Log RGB Images Provide Invariance to Intensity and Color Balance Variation for Convolutional Networks

Bruce A. Maxwell
b.maxwell@northeastern.edu
Sumegha Singhania
singhania.s@northeastern.edu
Heather Fryling
fryling.h@northeastern.edu
Haonan Sun
ha.sun@northeastern.edu

Northeastern University
Boston, MA
United States

Abstract

The interaction of light and matter follows physical rules that have been well-modeled in the vision community. These rules should be available to deep networks when learning vision tasks. However, typical signal processing pipelines, conversion to sRGB, and JPEG compression break the rules and make them unavailable for learning. This, in turn, makes color and intensity unreliable as features and more difficult to use. Using linear or log RGB images that preserve the rules of the physics of reflection should make certain visual tasks simpler to learn and increase robustness to certain types of visual variation.

We demonstrate that using linear RGB or log RGB improves the performance of a deep network on an image classification task when the same network architecture is trained on the same images but in different formats. Furthermore, the linear and log RGB networks are more robust to intensity and color balance variation. In particular, the network trained on log RGB inputs shows invariance to intensity and color balance variation when that variation is not included in the training set, while the network trained on the same images in JPEG format shows severe reductions in performance. We further explore why this difference exists by visualizing low-level features in log RGB, linear RGB, and JPEG data and show that log space preserves certain types of features across intensity and color balance variation.

1 Introduction

Reflection and the capture of images using sensors follow rules of physics that have been extensively studied in computer vision and related fields. Fundamental work focused on modeling the appearance of surfaces and analyzing the characteristics of reflection [24], [22], [25], [6], [17], [3], [32]. Subsequent work examined how to use the constraints and models provided by physics to segment images, recognize objects, detect highlights, calculate illumination, or identify shadows [16], [36], [12], [20], [42], [37], [49], [35].

The move to deep networks, and large convolution stacks to learn features, presumably removes the need to derive features from physical models, as the networks are designed to learn what features are useful for their specific tasks[27][46][21][47]. However, this assumes that the data being used actually exhibits the rules of physics and contains the structures and features identified in the foundational research.

Unfortunately, typical imaging pipelines are designed for human viewing. They modify or remove the structures and patterns that exist in the original images because they do not maintain the linearity of the data. In particular, operations such as conversion to sRGB with gamma compression, saturation enhancement, brightening, sharpening, and JPEG compression, break the rules of physics, often in a non-reversible manner. For example, gamma compression, brightening, and saturation enhancement tend to generate pixels that are pushed to the maximum or minimum possible value, in which case the actual measurement is lost.

Most data sets used for deep learning in computer vision are provided only in processed form, and in almost all cases the linear data was never available as the images were collected from the web. Some of the most commonly used data sets fit this category, including: ImageNet [9], COCO [29], Pascal VOC [11], Faces in the Wild [24], and Intrinsic Images in the Wild [2]. Even data sets for tasks like highlight detection, that traditionally have been solved using physics-based methods, default to scraping web images to create data sets [15].

A few data sets are available with either RAW or linear data, mostly in the field of color constancy where the processing normally occurs prior to conversion to sRGB and JPEG compression [19], [7], [10]. However, these data sets are not collected or annotated for tasks such as detection, classification, or recognition, so they have not contributed to mainstream vision tasks. It is worth noting that, because of the availability of linear data, some color constancy researchers have successfully used physics-based analysis in the design of the inputs to a deep network as a way of improving performance [45].

The PascalRAW and LOD data sets are two exceptions that support object detection or instance segmentation [38][23][6]. However, for our purpose of running basic experiments as to whether log RGB is a better alternative input these data sets are not well-suited due to the unbalanced categories in PascalRAW, or the special illumination situation in LOD.

While being able to process JPEG images from the web is important in the near term, this is not a reason to focus on JPEG images in the future. It is not difficult to take images in RAW format, and both Android and Apple phones are capable of capturing and processing RAW images [48][1]. Most images captured and processed by deep networks will never leave the devices on which they are taken, and most of the processed images will never be stored or seen by humans. Applications like autonomous driving, robotics, or recognition and detection tasks on cell phones all capture huge amounts of data and process it on the device. These processing pipelines have access to the linear data from the sensor, or can get access to that data with minimal changes. Therefore, if linear or log RGB data provides a benefit in terms of accuracy, training time, computation time, or robustness to illumination, then using data that preserves the physics of the world has huge potential benefits.

