

# DeepFuse: A Deep Unsupervised Approach for Exposure Fusion with Extreme Exposure Image Pairs

K. Ram Prabhakar, V Sai Srikar, and R. Venkatesh Babu  
Video Analytics Lab, Department of Computational and Data Sciences,  
Indian Institute of Science, Bangalore, India

## Abstract

We present a novel deep learning architecture for fusing static multi-exposure images. Current multi-exposure fusion (MEF) approaches use hand-crafted features to fuse input sequence. However, the weak hand-crafted representations are not robust to varying input conditions. Moreover, they perform poorly for extreme exposure image pairs. Thus, it is highly desirable to have a method that is robust to varying input conditions and capable of handling extreme exposure without artifacts. Deep representations have known to be robust to input conditions and have shown phenomenal performance in a supervised setting. However, the stumbling block in using deep learning for MEF was the lack of sufficient training data and an oracle to provide the ground-truth for supervision. To address the above issues, we have gathered a large dataset of multi-exposure image stacks for training and to circumvent the need for ground truth images, we propose an unsupervised deep learning framework for MEF utilizing a no-reference quality metric as loss function. The proposed approach uses a novel CNN architecture trained to learn the fusion operation without reference ground truth image. The model fuses a set of common low level features extracted from each image to generate artifact-free perceptually pleasing results. We perform extensive quantitative and qualitative evaluation and show that the proposed technique outperforms existing state-of-the-art approaches for a variety of natural images.

## 1. Introduction

High Dynamic Range Imaging (HDRI) is a photography technique that helps to capture better-looking photos in difficult lighting conditions. It helps to store all range of light (or brightness) that is perceivable by human eyes, instead of using limited range achieved by cameras. Due to this property, all objects in the scene look better and clear in HDRI, without being saturated (too dark or too bright) otherwise.

The popular approach for HDR image generation is called as Multiple Exposure Fusion (MEF), in which, a set

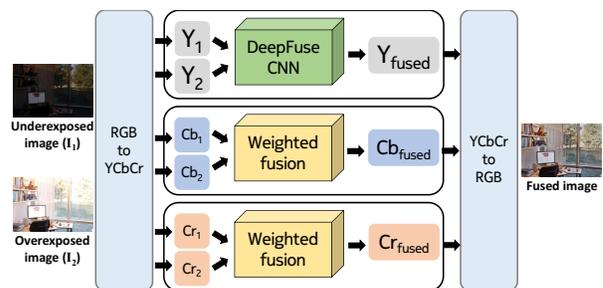


Figure 1. Schematic diagram of the proposed method.

of static LDR images (further referred as exposure stack) with varying exposure is fused into a single HDR image. The proposed method falls under this category. Most of MEF algorithms work better when the exposure bias difference between each LDR images in exposure stack is minimum<sup>1</sup>. Thus they require more LDR images (typically more than 2 images) in the exposure stack to capture whole dynamic range of the scene. It leads to more storage requirement, processing time and power. In principle, the long exposure image (image captured with high exposure time) has better colour and structure information in dark regions and short exposure image (image captured with less exposure time) has better colour and structure information in bright regions. Though fusing extreme exposure images is practically more appealing, it is quite challenging (existing approaches fail to maintain uniform luminance across image). Additionally, it should be noted that taking more pictures increases power, capture time and computational time requirements. Thus, we propose to work with exposure bracketed image pairs as input to our algorithm.

In this work, we present a data-driven learning method for fusing exposure bracketed static image pairs. To our knowledge this is the first work that uses deep CNN architecture for exposure fusion. The initial layers consists of a set of filters to extract common low-level features from each

<sup>1</sup>Exposure bias value indicates the amount of exposure offset from the auto exposure setting of an camera. For example, EV 1 is equal to doubling auto exposure time (EV 0).

input image pair. These low-level features of input image pairs are fused for reconstructing the final result. The entire network is trained end-to-end using a no-reference image quality loss function.

We train and test our model with a huge set of exposure stacks captured with diverse settings (indoor/outdoor, day/night, side-lighting/back-lighting, and so on). Furthermore, our model does not require parameter fine-tuning for varying input conditions. Through extensive experimental evaluations we demonstrate that the proposed architecture performs better than state-of-the-art approaches for a wide range of input scenarios.

The contributions of this work are as follows:

- A CNN based unsupervised image fusion algorithm for fusing exposure stacked static image pairs.
- A new benchmark dataset that can be used for comparing various MEF methods.
- An extensive experimental evaluation and comparison study against 7 state-of-the-art algorithms for variety of natural images.

The paper is organized as follows. Section 2, we briefly review related works from literature. Section 3, we present our CNN based exposure fusion algorithm and discuss the details of experiments. Section 4, we provide the fusion examples and then conclude the paper with an insightful discussion in section 5.

## 2. Related Works

Many algorithms have been proposed over the years for exposure fusion. However, the main idea remains the same in all the algorithms. The algorithms compute the weights for each image either locally or pixel wise. The fused image would then be the weighted sum of the images in the input sequence.

Burt *et al.* [3] performed a Laplacian pyramid decomposition of the image and the weights are computed using local energy and correlation between the pyramids. Use of Laplacian pyramids reduces the chance of unnecessary artifacts. Goshtasby *et al.* [5] take non-overlapping blocks with highest information from each image to obtain the fused result. This is prone to suffer from block artifacts. Mertens *et al.* [16] perform exposure fusion using simple quality metrics such as contrast and saturation. However, this suffers from hallucinated edges and mismatched color artifacts.

Algorithms which make use of edge preserving filters like Bilateral filters are proposed in [19]. As this does not account for the luminance of the images, the fused image has dark region leading to poor results. A gradient based approach to assign the weight was put forward by Zhang *et al.* [28]. In a series of papers by Li *et al.* [9], [10] different approaches to exposure fusion have been reported. In their early works they solve a quadratic optimization to extract finer details and fuse them. In one of their later works [10],

they propose a Guided Filter based approach.

