Let there be Color!: Joint End-to-end Learning of Global and Local Image Priors for Automatic Image Colorization with Simultaneous Classification

Satoshi Iizuka * Edgar Simo-Serra * Hiroshi Ishikawa

Waseda University

(a) Colorado National Park, 1941 *(b) Textile Mill, June 1937* *(c) Berry Field, June 1909* *(d) Hamilton, 1936*

Figure 1: *Colorization results of historical black and white photographs from the early 20th century. Our model can obtain realistic colorization results whether it is a picture of landscapes (a), man-made environments (b), human portraits (c), or close-ups (d). Photos taken by Ansel Adams (a) and Lewis Hines (b,c,d), and are taken from the US National Archives (Public Domain).*

Abstract

We present a novel technique to automatically colorize grayscale images that combines both global priors and local image features. Based on Convolutional Neural Networks, our deep network features a fusion layer that allows us to elegantly merge local information dependent on small image patches with global priors computed using the entire image. The entire framework, including the global and local priors as well as the colorization model, is trained in an end-to-end fashion. Furthermore, our architecture can process images of any resolution, unlike most existing approaches based on CNN. We leverage an existing large-scale scene classification database to train our model, exploiting the class labels of the dataset to more efficiently and discriminatively learn the global priors. We validate our approach with a user study and compare against the state of the art, where we show significant improvements. Furthermore, we demonstrate our method extensively on many different types of images, including black-and-white photography from over a hundred years ago, and show realistic colorizations.

Keywords: colorization, convolutional neural network**Concepts:** •**Computing methodologies** → **Image processing;** *Neural networks;*

*The authors assert equal contribution and joint first authorship.

SIGGRAPH '16 Technical Paper,, July 24-28, 2016, Anaheim, CA,
ISBN: 978-1-4503-4279-7/16/07
DOI: http://dx.doi.org/10.1145/2897824.2925974

1 Introduction

Traditional colorization requires significant user interaction whether in the form of placing numerous color scribbles, looking at related images, or performing segmentation. In this paper, we instead propose a fully automated data-driven approach for colorization of grayscale images. Our approach uses a combination of global image priors, which are extracted from the entire image, and local image features, which are computed from small image patches, to colorize an image automatically. Global priors provide information at an image level such as whether or not the image was taken indoors or outdoors, whether it is day or night, etc., while local features represent the local texture or object at a given location. By combining both features, we can leverage the semantic information to, for example, color the dusk sky or the human skin—all without requiring human interaction.

Our approach is based on Convolutional Neural Networks, which have a strong capacity for learning. We propose a novel architecture that can jointly extract global and local features from an image and then fuse them together to perform the final colorization. We will show that this approach greatly outperforms using local features only. Furthermore, we are able to exploit semantic class labels of existing datasets during training in order to learn more discriminative global features. However, the semantic class labels are not

needed when colorizing grayscale images. We use an existing large scale scene database to train our model to predict the chrominance of a grayscale image using the CIE L*a*b* colorspace. Our method requires neither pre-processing nor post-processing: everything is learnt in an end-to-end fashion.

Our model consists of four main components: a low-level features network, a mid-level features network, a global features network, and a colorization network. Conceptually, these networks function as follows: First, a common set of shared low-level features are extracted from the image. Using these features, a set of global image features and mid-level image features are computed. Then, the mid-level and the global features are both fused by our proposed "fusion layer" and used as the input to a colorization network that outputs the final chromimance map. Needless to say, this is not explicitly implemented as a sequential procedure; rather, it is realized as a single network. Note that no pre-processing nor post-processing is done: it is all computed in a single step. Additionally, as a side product of our approach, we can also perform classification of the scene. While the global features are computed using fixed-sized images, our novel approach for fusing the global and local features allows our model to be run on input images of arbitrary resolutions, unlike most Convolutional Neural Networks.

Due to the separation between the global and local features, it is possible to use global features computed on one image in combination with local features computed on another image, to change the style of the resulting colorization. For example, if the global features are computed on an image at dusk and combined with an image of a sunny beach, the resulting colorization will be of a beach at dusk. An image can also be made to seem as if it was taken during a different season. This highlights the flexibility of our model.

