# DualBLN: Dual Branch LUT-aware Network for Real-time Image Retouching [⋆]

Xiang Zhang[1] [0000−0003−3270−0020], Chengzhe Lu[1] [0000−0002−6203−8069], Dawei Yan[1] [0000−0001−5202−0255], Wei Dong[1] [0000−0003−0263−3584] [⋆⋆], and Qingsen Yan[2] [0000−0003−1010−3540] [⋆⋆]

[1] Xi'an University of Architecture and Technology
[2] Northwestern Polytechnical University

**Abstract.** The 3D Lookup Table (3D LUT) is an efficient tool for image retouching tasks, which models non-linear 3D color transformations by sparsely sampling them into a discrete 3D lattice. We propose **DualBLN** (Dual Branch LUT-aware Network) which innovatively incorporates the data representing the color transformation of 3D LUT into the real-time retouching process, which forces the network to learn the adaptive weights and the multiple 3D LUTs with strong representation capability. The estimated adaptive weights not only consider the content of the raw input but also use the information of the learned 3D LUTs. Specifically, the network contains two branches for feature extraction from the input image and 3D LUTs, to regard the information of the image and the 3D LUTs, and generate the precise LUT fusion weights. In addition, to better integrate the features of the input image and the learned 3D LUTs, we employ bilinear pooling to solve the problem of feature information loss that occurs when fusing features from the dual branch network, avoiding the feature distortion caused by direct concatenation or summation. Extensive experiments on several datasets demonstrate the effectiveness of our work, which is also efficient in processing high-resolution images. Our approach is not limited to image retouching tasks, but can also be applied to other pairwise learning-based tasks with fairly good generality. Our code is available at https://github.com/120326/DualBLN.

**Keywords:** 3D lookup table · Photo retouching · Color enhancement.

## 1 Introduction

Along with the promotion of digital image technology, it is common for people to use cameras or cell phones to take photos and record beautiful moments. However, most people will press the shutter randomly and the photos taken

[⋆] This work is supported by the Fundamental Research Funds for the Central Universities(No. D5000220444) and the Natural Science Basic Research Program of Shaanxi (2021JLM-16) and the Yulin Science and Technology Plan Project (CXY-2020-063)
[⋆⋆] Corresponding authors: Qingsen Yan (qingsenyan@gmail.com) and Wei Dong (dongwei156@outlook.com)

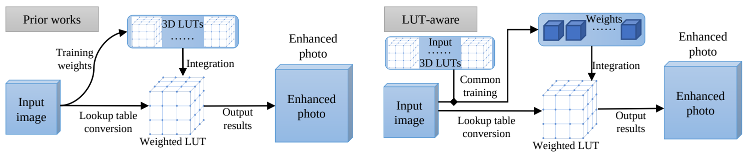

**Fig. 1.** Comparison between previous works and our method. Previous works (left) only uses images as the only input to the network when training the model, while our LUT-aware (right) applies look-up table information on the input side.

will be affected by external factors such as weather and time of day, and the original photos taken directly may have darker or overly bright areas. Due to this, the photos directly generated by the shooting equipment still need further post-processing to improve their visual quality. The retouching process requires the use of professional software that is very difficult to start, requires sufficient expertise reserves, and is quite time-consuming. These problems directly lead to the high threshold and tedious work for manual photo retouching.

The simplest solution for this task can be achieved to some extent by expert pre-stored color filters, but the poor generalization of fixed retouching methods is not a good solution. The 3D lookup table is a mapping relationship that can handle low-quality images well. The areas near the edges of the lookup table can be targeted for color conversion of light and dark areas to reduce noise and achieve better mapping results. However, the internal parameters of the lookup table depend on the expert's preset, and it is complex to target the optimization for a specific image. In recent years, combining traditional photo retouching techniques with deep learning has gained attention in academia. For example, Zeng *et al.* [45] combined multiple lookup tables with convolutional neural networks to propose an image-adaptive 3D lookup table method. Based on this, Liang *et al.* [24] focused on portrait photo retouching by segmenting foreground portraits and rear scenes, then realizing weighted optimization of different regions to give more attention to portrait regions. However, previous works are not flexible enough to use the information of 3DLUT for fusion.

Benefiting from the above works, we believe that lookup table parameters are a non-negligible element in image retouching tasks. No model directly uses the 3D LUT itself as a learnable object but only optimizes the parameters inside the 3D LUT using an optimizer from the field of deep learning to learn a pre-processed 3D LUT model for image retouching task. When performing transformations related to image modification, the network needs to know the internal data of the lookup table to perform targeted optimization. In other words, passively changing the lookup table parameters based on the model retouching results is not the optimal solution for image retouching since the information about these parameters is not fully used in the learning model.

To alleviate this problem, we propose the dual branch LUT-aware network named DualBLN which consists of image branch and LUT branch. In Fig. 2, the image input branch is used to process unretouched raw images. This branch

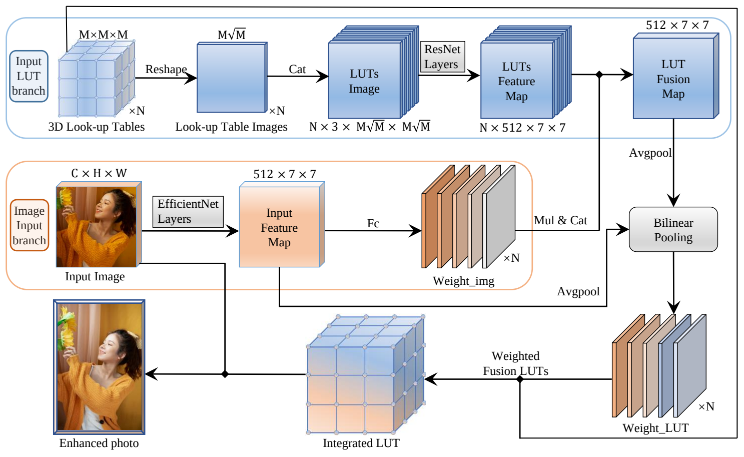

**Fig. 2.** Schematic diagram of the framework of the network structure. Given an unretouched input image, it is passed through the classifier network and used to generate weights and fuse the $N$ LUTs after flattening.