Shen *et al.* [22] proposed a fusion technique using quality metrics such as local contrast and color consistency. The random walk approach they perform gives a global optimum solution to the fusion problem set in a probabilistic fashion.

All of the above works rely on hand-crafted features for image fusion. These methods are not robust in the sense that the parameters need to be varied for different input conditions say, linear and non-linear exposures, filter size depends on image sizes. To circumvent this parameter tuning we propose a feature learning based approach using CNN. In this work we learn suitable features for fusing exposure bracketed images. Recently, Convolutional Neural Network (CNN) have shown impressive performance across various computer vision tasks [8]. While CNNs have produced state-of-the-art results in many high-level computer vision tasks like recognition ([7], [21]), object detection [11], Segmentation [6], semantic labelling [17], visual question answering [2] and much more, their performance on low-level image processing problems such as filtering [4] and fusion [18] is not studied extensively. In this work we explore the effectiveness of CNN for the task of multi-exposure image fusion.

To our knowledge, use of CNNs for multi-exposure fusion is not reported in literature. The other machine learning approach is based on a regression method called Extreme Learning Machine (ELM) [25], that feed saturation level, exposedness, and contrast into the regressor to estimate the importance of each pixel. Instead of using hand crafted features, we use the data to learn a representation right from the raw pixels.

## 3. Proposed Method

In this work, we propose an image fusion framework using CNNs. Within a span of couple years, Convolutional Neural Networks have shown significant success in high-end computer vision tasks. They are shown to learn complex mappings between input and output with the help of sufficient training data. CNN learns the model parameters by optimizing a loss function in order to predict the result as close as to the ground-truth. For example, let us assume that input  $\mathbf{x}$  is mapped to output  $\mathbf{y}$  by some complex transformation  $f$ . The CNN can be trained to estimate the function  $f$  that minimizes the difference between the expected output  $\mathbf{y}$  and obtained output  $\hat{\mathbf{y}}$ . The distance between  $\mathbf{y}$  and  $\hat{\mathbf{y}}$  is calculated using a loss function, such as mean squared error function. Minimizing this loss function leads to better estimate of required mapping function.

Let us denote the input exposure sequence and fusion operator as  $I$  and  $O(I)$ . The input images are assumed to be registered and aligned using existing registration algorithms, thus avoiding camera and object motion. We model

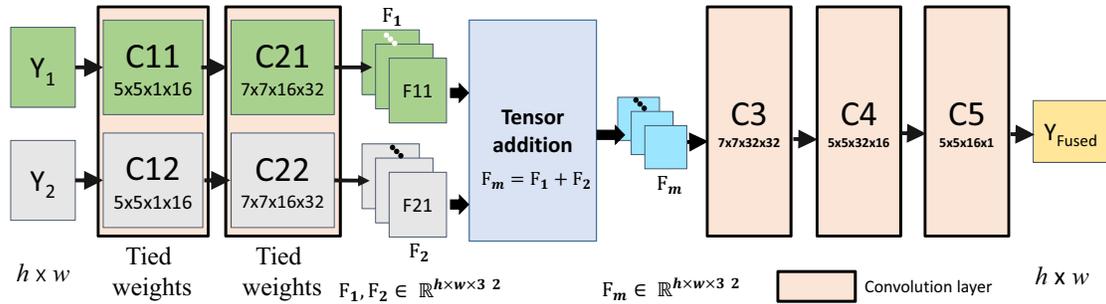


Figure 2. Architecture of proposed image fusion CNN illustrated for input exposure stack with images of size  $h \times w$ . The pre-fusion layers C1 and C2 that share same weights, extract low-level features from input images. The feature pairs of input images are fused into a single feature by merge layer. The fused features are input to reconstruction layers to generate fused image  $Y_{fused}$ .

$O(I)$  with a feed-forward process  $F_W(I)$ . Here,  $F$  denotes the network architecture and  $W$  denotes the weights learned by minimizing the loss function. As the expected output  $O(I)$  is absent for MEF problem, the squared error loss or any other full reference error metric cannot be used. Instead, we make use of no-reference image quality metric MEF SSIM proposed by Ma *et al.* [15] as loss function. MEF SSIM is based on structural similarity metric (SSIM) framework [27]. It makes use of statistics of a patch around individual pixels from input image sequence to compare with result. It measures the loss of structural integrity as well as luminance consistency in multiple scales (see section 3.1.1 for more details).

An overall scheme of proposed method is shown in Fig. 1. The input exposure stack is converted into YCbCr color channel data. The CNN is used to fuse the luminance channel of the input images. This is due to the fact that the image structural details are present in luminance channel and the brightness variation is prominent in luminance channel than chrominance channels. The obtained luminance channel is combined with chroma (Cb and Cr) channels generated using method described in section 3.3. The following subsection details the network architecture, loss function and the training procedure.

### 3.1. DeepFuse CNN

The learning ability of CNN is heavily influenced by right choice of architecture and loss function. A simple and naive architecture is to have a series of convolutional layers connected in sequential manner. The input to this architecture would be exposure image pairs stacked in third dimension. Since the fusion happens in the pixel domain itself, this type of architecture does not make use of feature learning ability of CNNs to a great extent.

