

INDIAN INSTITUTE OF TECHNOLOGY GANDHINAGAR

Deep Neural Networks for HDR imaging

Author:
Kshiteej Sheth

Supervisor:
Dr. Shanmuganathan Raman

November 3, 2016



Contents

1. Abstract
2. Introduction
3. CNN Architecture
4. Implementation details
5. Results
6. References

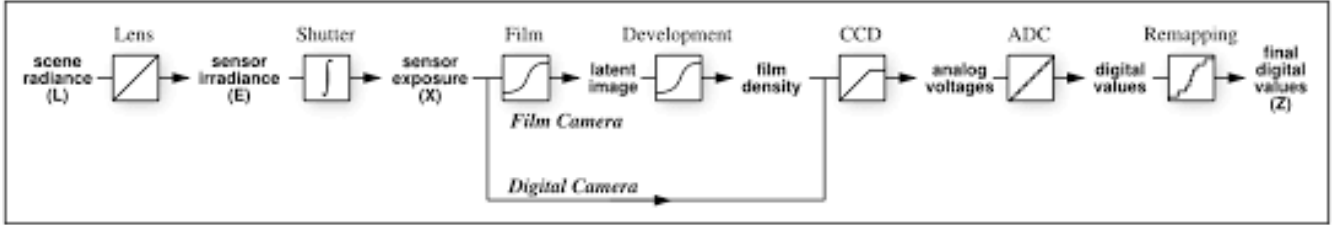
1 Abstract

We propose novel methods of solving two tasks using Convolutional Neural Networks, firstly the task of generating HDR map of a static scene using differently exposed LDR images of the scene captured using conventional cameras and secondly the task of finding an optimal tone mapping operator that would give a better score on the TMQI metric compared to the existing methods. We quantitatively show the performance of our networks and illustrate the cases where our networks performs good as well as bad.

2 Introduction

Natural scenes have a large range of intensity values (thus a large dynamic range) and conventional non-HDR cameras cannot capture this range in a single image. By controlling various factors, one of them being the exposure time of the shot we can capture a particular window in the total dynamic range of the scene. So we need multiple "Low dynamic range" images of the scene to get the complete information of the scene. Fig.1 illustrates an example. The internal processing pipeline of the camera is highly non linear i.e. the pixel intensity value at location (i, j) , Z_{ij} is equal to $f(E_{ij}\Delta t)$ where Δt is the exposure time of the shot and E_{ij} is the irradiance value at location (i, j) . $f(x)$ is known as the camera response function of that particular camera which is a non linear function. Given the values of Z_{ij} for differently exposed images of the same scene we can get a robust, noise free estimate of the E_{ij} 's (methods like Debevec et. al. (1997) use a weighted average of the $f^{-1}(Z_{ij})/\Delta t$ to get a robust estimate of the corresponding E_{ij} . We use a deep neural network to estimate the function taking as input the LDR pixel intensities of 5 LDR images of a static scene to estimate the irradiance values, i.e. the HDR map of the scene. Fig 1. shows the camera acquisition pipeline in modern cameras and the non-linear transforms involved.

We further conduct experiments of getting another similar convolutional neural network to approximate a tone mapping operator. Our training set includes HDR images of scenes and their corresponding tone mapped images generated by one of the Tone mapping operators provided in MATLAB's HDR-Toolbox (Banterle et al. 2011) which gives the highest value of the TMQI metric (Yeganeh et al. 2013). We try further experiments to improve the results, details of which are provided in the main report.



This is the end to end pipeline involved in digital image acquisition in modern digital cameras, which indicates that the mapping from the scene radiance to the pixel values at a spatial location in the image is non-linear.

