LLPackNet: A fast and light-weight approach to extreme low light image enhancement

A Project Report

submitted by

ATUL BALAJI (EE16B002)

in partial fulfilment of the requirements
for the award of the degree of

BACHELOR OF TECHNOLOGY

DEPARTMENT OF ELECTRICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY MADRAS.

Jan 2020 - Jun 2020

THESIS CERTIFICATE

This is to certify that the report titled **LLPackNet: A fast and light-weight approach to extreme low light image enhancement**, submitted by **Atul Balaji (EE16B002)**, to the Indian Institute of Technology, Madras, for the award of the degree of **Bachelor of Technology**, is a bona fide record of the research work done by him under our supervision.

Prof. Kaushik Mitra
Research Guide
Dept. of Electrical Engineering
IIT-Madras, 600 036

Place: Chennai

Date: 12th June 2020

TABLE OF CONTENTS

ACKNOWLEDGEMENTS

I wish to express my deep sense of gratitude to my guide Dr. Kaushik Mitra for his continued support and guidance throughout this project.

I would also like to thank Mohit Lamba (EE18D009), a PhD scholar at the Computational Imaging Lab, with whom I had worked on this project jointly. I acknowledge his important contributions to this project, in formulating, designing and implementing the key components of our approach, especially the amplifier module and the network architecture.

Finally, I would like to thank all the professors of this institute who taught me the subjects on which this project is based.

ABSTRACT

Keywords: Extreme low light image enhancement; Pack and UnPack operations; Downsampling and Upsampling; Color Correlation; Amplification

The ability to capture good quality images in extreme low-light has been a long-standing pursuit of the computer vision community. The seminal work by Chen *et al.* [6] has especially caused renewed interest in this area, resulting in methods that build on top of their work in a bid to improve the reconstruction. However, for practical utility and deployment of low-light enhancement algorithms on edge devices such as embedded systems, surveillance cameras and smartphones, the solution must respect additional constraints such as limited memory and processing power. With this in mind, we aim to develop a new deep neural network architecture that minimizes the network latency, memory utilization and no.of model parameters, while at the same time maintains a competitive image reconstruction quality.

The key idea to minimize processing time is to forbid any computation in the high resolution (HR) space and instead restrict most of the computations to a much lower resolution (LR) space. However, using standard techniques for such a large factor downsampling/upsampling causes a lot of artifacts and color distortions in the restored image, which arise due to information loss. To mitigate this, the *Pack* and *UnPack* operations are introduced to perform large factor downsampling/upsampling, thus greatly minimizing time, while at the same time achieving good color restoration.

Additionally, most of the state-of-the-art algorithms on low-light image enhancement need to pre-amplify the input image before processing it. However, they generally rely on the ground truth exposure information to estimate the amplification factor, which restricts their applicability to unknown scenes where such information is not available. We propose to solve this problem by designing a simple yet effective mechanism for automatically determining the amplification factor from the input image itself.

We show that we can enhance a full resolution, 2848×4256, extremely dark image in the ballpark of 3 seconds on a CPU. We achieve this with $2-7\times$ fewer model parameters, $2-3\times$ lower memory utilization, $5-20\times$ speed up, while maintaining a competitive image reconstruction quality compared to state-of-the-art algorithms.

CHAPTER 1

INTRODUCTION

The ability to swiftly capture high quality images with smartphones has led to the widespread proliferation of digital images. This quality is however limited to good lighting conditions and capturing good quality photos in low light is difficult, even with careful tuning of camera settings such as ISO, flash and exposure. However in many applications such as video surveillance, night-time photography and autonomous driving, it is necessary to work with low light images, which underlines the need for low-light image enhancement algorithms. While much of the work in this direction has focused on enhancing weakly illuminated images [16, 28, 27, 20, 18, 7, 12], enhancement of extreme low-light images, captured in near zero illumination conditions has received comparatively lesser attention.

Recently, however, a landmark paper by Chen *et al.* [6] has shown that using a fully convolutional network, it is possible to restore extreme low light high definition images, with good reconstruction quality. Following this work, several modifications have been proposed in a bid to improve the reconstruction quality. This includes the incorporation of attention units [1], recurrent units [5], the adoption of a multi-scale approach [11, 24] and the usage of deeper networks [23]. With these added complexities, most of these methods are constrained to run on devices with high computational capacity, such as desktop GPUs. However, real-world applications require image enhancement algorithms to run on embedded systems and edge devices, such as smartphones and microcontrollers which in most cases have only CPUs with limited RAM. Keeping these factors in mind, we aim to design a deep network that can restore an extreme low-light high-definition single-image with minimal CPU latency and low memory footprint, but at the same time has a competitive image restoration quality.

Given the fact that a neural network's complexity increases quadratically with spatial dimensions [36], a common approach to reduce the time complexity of a network is to downsample the feature maps in order to perform most of the computations in the low

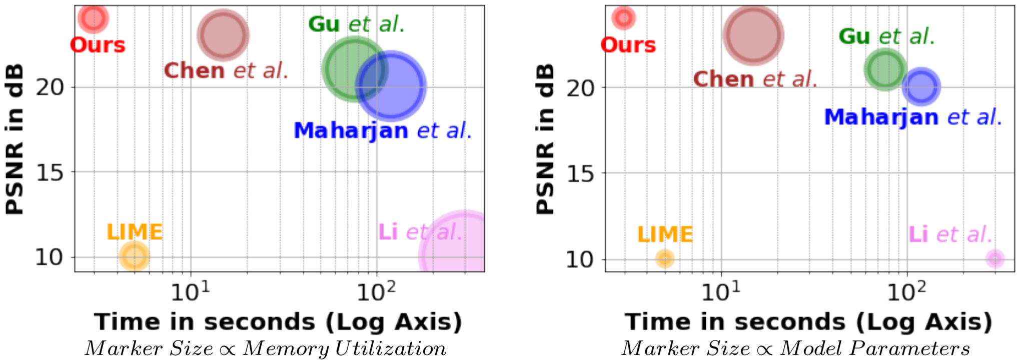


Figure 1.1: Performance comparison of the proposed method with state-of-the-art methods Chen *et al.* [6], Gu *et al.* [11], Maharjan *et al.* [23] and traditional methods LIME [12], and Li *et al.* [20] for extreme low-light single-image enhancement. Refer to Table 4.1 for more details.

resolution space. This is generally done in small steps (such as $2\times$ or $4\times$) [44, 21, 6, 43, 35] to prevent loss of information. However, in applications where fast and light-weight processing are crucial, this small downsampling factor is not sufficient and it is necessary to apply a large downsampling factor such as $8\times$ or $16\times$. But, for performing such large factor downsampling operations, popular choices such as max-pooling and strided convolution [9] cannot be used as they would cause a significant loss of information. In order to mitigate the problems involved with large factor downsampling, we propose the *Pack* $\alpha\times$ downsampling operation. (see Fig. 3.1). We show that this Pack operation bestows LLPackNet with an enormous receptive field which is not trivially possible by directly operating in the HR space.

To perform large factor upsampling, typical methods such as transposed convolution [9] and interpolation are not suitable as they are very slow when operating in high resolution. A faster way to perform upsampling is the PixelShuffle [33] operation, originally introduced in the context of super-resolution. However, it lacks proper correlation between the color channels and hence results in color cast and artifacts in the restored image as shown in Fig. 4.3. In light of this, we propose the complementary *UnPack* $\alpha\times$ operation as an improvement to the PixelShuffle operation, to perform large factor upsampling, while at the same time maintaining good color correlation. Essentially, the proposed *Pack* and *UnPack* operations allow us to operate in a much lower resolution space for computational advantages, without significantly affecting the restoration quality. See Fig. 1.1 for a qualitative comparison of our method with state-of-the-art algorithms.