The proposed network architecture for image fusion is illustrated in Fig. 2. The proposed architecture has three components: feature extraction layers, a fusion layer and re-

construction layers. As shown in Fig. 2, the under-exposed and the over-exposed images ( $Y_1$  and  $Y_2$ ) are input to separate channels (channel 1 consists of C11 and C21 and channel 2 consists of C12 and C22). The first layer (C11 and C12) contains  $5 \times 5$  filters to extract low-level features such as edges and corners. The weights of pre-fusion channels are *tied*, C11 and C12 (C21 and C22) share same weights. The advantage of this architecture is three fold: first, we force the network to learn the same features for the input pair. That is, the F11 and F21 are same feature type. Hence, we can simply combine the respective feature maps via fusion layer. Meaning, the first feature map of image 1 (F11) and the first feature map of image 2 (F21) are added and this process is applied for remaining feature maps as well. Also, adding the features resulted in better performance than other choices of combining features (see Table 1). In feature addition, similar feature types from both images are fused together. Optionally one can choose to concatenate features, by doing so, the network has to figure out the weights to merge them. In our experiments, we observed that feature concatenation can also achieve similar results by increasing the number of training iterations, increasing number of filters and layers after C3. This is understandable as the network needs more number of iterations to figure out appropriate fusion weights. In this tied-weights setting, we are enforcing the network to learn filters that are invariant to brightness changes. This is observed by visualizing the learned filters (see Fig. 8). In case of tied weights, few high activation filters have center surround receptive fields (typically observed in retina). These filters have learned to remove the mean from neighbourhood, thus effectively making the features brightness invariant. Second, the number of learnable filters is reduced by half. Third, as the network has low number of parameters, it converges quickly. The obtained features from C21 and C22 are fused by merge layer. The result of fuse layer is then passed through another

set of convolutional layers (C3, C4 and C5) to reconstruct final result ( $Y_{fused}$ ) from fused features.

### 3.1.1 MEF SSIM loss function

In this section, we will discuss on computing loss without using reference image by MEF SSIM image quality measure [15]. Let  $\{\mathbf{y}_k\}=\{\mathbf{y}_k|k=1,2\}$  denote the image patches extracted at a pixel location  $p$  from input image pairs and  $\mathbf{y}_f$  denote the patch extracted from CNN output fused image at same location  $p$ . The objective is to compute a score to define the fusion performance given  $\mathbf{y}_k$  input patches and  $\mathbf{y}_f$  fused image patch.

In SSIM [27] framework, any patch can be modelled using three components: structure ( $s$ ), luminance ( $l$ ) and contrast ( $c$ ). The given patch is decomposed into these three components as:

$$\begin{aligned} \mathbf{y}_k &= \|\mathbf{y}_k - \mu_{\mathbf{y}_k}\| \cdot \frac{\mathbf{y}_k - \mu_{\mathbf{y}_k}}{\|\mathbf{y}_k - \mu_{\mathbf{y}_k}\|} + \mu_{\mathbf{y}_k} \\ &= \|\tilde{\mathbf{y}}_k\| \cdot \frac{\tilde{\mathbf{y}}_k}{\|\tilde{\mathbf{y}}_k\|} + \mu_{\mathbf{y}_k} \\ &= c_k \cdot \mathbf{s}_k + l_k, \end{aligned} \quad (1)$$

where,  $\|\cdot\|$  is the  $\ell_2$  norm of patch,  $\mu_{\mathbf{y}_k}$  is the mean value of  $\mathbf{y}_k$  and  $\tilde{\mathbf{y}}_k$  is the mean subtracted patch. As the higher contrast value means better image, the desired contrast value ( $\hat{c}$ ) of the result is taken as the highest contrast value of  $\{c_k\}$ , (i.e.)

$$\hat{c} = \max_{\{k=1,2\}} c_k$$

The structure of the desired result ( $\hat{s}$ ) is obtained by weighted sum of structures of input patches as follows,

$$\bar{\mathbf{s}} = \frac{\sum_{k=1}^2 w(\tilde{\mathbf{y}}_k) \mathbf{s}_k}{\sum_{k=1}^2 w(\tilde{\mathbf{y}}_k)} \quad \text{and} \quad \hat{\mathbf{s}} = \frac{\bar{\mathbf{s}}}{\|\bar{\mathbf{s}}\|}, \quad (2)$$

where the weighting function assigns weight based on structural consistency between input patches. The weighting function assigns equal weights to patches, when they have dissimilar structural components. In the other case, when all input patches have similar structures, the patch with high contrast is given more weight as it is more robust to distortions. The estimated  $\hat{s}$  and  $\hat{c}$  is combined to produce desired result patch as,

$$\hat{\mathbf{y}} = \hat{c} \cdot \hat{\mathbf{s}} \quad (3)$$

As the luminance comparison in the local patches is insignificant, the luminance component is discarded from above equation. Comparing luminance at lower spatial resolution does not reflect the global brightness consistency. Instead, performing this operation at multiple scales would effectively capture global luminance consistency in coarser

Table 1. **Choice of blending operators:** Average MEF SSIM scores of 23 test images generated by CNNs trained with different feature blending operations. The maximum score is highlighted in bold. Results illustrate that adding the feature tensors yield better performance. Results by addition and mean methods are similar, as both operations are very similar, except for a scaling factor. Refer text for more details.

Product	Concatenation	Max	Mean	Addition
0.8210	0.9430	0.9638	0.9750	<b>0.9782</b>

scale and local structural changes in finer scales. The final image quality score for pixel  $p$  is calculated using SSIM framework,

$$Score(p) = \frac{2\sigma_{\hat{\mathbf{y}}\mathbf{y}_f} + C}{\sigma_{\hat{\mathbf{y}}}^2 + \sigma_{\mathbf{y}_f}^2 + C}, \quad (4)$$

where,  $\sigma_{\hat{\mathbf{y}}}^2$  is variance and  $\sigma_{\hat{\mathbf{y}}\mathbf{y}_f}$  is covariance between  $\hat{\mathbf{y}}$  and  $\mathbf{y}_f$ . The total loss is calculated as,

$$Loss = 1 - \frac{1}{N} \sum_{p \in P} Score(p) \quad (5)$$

where  $N$  is the total number of pixels in image and  $P$  is the set of all pixels in input image. The computed loss is back-propagated to train the network. The better performance of MEF SSIM is attributed to its objective function that maximizes structural consistency between fused image and each of input images.