focuses on three dimensions: network depth, width and image resolution, using EfficientNet [33] to extract features from the images and generate weights for the initial fusion of lookup table information. The LUT input branch is the module responsible for processing the 3D LUTs fed into the network. The 3D lookup table data is converted into 2D data that approximates the shape of the input image and fed into ResNet [12] with the fully connected layer removed for the generation of the feature map and the initial fusion with the image weights, resulting in the LUT fusion map. The inclusion of lookup table parameters allows for an adaptive image modification model based on LUT perception and thus generates more accurate lookup table fusion weights. To make the features fused as much as possible and retain the information, bilinear pooling [25] is used to generate 3D LUT fusion weights. Multiplied using the outer product at each location of the feature map and pooled across locations to combine the feature information. The outer product captures the pairwise correlations between feature channels and can model the interactions between dual branch features.

The contributions of this paper are as follows:

- In order to better utilize the internal information of the lookup table to achieve more exquisite image retouching effects, we propose a new two-branch network structure named DualBLN, which innovatively uses the LUT as part of the input to implement an adaptive LUT-aware model structure.
- In an effort to reasonably fuse the two intermediate feature matrices generated by the bifurcated structure, bilinear pooling is used for feature fusion to reduce the negative effects of two different feature matrices that are too different and also to improve the overall robustness of the model.
- Extensive experiments are conducted on three datasets. The results demonstrate that our method is reasonable and efficient.

## 2 Related work

**LUT in camera imaging process.** The process of camera imaging is very complex, with optical systems such as signal processing and imaging systems focusing the image on the imaging element. This process includes image enhancement modules, including adjustment of exposure, contrast, color style, and a number of other operations [43,5,8,21,3,29,20,27,23]. According to [16], most of these modules in real systems use a common technique, namely LUT (Look-up table). However, the use of these LUTs is not flexible enough require manual debugging by experts if they want to change their parameters. Therefore, [45] used this as their hair to study image adaptive 3D LUTs to improve the expressiveness and flexibility of photo enhancement in the imaging pipeline.

**Learning-based image enhancement.** Image retouching aims to improve the quality of the original image. With the development of deep learning networks, significant progress has been made in this task [4,7,11,14,17,18,28,30,32,46,19,35,41]. Gharbi *et al.* proposed the HDRNet [7] that put most of the computation on downsampled images. It can be regarded as a masterpiece along the lines of a bilateral filter. He *et al.* [11] proposed the CSRNet that extracted global features with normal CNN and modulated the features after 1×1 convolution. CSRNet [11] used an end-to-end approach that is easy to train.

Image restoration [3,10,36,26,44,6,47,42] aims to remove image distortions caused by various situations. Since manual retouching requires a mass of expert knowledge and subjective judgment, many learning-based image restoration methods have emerged recently. Wang *et al.* [36] proposed a neural network for photo enhancement using deep lighting estimation to fix underexposed photos. Recently Mahmoud *et al.* [1] proposed a model that can perform multi-scale exposure correction for images. Guo *et al.* [9] proposed a model based on a luminance enhancement curve, which is iterated continuously to achieve gradual contrast and luminance enhancement of low-light images. As for the noise reduction task, Huang *et al.* [13] presented a model which generates two independent noise-bearing images of similar scenes to train the noise-reduction network.

Some researchers [28,17,39,40,38] have adapted the network structure to pursue the enhancement of the original image. In addition to expert retouching, Han-Ul *et al.* [18] developed PieNet to solve the problem of personalization in image enhancement. These works demonstrate that image enhancement has splendid results based on paired datasets. However, the collection of paired datasets is expensive. Therefore, image enhancement methods based on unpaired datasets started to receive attention. Some GAN-based methods [4,14] do not require pairs of input and target images. Meanwhile, Jongchan *et al.* [30] proposed the Distort-and-Recover method focusing on image color enhancement, which only required high-quality reference images for training. While existing image enhancement models have good generality, our work focuses on exploring the role of look-up table information in the network architecture, a novel approach that makes it an under-explored task.

**Deep learning methods with 3D LUT.** In previous studies, Zeng *et al.* [45] combined the deep learning-based weight predictor and 3D LUT for the first time, which achieved outstanding results. Then, multiple look-up tables were dynamically combined to capture image characteristics and perform the enhancement. [45] was a representative work that combined the deep learning paradigm with the traditional image enhancement paradigm, which presented the community with a new perspective on how to implement image enhancement. Although the most significant contribution of Liang *et al.* [24] is to provide a high-quality paired portrait photo retouch dataset, they changed the network architecture to use ResNet [12] instead of the simple CNN in the network of Zeng *et al.* [45] as an image classifier and achieved some improvements in the results, which inspired us to replace some of the structures in the network, allowing the final generation of more reasonable weights for fusing LUTs. Subsequently, Jo *et al.* [15] implemented a similar look-up process from input to output values by training a deep super-resolution network and transferring the output values of the deep learned model into the LUT. Wang *et al.* [37] further optimized the whole structure by employing UNet to generate the output of the encoding part as the weights for fusing multiple LUTs and using the output of the decoding part as the spatially-aware weights for the whole image to further optimize the interpolation results of the 3D LUT.

## 3 Methodology

### 3.1 3D LUT of traditional image enhancement

Originally a traditional technique widely used in the field of screen color imaging display, 3D LUT has been combined with deep learning in recent years to perform image enhancement work.

As shown in Fig. 3(a), it can be simply interpreted as a 3D lattice consisting of three 1D LUTs, corresponding to the RGB color channels, and a total of $M^3$ elements. The input RGB value is mapped in three dimensions according to a look-up table to obtain the converted color. The input image is defined as $I$, and the converted image is $O$ through the look-up table, then the conversion process can be expressed by the formula:

$$O_{(i,j,k)} = \varphi(I^R_{(i,j,k)}, I^G_{(i,j,k)}, I^B_{(i,j,k)}), \tag{1}$$

where $(i, j, k)$ are the coordinates corresponding to the pixel color of the image in RGB space, and $\varphi$ is defined as the conversion method from the input color value to the output color value. Usually, $M$ is usually set to 33 in practice, and the range of coordinates (0 to 255) taken under the three channels is much larger than the set value $M$. As shown in Fig. 3(b), the introduction of trilinear interpolation can solve this problem. In this strategy, the LUT can be flattened and fed into the network, we make $M = 36$, and this part of the work will be described in detail in the following.