In this work we explore the use of linear or log RGB data as inputs to deep networks for a mainstream vision task. We test the hypothesis that if the data preserves the physics of reflection, then deep networks will be able to learn physics-based features that improve their performance and robustness. The innovations of our work include: (1) capturing and processing a RAW data set for an image classification task, (2) evaluating the performance of networks trained on JPEG sRGB, linear RGB, and log RGB data, (3) exploring the robustness of networks trained on different data types to intensity and color balance variation, and (4) providing guidance on why using log RGB shows improved performance and robustness.

2 Related Work

The structure of body and surface reflection under a single illuminant, the dichromatic reflection model, was initially proposed and demonstrated by Shafer [24] and Klinker, Shafer, and Kanade [25]. The dichromatic reflection model has been used to derive numerous color spaces and features that are invariant to illumination intensity or highlights [20][49].

Both Marchant and Onyango [32], and Finlayson *et al*. [13][14] showed there was structure in taking the log of chromaticity (R/G, B/G) such that it was possible to create a one-dimensional albedo estimate that was invariant to Planckian illuminants. As noted above, this analysis has been used to design deep networks for color constancy [45].

Maxwell, Friedhoff, and Smith [34] expanded the dichromatic reflection model to incorporate an ambient illuminant, proposing the bi-illuminant dichromatic reflection [BIDR] model. They further demonstrated the structure of real-world material appearance in linear RGB and log RGB. In particular, they showed that body reflection in log RGB demonstrates a regular structure across different materials that are under the same ambient/direct illumination pair. Maxwell *et al*. [35] further demonstrated that log RGB space could be used to make shadow-free versions of road images and simplify the task of road feature detection.

Log RGB has been used in other applications to enable illumination invariance. Wang *et al*. used log RGB HOG features for illumination invariance in person re-identification [50]. Both [39] and [31] used log of chromaticity for illumination invariant skin lesion identification. Liu *et al*. made use of the BIDR model and log RGB for intrinsic image decomposition [30], and Put *et al*. [40] computed material priors using log RGB histograms.

Physics-based features have also been used for object recognition. Nayar and Bolle [36] showed that Reflectance Ratios could be used for object recognition purposes and were invariant to illumination conditions. The relevance to modern deep networks is that subtraction of nearby pixel values in log space is computing ratios rather than differences.

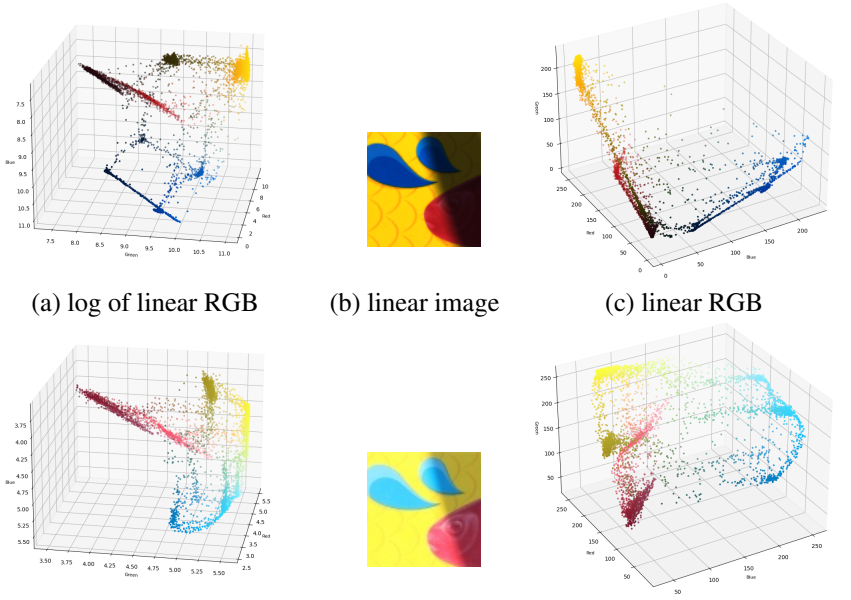
The most relevant recent work is the development of equivariant networks [8], which are a modification to convolutional neural networks that enable them to be invariant to offsets in the input data. For greyscale images, these offsets may be due to variations in brightness. For color images, the authors note that converting the data to log RGB turns multiplicative constants (color balance coefficients) to additive offsets. They show that using equivariant networks and log RGB enables the networks to maintain performance despite synthetically varying the illuminant on both CIFAR [26] and ImageNet [9] for object recognition and on the NUS data set for color constancy [7]. In our work, we demonstrate that invariance to image intensity and color balance **requires nothing more than training on log RGB data**. A standard CNN trained on log data exhibits invariance to intensity and color balance changes in new data without any modifications to the architecture.