### 3.2. Training

We have collected 25 exposure stacks that are available publicly [1]. In addition to that, we have curated 50 exposure stacks with different scene characteristics. The images were taken with standard camera setup and tripod. Each scene consists of 2 low dynamic range images with  $\pm 2$  EV difference. The input sequences are resized to  $1200 \times 800$  dimensions. We give priority to cover both indoor and outdoor scenes. From these input sequences, 30000 patches of size  $64 \times 64$  were cropped for training. We set the learning rate to  $10^{-4}$  and train the network for 100 epochs with all the training patches being processed in each epoch.

### 3.3. Testing

We follow the standard cross-validation procedure to train our model and test the final model on a disjoint test set to avoid over-fitting. While testing, the trained CNN takes the test image sequence and generates the luminance channel ( $Y_{fused}$ ) of fused image. The chrominance components of fused image,  $Cb_{fused}$  and  $Cr_{fused}$ , are obtained by weighted sum of input chrominance channel values.

The crucial structural details of the image tend to be present mainly in  $Y$  channel. Thus, different fusion strategies are followed in literature for  $Y$  and  $Cb/Cr$  fusion ([18], [24], [26]). Moreover, MEF SSIM loss is formulated to compute the score between 2 gray-scale ( $Y$ ) images. Thus,

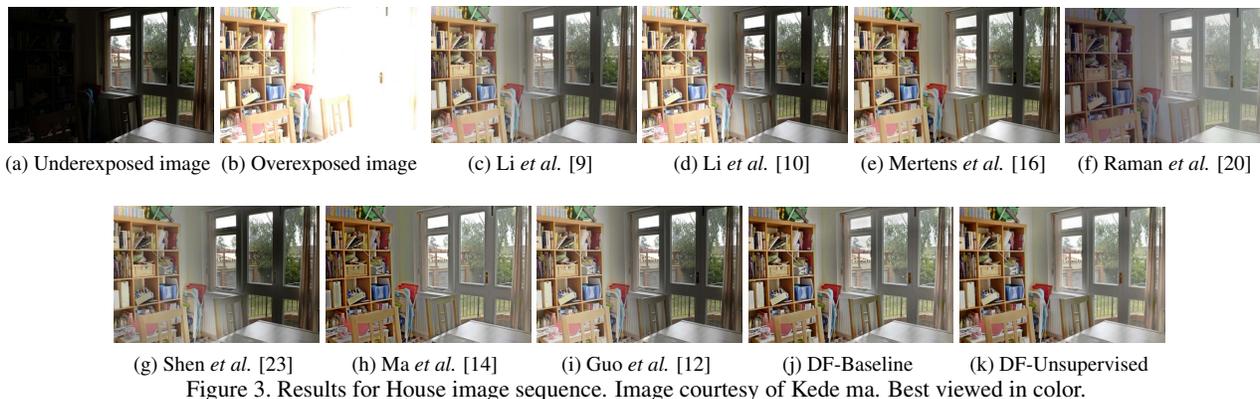


Figure 3. Results for House image sequence. Image courtesy of Kede ma. Best viewed in color.

measuring MEF SSIM for  $C_b$  and  $C_r$  channels may not be meaningful. Alternately, one can choose to fuse RGB channels separately using different networks. However, there is typically a large correlation between RGB channels. Fusing RGB independently fails to capture this correlation and introduces noticeable color difference. Also, MEF-SSIM is not designed for RGB channels. Another alternative is to regress RGB values in a single network, then convert them to a  $Y$  image and compute MEF SSIM loss. Here, the network can focus more on improving  $Y$  channel, giving less importance to color. However, we observed spurious colors in output which were not originally present in input.

We follow the procedure used by Prabhakar *et al.* [18] for chrominance channel fusion. If  $x_1$  and  $x_2$  denote the  $C_b$  (or  $C_r$ ) channel value at any pixel location for image pairs, then the fused chrominance value  $x$  is obtained as follows,

$$x = \frac{x_1(|x_1 - \tau|) + x_2(|x_2 - \tau|)}{|x_1 - \tau| + |x_2 - \tau|} \quad (6)$$

The fused chrominance value is obtained by weighing two chrominance values with  $\tau$  subtracted value from itself. The value of  $\tau$  is chosen as 128. The intuition behind this approach is to give more weight for good color components and less for saturated color values. The final result is obtained by converting  $\{Y_{fused}, C_b_{fused}, C_r_{fused}\}$  channels into RGB image.

#### 4. Experiments and Results

We have conducted extensive evaluation and comparison study against state-of-the-art algorithms for variety of natural images<sup>2</sup>. For evaluation, we have chosen standard image sequences to cover different image characteristics including indoor and outdoor, day and night, natural and artificial lighting, linear and non-linear exposure. The proposed algorithm is compared against seven best performing MEF algorithms, (1) Mertens09 [16], (2) Li13 [10] (3) Li12 [9]

<sup>2</sup>Due to space limitations, we have shown results for only few sequences. More results are shared in supplementary material

(4) Ma15 [14] (5) Raman11 [20] (6) Shen11 [23] and (7) Guo17 [12]. In order to evaluate the performance of algorithms objectively, we adopt MEF SSIM. Although number of other IQA models for general image fusion have also been reported, none of them makes adequate quality predictions of subjective opinions [15].

