UniFi: Universal Filter Model for Image Enhancement

A. Samarin[1], A. Nazarenko[1], A. Savelev[1], A. Toropov[1], A. Dzestelova[1], E. Mikhailova[1], A. Motyko[2], and V. Malykh[1]

[1] ITMO University, St. Petersburg, 197101 Russia
[2] St. Petersburg Electrotechnical University "LETI", St. Petersburg, 197022 Russia

Abstract. Image enhancement is becoming more and more popular, especially on mobile devices. Nowadays, it is a common approach to enhance an image using a convolutional neural network (CNN). Such a network should be of significant size, otherwise a possibility for the artifacts to occur is overgrowing. The existing large CNNs are computationally expensive which could be crucial for mobile devices. Another important flaw of such models is they are poorly interpretable. There is another approach to image enhancement, namely usage of predefined filters in combination with prediction of their applicability. We present an approach following this paradigm, which is outperforming both existing CNN-based and filter-based approaches in the image enhancement task. It is easily adoptable for mobile devices, since it has only 47 thousand parameters. It shows the best SSIM 0.919 on RANDOM250 (MIT Adobe FiveK) among small models and is thrice faster than previous models.

Keywords: Universal filter · Image enhancement · Neural networks · Computer vision.

1 Introduction

With digitization of photography the amount of photo images is growing exponentially. People are using digital photo cameras mostly integrated in mobile devices. People use these cameras in their everyday lives, tourism, and many other activities. This leads to a wide variety of shooting conditions, including light, angles, exposure, etc. Only a small percentage of people using digital cameras aware of importance of shooting conditions, which results in low quality photo images. To face this challenge there were presented a lot of photo retouching tools. Although these tools often require a skill to apply them. In the recent years there have been proposed several models to automate and ease retouching process, e.g. [16].

Nevertheless, the existing models do not solve the retouching problem in general. They address some specific features, including brightness, contrast, local artifacts, blur, and other aspects that affect human image perception. Below we explore existing approaches regarding to the features they address.

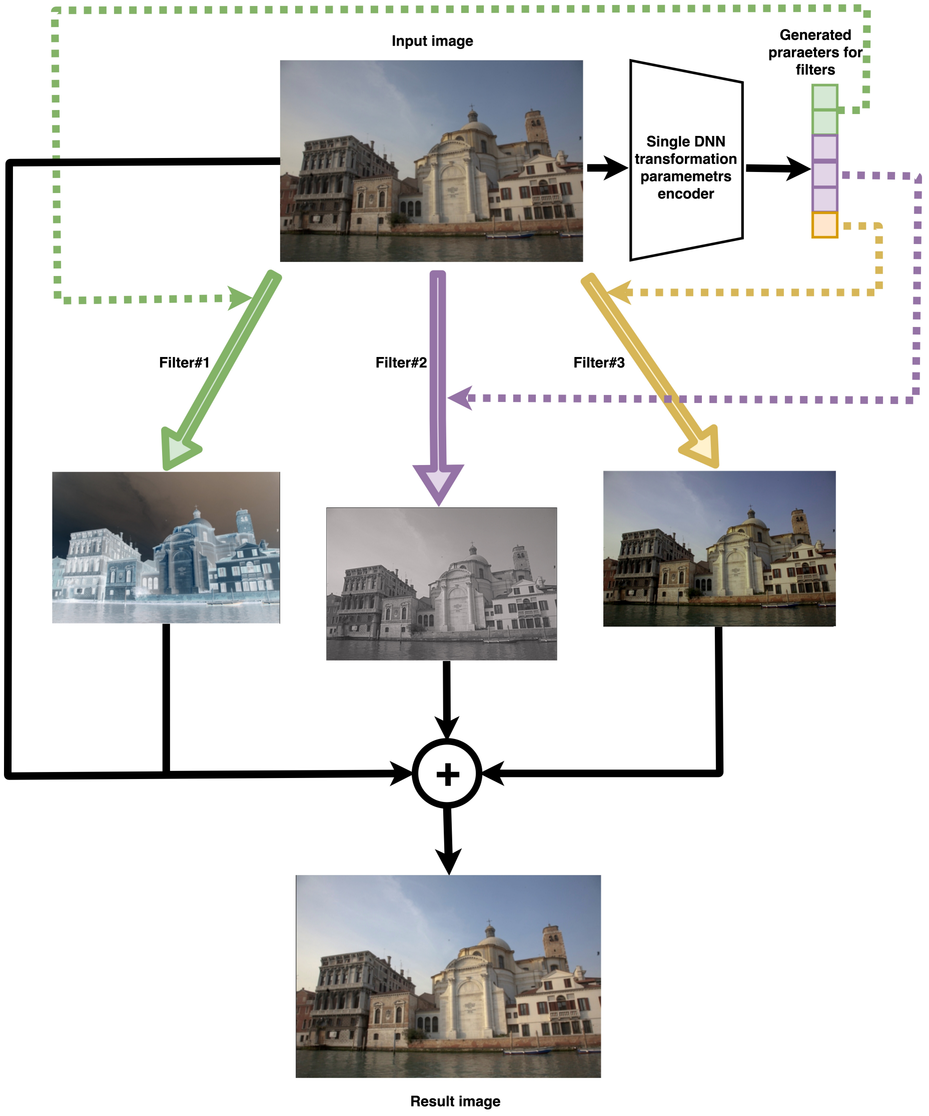

Input image
Single DNN transformation paramemetrs encoder
Generated praraeters for filters
Filter#1
Filter#2
Filter#3
+
Result image

Fig. 1. Sample image enhancement for a photo from MIT Adobe FiveK dataset. Our model does not produce artifacts and shows good results even on a detailed image.