Most other prior work on the impact of image quality on network performance focuses on noise, blur, and compression. Borel *et al*. examine the impact of blur, noise, resolution, and compression [4], and [18] examine the impact of compression on network performance.

3 Theory and Methods

As part of this work, we hope to explain why log RGB may be a better input for deep networks for computer vision tasks.

A common model for body reflection in images is the multiplicative model $I = LR$, where I is the captured image value, L is the direct illuminant, and R is the body reflection. The



(a) log of linear RGB (b) linear image (c) linear RGB

(d) log of JPEG sRGB (e) sRGB image (f) JPEG sRGB

Figure 1: 3D histograms showing the structure of the image in (a) log of linear RGB space, (b) linear image, (c) linear RGB space, (d) camera JPEG RGB values in log space, (e) camera JPEG image, and (f) camera JPEG RGB values in linear space.

BIDR model adds an ambient illumination term A, and a direct illuminant modifier γ that represents both geometric shading and shadows that modify the strength of L, giving the appearance model in (1).

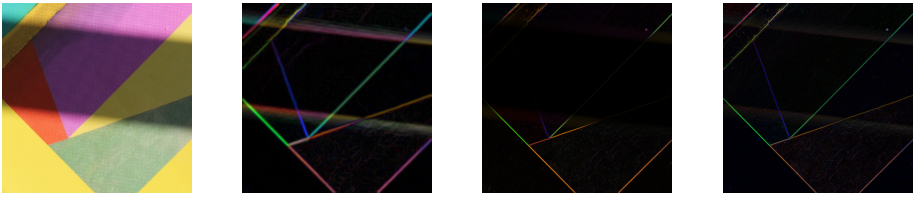
$$I = AR + \gamma LR = (A + \gamma L)R \tag{1}$$

Taking the log of the refactored equation gives two terms in log space, as in (2).

$$\log I = \log(R) + \log(A + \gamma L) \tag{2}$$

The first term is a constant for a single material. The second term varies according to the strength of the direct illuminant. The result is an approximate line segment, or cylinder in log space that represents the range of body reflection values for a single material. Because the second term contains only illumination terms, the orientation and length of the cylinder is the same for all materials under the same ambient/direct illumination pair: the cylinders representing each unique material are all translated versions of one another.

Conversion to sRGB, contrast enhancement, and JPEG compression all conspire to eliminate the linearity of the data and break the structure of material appearance in log space. Figure 1(b) and 1(e) show crops of an image with multiple materials under yellow sunlight in the lit areas and blue skylight in the shadows. Figure 1(b) is a linear image, and figure 1(e) is the camera JPEG version (the one people see, store, and share). Figures 1(a) and 1(d) show the log RGB structure as a 3D histogram, and figures 1(c) and 1(f) show the linear RGB structure as a 3D histogram. Note the clear linear structure present in the histogram for the linear data and the more convoluted structure in the histograms of the JPEG sRGB data.



(a) Original JPEG (b) JPEG Gradient (c) Linear Gradient (d) Log Gradient

Figure 2: (a) Original image shown as JPEG. Gradient magnitudes and colors for (b) sRGB JPEG, (c) linear RGB, and (d) log RGB image.

While surface reflection, blend pixels, and interreflection all add complexity to surface appearance, the log of linear and linear RGB histograms in figures 1(a) and (c) have more consistent structure to them than the equivalent JPEG sRGB histograms in figures 1(d) and (f). Our hypothesis is that if the structure exists, a deep network can learn to take advantage of it.