##### 4.1. DeepFuse - Baseline

So far, we have discussed on training CNN model in unsupervised manner. One interesting variant of that would be to train the CNN model with results of other state-of-art methods as ground truth. This experiment can test the capability of CNN to learn complex fusion rules from data itself without the help of MEF SSIM loss function. The ground truth is selected as best of Mertens [16] and GFF [10] methods based on MEF SSIM score<sup>3</sup>. The choice of loss function to calculate error between ground truth and estimated output is very crucial for training a CNN in supervised fashion. The Mean Square Error or  $\ell_2$  loss function is generally chosen as default cost function for training CNN. The  $\ell_2$  cost function is desired for its smooth optimization properties. While  $\ell_2$  loss function is better suited for classification tasks, they may not be a correct choice for image processing tasks [29]. It is also a well known phenomena that MSE does not correlate well with human perception of image quality [27]. In order to obtain visually pleasing result, the loss function should be well correlated with HVS, like Structural Similarity Index (SSIM) [27]. We have experimented with different loss functions such as  $\ell_1$ ,  $\ell_2$  and SSIM.

The fused image appear blurred when the CNN was trained with  $\ell_2$  loss function. This effect termed as *regression to mean*, is due to the fact that  $\ell_2$  loss function compares the result and ground truth in a pixel by pixel manner. The result by  $\ell_1$  loss gives sharper result than  $\ell_2$  loss but it has halo effect along the edges. Unlike  $\ell_1$  and  $\ell_2$ , results by CNN trained with SSIM loss function are both sharp and

<sup>3</sup>In a user survey conducted by Ma *et al.* [15], Mertens and GFF results are ranked better than other MEF algorithms

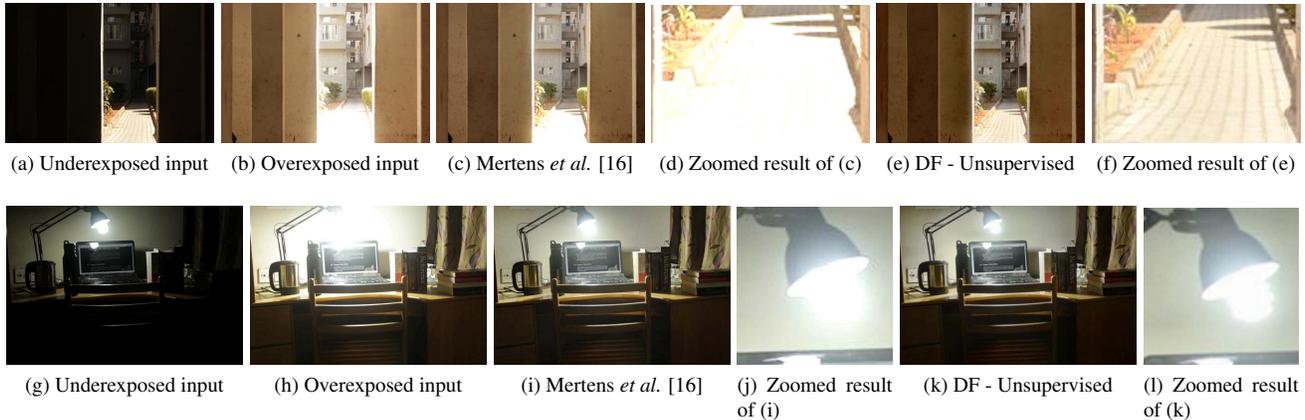


Figure 4. Comparison of the proposed method with Mertens *et al.* [16]. The Zoomed region of the result by Mertens *et al.* in (d) show that some highlight regions are not completely retained from input. The zoomed region of the result by Mertens *et al.* in (j) show that fine details of lamp are missing.

Table 2. MEF SSIM scores of different methods against DeepFuse (DF) for test images. Bolded values indicate the highest score by that corresponding column algorithm than others for that row image sequence.

	<i>Mertens09</i>	<i>Raman11</i>	<i>Li12</i>	<i>Li13</i>	<i>Shen11</i>	<i>Ma15</i>	<i>Guo17</i>	<i>DF-Baseline</i>	<i>DF-UnSupervised</i>
<i>AgiaGalini</i>	0.9721	0.9343	0.9438	0.9409	0.8932	0.9465	0.9492	0.9477	<b>0.9813</b>
<i>Balloons</i>	0.9601	0.897	0.9464	0.9366	0.9252	0.9608	0.9348	0.9717	<b>0.9766</b>
<i>Belgium house</i>	0.9655	0.8924	0.9637	0.9673	0.9442	0.9643	0.9706	0.9677	<b>0.9727</b>
<i>Building</i>	0.9801	0.953	0.9702	0.9685	0.9513	0.9774	0.9666	0.965	<b>0.9826</b>
<i>Cadik lamp</i>	0.9658	0.8696	0.9472	0.9434	0.9152	0.9464	0.9484	<b>0.9683</b>	0.9638
<i>Candle</i>	0.9681	0.9391	0.9479	0.9017	0.9441	0.9519	0.9451	0.9704	<b>0.9893</b>
<i>Chinese garden</i>	<b>0.990</b>	0.8887	0.9814	0.9887	0.9667	<b>0.990</b>	0.9860	0.9673	0.9838
<i>Corridor</i>	0.9616	0.898	0.9709	0.9708	0.9452	0.9592	0.9715	<b>0.9740</b>	<b>0.9740</b>
<i>Garden</i>	0.9715	0.9538	0.9431	0.932	0.9136	0.9667	0.9481	0.9385	<b>0.9872</b>
<i>Hostel</i>	0.9678	0.9321	0.9745	0.9742	0.9649	0.9712	0.9757	0.9715	<b>0.985</b>
<i>House</i>	<b>0.9748</b>	0.8319	0.9575	0.9556	0.9356	0.9365	0.9623	0.9601	0.9607
<i>Kluki Bartlomiej</i>	<b>0.9811</b>	0.9042	0.9659	0.9645	0.9216	0.9622	0.9680	0.9723	0.9742
<i>Landscape</i>	0.9778	0.9902	0.9577	0.943	0.9385	0.9817	0.9467	0.9522	<b>0.9913</b>
<i>Lighthouse</i>	0.9783	0.9654	0.9658	0.9545	0.938	0.9702	0.9657	0.9728	<b>0.9875</b>
<i>Madison capitol</i>	0.9731	0.8702	0.9516	0.9668	0.9414	0.9745	0.9711	0.9459	<b>0.9749</b>
<i>Memorial</i>	0.9676	0.7728	0.9644	<b>0.9771</b>	0.9547	0.9754	0.9739	0.9727	0.9715
<i>Office</i>	<b>0.9749</b>	0.922	0.9367	0.9495	0.922	0.9746	0.9624	0.9277	<b>0.9749</b>
<i>Room</i>	0.9645	0.8819	0.9708	<b>0.9775</b>	0.9543	0.9641	0.9725	0.9767	0.9724
<i>SwissSunset</i>	0.9623	0.9168	0.9407	0.9137	0.8155	0.9512	0.9274	0.9736	<b>0.9753</b>
<i>Table</i>	0.9803	0.9396	0.968	0.9501	0.9641	0.9735	0.9750	0.9468	<b>0.9853</b>
<i>TestChart1</i>	0.9769	0.9281	0.9649	0.942	0.9462	0.9529	0.9617	0.9802	<b>0.9831</b>
<i>Tower</i>	<b>0.9786</b>	0.9128	0.9733	0.9779	0.9458	0.9704	0.9772	0.9734	0.9738
<i>Venice</i>	<b>0.9833</b>	0.9581	0.961	0.9608	0.9307	0.9836	0.9632	0.9562	0.9787