There exist several approaches which work in end2end manner [7, 16, 21], i.e. an image is used as an input and an image is expected as an output. The advantage of these approaches is their conceptual simplicity and nonexistent requirement for the markup, only the image pairs are enough to train such a model. But these approaches has an innate disadvantage, as they could produce an unnatural distortion of the resulting image. The one is presumably caused by the difference in resolution between the image being processed and the training set samples. Another important flaw of these approaches is possible colour artifacts, both global and local. These artifacts could be caused by the fact of usage of unrestricted transformation space. More on that the aforementioned approaches are commonly heavyweight and thus cannot be used on mobile devices due to memory and power consumption limitations. There also exists a category of methods that are based on pre-defined filters. The models in these approaches are predict filter parameters for subsequent image processing [14,19]. The sample of such a method could be seen at Fig. 1, as our proposed method is also from this category.

These methods have the following flaw of lacking an analysis of available filters and their combinations. The lack of comprehensive comparative analysis of image transformations leaves open the question of the most optimal stack of filters for image enhancement. For the mobile devices this question is raised higher, since different filters have different resource consumption, and for the mobile device the whole task is to find not only the best possible combination of filters, but also a combination which is lightweight enough for a mobile device to process it. To address the aforementioned issues, we propose an automatic image enhancement model which predicts filter parameters using a single generator, lightweight, and faster than the previously presented approaches. Additionally, we offer an analysis of filters to construct the most optimal transformation stack and perform a comparative evaluation for our solution and other models.

Contributions. Overall our paper has two contributions:

1. We propose a new lightweight model which can be used on mobile devices. Our model shows best performance on MIT Adobe FiveK dataset [3] by SSIM among lightweight models and is twice lighter and thrice faster as closest lightweight counterpart.
2. We propose several new trainable image transformations and did an extensive study of them in application to image enhancement task.

2 Related Work

Currently, the field of automatic image enhancement features many works with different approaches to image modification for improving visual perception. Methods based on deep learning are considered the most successful in image enhancement.

The existing approaches could be divided into two main groups: the end-to-end approaches, where the output image is generated by a neural network, and compositional approaches, where the output image is generated by application of several pre-defined transformations, while neural network is predicting the parameters of these transformations. Let us first describe the methods from the second group.

Hu *et al.* [14] have introduced a similar filter-based approach that uses interpretable image transformations and provided their complete review. In its proposed implementation, their model is heavyweight, prone to unnatural color distortion and provides low inference speed without additional optimization.

Du *et al.* [9] have also presented an image enhancement framework based on specially designed self-interpretable image filters. The proposed model effectively solves the problem of the lack of paired samples and is resistant to distortions arising from the subjectivity of experts. Among the disadvantages of this approach, the most significant are the low numerical characteristics in such priority component as Luminance correction. For example, Pix2Pix [16] demonstrated better results. However, our model significantly outperforms Pix2Pix.

Another approach relies on a combined architecture that separates the enhancement process into channel-wise and pixel-wise refinement [21]. This solution uses the residual network backbone model [12] as a feature extractor, non-local attention block [27] for subsequent feature analysis and global linear mapping to improve the visual perception characteristics. Chai *et al.* [4] proposed a similar approach based on parameterized color transformation usage.

Such ideas have proven effectiveness in solving image enhancement problems so our combined model uses resembling techniques and contains at least 10 times less parameters. The one of the most advanced works of such kind is Tatanov et al. [23]. In this work authors use several well known filters and achieve improvement using the consistency regularization. The authors use separate generators for each filters which requires additional weights in the model.

One image enhancement approach [19] involves applying various retouching methods to the input image. It includes using a method based on deep reinforcement learning to find the optimal global enhancement sequence of image transformations. However, this solution is computationally intensive and very heavyweight, as it uses a VGG-16 model [22] and histogram estimation for evaluating the parameters for building the optimal image improvement strategy.

Particularly noteworthy are methods based on models with a small number of parameters. Moran *et al.* [18] proposed a lightweight filter-based solution that takes into account local features of images. This model uses non-universal parameter generator for the filter, which leads to suboptimal model selection due to oversized possible variants. Our model contains 8 times less parameters and due to usage of universal generator is able to show best SSIM results. Wang *et al.* [26] presented another lightweight engine that is based on global and local image feature estimation. In this work the authors are using separate generators for each filter which increases the model size and computation time.

As for the first group, the most widely known are the ones based on Generative Adversarial Networks (GAN) [11], a few to name [7,16]. A downside of these methods is that they are prone to artifacts. Generator with global features and Wasserstein GAN improvement with an adaptive weighting scheme have been used in a two-way GAN-based method for the task of image enhancement [7]. However, this method does not demonstrate competitive results on the MIT Adobe FiveK dataset.