In addition to the overall structure of images in log RGB, computing ratios of nearby pixels has been shown to be a useful feature in object recognition [36]. Standard CNNs can't compute ratios with linear data because convolution is adding or subtracting scaled input values. However, because subtraction of log data is calculating ratios, a standard CNN can learn to compute them from log RGB inputs. Ratios of pixels factor out the illumination signal from body reflection as long as both pixels are under the same illumination condition, which is usually the case for nearby pixels.

$$\text{ratio} = \frac{LR_1}{LR_2} = \frac{R_1}{R_2} \tag{3}$$

$$\log \frac{R_1}{R_2} = \log(R_1) - \log(R_2) \tag{4}$$

To support the hypothesis that ratios can be useful features, Figure 2 shows a visualization of the color and intensity of gradient magnitudes calculated in JPEG sRGB, linear RGB, and log RGB using 3x3 Sobel operators. Note that in log RGB, the gradient color and magnitude on material boundaries is constant across illumination conditions, while those features vary for the other two data types, making the local log data features more consistent.

4 Data and Data Preparation

Given the lack of data sets based on RAW imagery we collected a new data set for an image classification task. We captured 561 images that contain a Swedish Fish® candy box and 557 images taken in similar locations without the box. We use 100 images for the test set, evenly split. The images were captured with two devices–a Canon EOS Rebel T7 and an iPhone 13 Pro using the Halide app–and the camera and phone (via Mac Photos) both produced RAW and proprietary JPEG versions of the images. The RAW images were provided as either CR2 (Canon) or DNG (iPhone) format files. The images were captured in a variety of lighting situations and environments, including some with cast shadows on the box.

The RAW data was read and processed with the rawPy library [41] using the default de-Bayering algorithm, no gamma correction, auto-brightness adjustment on, percent saturated pixels at 0.001%, and using the camera white balance. The linear data was resized using the OpenCV `resize` function with the INTERP_AREA flag so the minimum spatial dimension was 64 pixels and then saved as 16-bit TIFF files. The log of linear data was generated from the resized linear data and saved as 32-bit EXR files.

To guarantee the linearity of the data, we captured an image of a MacBeth® chart using both devices and fit the green channel values from the grey sequence to the chart luminosities (L). The least squares line fit was $R^2 = 0.992$ for the iPhone and $R^2 = 0.994$ for the Canon, confirming the data is linear.

From the original data, we created three variations of the original test set: (A) random intensity variation, (B) random color balance, and (C) both random intensity and color balance. We applied the color balance as a diagonal matrix on linear RGB. We first calculated the minimum and maximum possible multiplier coefficients for each of the three color channels to avoid saturating the data to either 0 or the max value. We then picked one coefficient per channel, uniformly distributed between the min and max value for that channel. To generate random intensity variation, we found the max of the min coefficients and the min of the max coefficients across color channels and picked a single random multiplier uniformly distributed between the two values. When applying both modifications, we applied a random intensity variation, then recalculated the min/max multipliers and picked a random color balance.

When creating the JPEG images for the test set and augmented training set, we applied the color balance or intensity variation to the linear image, converted the data to sRGB, and saved the image in JPEG format using the defaults for the OpenCV save function.

5 Experiments and Results

5.1 Object Detection Experiment

We used a small CNN structure for the detection task, similar to Lecun *et al.* [28]. The network, built in pyTorch, has three convolution layers with 5x5 filters, stride 1, valid convolution, with 16, 32, and 32 channels, respectively. Each convolution layer is followed by a 2x2 max pooling layer and ReLU activation. The final pooling layer is 4x4 spatially with 32 channels and is fully connected to a 64 node linear layer, followed by the output layer with two nodes. A dropout layer with p = 0.7 sits after the final pooling layer.

We intentionally used a small CNN to guarantee that the data set was large enough to train the network from scratch without overfitting. Using a pre-trained standard backbone architecture–e.g. ResNet18–would have introduced bias into the procedure. Training a standard backbone architecture from scratch on just 1000 images would likely have resulted in overfitting, making comparative results suspect, because the network would not have needed to learn any underlying compact rules.

The network was trained with negative log likelihood as the loss and Adam as the optimizer [51]. The learning rate for the networks trained on the JPEG and linear data was 0.001 using an eps of 1e-8. The learning rate for the log network was 0.0003 with an eps of 0.1. The log-trained network would not consistently train with the JPEG/linear meta-parameters. For all networks we experimented with the meta-parameters to optimize performance on the validation set.