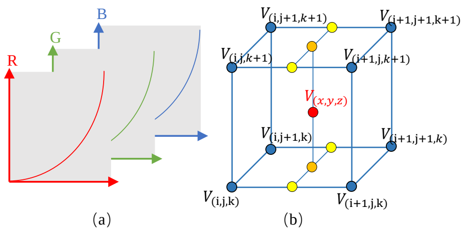

**Fig. 3.** (a) The 3D LUT is a cube containing $M^3$ nodes, but each coordinate system can be regarded as a separate 1D LUT. (b) The trilinear interpolation can compensate for the problem of an insufficient range of coordinates, and calculate the values of other points in the cube by giving the values of the vertices.

3D LUT is applied to conventional camera imaging to improve the image quality of the camera and also to reduce the effect of color difference between various devices. For image retouching tasks, the part of the lookup table that is closer to the black area inside the lookup table specializes in color mapping the dark areas of the original image, improving the details in dark scenes, and achieving better visual effects. Correspondingly, the part of the lookup table closer to white is also good at handling overexposed areas of the image. For other normal imaging areas, color conversion can also become more in line with human visual preferences. In summary, 3D LUT is a very suitable color conversion model for our work, as it can repair defects in photos and make images more brilliant.

The retouching style of the image shifts with the content of the shot, so a single 3D LUT does not work well with a wide variety of photos. As the prior art in 3D LUT paper, we use fused 3D LUTs, *i.e.*, we fuse multiple LUTs according to the weights generated by the classifier and thus achieve differentiation of retouching results.

## 3.2 Proposed method

Retouching images using 3D LUTs is efficient and fast, and the individual pixel color values of an image are found and mapped to produce a retouched image. Our model uses a two-branch network to train weights and fuse multiple 3D LUTs based on these weights to achieve a differentiated retouching result. The network contains two starting points, corresponding to the dual branches in the architecture, for input photos and lookup tables, respectively. Thus it can be subdivided into four segments based on their functions: *the input image branch module, the input LUT branch module, the feature map fusion module, and the weighted fusion of 3D LUTs module.*

**Image input branch.** Unretouched images are essential as the input to the network, and traditional deep learning methods generate weights directly from the input images for subsequent weighted fusion. We adopt EfficientNet [33] as the network layer of the image input branch, but we additionally reserve the middle feature map for the subsequent feature map fusion module, in addition to generating weights for fusing LUTs.

**Input LUT branch.** Since the structure of a 3D LUT is different from that of a 2D image, an additional deformation operation is required before feeding it into the network, which simply means flattening the data into a picture-like storage pattern and feeding the flattened LUTs image into the network as input. ResNet [12] is chosen as the network architecture for this branch, but we remove the final fully connected layer and keep only the LUT feature maps up to the middle, which is fused according to the weights generated by the image input branch to produce the final LUT fusion map.

**Feature map fusion.** The previous two branching modules ensure that the dimensions of the feature maps fed into the fusion module are the same. For the two feature maps extracted from Image and 3D LUT, the vector of the two fused features is obtained by bilinear pooling [25], which is used to generate the final weights and used to weight the different 3D LUTs.

**Weighted fusion of 3D LUTs.** This is the most intuitive part of image retouching, the original image is directly fused with the weighted 3D LUT to find the mapping relationship in the lookup table pixel by pixel, and finally, the image is mapped into a retouched and enhanced photo with better visual effect.

## 3.3 The image processing module

Simply using the set ResNet [12] does not achieve the best classification results, and if we want to further improve the accuracy of the classifier for image classification, it is a good way to replace it with EfficientNet [33], which is more suitable for image classification, it explores the effects of input resolution, depth, and width of the network, and uses Neural Architecture Search to search for a rationalized configuration of the above three parameters, which improves the image classification accuracy while achieving a smaller parameter computation. Considering all aspects, EfficientNet [33] is more suitable for our task and can be used for image processing to obtain intermediate feature matrices that are more conducive to subsequent computations and more accurate classification weights.

However, EfficientNet [33] is still not perfect. In our experiments, we found that although it looks fast and lightweight, it is very memory cost in practice. However, this problem can be avoided by reasonable settings.

EfficientNet [33] contains a total of eight branching categories from B0 to B7, and the accuracy of the classification and the number of parameters required for calculation are increasing in order. After consideration, we choose B0 as our image classifier, although the other models have better accuracy, their increased computational cost cannot be ignored. Considering the memory usage and the fact that B0 already has very good training results, this choice is an extremely cost-effective optimization decision.

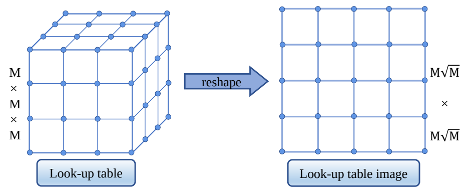

**Fig. 4.** A simple illustration of a look-up table directly subjected to reshapee operation to obtain a flattened LUT-image that can be fed into a neural network.

### 3.4 The LUT-aware module

Look-up tables (LUTs) are widely used for color correction and grading, but they are essentially a mapping of data between RGB values, with the coordinates still storing the RGB three-channel color values. In terms of data type, it is identical to the image that is fed into the network, the differences is that the image is a two-dimensional plane. Theoretically, just as a three-channel RGB image can be used as the input to a deep learning network, the data in the 3D LUT can also be fed to the network. In short, we believe that drawing the information from the 3D LUTs into the network can help the network to estimate better.