State-of-the-art deep learning solutions on extreme low-light image enhancement need to pre-amplify dark images before processing them [6, 23, 1, 11]. However, they use ground-truth (GT) exposure information for estimating the amplification factor. But, in a real-world setting, when an unknown image is provided, the GT exposure information will not be available, rendering the amplifier useless and causing degradation in performance. With this in mind, we propose an amplifier module, which will estimate the amplification factor directly from the input image histogram, without relying on the ground-truth information, making it applicable to new data. Putting these components together, we propose a novel, fast and light-weight deep neural network-based pipeline for extreme low-light image enhancement, called LLPackNet.

To summarize, the main contributions of this report are as follows:
1) We propose a deep neural network architecture, called *LLPackNet*, that enhances an extreme low-light image at high resolution even on a CPU with very low latency and computational load.
2) We propose *Pack* and *UnPack* operations for efficient large factor down/upsampling and better color restoration.
3) LLPackNet is equipped with an amplifier module that estimates the amplification factor just from the input image, without using ground truth information.
4) Our experiments show that compared to state-of-the-art solutions, we are able to restore high definition 2848×4256, extreme low-light RAW images with 2–7× fewer model parameters, 2–3× lower memory and 5–20× speed up, with a competitive restoration quality on CPU.

CHAPTER 2

LITERATURE REVIEW

2.1 Background

Over the years, low-light image enhancement has been an active area of research. Here we present a brief overview. Conventional approaches of low-light enhancement are chiefly comprised of histogram equalization (HE) techniques [16, 28, 27], which involve modifying the histogram of the low-light images and Retinex based methods [12, 20, 41, 26, 18, 10], which are inspired by the biological mechanism found in the early stages of the visual system and involve decomposing an image into illumination and reflectance components. Deep learning based methods [17, 29, 37, 8, 22, 38, 7, 32, 19, 42] have also come to be used in recent times, with some works such as MSRNet [32] and RetinexNet [7] trying to combine convolutional neural networks (CNNs) and retinex theory. One popular Retinex based method is LIME [12] which proposes a structure-aware smoothing model to estimate the illumination map. Li *et al.* [20] propose a robust Retinex model which additionally considers the noise map while estimating illumination, by using an optimization function to reveal the structure details in a low-light image. LLNet [22] uses a deep autoencoder to identify signal features of the low-light image for image enhancement and denoising. GLADNet [38] uses an encoder-decoder network to estimate global illumination and then employs a CNN for reconstruction.

2.2 Extreme low-light image enhancement

The methods discussed above are however, mostly limited to weakly illuminated images where a good representation of the scene is already available and they generally do not target high-definition, extreme low-light images, which have short exposure and severely limited illumination. More recently, Chen *et al.* [6] proposed an end-to-end pipeline using a Fully Convolutional Network (FCN), such as U-Net [31] to restore extreme low-light high-definition images. They also introduce a new dataset called See-in-the-Dark (SID), which consists of pairs of raw image data and corresponding ground-truth sRGB image. Their network learns the full image restoration pipeline, from the raw image data to the output restored image, including demosaicing, color transformations, etc. This landmark paper has subsequently spurred several other works in a similar direction [23, 1, 11, 24, 5, 15], in an effort to improve upon the image restoration quality. For example, Gu *et al.* [11] propose a multi-scale self-guided network (SGN) that combines contextual information at different resolutions. Maharjan *et al.* [23] propose a residual learning based end-to-end network to replace U-net and achieve better color and texture restoration. Most of these methods, however, still have significantly high processing time, memory utilization and model parameters and are not suitable for use on edge devices.

2.3 Input image amplification

As noted in Chap. 1, many of the deep learning based methods for low-light enhancement require ground truth information for pre-amplification of the input image, thereby limiting their applicability to unknown images where this information is not available. For example, in Chen *et al.* [6], the amplification ratio is set externally as a function of the ratio of reference image exposure to input image exposure, and is provided as input to the pipeline, for both training and testing.

By contrast, some conventional methods for weakly illuminated images, perform image enhancement by other means, without making use of the ground truth information. These include using the Camera Response Function (CRF) [30, 40], the image histogram [16, 28] or other assumptions [39, 12] to estimate the illumination, independent of any prior. In our proposed network, we use the histogram of the input dark image to automatically predict the amplification factor. To the best of our knowledge, this has not been attempted before for deep learning based low-light image enhancement.

CHAPTER 3

PROPOSED METHODOLOGY

We propose a new deep neural network architecture, called Low-Light Packing Network (LLPackNet) for enhancing an extreme low-light high-resolution image with low time–memory complexity. We first describe the overall network architecture, shown Fig. 3.1 and then analyze the core components of our network – the Pack and UnPack operations, in Sec. 3.2.

3.1 Network architecture

3.1.1 Image amplification

In general, dark images need to be pre-amplified before enhancing them. We estimate the amplification factor using the incoming RAW image $I_{i/p}^{HR}$ by constructing a 64 bin histogram, with the histogram bins being equidistant in the log domain. This provides a finer binning resolution for lower intensities and a coarser binning resolution for higher intensities. The histogram is used by a multilayer perceptron, with one hidden layer, to estimate the amplification factor.

3.1.2 Fast and light-weight enhancement

As discussed in Chap. 1, we want to perform most of the processing in LR space. Hence, our first step is to downsample the input image. For this purpose, we propose the *Pack* $\alpha\times$ operation, which downsamples the image by a factor of α along each dimension while increasing the number of channels by a factor of α^2. This is shown in Fig. 3.2 for $\alpha = 2$. Our goal is to perform $16\times$ downsampling, which we do in two stages. In the first stage, the Pack $2\times$ operation separates out the red, green and blue color components lying in the 2×2 Bayer pattern [13] of the amplified raw image $I_{i/p}^{HR}$. This reduces the spatial dimension to half and increases the number of channels from 1

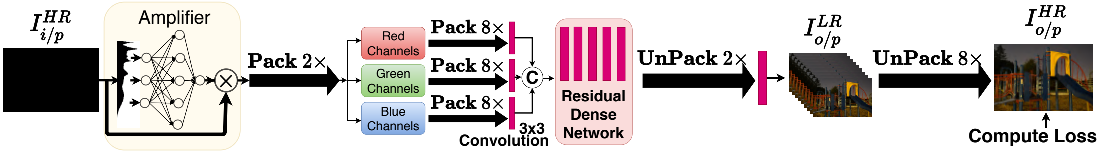


Figure 3.1: Our proposed network: LLPackNet.

to 4 (2^2). Once the colors are separated into these channels, a subsequent Pack $8\times$ operation is applied individually on each color channel, further reducing the spatial dimension by $8\times$ and increasing the number of channels from 1 to 64 (8^2). Now, using a 3×3 convolution kernel, the channel dimension of each color component is reduced such that on concatenation, the resulting feature map has 60 channels with $16\times$ lower resolution. The channel reduction at this stage is essential to prevent parameter and memory explosion in the downstream operations. It must also be noted that, if instead of RAW images, already demosaiced images sRGB images are fed to the network, the intial Pack $2\times$ operation can be omitted.