(1) Unmodified Train Set	JPEG	Linear RGB	Log RGB
Original Test Set	89.0% / 0.292	90% / 0.455	**91%** / 0.272
Random Color Balance	62% / 0.815	75% / 1.163	**89%** / 0.318
Random Intensity	73% / 0.669	82% / 0.748	**94%** / 0.247
Both	69% / 0.750	74% / 1.400	**89%** / 0.292
Validation	88.7% / 0.312	**93.6%** / 0.263	87.7% / 0.321
(2) Fixed Modified Train Set	JPEG	Linear RGB	Log RGB
Original Test Set	65% / 0.683	87% / 0.514	**90%** / 0.285
Random Color Balance	78% / 0.455	85% / 0.782	**92%** / 0.216
Random Intensity	71% / 0.681	92% / 0.392	**93%** / 0.192
Both	78% / 0.444	88% / 0.684	**93%** / 0.193
Validation	89.0% / 0.283	**95.6%** / 0.156	95.0% / 0.179
(3) Dynamic Train Set	JPEG	Linear RGB	Log RGB
Original Test Set	82% /0.527	87% / 0.320	**94%** / 0.197
Random Color Balance	56% / 1.090	84% / 0.339	**92%** / 0.227
Random Intensity	55% / 1.119	85% / 0.357	**92%** / 0.210
Both	55% / 1.122	85% / 0.324	**92%** / 0.213
Validation	90% / 0.361	**91.7%** / 0.313	87.3% / 0.344

Table 1: Accuracy / Loss for the JPEG, linear RGB, and Log RGB networks using three variations of the training data and four variations of the test set.

Given our focus on data integrity, we minimized pre-processing of the data prior to applying it to the network. The JPEG sRGB and linear data is normalized to the range [0, 1] by dividing by the max value for the data: 255 for JPEG and 65535 for the linear data. The log data is not normalized or shifted and is in the range [0, 11.1]. The images are 64 pixels on their short side, and we take a random square 64x64 crop from the image.

We ran three experiments, each evaluating four test set variations on networks trained on one of three image types: JPEG sRGB, linear RGB, and log RGB. In the first experiment, one network was trained on each type with no color balance or intensity augmentation. Each of the networks was then evaluated on each version of the test set.

In the second experiment, the networks were trained on the same training set, but with modified versions of the training set images with random color balance and intensity variation. Specifically, the training set contained 1/3 with random intensity variation, 1/3 with random color balance, and 1/3 with both types of variation. In order to create a fair experiment, the training set was augmented and then fixed so that all three networks trained on the same set of augmented data.

In the third experiment, we dynamically modified the inputs, with equal probabilities for no modification, intensity variation, color balance variation, and both color balance and intensity variation. Table 1 shows the results. For all experiments we trained the network at least twice and report results from the version with the best validation accuracy.

5.2 Object Detection Results

In all three experiments, the log network demonstrated consistent performance across both the original and modified test sets, outperforming the other networks despite not having the highest performance on the validation set. **The network trained on log data demonstrates**

Metric	JPEG-Linear	JPEG-Log	Linear-Log
↓ MSE / Stdev	4863 / 4057	5606 / 3438	**3374** / 2686
↑ SSIM / Stdev	0.42 / 0.21	0.34 / 0.17	**0.53** / 0.18

Table 2: Differences between grad-CAM heat maps for the three networks. For the MSE values, smaller is more similar. For SSIM, larger is more similar.

invariance to color balance and intensity even when trained only on the original data.

The linear network showed a drop in performance on the modified test sets when trained only on the original data. It demonstrated more sensitivity to color balance than to intensity variation in all three experiments, but it was able to perform more consistently when trained on the modified data. The linear network had the highest validation accuracy for all three experiments, but worse generalization to the unmodified test set than the log network, and the losses indicate the network was exhibiting more uncertainty despite the good accuracy.

The JPEG network showed similar performance on the validation set and the unmodified data in experiment 1, but did not generalize as well in experiments 2 or 3. In all three experiments the JPEG network performance decreased for the test sets with color balance or intensity variation applied. We were expecting the JPG network to be able to learn the task more effectively with the dynamic data set, and it did improve its performance on the unmodified test data, but it was unable to learn how to generalize to the modified test sets.

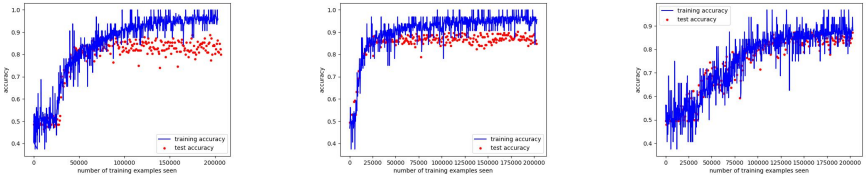
5.3 Analyzing the networks

One question we wanted to explore is whether the linear and log networks were activating on the same spatial features as the JPEG network. We implemented the grad-CAM [13] method of building activation maps, computing them for the positive detection class after the final convolution and pooling layers. We then computed both mean-squared error [MSE] and structural similarity [SSIM] metrics for the heat maps between all pairs of networks. Table 2 shows the MSE and SSIM comparisons.