EnhanceGAN [8] can be distinguished among other GAN-based methods for its competitive quantitative results on aesthetic-based color enhancement. Unfortunately, the encoder proposed by the authors is rather heavy, that significantly complicates the use of this architecture on mobile devices.The models of this group are called heavyweight since their size is one magnitude bigger than the models' from the second group one.

Above, we considered several modern methods for automated image enhancement, each with its own concept. the vast majority of considered models are unsuitable for deployment on mobile platforms due to a variety of reasons, such as being heavyweight [7, 8, 14, 16, 19, 21], computational complexity [14, 19], and low inference speed in practice because of lack of additional optimization [14]. It should also be noted that our solution outperforms most of the known methods for photos enhancement [5, 10, 15, 18, 20, 26] on MIT Adobe FiveK benchmark.

In this paper, we examine the image transformations used in various image enhancement methods and present the results of a comparative study that we conducted. Using a combination of researched techniques, we propose our own neural network model that demonstrates competitive results and is optimized for use on mobile devices.

3 Proposed Method

Our model is following the general design of LFIEM model proposed in [23]. Although, it has several important discrepancies. The model conceptually consists of a *single* parameter generator, several filters, and a summator. Each filter is used with the parameters generated by the generator. More formally, an original image I_o is resized to a smaller version of itself I_{so}. This image I_{so} is used as follows: it is fed to parameter generator h that produces the parameters p_i for the corresponding filter f_i which is applied to the original image I_o. Filter f_i produces an output image. Such output images from all the filters are summed with the original image I_o to produce the final enhanced image I_e. Overall, our model can be written as

$$\begin{aligned} p_{1..n} &= h(I_{so}) \\ I_e &= I_o + \sum_{i=1}^{n} f_i(I_o, p_i), \end{aligned}$$

where n is number of used filters. Thus, the generator is called once and produce the parameters for all the filters used. It is also important to mention that value in a pixel is clipped by maximum value of 1 after summation. We clip overflow in-pixel values due to we are working in RGB space.

In our setup we use *novel* universal filters, which are different from the previously presented and more importantly they allows our model to outperform LFIEM (and all the previous lightweight models). Another significant difference is that we *do not* use consistency regularization, which allows our model to run faster. We present a detailed overview of our model work at Fig. 2.

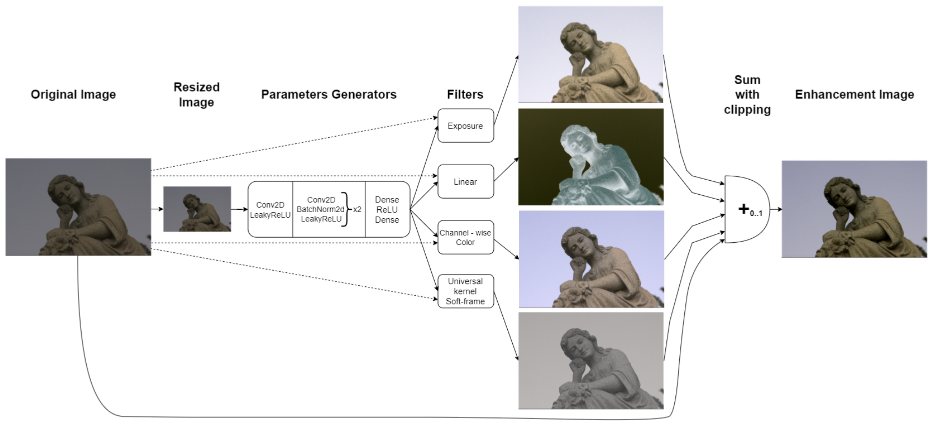


Fig. 2. Detailed representation of UniFi pipeline work. *The figure shows a visualization of an arbitrary architecture configuration.*

3.1 Parameter Generator

Following [23] we designed our parameter generator with two stages. The first stage includes three convolutional layers with strides equal to 2. Each convolutional layer, except for the first one, is followed by an batch normalization layer [25] and a LeakyReLU activation function [17]. We set the number of feature maps equal to 16, 32, and 128 in the first, second, and third convolutional layer respectively. The second stage contains two fully connected layers with ReLU activation function between them. There could be also a normalizing function, its type is dependent on input range of the filter the generator is used for. We used sigmoid function for $[0, 1]$ range, hyperbolic tangent function for $[-1, 1]$ range, and used no normalizing function if range is not limited for a filter. We present a detailed parameter generator architecture at Fig. 3.

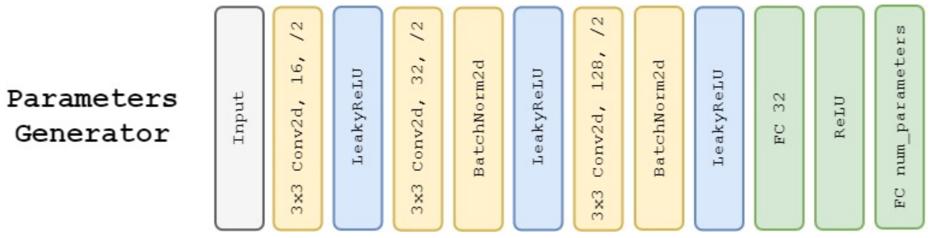


Fig. 3. Parameter generator architecture.