This downsampled representation is then processed by a series of convolution operations, in the Residual Dense Network [44] (RDN) – which consists of 3 residual dense blocks each with 6 convolutional layers and a growth rate of 32, but does not perform any down/up sampling operation. The output of the RDN now needs to be upsampled and for this we use the proposed *UnPack* $2\times$ operation, which is the inverse of *Pack* $2\times$. This reduces the number of channels from 60 to 15 ($60/2^2$). We then increase the channel-width of the feature map from 15 to 192 ($8^2 \times 3$) using 3×3 convolutions to enable subsequent upsampling. Except for this operation, all the computations are done in the $16\times$ lower resolution. We finally perform UnPack $8\times$ operation to get the restored image, $I_{o/p}^{HR}$.

The loss function consists of 3 components, as described by Ignatov *et al.* [14]:
1) Color loss: L1 loss between the ground-truth and the restored image after passing them through a Gaussian filter.
2) Content loss: L1 difference between the VGG-19 features of the ground truth and the restored image.
3) Total variation norm

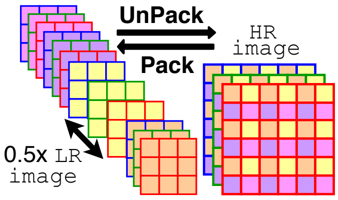


Figure 3.2: The Proposed Pack $\alpha\times$ and UnPack $\alpha\times$ operations, for down/up-sampling factor $\alpha = 2$.

3.2 Pack and UnPack operations

The last section discussed LLPackNet from the vantage point of network complexity. In this section we analyze the network from the standpoint of reconstruction quality and explain how the proposed *Pack* and *UnPack* operations result in better color restoration.

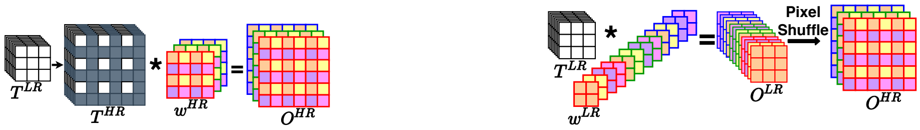
3.2.1 Improving color correlation with UnPack $\alpha\times$

Making abrupt transitions between LR and HR spaces, especially those with large difference in spatial resolutions, introduce several color artifacts in the restored image. In this subsection, we will show the effectiveness of Pack $\alpha\times$ and UnPack $\alpha\times$ operations in reducing these artifacts. Before going into the analysis of Pack and UnPack, it is necessary to understand the PixelShuffle operation [2, 34, 33], based on which they were formulated. Furthermore, we will also show that the Pack/UnPack operations lead to better color correlation than the PixelShuffle operation.

PixelShuffle:

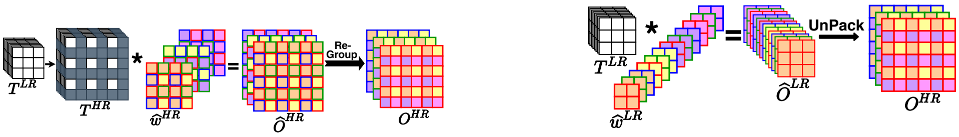
Consider the transposed convolution operation, with upsampling factor $\alpha = 2$ shown in Fig. 3.3 (a). T^{LR} refers to the penultimate feature map in the network, which is upsampled with zero padding and then convolved with w^{HR} in the HR space to obtain the restored image O^{HR}. We now explain the color coding used in the figure. When w^{HR} convolves with T^{HR}, for each shifted position of w^{HR}, only the weights in one set of colors in w^{HR} contribute to an output pixel in T^{HR}. This corresponding output pixel in O^{HR} has been labelled with the same color.

However, performing convolution operations in the high resolution (HR) space is computationally expensive. This problem can be solved by performing an equivalent



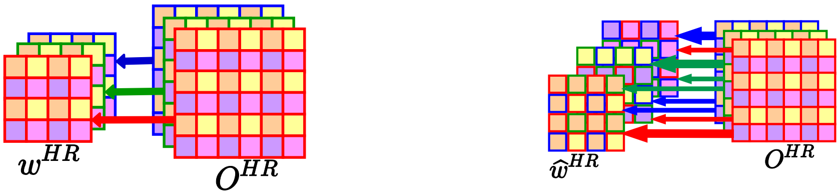
(a) The upsampling operation in HR. (b) Implementing (a) in LR using PixelShuffle.

Figure 3.3: Upsampling using PixelShuffle



(a) Upsampling, followed by regrouping. (b) Implementing (a) in LR using UnPack.

Figure 3.4: Upsampling using the UnPack operation



(a) Color correlation with PixelShuffle. (b) Color correlation with UnPack

Figure 3.5: Better color correlation with UnPack compared to PixelShuffle, since in UnPack all kernels of $\widehat{w}^{HR}$ are responsible for the colors in O^{HR} (Illustrated for $2\times$ upsampling).

operation in the LR space itself, as shown in Fig. 3.3 (b). This involves decomposing w^{HR} into spatially smaller kernels w^{LR} which are then convolved with T^{LR} to produce O^{LR}. Then, using the PixelShuffle operation, O^{LR} is rearranged to obtain O^{HR}. Thus, the same upsampling operation has been performed, but in a much shorter time, since the convolution operation has been restricted to the low resolution (LR) space.

In this scheme, the first one-third of the channels in O^{LR} always contribute to the red channel in O^{HR}, the next third to green and final third to blue, as shown in Fig. 3.3 (b). Translating this to Fig. 3.3 (a), we can observe that each kernel in w^{HR} maintains a monopoly on one of the red, green or blue color channels in the restored image O^{HR}. This means that there is little correlation among the color channels in restored image, see Fig. 3.5 (a). This weak correlation among the color channels of O^{HR}, leads to color artifacts as shown in Fig. 4.3.

UnPack:

The goal of UnPack operation is to improve upon PixelShuffle and enhance the correlation among the color channels of O^{HR}. Consider Fig. 3.4 (a) in which upsampling is performed using a color-shuffled version of weights $\widehat{w}^{HR}$. This provides the output $\widehat{O}^{HR}$, which is then re-grouped (unshuffled) in order to get the desired restored image.

This is a complicated two-stage operation in the HR space. But, the equivalent operation can be easily performed in the LR space, by just changing the order of the weights ($\widehat{w}^{LR}$) and then upsampling using the UnPack operation, as shown in Fig. 3.4 (b). Note that this operation has the same time-complexity as the PixelShuffle operation shown in Fig. 3.3 (b). The UnPack operation forces each consecutive triad of channels in O^{LR} to be a $2\times$ lower resolution version of the color image O^{HR} (also see Fig. 3.1). This way, the channels in O^{LR} corresponding to different colors of O^{HR} are nearby, which better preserves the color correlation because in a CNN nearby feature maps are heavily correlated [35]. Translating this to the high-resolution space weights ($\widehat{w}^{HR}$), we see that all the kernels of $\widehat{w}^{HR}$ are collectively responsible for all the colors in O^{HR}, as shown in Fig. 3.5 b). Thus, the UnPack operation leads to better color correlation than PixelShuffle. The practical effectiveness of the UnPack operation is demonstrated in Fig. 4.3.

Alternatively, this low color correlation can also be inferred by comparing the weights in the LR space (w^{LR}) of PixelShuffle (in Fig. 3.3 (b)) with the those ($\widehat{w}^{LR}$) of UnPack (in Fig. 3.4 (b)). In the case of PixelShuffle, the weight channels corresponding to each color occur together, i.e. the first four channels are red, the next four are green and the last four are blue, thus keeping weights for different colors apart. By contrast, in our UnPack operation, the RGB triads are always kept together in $\widehat{w}^{LR}$. Thus, noting a well known fact for a CNN, that nearby features bear a high correlation than spaced out ones, we can intuitively conclude that this is responsible for the better color restoration of our UnPack operation.

