengio** (2014). Generative adversarial nets. *In Advances in neural information processing systems.*

16. **He, K.**, **X. Zhang**, **S. Ren**, and **J. Sun** (2016). Deep residual learning for image recognition. *In Proceedings of the IEEE conference on computer vision and pattern recognition.*

17. **Hradis, M.**, **J. Kotera**, **P. Zemcık**, and **F. Sroubek** (2015). Convolutional neural networks for direct text deblurring. *In Proceedings of BMVC*, volume 10.

18. **Hu, J.**, **L. Shen**, and **G. Sun** (2018). Squeeze-and-excitation networks. *In Proceedings of the IEEE conference on computer vision and pattern recognition.*

19. **Hu, Z.**, **L. Xu**, and **M.-H. Yang** (2014). Joint depth estimation and camera shake removal from single blurry image. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.*

20. **Huang, G.**, **Z. Liu**, **L. Van Der Maaten**, and **K. Q. Weinberger** (2017). Densely connected convolutional networks. *In Proceedings of the IEEE conference on computer vision and pattern recognition.*

21. **Isola, P.**, **J.-Y. Zhu**, **T. Zhou**, and **A. A. Efros** (2017). Image-to-image translation with conditional adversarial networks. *In Proceedings of the IEEE conference on computer vision and pattern recognition.*

22. **Jaderberg, M.**, **K. Simonyan**, **A. Zisserman**, and **K. Kavukcuoglu** (2015). Spatial transformer networks. *arXiv preprint arXiv:1506.02025.*

23. **Johnson, J.**, **A. Alahi**, and **L. Fei-Fei** (2016). Perceptual losses for real-time style transfer and super-resolution. *In European conference on computer vision.* Springer.

24. **Joshi, N.**, **R. Szeliski**, and **D. J. Kriegman** (2008). Psf estimation using sharp edge prediction. *In 2008 IEEE Conference on Computer Vision and Pattern Recognition.* IEEE.

25. **Karras, T.**, **T. Aila**, **S. Laine**, and **J. Lehtinen** (2017). Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196.*

26. **Krishnan, D.**, **T. Tay**, and **R. Fergus** (2011). Blind deconvolution using a normalized sparsity measure. *In CVPR 2011.* IEEE.

27. **Kupyn, O.**, **V. Budzan**, **M. Mykhailych**, **D. Mishkin**, and **J. Matas** (2018). Deblurgan: Blind motion deblurring using conditional adversarial networks. *In Proceedings of the IEEE conference on computer vision and pattern recognition*.

28. **Kupyn, O.**, **T. Martyniuk**, **J. Wu**, and **Z. Wang** (2019). Deblurgan-v2: Deblurring (orders-of-magnitude) faster and better. *In Proceedings of the IEEE/CVF International Conference on Computer Vision*.

29. **Lai, W.-S.**, **J.-B. Huang**, **Z. Hu**, **N. Ahuja**, and **M.-H. Yang** (2016). A comparative study for single image blind deblurring. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

30. **Lee, C.-H.**, **Z. Liu**, **L. Wu**, and **P. Luo** (2020). Maskgan: Towards diverse and interactive facial image manipulation. *In IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

31. **Levin, A.**, **R. Fergus**, **F. Durand**, and **W. T. Freeman** (2007). Deconvolution using natural image priors. *Massachusetts Institute of Technology, Computer Science and Artificial Intelligence Laboratory*, **3**.

32. **Lu, B.**, **J.-C. Chen**, **C. D. Castillo**, and **R. Chellappa** (2019*a*). An experimental evaluation of covariates effects on unconstrained face verification. *IEEE Transactions on Biometrics, Behavior, and Identity Science*, **1**(1), 42–55.

33. **Lu, B.**, **J.-C. Chen**, and **R. Chellappa** (2019*b*). Unsupervised domain-specific deblurring via disentangled representations. *In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

34. **Madam Nimisha, T.**, **K. Sunil**, and **A. Rajagopalan** (2018). Unsupervised class-specific deblurring. *In Proceedings of the European Conference on Computer Vision (ECCV)*.