3.2 Filters

In work [23] there were proposed two types of the filters, in this work we add several new ones, allowing our model to outperform the previous approaches.

Throughout this section, we use a unified notation for the filters: I_{in} is the input image, I_{out} is the output image, (x, y) stands for image pixel coordinates, c stands for a color channel (red, green, or blue) and p, q, r, s, t, u are trainable parameters unless otherwise is specified. To achieve good time performance, we used only the RGB color space for image representation. For uniform description, we used channel values normalized to $[0, 1]$ range.

We use an automatic ***saturation*** correction applying $p \in [-1, 1]$ that defines the transformation strength to each pixel of the input image. The saturation filter is formulated as follows:

$$\Delta[x,y] = \begin{cases} (m - I_{in}[x,y]) \cdot (1 - \frac{1}{1-p}), & \text{if } p > 0 \\ -(m - I_{in}[x,y]) \cdot p, & \text{otherwise;} \end{cases}$$
$$I_{out}[x,y] = I_{in}[x,y] + \Delta[x,y],$$

where m stands for the mean by channel-wise of the input image.

We used automatic ***contrast*** correction (analogously, with $p \in [-1, 1]$). Its formula is provided below:

$$I_{out}[x,y] = \begin{cases} (I_{in}[x,y] - 0.5) \cdot \frac{1}{1-r}, & \text{if } r > 0 \\ (I_{in}[x,y] - 0.5) \cdot (1 - r), & \text{otherwise;} \end{cases}$$

White balance transformation is using a trainable parameter s_c for each color channel of a pixel in the input image. These trainable parameters are in $[0, 1]$ range:

$$I^c_{out}[x,y] = I_{in}[x,y] \cdot s_c.$$

We use three trainable parameters for red, green, and blue channels respectively.

The image transformation performing automatic ***exposure*** correction is written as follows:

$$I_{out}[x,y] = I_{in}[x,y] \cdot 2^t,$$

where t is trainable parameter with real value.

The trainable ***linear*** image transformation, described in [21], is an additional important mapping. It can be described with the expression:

$$I_{out}[x,y] = P \cdot I_{in}[x,y] + b,$$

where $P \in \mathbb{R}^{3\times3}$ stands for the trainable affine mapping matrix, $b \in \mathbb{R}^3$ – trainable vector in RGB color space.

The ***channel-wise*** image ***color*** transformation that was described in [2] was also used. The transformation is composed of a triplet of functions that are applied to the red, green, and blue color channels respectively. Each function is a linear combination of the elements $f_1, f_2, ...f_n$ of a n-dimensional basis, the coefficients for which are calculated from the output of the neural network. Therefore, a channel value for each pixel of the input image is evaluated by the formula:

$$I^c_{out}[x,y] = I^c_{in}[x,y] + \sum_{i=1}^{n} u_{ic} \cdot f_i(I^c_{in}[x,y]),$$

where $f_1, f_2, ...f_n$ – functional basis mentioned above, and u_c – trainable parameters (one parameter for each channel).

Because of its proven effectiveness [2], we considered only the piece-wise basis and used the set of functions:

$$f_i(x) = \max(0, 1 - |(n-1) \cdot x - i + 1|), i \in \{1, 2, ...n\},$$

where x is a value of the current pixel of the input image.

We use kernel-based filters. The most significant kernel-based filtering results were obtained using a fully trainable filter operation that we called the universal kernel, and sharp filter modification built on kernel-based mapping.

We used the classic definition for the ***universal kernel*** filter:

$$I_{out} = I_{in} \circledast \frac{1}{\nu} P,$$

where $P \in \mathbb{R}^{5\times5}$ is fully trainable filter kernel matrix and ν is sum of elements of P for matrix normalization.

The ***sharp*** filter is defined using auxiliary formula:

$$I_{out} = I_{in} \circledast \frac{1}{\nu}(K + M \cdot q),$$

where K – filter kernel matrix, M – map matrix with the same shape as K and ν is sum of elements of $(K + M \cdot q)$ for kernel matrix normalization. The above formula is applied to red, green, and blue channels independently with its own trainable parameter.