3.2.2 Increasing receptive field with Pack $\alpha\times$

Having a large receptive field is essential for capturing the contextual information in an image. A low receptive field leads to staircasing and artificial shocks [3, 4] in the restored image. Large receptive fields are also beneficial for good color restoration because they gather more contextual information.

Downsampling the incoming feature map using the novel Pack $\alpha\times$ operation equips LLPackNet with a large receptive field. To illustrate this fact, let us consider a large feature map I^{HR} which is downsampled to I^{LR} using Pack $10\times$ operation. Note that the neighboring pixels in I^{LR} are actually 10 pixels apart in I^{HR}. Also, the pixels along the channel dimension of I^{LR} are in a 10×10 neighborhood in I^{HR}. Thus, even using a 3×3 convolution kernel on I^{LR} with a stride of 1 leads to a receptive field of 900 pixels in I^{HR}. In contrast, to do a similar operation directly on I^{HR}, requires a 30×30 kernel with a stride of 10, which is impractical.

3.3 Numerical example of UnPack operation

Pack and UnPack operators perform intermixing of pixels for better color correlation, as shown in Fig. 3.2. To further motivate how the UnPack $2\times$ shuffling works, we display a worked-out example below.

Consider an input tensor of shape $2 \times 2 \times 12$ (spatial resolution 2×2 with 12 channels) as shown below.

Channel Count	Channel 1	Channel 2	Channel 3	$\cdots$	Channel 12
Channel	1 2	5 6	9 10	$\cdots$	45 46
Values	3 4	7 8	11 12	$\cdots$	47 48

Then, applying the UnPack $2\times$ operation we get a tensor of shape $4 \times 4 \times 3$ (spatial resolution 4×4 with 3 channels) as shown below.

```
Red (first) Channel
             [ 1, 13,  2, 14]
             [25, 37, 26, 38]
             [ 3, 15,  4, 16]
             [27, 39, 28, 40],

Green (second) Channel
             [ 5, 17,  6, 18]
             [29, 41, 30, 42]
```

```
        [ 7, 19,  8, 20]
        [31, 43, 32, 44],

Blue (third) Channel
        [ 9, 21, 10, 22]
        [33, 45, 34, 46]
        [11, 23, 12, 24]
        [35, 47, 36, 48]
```

3.4 Comparison of Pack/UnPack with other down/up-sampling methods

3.4.1 Upsampling

We have already shown the effectiveness of UnPack operation over the PixelShuffle operation in Sec. 4.3. Here, we compare with two other popular approaches – Transposed convolution as used by Chen *et al.* and Interpolation suggested by Odena *et al.* [25]. The transposed convolution is very slow as compared to the UnPack operation because it has to iterate the convolution kernel over the entire feature map. Moreover it increases the parameter count of the network. On the other hand, the interpolation technique suggested by Odena *et al.* has no learnable parameters but is still a slower operation. This can be seen in Table 3.1.

H×W; Channels	**Execution Time in Seconds**			**Number of Learnable Parameters**		
	TransposeConv2D	**UnPack**	**Interpolation**	**TransposeConv2D**	**UnPack**	**Interpolation**
1024 × 1024; 32 -> 2048 × 2048; 8	0.18	0.05	0.13	1032	–	–
256 × 256; 128 -> 512 × 512; 32	0.04	0.01	0.04	16416	–	–
32 × 32; 512 -> 64 × 64; 128	0.0025	0.0006	0.0025	262272	–	–

Table 3.1: We compare the complexity of three upsampling methods – Transposed Convolution (TransposeConv2D), Interpolation [25] and UnPack to perform 2× upsampling. The execution time and model parameters are provided for input feature maps of different spatial resolutions and channel dimensions, as listed in the first column. UnPack is 3–4× faster than the other techniques.

3.4.2 Downsampling

Max-pooling is the most popular technique for downsampling feature maps. This has been used in many deep learning methods, including the pipeline proposed by Chen *et al.* [6]. But for a large downsampling factor, max-pooling will cause a significant loss of information. For example, when doing an $8\times$ downsampling, max-pooling will choose only a single element from an 8×8 block. In addition, max-pooling also causes gradient sparsity during backpropagation.

Another popular downsampling technique is strided convolution, usually done with small kernels such as 3×3 or 5×5. But, for a large downsampling factor, say 8, a stride of 8 is required, and would lead to loss of information with such small kernels. To alleviate these issues of information loss and gradient sparsity, we used the Pack operation for downsampling feature maps which ensures a better restoration.

CHAPTER 4

EXPERIMENTS

4.1 Experimental settings

For extreme low-light single-image enhancement, we compare with Chen *et al.* [6], Gu *et al.* [11] and Maharjan *et al.* [23]. In addition, we also tried conventional techniques such as LIME [12] and Li *et al.* [20] but they did not work well for extreme low-light images. Both the training and test codes of these methods are publicly available and have been used for comparisons.

4.1.1 SID (See-in-the-Dark) dataset

For experiments on extreme low-light high resolution images, we use the See-in-the-Dark (SID) dataset [6], introduced by Chen *et al.* , which contains pairs of extreme low-light images and the corresponding ground truth reference images. The dataset has 5094 such pairs of images, of size 2848×4256, captured with a high definition full-frame Sony α7S II sensor. Unlike some methods that collect their dataset by simulating pairs of low-light and ground truth images [22, 26, 37, 29, 19, 7], SID provides physically captured images. Also, rather than using sRGB images, they provide images in the raw format. This is done so as to prevent compression artifacts and loss of crucial information incurred in the standard image processing pipeline of a camera, especially when applied in extreme low-light conditions.

4.1.2 LOL (LOw Light paired) dataset

We additionally show comparisons on the LOL dataset [7] to evaluate the performance of LLPackNet on low resolution images. It contains weakly illuminated VGA resolution (400×600) PNG compressed images. Additionally, the SID dataset comes with the ground truth and low-light image exposure information, which can be used for

Model	**Processing Time**	**Memory**	**Parameters**	**PSNR(dB) / SSIM**	
	(in seconds)	(in GB)	(in million)	**w/o GT exposure**	**using GT exposure**
Maharjan *et al.* [23]	120	10	2.5	20.98 / 0.49	28.41 / **0.81**
Gu *et al.* [11]	77	8	3.5	21.90 / 0.59	**28.53 / 0.81**
Chen *et al.* [6]	17	5	7.75	22.93 / 0.70	28.30 / 0.79
Chen *et al.* [6] + Our Amplifier	17	5	7.76	22.98 / **0.71**	28.30 / 0.79
LLPackNet (Ours)	**3**	**3**	**1.1**	**23.27** / 0.69	27.83 / 0.75

Table 4.1: Results on the SID dataset [6] for extreme low-light 2848×4256 images. Compared to existing approaches, we have 2–7× fewer model parameters, 2–3× lower memory, 5–20× speed up with competitive restoration quality.

estimating the amplification factor, but LOL has no such information. Therefore the amplification factor is found using our approach directly from the input image.

We use the train/test split as given in the respective datasets. For LLPackNet, patches of size 512×512 are used for training and full resolution for testing. For benchmarking, we use the PyTorch framework on Intel Xeon E5-1620V4 @ 3.50 GHz CPU with 64 GB RAM in order to accommodate computationally intensive methods. Adam optimizer was used for training, with a learning rate of 10^{-4}. Also, kernels of size 3x3 were used for the convolution operations.