artifact-free. Therefore, SSIM is used as loss function to calculate error between generated output and ground truth in this experiment<sup>4</sup>.

The quantitative comparison between DeepFuse baseline and unsupervised method is shown in Table 2. The MEF SSIM scores in Table 2 shows the superior performance of DeepFuse unsupervised over baseline method in almost all test sequences. The reason is due to the fact that for baseline method, the amount of learning is upper bound by the other algorithms, as the ground truth for baseline method is from Mertens *et al.* [16] or Li *et al.* [10]. We see from Table 2

<sup>4</sup>Refer to supplementary material for more detailed analysis on results by network trained with different loss functions

that the baseline method does not exceed both of them.

The idea behind this experiment is to combine advantages of all previous methods, at the same time avoid shortcomings of each. From Fig. 3, we can observe that though DF-baseline is trained with results of other methods, it can produce results that do not have any artifacts observed in other results.

## 4.2. Comparison with State-of-the-art

*Comparison with Mertens et al.:* Mertens *et al.* [16] is a simple and effective weighting based image fusion technique with multi resolution blending to produce smooth results. However, it suffers from following shortcomings: (a) it picks “best” parts of each image for fusion using hand

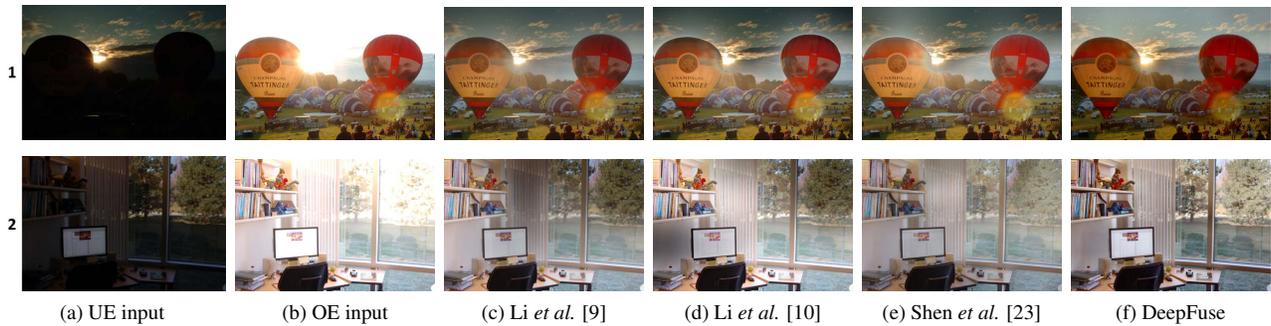


Figure 5. Comparison of the proposed method with Li *et al.* [9], Li *et al.* [10] and Shen *et al.* [23] for *Balloons* and *Office*. Image courtesy of Kede ma.

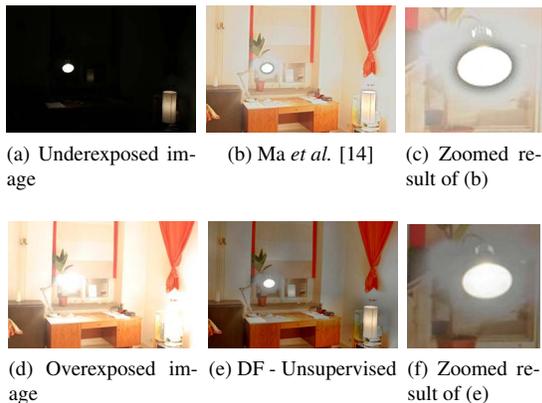


Figure 6. Comparison of the proposed method with Ma *et al.* [14] for *Table* sequence. The zoomed region of result by Ma *et al.* [14] shows the artificial halo artifact effect around edges of lamp. Image courtesy of Kede ma.



Figure 7. Comparison of the proposed method with Ma *et al.* [14]. A close-up look on the results for *Lighthouse* sequence. The results by Ma *et al.* [14] show a halo effect along the roof and lighthouse. Image courtesy of Kede Ma.

crafted features like saturation and well-exposedness. This approach would work better for image stacks with many exposure images. But for exposure image pairs, it fails to maintain uniform brightness across whole image. Compared to Mertens *et al.*, DeepFuse produces images with

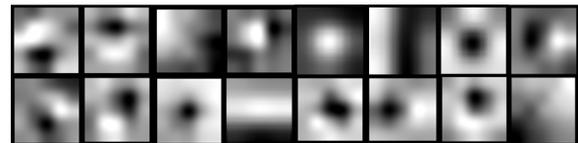


Figure 8. **Filter Visualization.** Some of the filters learnt in first layer resemble Gaussian, Difference of Gaussian and Laplacian of Gaussian filters. Best viewed electronically, zoomed in.