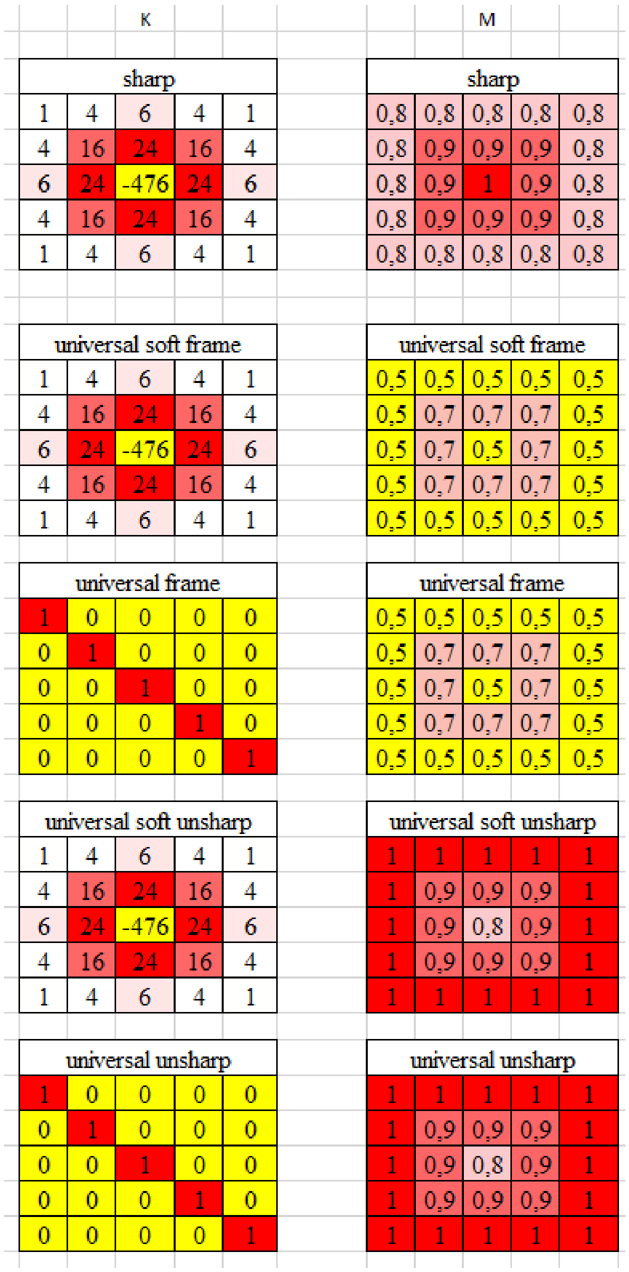
In addition to previously presented *sharp* filter, we introduce several new universal kernel variants, namely *universal soft frame*, *universal frame*, *universal soft unsharp*, and *universal unsharp*. We present the parameters for all the mentioned filters at Fig. 4.

3.3 Loss Function

The parameter generators in our model are meant to give the same results for similar images, hence the outputs of the generator have to be invariant with respect to any weak image augmentations.

The loss functions we use are L_1 and L_{SSIM}, which are defined as follows:

$$L_1 = \|I_e - I_{gt}\|_1,$$
$$L_{SSIM} = 1 - SSIM(I_e, I_{gt}),$$



K

sharp				
1	4	6	4	1
4	16	24	16	4
6	24	-476	24	6
4	16	24	16	4
1	4	6	4	1

universal soft frame				
1	4	6	4	1
4	16	24	16	4
6	24	-476	24	6
4	16	24	16	4
1	4	6	4	1

universal frame				
1	0	0	0	0
0	1	0	0	0
0	0	1	0	0
0	0	0	1	0
0	0	0	0	1

universal soft unsharp				
1	4	6	4	1
4	16	24	16	4
6	24	-476	24	6
4	16	24	16	4
1	4	6	4	1

universal unsharp				
1	0	0	0	0
0	1	0	0	0
0	0	1	0	0
0	0	0	1	0
0	0	0	0	1

M

sharp				
0,8	0,8	0,8	0,8	0,8
0,8	0,9	0,9	0,9	0,8
0,8	0,9	1	0,9	0,8
0,8	0,9	0,9	0,9	0,8
0,8	0,8	0,8	0,8	0,8

universal soft frame				
0,5	0,5	0,5	0,5	0,5
0,5	0,7	0,7	0,7	0,5
0,5	0,7	0,5	0,7	0,5
0,5	0,7	0,7	0,7	0,5
0,5	0,5	0,5	0,5	0,5

universal frame				
0,5	0,5	0,5	0,5	0,5
0,5	0,7	0,7	0,7	0,5
0,5	0,7	0,5	0,7	0,5
0,5	0,7	0,7	0,7	0,5
0,5	0,5	0,5	0,5	0,5

universal soft unsharp				
1	1	1	1	1
1	0,9	0,9	0,9	1
1	0,9	0,8	0,9	1
1	0,9	0,9	0,9	1
1	1	1	1	1

universal unsharp				
1	1	1	1	1
1	0,9	0,9	0,9	1
1	0,9	0,8	0,9	1
1	0,9	0,9	0,9	1
1	1	1	1	1

Fig. 4. Matrices K and M for all presented filters.

where $\|\cdot\|_1$ is L^1 the norm of a given vector, SSIM is the structural similarity index measure [28], I_e and I_{gt} stand for the enhanced image and the target image, respectively.

4 Experimental Setup

4.1 Dataset

We used the MIT Adobe FiveK dataset [3] for our model evaluation. This dataset contains 5,000 original images and 5 enhanced versions per each original image, provided by the image retouching experts. Each expert has her/his own code name, namely (A, B, C, D, E). The C expert's variant has been used in the vast majority of works on automatic image enhancement as a target, we follow this common approach. We used RANDOM250 [19, 29] subset of the MIT Adobe FiveK dataset for model validation, while the rest of 4,750 image pairs were used as a training set. We used only RGB color space for models' training, validation, and comparison. In the data pre-processing we follow [21]: the images were padded to 500×500 and after that normalized to $[0, 1]$.