4.2 Restoration results for extreme low-light images

We compare our algorithm with Chen *et al.* [6], Gu *et al.* [11] and Maharjan *et al.* [23] on the SID dataset, see Table 4.1 and Fig. 4.1. These methods use the ratio of GT exposure to that of the input dark image, available in the SID dataset, to pre-amplify the images. The corresponding results are shown under the label "Amplification using GT exposure" in Table 4.1 and Fig. 4.1. But, since the GT information will not be readily available in a real-world setting, we additionally show results in the absence of GT information. This is shown under the heading "w/o GT exposure". We additionally show results for "Chen *et al.* + Our Amplifier" in which our proposed amplifier is added to their algorithm. We have chosen Chen *et al.* since compared to the other existing methods, they have the least time and memory complexity. All the methods are appropriately retrained before evaluation. The metrics used for comparison are Peak Signal to Noise Ratio (PSNR) and Structural Similarity (SSIM).

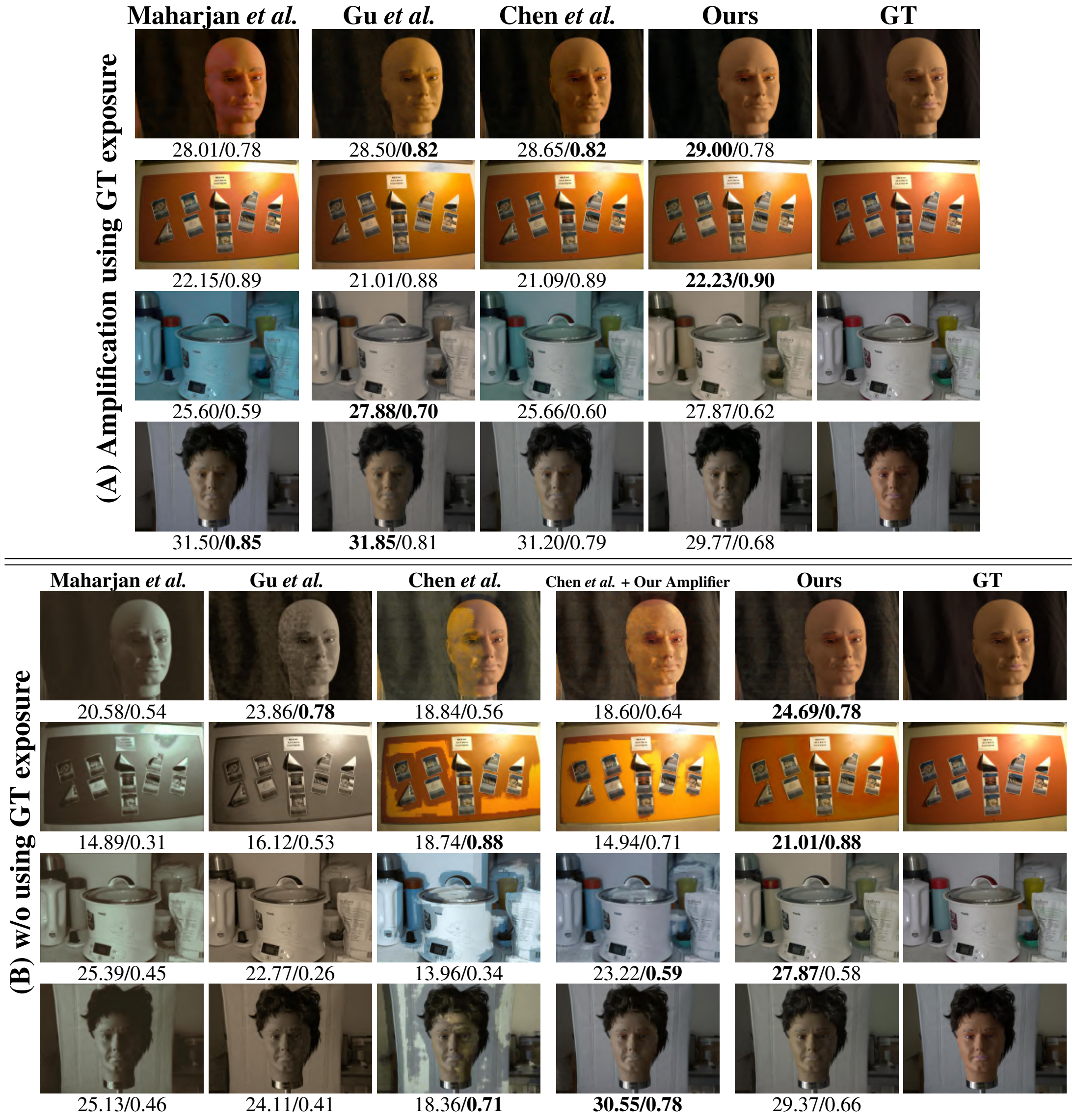


Figure 4.1: Performance comparison of the proposed LLPackNet with state-of-the-art algorithms corresponding to Table 4.1. **(A):** All the methods work well with GT exposure. **(B):** In a realistic scenario, where GT exposure is not available during inference, our LLPackNet gives the best restoration.

4.2.1 Network speed and memory utilization

As shown in Table 4.1, LLPackNet is $5-20\times$ faster with $2-3\times$ lower memory and $2-7\times$ lesser model parameters. We achieve this because we do the bulk of operations in $16\times$ lower resolution. Opposed to this, Maharjan *et al.* [23] do not perform any downsampling operation and hence, the feature maps propagating through their network are huge. This results in very high network latency and memory consumption. Likewise, Gu *et al.* [11] adopt a multi-scale approach that requires feature map propagation at $2\times$ and $4\times$ lower resolution. But this marginal downsampling is not sufficient to contain the network latency and memory consumption. Chen *et al.* [6]

have relatively better metrics by performing up to 32× downsampling. But this is done only in steps of 2 requiring five downsampling and five upsampling operations. Further, four out of five upsampling operations are done using transposed convolution [9], which is much slower than the proposed UnPack operation, see Sec. 3.4. Thus, Chen *et al.* have a moderately high processing time and memory utilization.