As shown in Fig. 4, before training, we set $N$ 3D LUTs, each of which has a parametric number of $M^3$. We noted that the data in the form of 3D storage cannot be fed directly into the network for learning, but could be converted into a form similar to an image. This is the reason why we change the value of $M$ to 36, under which the data can be directly expanded into a plane, and this step can be expressed by the formula:

$$LUT_{orig}(M^3) \xrightarrow{flatten} LUT_{flat}(M\sqrt{M} \times M\sqrt{M}), \tag{2}$$

where $LUT_{orig}$ is the original 3D LUT storage form, and $LUT_{flat}$ is the flattened storage form. We call it LUT image because it has a similar appearance to the image. Obviously after this step, the idea of feeding the look-up table into the network learning becomes possible.

As for how to handle the 3D LUT after the transformation, we chose an incomplete ResNet [12] as the network model. Since the current module only needs to obtain feature maps for the fusion module afterward, the network model does not need the final fully connected layer for outputting weights. In addition, ResNet [12] is mature enough, which is very friendly used in this task. After the network layer, the LUT feature map is obtained.

Although it is possible to use the LUT feature map for the subsequent feature map fusion module, it will eventually produce poor fusion results because of the large gap between the image and the LUT feature maps. In order to reduce the gap between the feature maps, we fuse the LUT feature maps initially according to the image weights to ensure the effectiveness of the subsequent feature map fusion module. The effects of this section will be compared and analyzed in the subsequent experimental section. (see section 4.3)

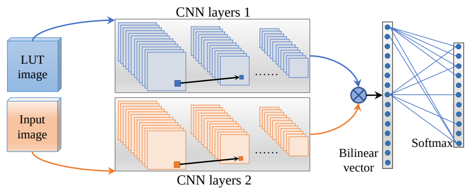

**Fig. 5.** Bilinear pooling is mainly used for feature fusion, for features $x$ and $y$ extracted from the same sample, the two features are fused by bilinear pooling to obtain the vector. The final output of the learned weights through the fully connected layer.

## 3.5 Feature fusion using bilinear pooling

Our network is constructed with two different inputs, the image, and the LUT image. The overall framework is shown in Fig. 2, we keep the feature map before feeding it to the fully connected layer for subsequent operations; the LUTs Feature Map is combined with the weights generated by the image classifier and fused into a LUT fusion map with the same size as the Input Feature Map, which is further mingled with it by averaging pooling and bilinear pooling [25]. For image $\mathcal{I}$ and 3D LUT $\mathcal{L}$ at the position $l$ of two features $f_{img} \in \mathbb{R}^{T\times M}$ and $f_{lut} \in \mathbb{R}^{T\times N}$, the bilinear pooling can be expressed by the following equation:

$$\begin{aligned}
b(l,\mathcal{I},\mathcal{L},f_{img},f_{LUT}) &= f_{img}^{T}(\mathcal{I})f_{LUT}(\mathcal{L}) && \in \mathbb{R}^{M\times N},\\
\xi(\mathcal{I},\mathcal{L}) &= \sum_{l} b(l,\mathcal{I},\mathcal{L},f_{img},f_{LUT}) && \in \mathbb{R}^{M\times N},\\
x &= vec(\xi(\mathcal{I},\mathcal{L})) && \in \mathbb{R}^{MN\times 1},\\
y &= sign(x)\sqrt{|x|} && \in \mathbb{R}^{MN\times 1},\\
z &= y/\|y\|_2 && \in \mathbb{R}^{MN\times 1}.
\end{aligned} \tag{3}$$

Intuitively, bilinear pooling [25] is the bilinear fusion of two features to obtain the matrix $b$. Sum pooling, or max pooling, is performed on $b$ at all positions, after which the matrix $\xi$ is obtained, and then, $\xi$ is expanded into a vector, which is called the bilinear vector $x$. After the moment normalization operation on $x$, we could get smoothed feature $y$, and after further L2 normalization operation, the fused feature $z$ is obtained.

Besides, we have considered other possibilities - whether it is possible to use a simpler approach (*e.g.*matrix stitching or element summation). But experience and experiments led us to conclude that bilinear pooling, which is specifically made for feature fusion, is more effective than the simple approach.

At this point, we can simply use the fully connected layer to generate weights that can be better fused against the original lookup table rather than just through optimization methods. Extensive experiments (see section 4.2) have shown that our proposed LUT-aware module can significantly improve image enhancement, not only for optimal retouching, but also for other image-related tasks, and has shown advantageous results on several datasets.

### 3.6 Loss function

As we adopt the look-up table paradigm to retouch photos, we follow [45], adopt the MSE loss $\mathcal{L}_{mse}$, the smooth regularization loss $\mathcal{R}_s$ and monotonicity regularization loss $\mathcal{R}_m$ as the basic loss terms, which can be calculated as:

$$\mathcal{L}_{LUT} = \mathcal{L}_{mse} + \lambda_s \mathcal{R}_s + \lambda_m \mathcal{R}_m, \tag{4}$$

where $\lambda_s$ and $\lambda_m$ are trade-off coefficients, we follow [45] and set $\lambda_s = 1 \times 10^{-4}$, $\lambda_m = 10$. The MSE Loss $\mathcal{L}_{mse}$ ensures the content consistency between enhancing result and target photo. Smooth regularization $\mathcal{R}_s$ and monotonicity regularization $\mathcal{R}_m$ are used to ensure the retouching process of the LUT are smoothed and monotonic, and the relative brightness and saturation of the input RGB values are maintained, ensuring natural enhancement results.

## 4 Experiments

### 4.1 Experiment setups

**Datasets.** We complete all experiments and compare the data on PPR10K [24], MIT-Adobe FiveK [2], and HDR+ [10] datasets, respectively. PPR10K [24] contains high-quality raw portrait photos and its corresponding retouched result pairs. We use randomly scrambled, a-expert-retouched, low-resolution photos for training according to the authors' recommendation, and divide the dataset into a training set with 8875 photos, and a test set with 2286 photos. Also, to demonstrate that our work can be used for high-resolution photos, the PPR10K-HR dataset are used. MIT-Adobe FiveK [2] is a database often used in many studies on image enhancement and image retouching, and we set 4500 converted photos for training and the remaining 500 for testing. The HDR+ dataset [10] is a continuous shooting dataset collected by Google Camera Group to study high dynamic range (HDR) and low-light imaging of mobile cameras, and we use this dataset to study the generalizability of the model.