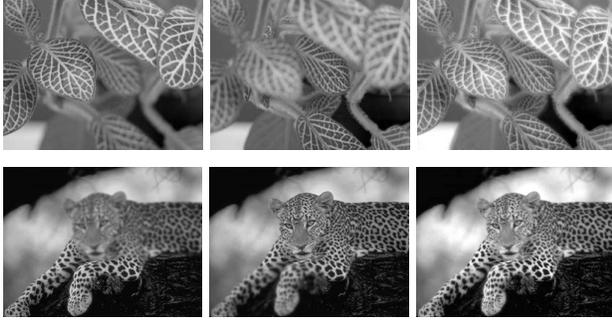
consistent and uniform brightness across whole image. (b) Mertens *et al.* does not preserve complete image details from under exposed image. In Fig. 4(d), the details of the tile area is missing in Mertens *et al.*'s result. The same is the case in Fig. 4(j), the fine details of the lamp are not present in the Mertens *et al.* result. Whereas, DeepFuse has learned filters that extract features like edges and textures in C1 and C2, and preserves finer structural details of the scene.

*Comparison with Li et al.* [9] [10]: It can be noted that, similar to Mertens *et al.* [16], Li *et al.* [9] [10] also suffers from non-uniform brightness artifact (Fig. 5). In contrast, our algorithm provides a more pleasing image with clear texture details.

*Comparison with Shen et al.* [23]: The results generated by Shen *et al.* show contrast loss and non-uniform brightness distortions (Fig. 5). In Fig. 5(e1), the brightness distortion is present in the cloud region. The cloud regions in between balloons appear darker compared to other regions. This distortion can be observed in other test images as well in Fig. 5(e2). However, the DeepFuse (Fig. 5(f1) and (f2) ) have learnt to produce results without any of these artifacts.

*Comparison with Ma et al.* [14]: Fig. 6 and 7 shows comparison between results of Ma *et al.* and DeepFuse for *Lighthouse* and *Table* sequences. Ma *et al.* proposed a patch based fusion algorithm that fuses patches from input images based on their patch strength. The patch strength is calculated using a power weighting function on each patch. This method of weighting would introduce unpleasant halo effect along edges (see Fig. 6 and 7).

*Comparison with Raman et al.* [20]: Fig. 3(f) shows



(a) Near focused image (b) Far focused image (c) DF result  
 Figure 9. Application of DeepFuse CNN to multi-focus fusion. The first two column images are input varying focus images. The All-in-focus result by DeepFuse is shown in third column. Images courtesy of Liu *et al.* [13]. Image courtesy of Slavica savic.

the fused result by Raman *et al.* for House sequence. The result exhibit color distortion and contrast loss. In contrast, proposed method produces result with vivid color quality and better contrast.

After examining the results by both subjective and objective evaluations, we observed that our method is able to faithfully reproduce all the features in the input pair. We also notice that the results obtained by DeepFuse are free of artifacts such as darker regions and mismatched colors. Our approach preserves the finer image details along with higher contrast and vivid colors. The quantitative comparison between proposed method and existing approaches in Table 2 also shows that proposed method outperforms others in most of the test sequences. From the execution times shown in Table 3 we can observe that our method is roughly 3-4 $\times$  faster than Mertens *et al.* DeepFuse can be easily extended to more input images by adding additional streams before merge layer. We have trained DeepFuse for sequences with 3 and 4 images. For sequences with 3 images, average MEF SSIM score for DF is 0.987 and 0.979 for Mertens *et al.* For sequences with 4 images, average MEF SSIM score for DF is 0.972 and 0.978 for Mertens *et al.* For sequences with 4 images, we attribute dip in performance to insufficient training data. With more training data, DF can be trained to perform better in such cases as well.

### 4.3. Application to Multi-Focus Fusion

In this section, we discuss the possibility of applying our DeepFuse model for solving other image fusion problems. Due to the limited depth-of-field in the present day cameras, only object in limited range of depth are focused and the remaining regions appear blurry. In such scenario, Multi-Focus Fusion (MFF) techniques are used to fuse images taken with varying focus to generate a single all-in-focus image. MFF problem is very similar to MEF, except that the input images have varying focus than varying exposure for MEF. To test the generalizability of CNN, we have used the

Table 3. **Computation time:** Running time in seconds of different algorithms on a pair of images. The numbers in bold denote the least amount of time taken to fuse. ‡: tested with NVIDIA Tesla K20c GPU, †: tested with Intel<sup>®</sup>Xeon @ 3.50 GHz CPU

Image size	Ma15 <sup>†</sup>	Li13 <sup>†</sup>	Mertens07 <sup>†</sup>	DF <sup>‡</sup>
512*384	2.62	0.58	0.28	<b>0.07</b>
1024*768	9.57	2.30	0.96	<b>0.28</b>
1280*1024	14.72	3.67	1.60	<b>0.46</b>
1920*1200	27.32	6.60	2.76	<b>0.82</b>

already trained DeepFuse CNN to fuse multi-focus images without any fine-tuning for MFF problem. Fig. 9 shows that the DeepFuse results on publicly available multi-focus dataset show that the filters of CNN have learnt to identify proper regions in each input image and successfully fuse them together. It can also be seen that the learnt CNN filters are generic and could be applied for general image fusion.