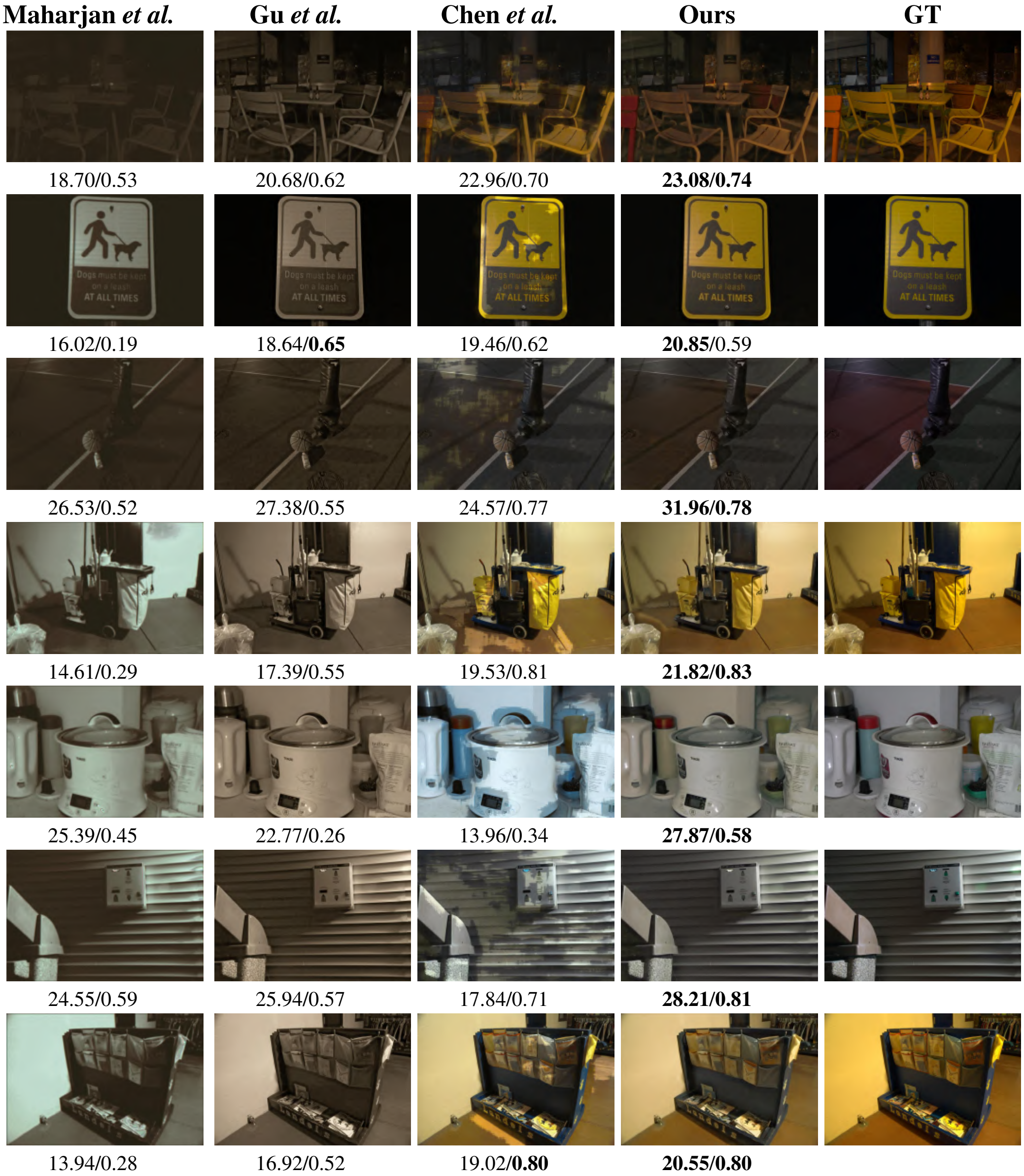


Figure 4.2: More visual comparisons on the SID dataset, without using the GT exposure information for amplification.

4.2.2 Restoration quality

All methods perform notably well when the GT exposure is available. But in a practical setting when GT exposure is not readily available, except for our network, the other methods struggle to restore proper colors. The results for this practical setting are also shown in Fig 1.1. Adding our amplifier module to Chen *et al.* improves their performance to some extent, but the restored images still exhibit noisy patches, artifacts and color cast. This is because amplification is not the only factor in improving the performance of a network. Rather, having a large receptive field, which provides more contextual information, and better correlation among the color channels is more important than the correct amplification factor in terms of restoration quality. To further assess these claims, refer to the ablation studies in section 4.3, in which show that LLPackNet continues to give structurally consistent results even when the amplifier is removed.

4.3 Ablation studies on LLPackNet

We now show ablation studies on LLPackNet to better understand the contribution of individual components of the network. For each ablation study the network is appropriately retrained.

4.3.1 UnPack vs. PixelShuffle

As a first ablation study, we replace the UnPack operation in the proposed LLPackNet with the PixelShuffle operation [33] and the results are shown in Fig. 4.3. We notice that this leads to color cast and abrupt change in colors, which is not characteristic of photographic images. On the other hand, UnPack operation enhances the color correlation in the restored image which mitigates the color artifacts and color cast to a significant extent. Using the UnPack operation in place of PixelShuffle improves the PSNR/SSIM from 22.72 dB/0.68 to 23.27dB/0.69.

4.3.2 Importance of the amplifier module

Fig. 4.4 shows the restoration results using LLPackNet with and without the amplifier. Without the amplifier, the colors are paler, tend to be monochromatic and do not match the desired color hue. It is worth noting that the absence of the amplifier affects only

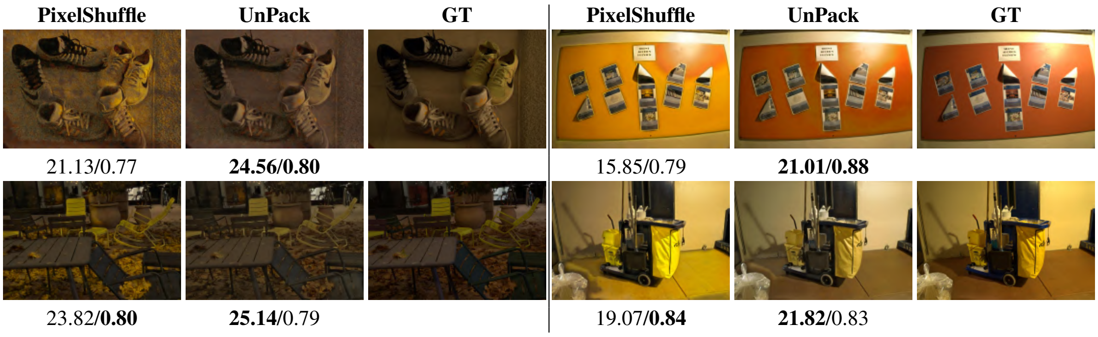


Figure 4.3: Comparison between the two upsampling methods with LLPackNet – PixelShuffle and UnPack, evaluated on the SID dataset. The UnPack operation helps to achieve better color restoration by reducing color distortions.

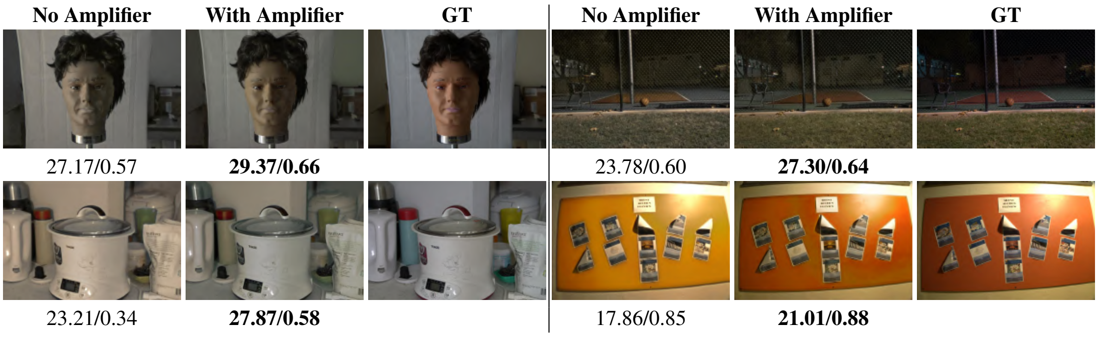


Figure 4.4: Results using the proposed LLPackNet on the SID dataset with and without amplification estimation. Without amplification, the colors are pale and tend to be monochromatic.

the color restoration. The distortions and artifacts exhibited by Chen *et al.* (refer Fig. 4.1 (B)) are still absent. This is due to the large receptive field provided by the Pack operation. With the amplifier, the performance of the network improves from 22.53dB/0.66 to 23.27dB/0.69.

Finally, we trained and tested LLPackNet by simultaneously applying both modifications. The combined effect of using the proposed UnPack operation, instead of PixelShuffle, and estimating proper amplification, boosts the average PSNR/SSIM from 21.35dB/0.60 to 23.27dB/0.69.