**Implementation details.** We perform our experiments on NVIDIA RTX 3090 GPUs with Pytorch framework [31]. The number of parameters in the model is only 13.18M, and it takes only 13 ms to retouch a single image (360P). All of the training photos are LR photos, so as to improve the training speed, and the testing photos include HR photos (4K $\sim$ 8K) and LR photos (360P). We follow the data augment setting in [24]. We use Adam [22] as the network optimizer and the learning rate is fixed at $1 \times 10^{-4}$ following [45].

**Evaluation metrics.** Three metrics are employed to evaluate the performance of different methods quantitatively. In addition to the basic $PSNR$ and $SSIM$ , the color difference between the retouched photo $R$ and the target photo $T$ is defined as the L2-distance in CIELAB color space with

$$\Delta E_{ab} = \| R^{Lab} - T^{Lab} \|_2 \,. \tag{5}$$

**Table 1.** Comparison of photo enhancement results on several different datasets. The $\uparrow$ and $\downarrow$ denote that larger or smaller is better.

| Method | Dataset | $PSNR\uparrow$ | $\Delta E_{ab}\downarrow$ | $SSIM\uparrow$ | Method | Dataset | $PSNR\uparrow$ | $\Delta E_{ab}\downarrow$ | $SSIM\uparrow$ |
|---|---|---|---|---|---|---|---|---|---|
| Dis-Rec [30] | FiveK | 21.98 | 10.42 | 0.856 | Camera Raw | HDR+ | 19.86 | 14.98 | 0.791 |
| HDRNet [7] | FiveK | 24.32 | 8.49 | 0.912 | UPE [4] | HDR+ | 21.21 | 13.05 | 0.816 |
| DeepLPF [28] | FiveK | 24.73 | 7.99 | 0.916 | DPE [36] | HDR+ | 22.56 | 10.45 | 0.872 |
| CSRNet [11] | FiveK | 25.17 | 7.75 | **0.924** | HDRNet [7] | HDR+ | 23.04 | 8.97 | 0.879 |
| 3D LUT [45] | FiveK | 25.21 | 7.61 | 0.922 | 3D LUT [45] | HDR+ | 23.54 | 7.93 | **0.885** |
| Ours | FiveK | **25.42** | **7.29** | 0.917 | Ours | HDR+ | **23.77** | **7.89** | 0.866 |
| Method | Dataset | $PSNR\uparrow$ | $\Delta E_{ab}\downarrow$ | $PSNR^{HC}\uparrow$ | Method | Dataset | $PSNR\uparrow$ | $\Delta E_{ab}\downarrow$ | $PSNR^{HC}\uparrow$ |
| HDRNet [7] | PPR-a | 23.93 | 8.70 | 27.21 | HDRNet [7] | PPR-b | 23.96 | 8.84 | 27.21 |
| CSRNet [11] | PPR-a | 22.72 | 9.75 | 25.90 | CSRNet [11] | PPR-b | 23.76 | 8.77 | 27.01 |
| 3D LUT [45] | PPR-a | 25.64 | 6.97 | 28.89 | 3D LUT [45] | PPR-b | 24.70 | 7.71 | 27.99 |
| HRP [24] | PPR-a | 25.99 | 6.76 | 28.29 | HRP [24] | PPR-b | 25.06 | 7.51 | 28.36 |
| Ours | PPR-a | **26.51** | **6.45** | **29.74** | Ours | PPR-b | **25.40** | **7.24** | **28.66** |
| HDRNet [7] | PPR-c | 24.08 | 8.87 | 27.32 | HDRNet [7] | PPR-HR | 23.06 | 9.13 | 26.58 |
| CSRNet [11] | PPR-c | 23.17 | 9.45 | 26.47 | CSRNet [11] | PPR-HR | 22.01 | 10.20 | 25.19 |
| 3D LUT [45] | PPR-c | 25.18 | 7.58 | 28.49 | 3D LUT [45] | PPR-HR | 25.15 | 7.25 | 28.39 |
| HRP [24] | PPR-c | 25.46 | 7.43 | 28.80 | HRP [24] | PPR-HR | 25.55 | 7.02 | 28.83 |
| Ours | PPR-c | **25.89** | **7.21** | **29.15** | Ours | PPR-HR | **26.21** | **6.62** | **29.44** |

In the training part of the implementation, we use $PSNR$ as the only filtering metric, based on which the model corresponding to the epoch with the highest score is selected for testing, while the metrics $\Delta E_{ab}$ and $SSIM$ are computed only in the testing part. In general, a better $PSNR$ will correspond to a better $\Delta E_{ab}$ and $SSIM$ score, as this represents a better learning outcome of the model and a higher quality of the augmented picture.

## 4.2 Visualization of results.

**Quantitative Comparisons.** Table 1 reports the comparison of our proposed work with state-of-the-art methods. We obtained the results of these methods using the code and default configuration provided by the existing methods, it can be visually seen from the data presented in the table that with the introduction of LUT-aware, the values of the metrics $PSNR$ and $SSIM$ have improved more apparently, while the $\Delta E_{ab}$ metric is well optimized, and these data show the effectiveness of our work.

**Qualitative Comparisons.** The effectiveness of our work can be clearly seen by comparing the photos before and after the optimization of each dataset. Among Fig. 6, our output is the closest to the target photos provided in the dataset, not only in terms of visual effect but also in terms of indicators that demonstrate the validity of our work. For the two most obvious image retouching datasets, PPR10K [24] and MIT-Adobe FiveK [2], our method is more friendly to non-professionals than manually adjusting image parameters. On the HDR+ dataset [10], there are still some gaps compared to the targets provided in the dataset, especially in the case of severe underexposure, and our enhancement results are not natural. This may be due to not taking into account the local contextual information. In future work, we will consider solving such problems.