Table 2 shows that the linear and log trained networks are learning activation patterns more similar to one another than to the JPEG trained network. A pairwise ranking using SSIM shows that the log map is more similar to the linear map in 83 of 100 cases, and the linear map is more similar to the log map in 67 of 100. The JPEG map is more similar to the linear map in 69 of 100. These results support the hypothesis that the log and linear networks are learning different spatial features than the JPEG network, and the learned features for the linear and log networks are more similar to one another.

5.4 Training Log Space Networks

Using log RGB as input to a convolutional network changes what the network is computing in the first layer. Applying standard convolution on log data means the filters are computing ratios (subtraction in log space) or products (addition in log space). Therefore, it is important to avoid pre-processing steps that change differences. In particular, we had to avoid normalizing the log inputs, such as dividing by the max input value. The log network will not train if the data is normalized by a scalar. Likewise, we found that subtracting the mean of the log data to center it around zero in log space–which is identical to dividing by the mean in linear space–also caused the log network to underperform.



(a) JPEG training (b) Linear training (c) Log training

Figure 3: Training curves for the JPEG, linear, and log networks showing accuracy for the train and test sets.

Possibly because of the increased range of the input data, the log network training was more consistent if the learning rate was smaller. Training was also more consistent if we increased the value of the eps coefficient of the Adam scheduler, which is used to avoid division by zero errors and modifies the learning rate [5].

A final note about training log space networks is that the training curves tend to be different than the curves for JPEG or linear networks. In particular, the network does not display typical overfitting behavior, as shown in figure 3. Both the JPEG and linear networks display typical overfitting where the training and test data diverge. This suggests that the log space network may be learning more robust general features than the other two networks.

6 Discussion

Our results and analysis support two hypotheses. (1) *Using linear or log data provides better results on an image classification task when using the same network and training set.* (2) *Using log data provides invariance to color balance and intensity variation with no additional training.* These hypotheses have potentially significant broader impacts and suggest that log RGB inputs should be evaluated on other computer vision tasks. The key missing element is the need for data sets large enough to support networks of more typical size (e.g. ResNet18).

Some questions that may arise from our experiments include the following.

(1) Why not try a standard pre-trained CNN architecture (e.g. ResNet18) on this task? It's important to train from scratch, as the log space input will likely compute different types of features, and the data set is not large enough to train a large network. We wanted to avoid transfer learning with a network trained on standard data, as it would bias the network features if not pre-trained on linear or log data.

(2) Is the data set is too small to be significant? We have >500 images of a unique object, and >1000 images total. That's 50% of the number of images used for a whole category (e.g. airplane) in ImageNet. The network we used should have enough data to learn how to detect the object. We used 64x64 inputs in order to ensure the object was big enough for the network to have multiple pixels on each color of the object in every image.

(3) How does this work relate to the equivariant networks [8], which used large standard data sets? Given our results, it's not clear if the equivariant networks are able to achieve color balance invariance from the equivariant modification to CNNs or by executing the inverse sRGB and log conversion. They would need to do two additional experiments in order to separate the effects of the two modifications: (1) use the log space conversion and standard

networks, and (2) use the equivariant networks but no log space conversion. In addition, the JPEG data they use is only 8-bits and likely corrupted by more transformations than sRGB, meaning the conversion to log space is most likely producing only approximate log data.

Future work on this topic should explore larger data sets carefully captured to preserve the linearity of the data, other computer vision tasks, and larger networks (enabled by larger data sets). It is also important to explore whether objects for which color is not a defining feature of the category–such as mugs or chairs–also receive a boost in performance from using log RGB. Reflectance ratios, for example, can be both a characteristic of an object–such as a boundary between two colors on the same object–and an indicator that two adjacent regions are not part of the same surface [33]. Giving deep networks access to the structure and information present in linear and log RGB images may provide across-the-board improvements in many vision tasks with smaller and simpler networks.