3 CNN Architecture

3.1 LDR2HDR network architecture

We have collected a dataset of 957 HDR images from the internet from the websites of various groups around the world working on HDR imaging and the camera response functions of the cameras with which those photos were taken. We use these to generate LDR images from the HDR's whose exposure time values are taken from a list of exposure times (a geometric progression with the first term 1, common ratio 4 and the last term equal to 4^9). Our network takes as input a stack of 5 RGB LDR images of a particular scene, each having a different exposure. In the initial experiments we fixed the exposure of these LDR images to be $[1, 8, 64, 512, 4096]$, and then we moved on to adaptive exposure based method in which we choose first the LDR image of a scene which gives the maximum value of entropy (entropy of a single channel image is defined as $-\sum p \log p$ where p is the histogram count of a particular intensity value in the image and the sum is calculated for all the possible intensities in the image) and we take two images of the previous and next two intensity values. We have 3 networks, one for each of the R, G and B channels of the inputs, so we conducted many experiments with both the former and the latter case approach but we were able to obtain plausible results only in the first case where the exposure times were fixed. The graphs of training error vs. epochs for 3 different models which turned out to be the best after testing many different sets of hyperparameters are shown below for the case where the exposure times are kept constant. The final test error that we get for the best model is . We conducted experiments in models with dropout in each layer with $p = 0.4$ in order to improve generalization. We also added spatial batch normalization just before passing the activations of each layer through the ReLU non linearity. Batch size was kept 40 (decided by the memory limitations of the GPU). BatchNorm strictly improved the results as the same training error was attained in less number of epochs. The architecture of the network is illustrated in the Fig 2

3.2 HDR2ToneMap network architecture

We first create a dataset using the existing 957 HDR images. We then use the tone mapping operators provided in the HDR-toolbox by Francesco Banterle et al. and use them to create different tone maps of each HDR and run the TMQI metric on each of the tone maps and choose the one which gives the highest TMQI score. Some are local and some are global tone mapping operators, so our approach is not fully justified. We then train a convolutional neural network whose input is the HDR image and its corresponding truth is the best Tone map corresponding to the TMQI metric. We use 3x3 conv in the first layer followed by 1x1 convs in the subsequent

layers. We get this intuition from the works of Mertens et al. whose method most of the time gave the best TMQI score. In their work the final intensity value at location (x, y) depends only on the 3×3 neighborhood of radiance value in its corresponding HDR image at location (x, y) . Further study of the other tone mapping works is required in order to improve the architecture after the pilot testing that we have done in the course of the summer. After preliminary results it was observed that the network was not able to deal with high frequency inputs simultaneously with low frequency ones so in order to tackle that problem we first convert both the input and output pairs to Lab space, apply a Bilateral Filter to the L channel, create a new channel $L_{original} - L_{filtered}$ and train 4 networks each for these new channels as well as for a and b channels. We obtain better results for this method. In order to obtain a good estimate of the hyperparameters of the network, we test out several values of them by training their corresponding architectures for 2 epochs and observing the validation error. Due to computational constraints, this could not be afforded for more epochs and only 4 sets of hyperparameters could be tested.