4.2 Evaluation Metrics

For calculating the distance between the enhancement result and the corresponding target image provided by expert C from the MIT Adobe FiveK dataset, we used two common image difference metrics: the peak-signal-to-noise ratio (PSNR) and the structural similarity index measure (SSIM) [28]. SSIM aims to compare structural information from inter-dependencies of pixel values that allows to take into account structural features of scene and presented objects. While SSIM is more suitable for local artifacts comparison [13], PSNR is standard mean squared error for per pixel comparison.

4.3 Implementation Details

Our model was developed using TensorFlow 2 library [1]. For time measurement the model was converted using Torch package. We trained the models using the Adam optimizer with the following parameters: $\beta_1 = 0$, $\beta_2 = 0.9$ and a batch size of 40. The learning rate was initialized to $1e^{-3}$ with decay factor of 0.95 for every 1,200 steps. For all experiments, we stopped training after 10k steps. All our experiments have been run on 1 x NVIDIA GeForce RTX 3060 and Intel Core i5-10400 CPU with 2.90GHz.

5 Results

In Tab. 1 we present the results for comparison of our best model with existing state of the art models. These models were shortly described in Section 2. Although our approach is targeted for use on mobile devices, we compare our model to full-size models alongside with lightweight ones. All the numbers in this table are adopted from the respective papers.

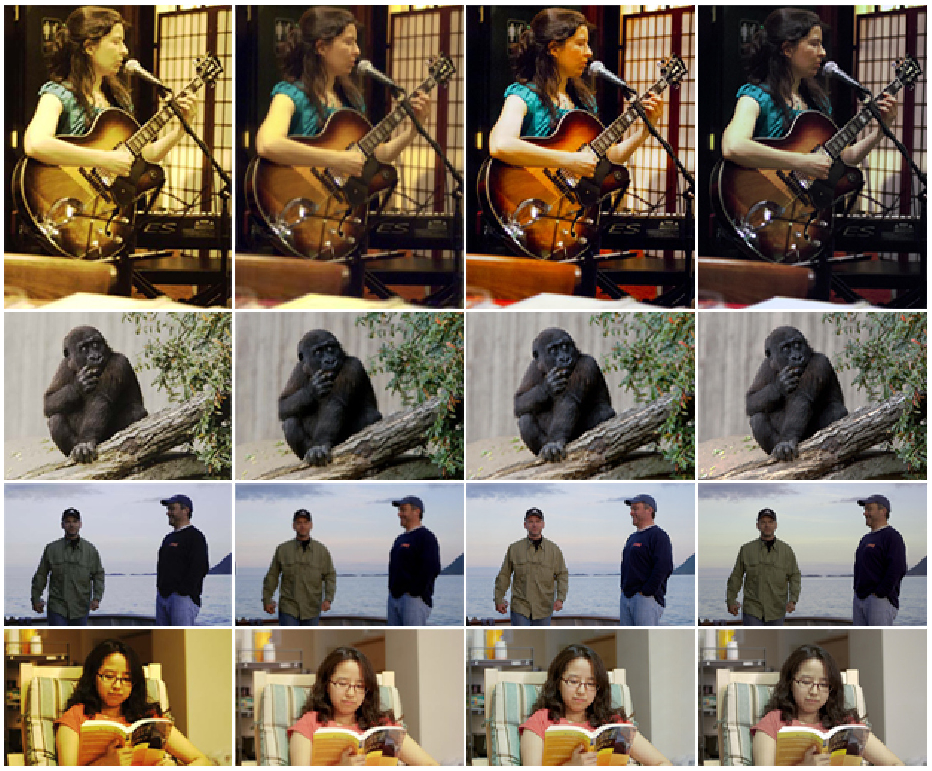


Fig. 5. Visual comparison of output for Exposure filter, LFIEM [23], MAXIM [24], and UniFi (our model) by columns.

As one can see, our model is comparable to the best lightweight models, namely LFIEM [23] and SULPCE [4], being close both by PSNR and SSIM. Our model could be directly compared only to LFIEM, since this model uses the same setup, while SULPCE model is using non-standard setup and thus is incomparable with all the previous art. But our model is more than twice smaller then LFIEM model and more than 20 times smaller than SUPLCE, which makes it state of the art in the lightweight group. We also present a visual comparison of the models' output at Fig. 5.

Table 1. Comparisons of different methods with our best model on MIT Adobe FiveK dataset (RGB color space).

Method	# params	PSNR	SSIM	Train-test split
Heavyweight				
CE+PRNL [21]	>30M	24.19	0.915	4750-250
Pix2Pix [16]	54M	-	0.857	4750-250
Distort-and-Recover [19]	153M	-	0.905	4750-250
DPED [15]	-	21.76	0.871	2250-500
8RESBLK [5]	-	23.42	0.875	2250-500
CRN [6]	-	22.38	0.877	2250-500
HDRNet [10]	-	21.96	0.866	4500-500
MAXIM [24]	14.1M	**26.15**	**0.945**	4500-500
Lightweight				
U-Net [20]	1.3M	22.24	0.850	4500-500
DeepUPE [26]	1.0M	23.04	0.893	4500-500
DeepLPF [18]	800K	24.48	0.887	4500-500
DPE [7]	2.2M	23.89	0.906	4750-250
SULPCE [4]	>1M	23.93	0.920	4000-1000
LFIEM [23]	101K	24.77	0.911	4750-250
UniFi *(ours)*	**47k**	24.18	*0.919*	4750-250

Table 2. Comparison of GPU running time.