35. **Mittal, A.**, **A. K. Moorthy**, and **A. C. Bovik** (2012). No-reference image quality assessment in the spatial domain. *IEEE Transactions on image processing*, **21**(12), 4695–4708.

36. **Mnih, V.**, **N. Heess**, **A. Graves**, and **K. Kavukcuoglu** (2014). Recurrent models of visual attention. *arXiv preprint arXiv:1406.6247*.

37. **Nah, S.**, **T. Hyun Kim**, and **K. Mu Lee** (2017). Deep multi-scale convolutional neural network for dynamic scene deblurring. *In Proceedings of the IEEE conference on computer vision and pattern recognition*.

38. **Nimisha, T. M.**, **A. Kumar Singh**, and **A. N. Rajagopalan** (2017). Blur-invariant deep learning for blind-deblurring. *In Proceedings of the IEEE International Conference on Computer Vision*.

39. **Noroozi, M.**, **P. Chandramouli**, and **P. Favaro** (2017). Motion deblurring in the wild. *In German conference on pattern recognition*. Springer.

40. **Pan, J.**, **Z. Hu**, **Z. Su**, and **M.-H. Yang** (2014*a*). Deblurring face images with exemplars. *In European conference on computer vision*. Springer.

41. **Pan, J.**, **Z. Hu**, **Z. Su**, and **M.-H. Yang** (2014*b*). Deblurring text images via l0-regularized intensity and gradient prior. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

42. **Pei, Y.**, **Y. Huang**, **Q. Zou**, **H. Zang**, **X. Zhang**, and **S. Wang** (2018). Effects of image degradations to cnn-based image classification. *arXiv preprint arXiv:1810.05552*.

43. **Purohit, K.** and **A. Rajagopalan** (2020). Region-adaptive dense network for efficient motion deblurring. *In Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34.

44. **Radford, A.**, **L. Metz**, and **S. Chintala** (2015). Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*.

45. **Rajagopalan, A.** and **R. Chellappa** (2014). *Motion Deblurring: Algorithms and Systems*. Cambridge University Press. ISBN 9781107044364. URL `https://books.google.co.in/books?id=_tFkAwAAQBAJ`.

46. **Ramakrishnan, S.**, **S. Pachori**, **A. Gangopadhyay**, and **S. Raman** (2017). Deep generative filter for motion deblurring. *In Proceedings of the IEEE International Conference on Computer Vision Workshops*.

47. **Sahu, S.**, **M. K. Lenka**, and **P. K. Sa** (2019). Blind deblurring using deep learning: A survey. *arXiv preprint arXiv:1907.10128*.

48. **Schuler, C. J.**, **M. Hirsch**, **S. Harmeling**, and **B. Schölkopf** (2015). Learning to deblur. *IEEE transactions on pattern analysis and machine intelligence*, **38**(7), 1439–1451.

49. **Shan, Q.**, **J. Jia**, and **A. Agarwala** (2008). High-quality motion deblurring from a single image. *Acm transactions on graphics (tog)*, **27**(3), 1–10.

50. **Shen, Z.**, **W.-S. Lai**, **T. Xu**, **J. Kautz**, and **M.-H. Yang** (2018). Deep semantic face deblurring. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

51. **Shi, X.**, **Z. Chen**, **H. Wang**, **D.-Y. Yeung**, **W.-K. Wong**, and **W.-c. Woo** (2015). Convolutional lstm network: A machine learning approach for precipitation nowcasting. *arXiv preprint arXiv:1506.04214*.

52. **Simonyan, K.** and **A. Zisserman** (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

53. **Stuchi, J. A.**, **L. Boccato**, and **R. Attux** (2020). Frequency learning for image classification. *arXiv preprint arXiv:2006.15476*.

54. **Suin, M.**, **K. Purohit**, and **A. Rajagopalan** (2020). Spatially-attentive patch-hierarchical network for adaptive motion deblurring. *In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

55. **Suin, M.**, **K. Purohit**, and **A. N. Rajagopalan** (2021). Degradation aware approach to image restoration using knowledge distillation. *IEEE Journal of Selected Topics in Signal Processing*, **15**(2), 162–173, doi:10.1109/JSTSP.2020.3043622.