7 Summary

Log of linear RGB contains consistent structure that is not present in data converted to JPEG sRGB, especially with other potential image processing applied. Using log data also makes it possible for a standard convolution layer to compute pixel ratios, which have been shown to be useful as illumination invariant features of objects. There is still much more exploration to be done, but this work suggests that log RGB space may have important benefits as an input to deep networks for computer vision tasks.

References

[1] Apple. About apple proraw, 2022. URL https://support.apple.com/en-us/HT211965.

[2] Sean Bell, Kavita Bala, and Noah Snavely. Intrinsic images in the wild. *ACM Trans. Graphics (SIGGRAPH)*, 33(4), 2014.

[3] J. Berens and G. D. Finlayson. Log-opponent chromaticity coding of colour space. In *15th Int'l Conf. on Pattern Recognition*, Barcelona, Spain, 2000.

[4] Christoph Borel-Donohue and S. Susan Young. Image quality and super resolution effects on object recognition using deep neural networks. In Tien Pham, editor, *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications*, volume 11006, page 110061M. International Society for Optics and Photonics, SPIE, 2019. doi: 10.1117/12.2518524.

[5] Carlos F. Borges. A trichromatic approximation method for surface illumination. *J. of Optical Society of America*, 8(8):1319–1323, August 1991.

[6] Linwei Chen, Ying Fu, Kaixuan Wei, Dezhi Zheng, and Felix Heide. Instance segmentation in the dark. *International Journal of Computer Vision*, 131, 2023.

[7] D. Cheng, D.K. Prasad, and M.S. Brown. Illuminant estimation for color constancy: why spatial-domain methods work and the role of the color distribution. *Journal of the Optical Society of America A*, 31(5):1049–1058, 2014.

[8] Marco Cotogni and Claudio Cusano. Offset equivariant networks and their applications. *Neurocomputing*, 502:110–119, 2022.

[9] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE conference on computer vision and pattern recognition*. IEEE, 2009.

[10] Egor Ershov, Alexey Savchik, Illya Semenkov, Nikola Banić, Alexander Belokopytov, Daria Senshina, Karlo Koščević, Marko Subašić, and Sven Lončarić. The cube++ illumination estimation dataset. *IEEE Access*, 8, 2020.

[11] Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman. The Pascal Visual Object Classes (VOC) Challenge. *International Journal of Computer Vision*, 88(2):303–338, June 2010. ISSN 1573-1405. doi: 10.1007/s11263-009-0275-4. URL https://doi.org/10.1007/s11263-009-0275-4.

[12] G. D. Finlayson and S. D. Hordley. Color constancy at a pixel. *J. of Optical Society of America A*, 18(2):253–264, February 2001.

[13] G. D. Finlayson, S. D. Hordley, and M. S. Drew. Removing shadows from images. In *Proc. of European Conf. on Computer Vision*, pages 823–836, London, UK, 2002. Springer-Verlag. ISBN 3-540-43748-7.

[14] G. D. Finlayson, M. S. Drew, and L. Cheng. Intrinsic images by entropy minimization. In T. Pajdla and J. Matas, editors, *Proc. of European Conf. on Computer Vision*, LNCS 3023, pages 582–595, 2004.

[15] Gang Fu, Qing Zhang, Qifeng Lin, Lei Zhu, and Chunxia Xiao. Learning to detect specular highlights from real-world images. In *Proceedings of the 28th ACM International Conference on Multimedia*, MM '20, pages 1873–1881. Association for Computing Machinery, 2020.

[16] G. D. Funka-Lea and R. Bajcsy. Combining color and geometry for the active, visual recognition of shadows. In *Proc. Fifth Int'l Conf. on Computer Vision*, Cambridge, 1995.

[17] B. V. Funt and M. S. Drew. Color space analysis of mutual illumination. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 15(12):1319–1326, 1993.

[18] Tomasz Gandor and Jakub Nalepa. First gradually, then suddenly: Understanding the impact of image compression on object detection using deep learning. *Sensors*, 22, 2022.

[19] P.V. Gehler, C. Rother, A. Blake, T. Minka, and T. Sharp. Bayesian color constancy revisited. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[20] Jan-Mark Geusebroek, Rein van den Boomgaard, Arnold W. M. Smeulders, and Hugo Geerts. Color invariance. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 23(12):1338–1350, December 2001.

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[22] Glenn E. Healey. Using color for geometry-insensitive segmentation. *J. of the Optical Society of America A*, 6(6):920–937, June 1989.