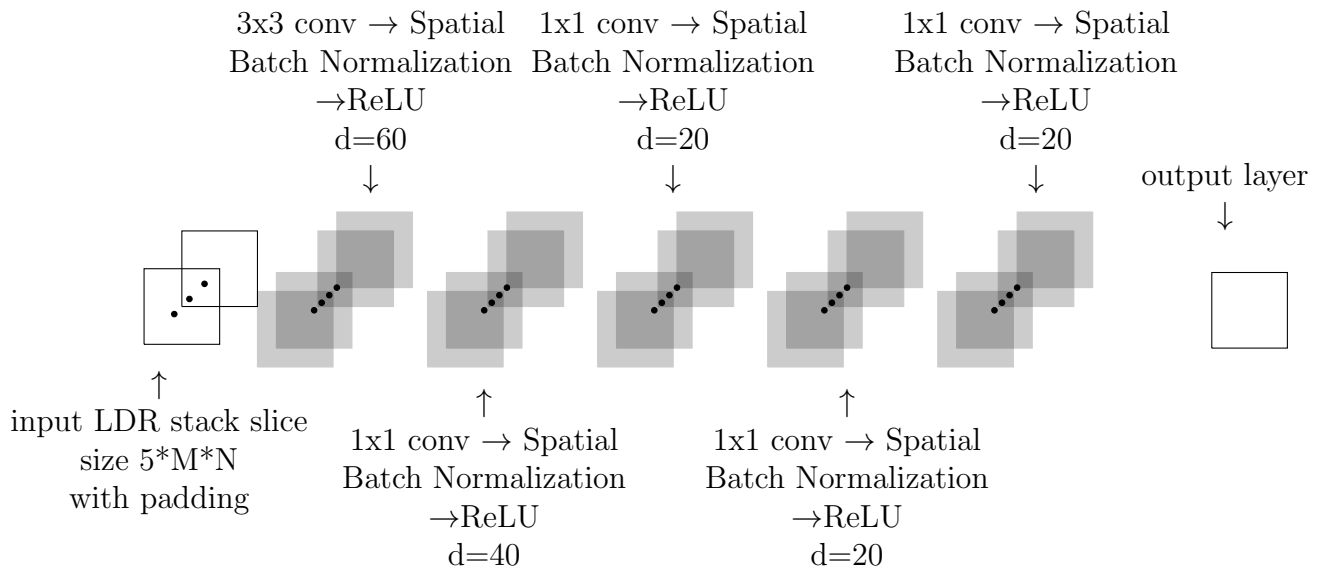


Figure 1: This is the network architecture which is used to generate the HDR image from the LDR stack. There are 3 networks for each of the R,G and B channels.

4 Implementation Details

For all the data processing tasks we use MATLAB and for implementing and testing our neural networks we use the Torch framework in the Lua scripting language, and most of the models are trained on a single NVIDIA GeForce GT 730 graphics processor, although for a brief amount of time during which we had access to a HPC node which had 2 GPU's, a Tesla K20C and a Titan X, we did multi-GPU training of our models using the following algorithm -

- Have the same network on the 2 GPU's at the beginning of every iteration in an epoch.
- Independently processing two different batches on the two GPU's and then copying over the accumulated gradients in the backward pass on one of the GPU's to the other, adding them to the accumulated gradients on the other GPU during the backward pass on it .

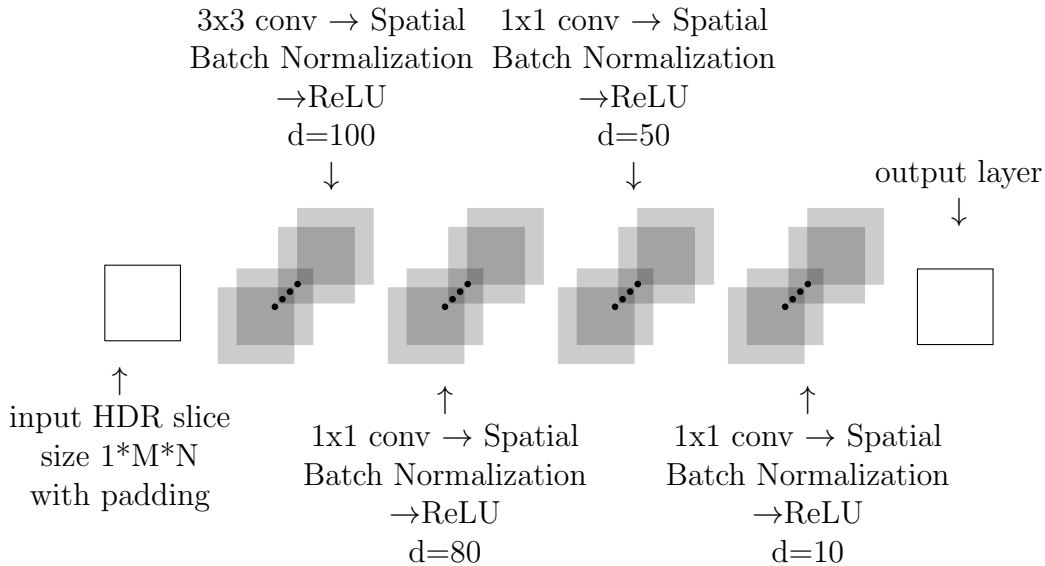


Figure 2: This is the network architecture which is used to approximate the tone mapping operator, there are four networks, one for the bilaterally filtered L channel, Original L minus the filtered L channel, a and b channel of the of the HDR image

- Updating the parameters of the model on the GPU to which the gradients were copied.
- Process the next set of batches

One drawback with this approach was that the inter GPU communication overhead outweighed the almost 2X gain time in the actual training of the networks (the time required to the forward-backward pass). During our other experiments, in order to save time in loading of the data, we implemented a multi-threaded approach to load our mini-batch.