56. **Sun, J.**, **W. Cao**, **Z. Xu**, and **J. Ponce** (2015). Learning a convolutional neural network for non-uniform motion blur removal. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

57. **Tao, X.**, **H. Gao**, **X. Shen**, **J. Wang**, and **J. Jia** (2018). Scale-recurrent network for deep image deblurring. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

58. **Vasu, S.**, **V. R. Maligireddy**, and **A. Rajagopalan** (2018). Non-blind deblurring: Handling kernel uncertainty with cnns. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

59. **Vasu, S.** and **A. Rajagopalan** (2017). From local to global: Edge profiles to camera motion in blurred images. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

60. **Vaswani, A.**, **N. Shazeer**, **N. Parmar**, **J. Uszkoreit**, **L. Jones**, **A. N. Gomez**, **L. Kaiser**, and **I. Polosukhin** (2017). Attention is all you need. *arXiv preprint arXiv:1706.03762*.

61. **Venkatanath, N.**, **D. Praneeth**, **M. C. Bh**, **S. S. Channappayya**, and **S. S. Medasani** (2015). Blind image quality evaluation using perception based features. *In 2015 Twenty First National Conference on Communications (NCC)*. IEEE.

62. **Wang, T.-C.**, **M.-Y. Liu**, **J.-Y. Zhu**, **A. Tao**, **J. Kautz**, and **B. Catanzaro** (2018). High-resolution image synthesis and semantic manipulation with conditional gans. *In Proceedings of the IEEE conference on computer vision and pattern recognition*.

63. **Woo, S.**, **J. Park**, **J.-Y. Lee**, and **I. S. Kweon** (2018). Cbam: Convolutional block attention module. *In Proceedings of the European conference on computer vision (ECCV)*.

64. **Xu, K.**, **J. Ba**, **R. Kiros**, **K. Cho**, **A. Courville**, **R. Salakhudinov**, **R. Zemel**, and **Y. Bengio** (2015). Show, attend and tell: Neural image caption generation with visual attention. *In International conference on machine learning*. PMLR.

65. **Xu, K.**, **M. Qin**, **F. Sun**, **Y. Wang**, **Y.-K. Chen**, and **F. Ren** (2020). Learning in the frequency domain. *In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

66. **Xu, L.** and **J. Jia** (2010). Two-phase kernel estimation for robust motion deblurring. *In European conference on computer vision*. Springer.

67. **Xu, L.**, **S. Zheng**, and **J. Jia** (2013). Unnatural l0 sparse representation for natural image deblurring. *In Proceedings of the IEEE conference on computer vision and pattern recognition*.

68. **Yan, Y.**, **W. Ren**, **Y. Guo**, **R. Wang**, and **X. Cao** (2017). Image deblurring via extreme channels prior. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

69. **Yang, J.**, **Y. Zhang**, and **W. Yin** (2009). An efficient tvl1 algorithm for deblurring multichannel images corrupted by impulsive noise. *SIAM Journal on Scientific Computing*, **31**(4), 2842–2865.

70. **Zhang, H.**, **Y. Dai**, **H. Li**, and **P. Koniusz** (2019*a*). Deep stacked hierarchical multi-patch network for image deblurring. *In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

71. **Zhang, J.**, **J. Pan**, **J. Ren**, **Y. Song**, **L. Bao**, **R. W. Lau**, and **M.-H. Yang** (2018). Dynamic scene deblurring using spatially variant recurrent neural networks. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

72. **Zhang, Q.**, **H. Xiao**, **F. Xue**, **W. Lu**, **H. Liu**, and **F. Huang** (2019*b*). Digital image forensics of non-uniform deblurring. *Signal Processing: Image Communication*, **76**, 167–177.

73. **Zhao, H.**, **X. Kong**, **J. He**, **Y. Qiao**, and **C. Dong** (2020). Efficient image super-resolution using pixel attention. *arXiv preprint arXiv:2010.01073*.

74. **Zhu, J.-Y.**, **T. Park**, **P. Isola**, and **A. A. Efros** (2017). Unpaired image-to-image translation using cycle-consistent adversarial networks. *In Proceedings of the IEEE international conference on computer vision*.