**Table 2.** (a)(b)The effect of the size and number of each LUT cell of the 3D LUT was verified. (c)A comparison test of three combined methods of feature maps output by two CNN layers. (d)A comparison experiment of two CNN layers in the network architecture. (e)Comparative experiments demonstrate which of the two feature map fusion methods is more effective. All experiments are trained on the PPR10K [24].

| (a) | $M$ | $N$ | $PSNR$ | $\Delta E_{ab}$ | $SSIM$ | (c) | $Method$ | $PSNR$ | $\Delta E_{ab}$ | $SSIM$ |
|---|---|---|---|---|---|---|---|---|---|---|
| | 36 | 3 | 26.09 | 6.72 | 0.915 | | Cat | 25.62 | 7.12 | 0.901 |
| | 36 | 4 | 26.17 | 6.55 | 0.921 | | Add | 25.52 | 7.27 | 0.899 |
| | 36 | **5** | **26.51** | **6.45** | **0.916** | | Bilinear Pooling | **26.51** | **6.45** | **0.916** |
| | 36 | 6 | 26.33 | 6.50 | 0.912 | (d) | $Network$ | $PSNR$ | $\Delta E_{ab}$ | $SSIM$ |
| | 36 | 7 | 26.16 | 6.70 | 0.915 | | 2 ResNet | 25.84 | 7.04 | 0.903 |
| (b) | $M$ | $N$ | $PSNR$ | $\Delta E_{ab}$ | $SSIM$ | | ResNet + EfficientNet | 25.90 | 6.97 | 0.907 |
| | 16 | 5 | 26.03 | 6.72 | 0.911 | | EfficientNet + ResNet | **26.51** | **6.45** | **0.916** |
| | 25 | 5 | 26.23 | 6.62 | 0.915 | | 2 EfficientNet | 25.93 | 6.93 | 0.911 |
| | **36** | 5 | **26.51** | **6.45** | **0.916** | (e) | $Method$ | $PSNR$ | $\Delta E_{ab}$ | $SSIM$ |
| | 49 | 5 | 26.14 | 6.67 | 0.909 | | Direct Fusion | 25.94 | 7.01 | 0.909 |
| | 64 | 5 | 26.23 | 6.64 | 0.914 | | Pre-weighted Fusion | **26.51** | **6.45** | **0.916** |

## 4.3 Ablation study

We conduct extensive ablation experiments on the PPR10K dataset [24] to determine the validity of the components and verify the influence of each parameter, and analyze the effects of each component of our model.

**Efficacy of each component.** As shown in Table 2, we chose to use the model in 3D LUT [45] as the baseline model in our experiments and deploy the LUT-aware model on it, and design the ablation experiments. From the experimental data, we try to identify the most suitable experimental setup on all three different metrics. In addition, we enumerated four different ways of using CNN layers and apply them to both the input image and the input LUT, comparing four sets of experiments. The results demonstrate that the combination of using the EfficientNet [33] layer for image input and ResNet [12] for LUT input is the most reasonable and effective.

In [34], layers were designed for image classification and explores the effects of input resolution, depth, and width of the network at the same time, while the LUT-transformed input does not have a resolution as a feature-level meaning. Therefore, more suitable for using the ResNet [12] layer, which has better generalization. In contrast, if the ResNet [12] layer is used for both CNN layers, although it does reduce the training cost substantially, the training results become unsatisfactory. Although there is an improvement compared to the baseline, this proves that the LUT is valuable as an input, but it is obvious that the feature matrix after EfficientNet [33] layer processing has better training effect.

As future research continues, network architectures more suitable for LUT-aware models than EfficientNet [33] layer and ResNet [12] layer may emerge, but the flexibility of the model makes network replacement feasible and convenient, which makes our approach not limited to the performance of the classifier at the moment, but constantly evolving and improving.

**Feature fusion method.** In the previous section, we illustrate that the LUT-aware model uses two different CNN layers, which means that our model architecture has two intermediate feature matrices corresponding to the two inputs.

After the LUTs image is passed through ResNet [12] Layers, the first intermediate feature (LUT feature map) on the lookup table is generated. According to our initial idea, we can directly fuse the double matrix with the features at this point, however, this does not prove to be reasonable. We first perform a weight-based feature map fusion before the feature map fusion. The results of the comparison experiments are shown in Table 2, and the results after the initial fusion are much better than the direct fusion, which proves that our design reasonably eliminates the difference between the image information and the lookup table information and makes the subsequent fusion more effective.

In order to subsequently generate 3D LUT fusion weights, we need to fuse the two feature matrices and put them into the subsequent fully connected layer to learn the fusion weights. We initially consider simply using splicing (cat) or summing (add) to process the matrix data, but the results were unsatisfactory. Therefore, to solve this problem, bilinear pooling [25] is introduced to better handle the fusion operation of feature matrix. The experimental results in Table 2 show that the combination of bilinear pooling [25] can achieve better optimization results, and the improvement is surprising.

However, with the introduction of bilinear pooling, the fully connected layer where our network finally generates the weights becomes more cumbersome. This is due to the fact that bilinear pooling is different from simple splicing or element summing, which makes the dimensionality of the matrix grow substantially.

As scientific research progresses, many contributions have been made to related research. When introducing bilinear pooling into our design, many variants were considered, but this would make the already cumbersome weight output layer more complex and would have limited improvement in the final result. The most original architecture of bilinear pooling [25] was finally adopted.

**The parameter of 3D LUTs.** In order to ensure that the experiment settings are appropriate, we change the dimension of LUT for different experiments while keeping other conditions constant. Results of the experiments are shown in Table 2. The ablation results indicate that the most suitable number of 3D LUTs is 5, while the suitable dimension is 36.

Theoretically, the number and size of LUT become relatively larger will have a more detailed mapping of pixel values and thus achieve better retouching. However, a more moderate choice would achieve more desirable results. In fact, if the number $N$ is set too much, it will make the network layer generate too much weight on the image, which is close to the weight generated by image classification and will trigger overfitting-like results, affecting the final fusion effect instead. Besides, the LUT size $M$ is due to the refinement of the interpolation algorithm, which means finer inputs will sightly improve the accuracy, and the larger size of LUT, the more difficult for model learning.