Method	GPU Time, ms	δ time, ms
MAXIM [24]	8948.97	204.85
LFIEM [23]	2.08	0.57
UniFi *(ours)*	0.66	0.18

5.1 Ablation Study

Since our model is compositional and dependent on the filters used, we were experimented with significant number of filter setups. We have combined filters described in Section 3.2 in combinatorial way, including combinations of one to four filters and evaluated each configuration using SSIM and PSNR metrics.

Given the results of this experiment series, we can conclude that channel-wise color and linear filters have the most impact on image enhancement quality. In contrast to [23] finding, the universal kernels has a crucial importance for our architecture. It is interesting to mention that despite the fact that PSNR and SSIM are roughly correlated the best PSNR among the compared setups is achieved by the second-best by SSIM one.

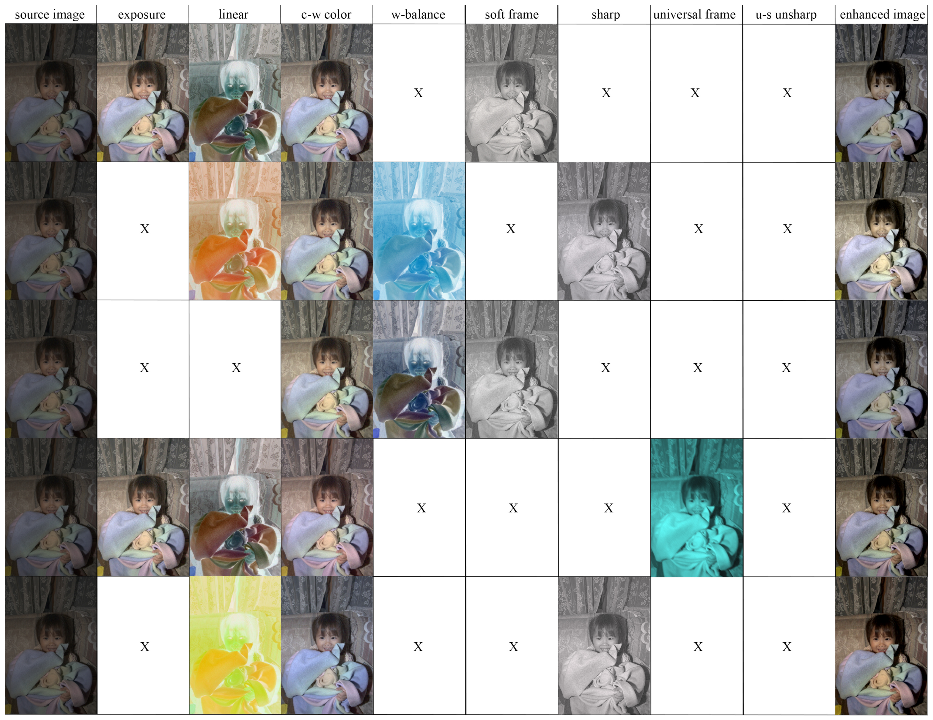


Fig. 6. Comparison of different filter combinations applied to a single image.

We selected only 15 top-performing by SSIM setups to show in Table 3. It is remarkable that the vast majority of top performing model configurations include universal filters. For brevity we use several shorthand names for the filters listed in this section: *c-w color* = channel-wise color, *w-balance* = white balance, *soft frame* = universal kernel variant (U) "soft frame", *sharp* = U "sharp", *universal frame* = U "universal frame", *u-s unsharp* = U "universal soft unsharp". We present visual comparison for several configurations at Fig. 6.

Table 3. Filter configuration comparative study on RANDOM250. Top-15 only.

Used Filters	SSIM	PSNR
w-balance, c-w color, sharp	0.9103	24.12
universal frame, linear, c-w color	0.9104	23.87
linear, c-w color, w-balance, universal frame	0.9114	23.99
w-balance, c-w color, universal frame	0.9119	24.10
u-s unsharp, linear, c-w color	0.9122	23.94
saturation, c-w color, w-balance, u-s unsharp	0.9125	24.34
linear, c-w color, soft frame	0.9125	23.67
saturation, c-w color, w-balance, soft frame	0.9127	24.15
linear, c-w color, w-balance, u-s unsharp	0.9129	24.31
exposure, linear, c-w color, u-s unsharp	0.9131	24.03
sharp, linear, c-w color	0.9146	23.92
exposure, linear, c-w color, universal frame	0.9156	24.24
w-balance, c-w color, soft frame	0.9168	24.11
linear, c-w color, w-balance, sharp	0.9178	**24.47**
exposure, linear, c-w color, soft frame	**0.9188**	24.18

6 Conclusion

In this paper, we have proposed a novel lightweight image enhancement model. Our model is outperforming all the lightweight models, being comparable by PSNR and best by SSIM. Although, it contains only 47 thousand of trainable parameters, being the smallest model in lightweight group (*twice* as less parameters comparing to the previous state of the art). Due to the simple architecture of our model it is able to outperform the previous lightweight state of the art model by the running time, being *three times* faster, which could be crucial for mobile devices. In comparison to state of the art heavyweight model MAXIM, our model is 3 *hundred* times lighter and 13 *thousand* times faster.