Another important thing to note is that we were not able to process even a single input example of dimensions $15 \times M \times N$ as our images were of quite high resolution and during a forward pass of the network since every individual module in Torch caches its local output, the GPU's memory didn't turn out to be sufficient, so we broke each image into patches of 64×64 and after that we were able to keep the minibatch size to be 40 without overloading the GPU's memory. Due to computing power issues we were not able to test our models that efficiently. The code will shortly be made available at my github repository.

5 Results

5.1 LDR2HDR Results

In the results we present the graph of the training error vs no. of epochs for the best three models(Fig 3.). The test error for the best model is 0.09345. Visual results are shown below. It is clear that further experiments with validating the hyperparameters are required to find the optimal architecture for the task. The network is clearly able to generate plausible results for some of the colors but not all(Fig. 3 and 4). Also it is clear that the network is able to generate outputs without under/over saturation of regions that have high and low radiance values in the same image which hence proves that the dynamic range of the output is quite high.

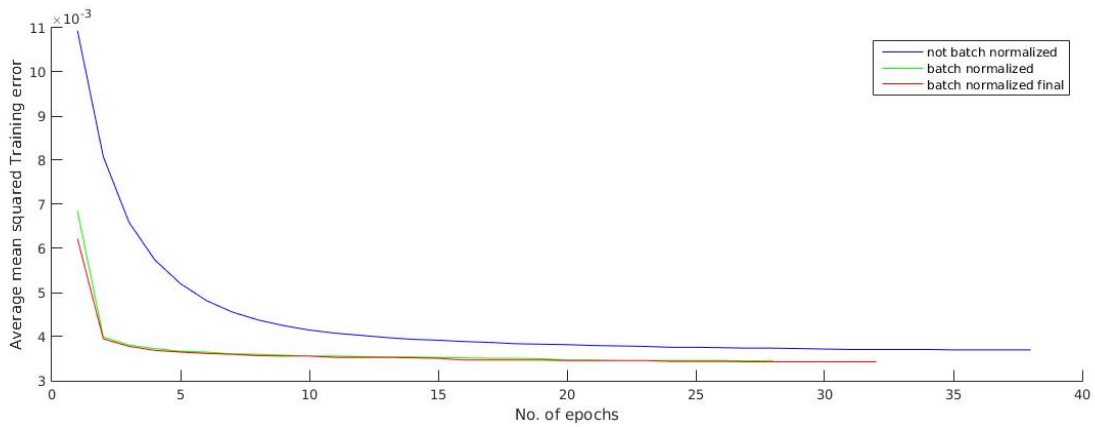


Figure 3: The training error plot for the three best architectures, blue and green share the same architecture, whereas the final one, red, is one more no. of activations in the first layer. The architecture of the final network has been illustrated in Fig 1.)