4.4 LLPackNet for low-resolution images

The SID dataset contains high definition images, thereby, giving us the liberty to chose a large downsampling factor of 16. This leads us to the question: Can LLPackNet also work for low-resolution images? When low-resolution images are downsampled using a large factor, the intra-channel correlation in the downsampled image is reduced,

Model	Processing Time	PSNR (dB)	SSIM
Chen *et al.* [6]	0.21 sec.	18.82	0.73
LIME [12]	0.19 sec.	16.94	0.60
Li *et al.* [20]	17.89 sec.	13.85	0.65
Gu *et al.* [11]	0.41 sec.	19.46	**0.75**
LLPackNet-8× (Proposed)	**0.06** sec.	**19.61**	0.69
LLPackNet-4× (Proposed)	0.24 sec.	19.60	0.74

Table 4.2: Results on the LOL dataset [7] consisting of weakly illuminated sRGB images of resolution 400 × 600. LLPackNet with 8× downsampling (LLPackNet-8×) is very fast, but has a low SSIM due to large factor downsampling of an already small image. LLPackNet-4× opts for 4× downsampling to achieve better reconstruction as reflected in the SSIM value.

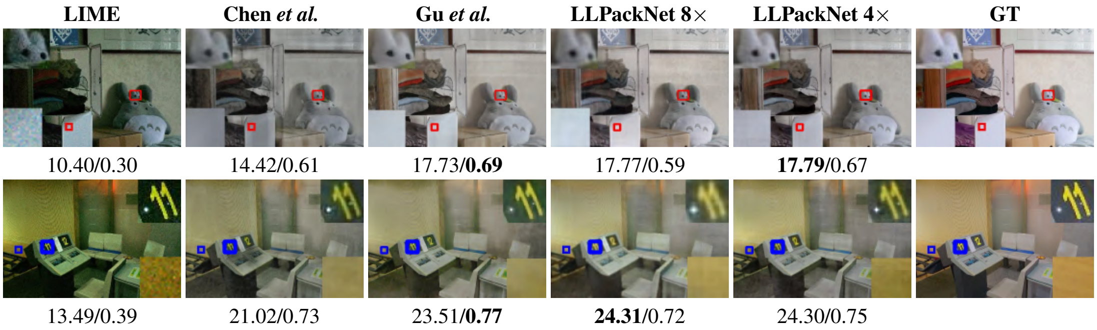


Figure 4.5: Visual results on the LOL dataset corresponding to Table 4.2. LLPackNet-8× is very fast with good color restoration but exhibits a slight blur due to large factor downsampling on a low resolution image. LLPackNet-4×, with a smaller downsampling factor reduces the blur improves the SSIM.

which negatively impacts the restoration. To investigate this, we conducted experiments on the LOL dataset [7] containing weakly illuminated images at VGA resolution of 400×600. As the images in the LOL dataset are already in the compressed PNG (sRGB) format, the 2× downsampling at the beginning of LLPackNet to separate out the Bayer pattern [13] is not required. Thus, the effective downsampling is only 8× and we denote this network by LLPackNet-8×. The results are shown in Table 4.2. Once again, LLPackNet has the lowest processing time since it operates in the LR space.

We observe that the large receptive field of LLPackNet enhances the denoising and color restoration capabilities. But, at the same time, a slight blur is introduced, since we are using a large downsampling factor for images which are already of low-resolution. To verify that the blur is because of large downsampling, we retrain LLPackNet on the LOL dataset with 4× downsampling, which we denote as LLPackNet-4×. With this model, we obtain sharper results with higher SSIM values, as shown in Fig. 4.5.

CHAPTER 5

SCOPE OF FUTURE WORK

The presented work opens many opportunities for future research. Through the amplifier module, we have introduced a way to perform amplification without relying on ground truth information. However, the PSNR/SSIM using our amplifier module is significantly lower compared to the case where ground truth (GT) exposure based amplification is done, as can be seen in Table 4.1. For our amplifier module, we have made use of the histogram of the input image to estimate the amplification factor, using a small neural network. Further research into this area can be conducted in order to improve the metrics of the amplifier and bring it closer to the level of manual GT-based amplification.

Another possible extension of our work is to apply LLPackNet to raw images captured with different color filters. As part of the SID dataset, Chen *et al.* [6] provide two datasets - the Sony dataset captured using the Bayer color filter array and the Fuji dataset captured using the X-Trans color filter array. Our experiments have been performed using only the Sony dataset. However, it can also be extended to the Fuji dataset, albeit with some modifications in the pipeline, such as replacing Pack $2\times$ with Pack $3\times$ and then applying a pixel permutation operation, similar to what was applied by Chen *et al.* [6].

Furthermore, our Pack and UnPack operations can be applied to applications outside of low light enhancement. For example, UnPack could be applied in the context of super resolution as a replacement to PixelShuffle [33]. Since we have shown that our UnPack operation provides a better color restoration quality than PixelShuffle, it is expected that this advantage will also be seen while applying it to the problem of super-resolution, or any other application which demands better color restoration.

CHAPTER 6

CONCLUSION

Attempts to improve the restoration of extreme low-light images have lead to increasingly complex networks. But, for image enhancement solutions to become robust to common low-end devices, they must operate in a limited time-memory budget. Accordingly, we proposed *LLPackNet* which restricts the bulk of computations to a low-resolution space by performing large factor down-sampling and up-sampling using the novel Pack and UnPack operations. The proposed Pack and UnPack operations provide LLPackNet with more contextual information and better-correlated color channels resulting in a better restoration, free from unnatural artifacts or color cast. In addition to faster restoration, we also introduced a novel amplifier module, which can perform amplification using only the input image, without relying on ground truth exposure information, making it suitable for unknown scenes. As a result of these features, LLPackNet is 5–20× faster and 2–3× lighter and yet maintains a competitive restoration quality compared to the state-of-the-art algorithms.

REFERENCES

[1] **Ai, S.** and **J. Kwon** (2020). Extreme low-light image enhancement for surveillance cameras using attention u-net. *Sensors*, **20**(2), 495.

[2] **Aitken, A.**, **C. Ledig**, **L. Theis**, **J. Caballero**, **Z. Wang**, and **W. Shi** (2017). Checkerboard artifact free sub-pixel convolution: A note on sub-pixel convolution, resize convolution and convolution resize. *arXiv preprint arXiv:1707.02937*.

[3] **Buades, A.**, **B. Coll**, and **J. . Morel** (2006). The Staircasing Effect in Neighborhood Filters and its Solution. *IEEE Transactions on Image Processing*, **15**(6), 1499–1505.

[4] **Buades, A.**, **B. Coll**, and **J.-M. Morel**, Image Denoising by Non-Local Averaging. *In International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 2. IEEE, 2005.

[5] **Cai, Y.** and **U. Kintak**, Low-light image enhancement based on modified u-net. *In 2019 International Conference on Wavelet Analysis and Pattern Recognition (ICWAPR)*. IEEE, 2019.

[6] **Chen, C.**, **Q. Chen**, **J. Xu**, and **V. Koltun**, Learning to see in the dark. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018.

[7] **Chen Wei, W. Y. J. L., Wenjing Wang**, Deep retinex decomposition for low-light enhancement. *In British Machine Vision Conference*. 2018.

[8] **Cheng, Y.**, **J. Yan**, and **Z. Wang**, Enhancement of weakly illuminated images by deep fusion networks. *In 2019 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2019.

[9] **Dumoulin, V.** and **F. Visin** (2016). A guide to convolution arithmetic for deep learning. *arXiv preprint arXiv:1603.07285*.

[10] **Ghosh, S.** and **K. N. Chaudhury**, Fast bright-pass bilateral filtering for low-light enhancement. *In 2019 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2019.

[11] **Gu, S.**, **Y. Li**, **L. V. Gool**, and **R. Timofte**, Self-guided network for fast image denoising. *In Proceedings of the IEEE International Conference on Computer Vision*. 2019.