As a part of our model we presented several variations of universal kernel filters and made a thorough analysis of their combination performance. We hope that our work will foster further research on universal kernels in application to image enhancement.

References

1. Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al.: Tensorflow: A system for large-scale machine learning. In: OSDI. pp. 265–283 (2016)
2. Bianco, S., Cusano, C., Piccoli, F., Schettini, R.: Learning parametric functions for color image enhancement. In: International Workshop on Computational Color Imaging. pp. 209–220. Springer (2019)
3. 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)
4. Chai, Y., Giryes, R., Wolf, L.: Supervised and unsupervised learning of parameterized color enhancement. pp. 981–989 (03 2020). https://doi.org/10.1109/WACV45572.2020.9093321
5. Chen, J., Adams, A., Wadhwa, N., Hasinoff, S.: Bilateral guided upsampling. ACM Transactions on Graphics **35**, 1–8 (11 2016). https://doi.org/10.1145/2980179.2982423
6. Chen, Q., Koltun, V.: Photographic image synthesis with cascaded refinement networks. pp. 1520–1529 (10 2017). https://doi.org/10.1109/ICCV.2017.168
7. 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: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 6306–6314 (2018)
8. Deng, Y., Loy, C.C., Tang, X.: Aesthetic-driven image enhancement by adversarial learning. pp. 870–878 (10 2018). https://doi.org/10.1145/3240508.3240531
9. Du, X., Yang, X., Qin, Z., Tang, J.: Progressive image enhancement under aesthetic guidance. pp. 349–353 (06 2019). https://doi.org/10.1145/3323873.3325055
10. Gharbi, M., Chen, J., Barron, J., Hasinoff, S., Durand, F.: Deep bilateral learning for real-time image enhancement. ACM Transactions on Graphics **36** (07 2017). https://doi.org/10.1145/3072959.3073592
11. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: Advances in neural information processing systems. pp. 2672–2680 (2014)
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. Hore, A., Ziou, D.: Image quality metrics: Psnr vs. ssim. In: 2010 20th International Conference on Pattern Recognition. pp. 2366–2369. IEEE (2010)
14. Hu, Y., He, H., Xu, C., Wang, B., Lin, S.: Exposure: A white-box photo post-processing framework. ACM Transactions on Graphics (TOG) **37**(2), 26 (2018)
15. Ignatov, A., Kobyshev, N., Timofte, R., Vanhoey, K., Van Gool, L.: Dslr-quality photos on mobile devices with deep convolutional networks. pp. 3297–3305 (10 2017). https://doi.org/10.1109/ICCV.2017.355
16. Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A.: Image-to-image translation with conditional adversarial networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1125–1134 (2017)
17. Maas, A.L., Hannun, A.Y., Ng, A.Y.: Rectifier nonlinearities improve neural network acoustic models. In: ICML (2013)
18. Moran, S., Marza, P., McDonagh, S., Parisot, S., Slabaugh, G.: Deeplpf: Deep local parametric filters for image enhancement (03 2020)

19. Park, J., Lee, J.Y., Yoo, D., So Kweon, I.: Distort-and-recover: Color enhancement using deep reinforcement learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5928–5936 (2018)
20. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation (05 2015)
21. Shan, C., Zhang, Z., Chen, Z.: A coarse-to-fine framework for learned color enhancement with non-local attention. In: 2019 IEEE International Conference on Image Processing (ICIP). pp. 949–953 (Sep 2019). https://doi.org/10.1109/ICIP.2019.8803052
22. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. CoRR **abs/1409.1556** (2014)
23. Tatanov, O., Samarin, A.: Lfiem: Lightweight filter-based image enhancement model. 2020 25th International Conference on Pattern Recognition (ICPR) pp. 873–878 (2021)
24. Tu, Z., Talebi, H., Zhang, H., Yang, F., Milanfar, P., Bovik, A., Li, Y.: Maxim: Multi-axis mlp for image processing. arXiv preprint arXiv:2201.02973 (2022)
25. Ulyanov, D., Vedaldi, A., Lempitsky, V.: Instance normalization: The missing ingredient for fast stylization. arXiv preprint arXiv:1607.08022 (2016)
26. Wang, R., Zhang, Q., Fu, C.W., Shen, X., Zheng, W.S., Jia, J.: Underexposed photo enhancement using deep illumination estimation. pp. 6842–6850 (06 2019). https://doi.org/10.1109/CVPR.2019.00701
27. Wang, X., Girshick, R., Gupta, A., He, K.: Non-local neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 7794–7803 (2018)
28. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P., et al.: Image quality assessment: from error visibility to structural similarity. IEEE transactions on image processing **13**(4), 600–612 (2004)
29. Yan, Z., Zhang, H., Wang, B., Paris, S., Yu, Y.: Automatic photo adjustment using deep neural networks. ACM Transactions on Graphics (TOG) **35**(2), 11 (2016)

Response to review

1. "Page 3 in phrase "The existing approaches could be into two main groups" word devides is omitted" — **Fixed**.
2. "Page 4 in phrase "In this word the authors are using .." it should be work instead of word" — **Fixed**.