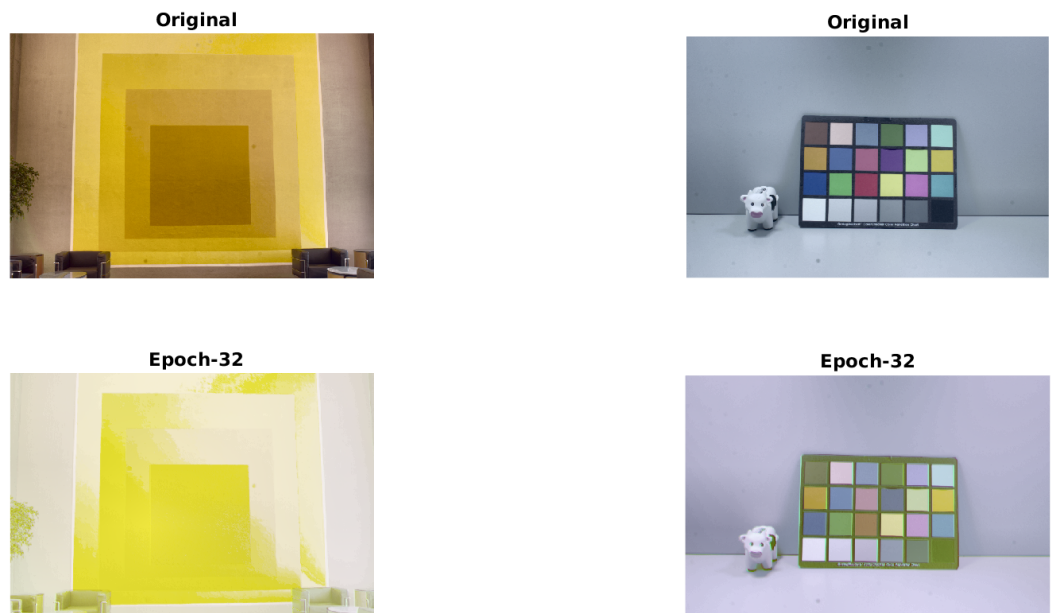


Figure 4: Some examples from test set where the LDR2HDR network gives plausible results.

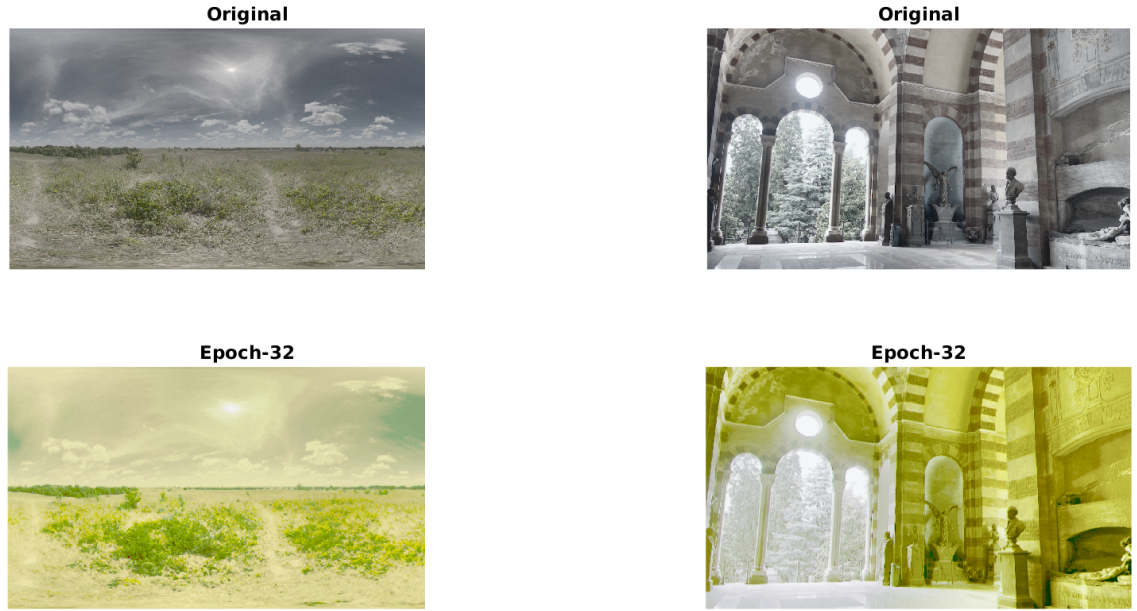


Figure 5: Some examples from test set where the LDR2HDR network does not give plausible results.