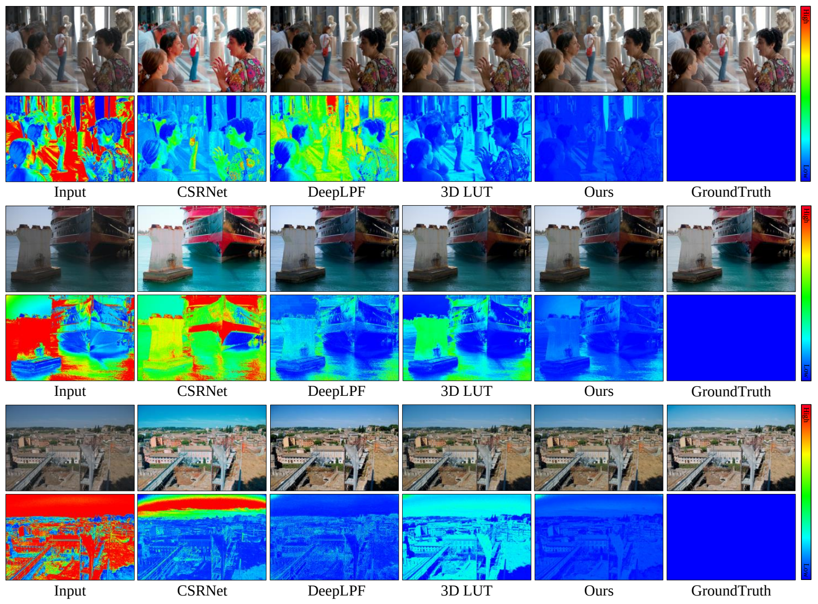

**Fig. 6.** Qualitative comparisons with corresponding error maps on the FiveK dataset [2] for photo retouching. Blue indicates good effect and red indicates large differences. Our model has the best visualization with the output image closest to the target images.

## 5 Conclusion

Since the existing deep learning work on image enhancement based on 3D LUT simply uses the look-up table as a parameter for optimization, ignoring that the look-up table itself is also valuable information that can be sent to the network input port for training, this paper designs a lightweight LUT-aware module, which feeds the 3D LUT into the network module as a parameter after a reasonable deformation. In order to obtain more accurate and reasonable adaptive fusion LUT weights. In addition, we improve the network architecture necessary for training the model, so that the input images and the look-up table information can be effectively combined. Since the final training result contains only one image classifier and some 3D LUTs fused based on the classification results, our approach still maintains the advantage of lightweight look-up table-based photo enhancement. Extensive experiments on several different datasets demonstrate that our approach is effective and can be applied to other image enhancement tasks (*e.g.*low-light enhancement) rather than being limited to image retouching tasks. It is worth mentioning that the unintended using of our method for surveillance may violate personal privacy. In the future, we consider applying this work to other areas as the next research direction, such as image denoising, underwater image enhancement, and super-resolution.

## References

1. Afifi, M., Derpanis, K.G., Ommer, B., Brown, M.S.: Learning multi-scale photo exposure correction. In: CVPR. pp. 9157–9167 (2021)
2. Bychkovsky, V., Paris, S., Chan, E., Durand, F.: Learning photographic global tonal adjustment with a database of input/output image pairs. In: CVPR 2011. pp. 97–104. IEEE (2011)
3. Cai, J., Gu, S., Zhang, L.: Learning a deep single image contrast enhancer from multi-exposure images. IEEE TIP **27**(4), 2049–2062 (2018)
4. Chen, Y.S., Wang, Y.C., Kao, M.H., Chuang, Y.Y.: Deep photo enhancer: Unpaired learning for image enhancement from photographs with gans. In: ICCV. pp. 6306–6314 (2018)
5. Finlayson, G.D., Trezzi, E.: Shades of gray and colour constancy. In: Color and Imaging Conference. vol. 2004, pp. 37–41. Society for Imaging Science and Technology (2004)
6. Fu, L., Zhou, C., Guo, Q., Juefei-Xu, F., Yu, H., Feng, W., Liu, Y., Wang, S.: Auto-exposure fusion for single-image shadow removal. In: CVPR. pp. 10571–10580 (2021)
7. Gharbi, M., Chen, J., Barron, J.T., Hasinoff, S.W., Durand, F.: Deep bilateral learning for real-time image enhancement. ACM TOG **36**(4), 1–12 (2017)
8. Gijsenij, A., Gevers, T., Van De Weijer, J.: Computational color constancy: Survey and experiments. IEEE transactions on image processing **20**(9), 2475–2489 (2011)
9. Guo, C.G., Li, C., Guo, J., Loy, C.C., Hou, J., Kwong, S., Cong, R.: Zero-reference deep curve estimation for low-light image enhancement. In: CVPR. pp. 1780–1789 (June 2020)
10. Hasinoff, S.W., Sharlet, D., Geiss, R., Adams, A., Barron, J.T., Kainz, F., Chen, J., Levoy, M.: Burst photography for high dynamic range and low-light imaging on mobile cameras. ACM TOG **35**(6), 1–12 (2016)
11. He, J., Liu, Y., Qiao, Y., Dong, C.: Conditional sequential modulation for efficient global image retouching. In: ECCV. pp. 679–695. Springer (2020)
12. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
13. Huang, T., Li, S., Jia, X., Lu, H., Liu, J.: Neighbor2neighbor: Self-supervised denoising from single noisy images. In: CVPR. pp. 14781–14790 (June 2021)
14. Ignatov, A., Kobyshev, N., Timofte, R., Vanhoey, K., Van Gool, L.: Dslr-quality photos on mobile devices with deep convolutional networks. In: ICCV. pp. 3277–3285 (2017)
15. Jo, Y., Kim, S.J.: Practical single-image super-resolution using look-up table. In: CVPR. pp. 691–700 (2021)
16. Karaimer, H.C., Brown, M.S.: A software platform for manipulating the camera imaging pipeline. In: European Conference on Computer Vision. pp. 429–444. Springer (2016)
17. Kim, H.U., Koh, Y.J., Kim, C.S.: Global and local enhancement networks for paired and unpaired image enhancement. In: ECCV. pp. 339–354. Springer (2020)
18. Kim, H.U., Koh, Y.J., Kim, C.S.: Pienet: Personalized image enhancement network. In: ECCV. pp. 374–390. Springer (2020)
19. Kim, H., Choi, S.M., Kim, C.S., Koh, Y.J.: Representative color transform for image enhancement. In: ICCV. pp. 4459–4468 (2021)