## 5. Conclusion and Future work

In this paper, we have proposed a method to efficiently fuse a pair of images with varied exposure levels to produce an output which is artifact-free and perceptually pleasing. DeepFuse is the first ever unsupervised deep learning method to perform static MEF. The proposed model extracts set of common low-level features from each input images. Feature pairs of all input images are fused into a single feature by merge layer. Finally, the fused features are input to reconstruction layers to get the final fused image. We train and test our model with a huge set of exposure stacks captured with diverse settings. Furthermore, our model is free of parameter fine-tuning for varying input conditions. Finally, from extensive quantitative and qualitative evaluation, we demonstrate that the proposed architecture performs better than state-of-the-art approaches for a wide range of input scenarios.

In summary, the advantages offered by DF are as follows: 1) Better fusion quality: produces better fusion result even for extreme exposure image pairs, 2) SSIM over  $\ell_1$ : In [29], the authors report that  $\ell_1$  loss outperforms SSIM loss function. In their work, the authors have implemented approximate version of SSIM and found it to perform sub-par compared to  $\ell_1$ . We have implemented the exact SSIM formulation and observed that SSIM loss function perform much better than MSE and  $\ell_1$ . Further, we have shown that a complex perceptual loss such as MEF SSIM can be successfully incorporated with CNNs in absence of ground truth data. The results encourage the research community to examine other perceptual quality metrics and use them as loss functions to train a neural net. 3) Generalizability to other fusion tasks: The proposed fusion is generic in nature and could be easily adapted to other fusion problems as well. In our current work, DF is trained to fuse static images. For future research, we aim to generalize DeepFuse to fuse images with object motion as well.

## References

- [1] EMPA HDR image database. <http://www.empamedia.ethz.ch/hdrdatabase/index.php>. Accessed: 2016-07-13.
- [2] S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. Lawrence Zitnick, and D. Parikh. VQA: Visual question answering. In *Proceedings of the IEEE International Conference on Computer Vision*, 2015.
- [3] P. J. Burt and R. J. Kolczynski. Enhanced image capture through fusion. In *Proceedings of the International Conference on Computer Vision*, 1993.
- [4] N. Divakar and R. V. Babu. Image denoising via CNNs: An adversarial approach. In *New Trends in Image Restoration and Enhancement, CVPR workshop*, 2017.
- [5] A. A. Goshtasby. Fusion of multi-exposure images. *Image and Vision Computing*, 23(6):611–618, 2005.
- [6] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask R-CNN. *arXiv preprint arXiv:1703.06870*, 2017.
- [7] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.
- [8] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.
- [9] S. Li and X. Kang. Fast multi-exposure image fusion with median filter and recursive filter. *IEEE Transaction on Consumer Electronics*, 58(2):626–632, May 2012.
- [10] S. Li, X. Kang, and J. Hu. Image fusion with guided filtering. *IEEE Transactions on Image Processing*, 22(7):2864–2875, July 2013.
- [11] Y. Li, K. He, J. Sun, et al. R-fcn: Object detection via region-based fully convolutional networks. In *Advances in Neural Information Processing Systems*, 2016.
- [12] Z. Li, Z. Wei, C. Wen, and J. Zheng. Detail-enhanced multi-scale exposure fusion. *IEEE Transactions on Image Processing*, 26(3):1243–1252, 2017.
- [13] Y. Liu, S. Liu, and Z. Wang. Multi-focus image fusion with dense SIFT. *Information Fusion*, 23:139–155, 2015.
- [14] K. Ma and Z. Wang. Multi-exposure image fusion: A patch-wise approach. In *IEEE International Conference on Image Processing*, 2015.
- [15] K. Ma, K. Zeng, and Z. Wang. Perceptual quality assessment for multi-exposure image fusion. *IEEE Transactions on Image Processing*, 24(11):3345–3356, 2015.
- [16] T. Mertens, J. Kautz, and F. Van Reeth. Exposure fusion. In *Pacific Conference on Computer Graphics and Applications*, 2007.
- [17] P. H. Pinheiro and R. Collobert. Recurrent convolutional neural networks for scene parsing. *arXiv preprint arXiv:1306.2795*, 2013.
- [18] K. R. Prabhakar and R. V. Babu. Ghosting-free multi-exposure image fusion in gradient domain. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2016.
- [19] S. Raman and S. Chaudhuri. Bilateral filter based compositing for variable exposure photography. In *Proceedings of EUROGRAPHICS*, 2009.
- [20] S. Raman and S. Chaudhuri. Reconstruction of high contrast images for dynamic scenes. *The Visual Computer*, 27:1099–1114, 2011. 10.1007/s00371-011-0653-0.
- [21] R. K. Sarvadevabhatla, J. Kundu, et al. Enabling my robot to play pictictionary: Recurrent neural networks for sketch recognition. In *Proceedings of the ACM on Multimedia Conference*, 2016.
- [22] J. Shen, Y. Zhao, S. Yan, X. Li, et al. Exposure fusion using boosting laplacian pyramid. *IEEE Trans. Cybernetics*, 44(9):1579–1590, 2014.
- [23] R. Shen, I. Cheng, J. Shi, and A. Basu. Generalized random walks for fusion of multi-exposure images. *IEEE Transactions on Image Processing*, 20(12):3634–3646, 2011.
- [24] M. Tico and K. Pulli. Image enhancement method via blur and noisy image fusion. In *IEEE International Conference on Image Processing*, 2009.
- [25] J. Wang, B. Shi, and S. Feng. Extreme learning machine based exposure fusion for displaying HDR scenes. In *International Conference on Signal Processing*, 2012.
- [26] J. Wang, D. Xu, and B. Li. Exposure fusion based on steerable pyramid for displaying high dynamic range scenes. *Optical Engineering*, 48(11):117003–117003, 2009.
- [27] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.
- [28] W. Zhang and W.-K. Cham. Reference-guided exposure fusion in dynamic scenes. *Journal of Visual Communication and Image Representation*, 23(3):467–475, 2012.
- [29] H. Zhao, O. Gallo, I. Frosio, and J. Kautz. Loss functions for neural networks for image processing. *arXiv preprint arXiv:1511.08861*, 2015.