[23] Yang Hong, Kaixuan Wei, Linwei Chen, and Ying Fu. Crafting object detection in very low light. In *BMVC*, 2021.

[24] Gary B. Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49, University of Massachusetts, Amherst, October 2007.

[25] Gudrun J. Klinker, Steven A. Shafer, and Takeo Kanade. A physical approach to image understanding. *Int'l J. of Computer Vision*, 4(1):7–38, January 1990.

[26] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

[27] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *NeuroIPS*, 2012.

[28] Yan LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, November 1998.

[29] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft COCO: Common Objects in Context, February 2015. URL http://arxiv.org/abs/1405.0312. arXiv:1405.0312 [cs].

[30] Yuanliu Liu, Ang Li, Zejian Yuan, Badong Chen, and Nanning Zheng. Consistency-aware shading orders selective fusion for intrinsic image decomposition. *ArXiv*, abs/1810.09706, 2018.

[31] Ali Madooei, Mark S. Drew, Maryam Sadeghi, and M. Stella Atkins. Intrinsic melanin and hemoglobin colour components for skin lesion malignancy detection. *Med Image Comput Comput Assist Interv*, 15, 2012.

[32] John A. Marchant and Christine M. Onyango. Shadow-invariant classification for scenes illuminated by daylight. *J. of the Optical Society of America A*, 17(11), November 2000.

[33] B. A. Maxwell and S. A. Shafer. Physics-based segmentation of complex objects using multiple hypotheses of image formation. *Computer Vision and Image Understanding*, 65(2):269–295, February 1997.

[34] Bruce A Maxwell, Richard M Friedhoff, and Casey A Smith. A bi-illuminant dichromatic reflection model for understanding images. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[35] Bruce A. Maxwell, Casey A. Smith, Maan Qraitem, Ross Messing, Spencer Whitt, Nicolas Thien, and Richard M. Friedhoff. Real-time physics-based removal of shadows and shading from road surfaces. In *CVPR Workshop on Autonomous Driving*. IEEE / CVF, 2019.

[36] S. K. Nayar and R. M. Bolle. Reflectance based object recognition. *Int'l J. of Computer Vision*, 17(3):219–240, March 1996.

[37] Ido Omer and Michael Werman. Color lines: Image specific color representation. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition*, pages II: 946–953, June 2004.

[38] Alex Omid-Zohoor, David Ta, and Boris Murmann. Pascalraw: Raw image database for object detection, 2015. URL http://purl.stanford.edu/hq050zr7488.

[39] Luisa F. Polanía, Raja Bala, Ankur Purwar, Paul Matts, and Martin Maltz. Skin chromophore estimation from mobile selfie images using constrained independent component analysis. *Electronic Imaging*, 14, 2020.

[40] Jerome Put, Nick Michiels, and Philippe Bekaert. Material-specific chromaticity priors. In *British Machine Vision Conference*, 2016.

[41] Mike Reichert. rawpy, 2023. URL https://letmaik.github.io/rawpy/index.html.

[42] Imari Sato, Yoichi Sato, and Katsushi Ikeuchi. Illumination from shadows. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 25(3):290–300, March 2003.

[43] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[44] Steven A. Shafer. Using color to separate reflection components. *Color Research Applications*, 10:210–218, 1985.

[45] Wu Shi, Chen Change Loy, and Xiaoou Tang. Deep specialized network for illuminant estimation. In *European Conf. on Computer Vision*, 2016.

[46] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.

[47] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[48] New York Times. How to make your smartphone photos so much better, 2023. URL https://www.nytimes.com/2023/01/11/technology/personaltech/smartphone-cameras-tips.html.

[49] Joost van de Weijer and Cordelia Schmid. Coloring local feature extraction. In *European Conf. on Computer Vision*, volume 3952 of *LNCS*, pages 334–348. Springer, 2006.

[50] X. Wang, G. Doretto, T. Sebastian, J. Rittscher, and P. Tu. Shape and appearance context modeling. In *International Conference on Computer Vision*, 2007.

[51] Dokkyun Yi, Jaehyun Ahn, and Sangmin Ji. An Effective Optimization Method for Machine Learning Based on ADAM. *Applied Sciences*, 10(3):1073, January 2020. ISSN 2076-3417. doi: 10.3390/app10031073. URL https://www.mdpi.com/2076-3417/10/3/1073. Number: 3 Publisher: Multidisciplinary Digital Publishing Institute.