20. Kim, S.J., Lin, H.T., Lu, Z., Süsstrunk, S., Lin, S., Brown, M.S.: A new in-camera imaging model for color computer vision and its application. IEEE Transactions on Pattern Analysis and Machine Intelligence **34**(12), 2289–2302 (2012)
21. Kim, Y.T.: Contrast enhancement using brightness preserving bi-histogram equalization. IEEE transactions on Consumer Electronics **43**(1), 1–8 (1997)
22. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. ICLR (2015)
23. Li, J., Han, K., Wang, P., Liu, Y., Yuan, X.: Anisotropic convolutional networks for 3d semantic scene completion. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3351–3359 (2020)
24. Liang, J., Zeng, H., Cui, M., Xie, X., Zhang, L.: Ppr10k: A large-scale portrait photo retouching dataset with human-region mask and group-level consistency. In: CVPR. pp. 653–661 (2021)
25. Lin, T.Y., RoyChowdhury, A., Maji, S.: Bilinear cnn models for fine-grained visual recognition. In: Proceedings of the IEEE international conference on computer vision. pp. 1449–1457 (2015)
26. Liu, H., Wan, Z., Huang, W., Song, Y., Han, X., Liao, J.: Pd-gan: Probabilistic diverse gan for image inpainting. In: CVPR. pp. 9371–9381 (2021)
27. Mantiuk, R., Daly, S., Kerofsky, L.: Display adaptive tone mapping. In: ACM SIGGRAPH 2008 papers, pp. 1–10 (2008)
28. Moran, S., Marza, P., McDonagh, S., Parisot, S., Slabaugh, G.: Deeplpf: Deep local parametric filters for image enhancement. In: CVPR. pp. 12826–12835 (2020)
29. Mukherjee, J., Mitra, S.K.: Enhancement of color images by scaling the dct coefficients. IEEE Transactions on Image processing **17**(10), 1783–1794 (2008)
30. Park, J., Lee, J.Y., Yoo, D., Kweon, I.S.: Distort-and-recover: Color enhancement using deep reinforcement learning. In: CVPR. pp. 5928–5936 (2018)
31. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: Pytorch: An imperative style, high-performance deep learning library. NeurIPS **32**, 8026–8037 (2019)
32. Song, Y., Qian, H., Du, X.: Starenhancer: Learning real-time and style-aware image enhancement. In: ICCV. pp. 4126–4135 (2021)
33. Tan, M., Le, Q.: EfficientNet: Rethinking model scaling for convolutional neural networks. In: Chaudhuri, K., Salakhutdinov, R. (eds.) Proceedings of the 36th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 97, pp. 6105–6114. PMLR (09–15 Jun 2019)
34. Tan, M., Le, Q.: Efficientnet: Rethinking model scaling for convolutional neural networks. In: International conference on machine learning. pp. 6105–6114. PMLR (2019)
35. Wang, P., Liu, L., Shen, C., Shen, H.T.: Order-aware convolutional pooling for video based action recognition. Pattern Recognition **91**, 357–365 (2019)
36. Wang, R., Zhang, Q., Fu, C.W., Shen, X., Zheng, W.S., Jia, J.: Underexposed photo enhancement using deep illumination estimation. In: CVPR. pp. 6849–6857 (2019)
37. Wang, T., Li, Y., Peng, J., Ma, Y., Wang, X., Song, F., Yan, Y.: Real-time image enhancer via learnable spatial-aware 3d lookup tables. In: ICCV. pp. 2471–2480 (2021)
38. Yan, Q., Gong, D., Shi, J.Q., van den Hengel, A., Sun, J., Zhu, Y., Zhang, Y.: High dynamic range imaging via gradient-aware context aggregation network. Pattern Recognition **122**, 108342 (2022)
39. Yan, Q., Gong, D., Shi, Q., Hengel, A.v.d., Shen, C., Reid, I., Zhang, Y.: Attention-guided network for ghost-free high dynamic range imaging. In: Proceedings of the

IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 1751–1760 (2019)
40. Yan, Q., Gong, D., Zhang, Y.: Two-stream convolutional networks for blind image quality assessment. IEEE Transactions on Image Processing **28**(5), 2200–2211 (2018)
41. Yan, Q., Zhang, L., Liu, Y., Zhu, Y., Sun, J., Shi, Q., Zhang, Y.: Deep hdr imaging via a non-local network. IEEE Transactions on Image Processing **29**, 4308–4322 (2020)
42. Yu, L., Yang, Y., Huang, Z., Wang, P., Song, J., Shen, H.T.: Web video event recognition by semantic analysis from ubiquitous documents. IEEE Transactions on Image Processing **25**(12), 5689–5701 (2016)
43. Yuan, L., Sun, J.: Automatic exposure correction of consumer photographs. In: European Conference on Computer Vision. pp. 771–785. Springer (2012)
44. Zamir, S.W., Arora, A., Khan, S., Hayat, M., Khan, F.S., Yang, M.H., Shao, L.: Multi-stage progressive image restoration. In: CVPR. pp. 14821–14831 (2021)
45. Zeng, H., Cai, J., Li, L., Cao, Z., Zhang, L.: Learning image-adaptive 3d lookup tables for high performance photo enhancement in real-time. IEEE TPAMI (2020)
46. Zhang, Z., Jiang, Y., Jiang, J., Wang, X., Luo, P., Gu, J.: Star: A structure-aware lightweight transformer for real-time image enhancement. In: ICCV. pp. 4106–4115 (2021)
47. Zheng, Z., Ren, W., Cao, X., Wang, T., Jia, X.: Ultra-high-definition image hdr reconstruction via collaborative bilateral learning. In: ICCV. pp. 4449–4458 (2021)