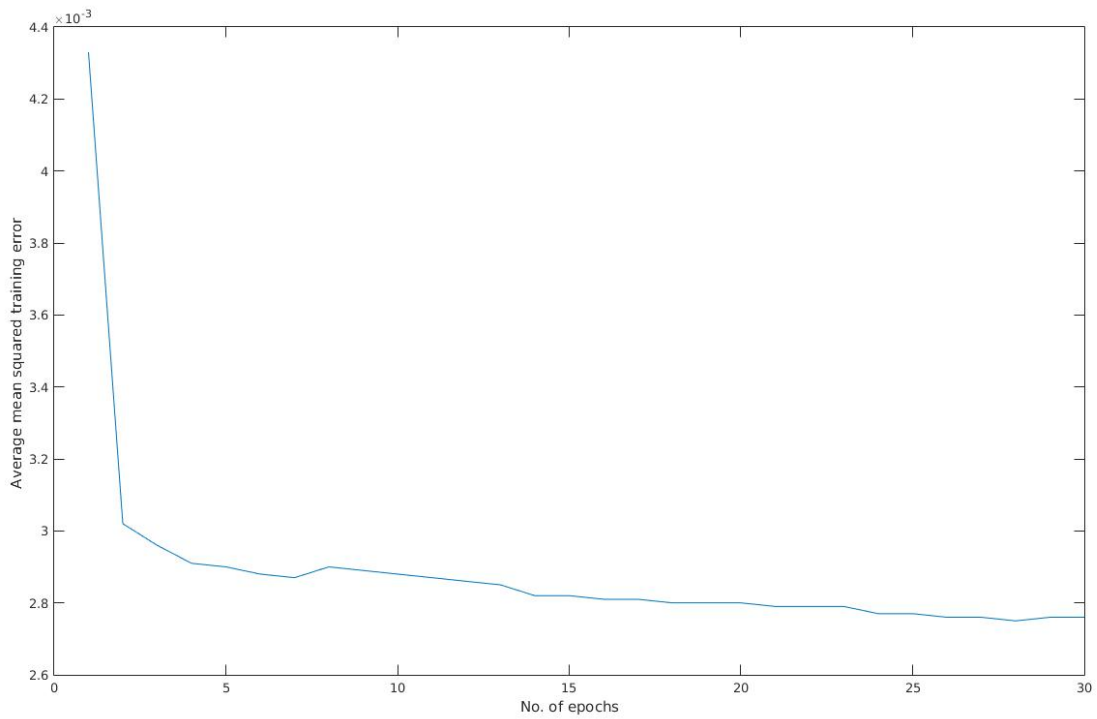


Figure 6: The training error plot for the final tone mapping network whose architecture has been illustrated in Fig 2.

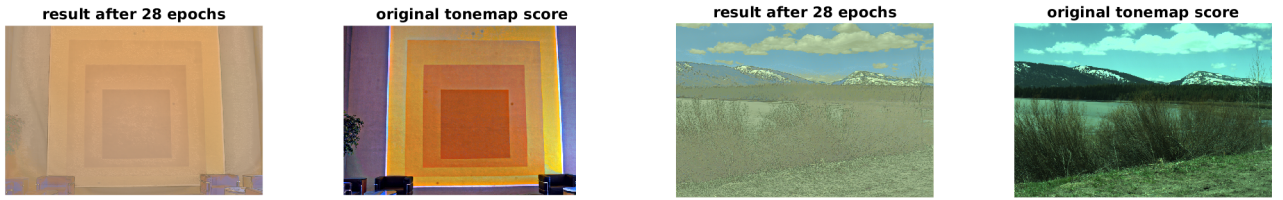


Figure 7: Some examples from test set where the tone mapping network gives plausible results.



Figure 8: Some examples from test set where the tone mapping network does not give plausible results.

5.2 HDR2Tonemap Results

In the results we present the outputs of the final validated architecture for the cases where the network performs good as well as bad (Fig 7. and 8.) . The final test error for that model after 28 epochs of training is 0.002764 (Fig 6.). We also present the plot of training error vs no. of epochs for that model.

6 References

- 1.) Debevec, Paul E., and Jitendra Malik. "Recovering high dynamic range radiance maps from photographs." ACM SIGGRAPH 2008 classes. ACM, 2008.
- 2.) Mitsunaga, Tomoo, and Shree K. Nayar. "Radiometric self calibration." Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.. Vol. 1. IEEE, 1999.
- 3.) Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." Advances in neural information processing systems. 2012.
- 4.) Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." Advances in neural information processing systems. 2012.
- 5.) Wang, Zhou, et al. "Image quality assessment: from error visibility to structural similarity." IEEE transactions on image processing 13.4 (2004): 600-612.