[12] **Guo, X.**, **Y. Li**, and **H. Ling** (2016). Lime: Low-light image enhancement via illumination map estimation. *IEEE Transactions on Image Processing*, **26**(2), 982–993.

[13] **Hirakawa, K.** and **T. W. Parks** (2005). Adaptive homogeneity-directed demosaicing algorithm. *IEEE Transactions on Image Processing*, **14**(3), 360–369.

[14] **Ignatov, A.**, **N. Kobyshev**, **R. Timofte**, **K. Vanhoey**, and **L. Van Gool**, Dslr-quality photos on mobile devices with deep convolutional networks. *In Proceedings of the IEEE International Conference on Computer Vision*. 2017.

[15] **Jenicek, T.** and **O. Chum**, No fear of the dark: Image retrieval under varying illumination conditions. *In Proceedings of the IEEE International Conference on Computer Vision*. 2019.

[16] **Kim, Y.-T.** (1997). Contrast enhancement using brightness preserving bi-histogram equalization. *IEEE transactions on Consumer Electronics*, **43**(1), 1–8.

[17] **Lee, H.**, **K. Sohn**, and **D. Min** (2020). Unsupervised low-light image enhancement using bright channel prior. *IEEE Signal Processing Letters*, **27**, 251–255.

[18] **Li, B.**, **S. Wang**, and **Y. Geng**, Image enhancement based on retinex and lightness decomposition. *In 2011 18th IEEE International Conference on Image Processing*. IEEE, 2011.

[19] **Li, C.**, **J. Guo**, **F. Porikli**, and **Y. Pang** (2018). Lightennet: a convolutional neural network for weakly illuminated image enhancement. *Pattern Recognition Letters*, **104**, 15–22.

[20] **Li, M.**, **J. Liu**, **W. Yang**, **X. Sun**, and **Z. Guo** (2018). Structure-revealing low-light image enhancement via robust retinex model. *IEEE Transactions on Image Processing*, **27**(6), 2828–2841.

[21] **Lim, B.**, **S. Son**, **H. Kim**, **S. Nah**, and **K. Mu Lee**, Enhanced deep residual networks for single image super-resolution. *In Proceedings of the IEEE conference on computer vision and pattern recognition workshops*. 2017.

[22] **Lore, K. G.**, **A. Akintayo**, and **S. Sarkar** (2017). Llnet: A deep autoencoder approach to natural low-light image enhancement. *Pattern Recognition*, **61**, 650–662.

[23] **Maharjan, P.**, **L. Li**, **Z. Li**, **N. Xu**, **C. Ma**, and **Y. Li**, Improving extreme low-light image denoising via residual learning. *In 2019 IEEE International Conference on Multimedia and Expo (ICME)*. IEEE, 2019.

[24] **Malik, S.** and **R. Soundararajan**, Llrnet: A multiscale subband learning approach for low light image restoration. *In 2019 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2019.

[25] **Odena, A.**, **V. Dumoulin**, and **C. Olah** (2016). Deconvolution and checkerboard artifacts. *Distill*. URL `http://distill.pub/2016/deconv-checkerboard`.

[26] **Park, S.**, **S. Yu**, **B. Moon**, **S. Ko**, and **J. Paik** (2017). Low-light image enhancement using variational optimization-based retinex model. *IEEE Transactions on Consumer Electronics*, **63**(2), 178–184.

[27] **Pisano, E. D.**, **S. Zong**, **B. M. Hemminger**, **M. DeLuca**, **R. E. Johnston**, **K. Muller**, **M. P. Braeuning**, and **S. M. Pizer** (1998). Contrast limited adaptive histogram equalization image processing to improve the detection of simulated spiculations in dense mammograms. *Journal of Digital imaging*, **11**(4), 193.

[28] **Pizer, S. M.**, **E. P. Amburn**, **J. D. Austin**, **R. Cromartie**, **A. Geselowitz**, **T. Greer**, **B. ter Haar Romeny**, **J. B. Zimmerman**, and **K. Zuiderveld** (1987). Adaptive histogram equalization and its variations. *Computer vision, graphics, and image processing*, **39**(3), 355–368.

[29] **Ren, W.**, **S. Liu**, **L. Ma**, **Q. Xu**, **X. Xu**, **X. Cao**, **J. Du**, and **M.-H. Yang** (2019). Low-light image enhancement via a deep hybrid network. *IEEE Transactions on Image Processing*, **28**(9), 4364–4375.

[30] **Ren, Y.**, **Z. Ying**, **T. H. Li**, and **G. Li** (2018). Lecarm: low-light image enhancement using the camera response model. *IEEE Transactions on Circuits and Systems for Video Technology*, **29**(4), 968–981.

[31] **Ronneberger, O.**, **P.Fischer**, and **T. Brox**, U-net: Convolutional networks for biomedical image segmentation. *In Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, volume 9351 of *LNCS*. Springer, 2015.

[32] **Shen, L.**, **Z. Yue**, **F. Feng**, **Q. Chen**, **S. Liu**, and **J. Ma** (2017). Msr-net: Low-light image enhancement using deep convolutional network. *arXiv preprint arXiv:1711.02488*.

[33] **Shi, W.**, **J. Caballero**, **F. Huszár**, **J. Totz**, **A. P. Aitken**, **R. Bishop**, **D. Rueckert**, and **Z. Wang**, Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. *In Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.

[34] **Shi, W.**, **J. Caballero**, **L. Theis**, **F. Huszar**, **A. Aitken**, **C. Ledig**, and **Z. Wang** (2016). Is the deconvolution layer the same as a convolutional layer? *arXiv preprint arXiv:1609.07009*.

[35] **Szegedy, C.**, **V. Vanhoucke**, **S. Ioffe**, **J. Shlens**, and **Z. Wojna**, Rethinking the inception architecture for computer vision. *In Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.

[36] **Wang, M.**, **B. Liu**, and **H. Foroosh**, Factorized convolutional neural networks. *In Proceedings of the IEEE International Conference on Computer Vision Workshops*. 2017.

[37] **Wang, R.**, **Q. Zhang**, **C.-W. Fu**, **X. Shen**, **W.-S. Zheng**, and **J. Jia**, Underexposed photo enhancement using deep illumination estimation. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019.

[38] **Wang, W.**, **C. Wei**, **W. Yang**, and **J. Liu**, Gladnet: Low-light enhancement network with global awareness. *In 2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)*. IEEE, 2018.

[39] **Yang, K.-F.**, **X.-S. Zhang**, and **Y.-J. Li** (2019). A biological vision inspired framework for image enhancement in poor visibility conditions. *IEEE Transactions on Image Processing*, **29**, 1493–1506.

[40] **Ying, Z.**, **G. Li**, **Y. Ren**, **R. Wang**, and **W. Wang**, A new low-light image enhancement algorithm using camera response model. *In Proceedings of the IEEE International Conference on Computer Vision Workshops*. 2017.

[41] **Yu, S.-Y.** and **H. Zhu** (2017). Low-illumination image enhancement algorithm based on a physical lighting model. *IEEE Transactions on Circuits and Systems for Video Technology*, **29**(1), 28–37.

[42] **Zhang, J.**, **R. Liu**, **L. Ma**, **W. Zhong**, **X. Fan**, and **Z. Luo**, Principle-inspired multi-scale aggregation network for extremely low-light image enhancement. *In IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2020.

[43] **Zhang, S.**, **Y. Lin**, and **H. Sheng**, Residual networks for light field image super-resolution. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019.

[44] **Zhang, Y.**, **Y. Tian**, **Y. Kong**, **B. Zhong**, and **Y. Fu**, Residual dense network for image super-resolution. *In Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018.