We train and evaluate our model on a large-scale scene database. For evaluation, in a user study we compare our method against a strong Convolutional Neural Network baseline. The colorization of our proposed approach is considered "natural" 92.6% of the time, while the baseline only achieves roughly 70%. We also provide a comparison against the state of the art. Finally, we demonstrate our model extensively on early-20th-century black-and-white photography and show convincing results. Some examples are shown in Fig. 1.

In summary, in this paper we present:

- A user-intervention-free approach to colorize grayscale images.
- A novel end-to-end network that jointly learns global and local features for an image.
- A learning approach that exploits classification labels to increase performance.
- A style transfer technique based on exploiting the global features.
- In-depth evaluation of our model with user study and many diverse examples, including hundred-year-old black-and-white photographs.

2 Related Work

One of the more traditional approaches to image colorization consists of propagating colored user-specific scribbles to the whole image. Levin *et al*. [2004] proposed an optimization-based framework for colorizing a grayscale image using this approach. This was done by solving a quadratic cost function derived from differences of intensities between a pixel and its neighboring pixels. This method was improved by Hunang *et al*. [2005] to prevent the color bleeding over object boundaries. Yatziv and Sapiro [2004] proposed a fast colorization technique using chrominance blending based on weighted geodesic distances. Luan *et al*. [2007] employed texture similarity for more effective color propagation. Texture classification has also been used for cartoon colorization [Qu et al. 2006]. Sýkora *et al*. [2009] proposed a flexible colorization tool for hand-drawn cartoons based on a graph-cut-based optimization framework that is easily applicable to various drawing styles. To enable long-range propagation for image recolorization or tonal editing, various affinity-based methods have also been proposed, such as global optimization with all-pair constraints [An and Pellacini 2008; Xu et al. 2009], Radial Basis Function interpolation [Li et al. 2010], manifold learning [Chen et al. 2012], and stochastic modeling of appearance similarities between user-specified pixels and other pixels [Xu et al. 2013]. However, these methods heavily depend on user input and require trial and error to obtain an acceptable result.

Unlike the scribble-based methods that utilize user-supplied colors, example-based colorization techniques exploit the colors of a reference image that are similar to the input image. For recoloring a color image, *color transfer* techniques [Reinhard et al. 2001; Tai et al. 2005; Pitié et al. 2007; Wu et al. 2013] are widely used. These compute color statistics in both input and reference images and then establish mapping functions that map the color distribution of a reference image to the input image. Inspired by the color transfer, Welsh *et al*. [2002] proposed a general technique to colorize grayscale images by matching the luminance and texture information between images. This technique was improved by using a supervised classification scheme that analyzed low-level features [Irony et al. 2005]. Charpiat *et al*. [2008] proposed a global optimization framework that deals with multi-modality to predict probability of possible colors at each pixel. Gupta *et al*. [2012] match superpixels between the input image and the reference image using feature matching and space voting to perform the colorization. However, these methods require the user to supply suitable reference images that are similar to the input image, which is a time-consuming task. In comparison, our model does not require any user annotation at all.

Instead of requiring the user to provide reference images, Liu *et al*. [2008] proposed an example-based colorization robust to illumination differences between input and reference images that are obtained directly from web search. Its applicability is, however, limited to famous landmarks where exact matches can be found. Chia *et al*. [2011] extend this to general objects and scenes where exact matches are in general not available by filtering the reference images so that only the most appropriate parts of the images are used. However, they still require the user to input the search query to find the reference images.

More recently, Cheng *et al*. [2015] proposed a fully automatic approach in which various features are extracted and the different patches of the image are colorized using a small neural network. Joint bilateral filtering is used to improve the results. Unlike our approach, they use very little training data which greatly limits the type of images it is applicable to. Furthermore, they require a high-performance segmentation model to provide a segmentation of the image. Because of this dependency on segmentation, result is very poor for images in which none of the segmentation classes appear. This limits the application of the approach to simple outdoor scenes. On the other hand, our approach does not rely on any hand-crafted or pre-trained model. We learn everything in an end-to-end fashion from a large dataset which allows our model to generalize to many types of images.

With the advent of back-propagation roughly three decades ago [Rumelhart et al. 1986], neural networks have been exploited for a diversity of tasks. Initially, research focused on small set of outputs, in particular the classification task; however, they are now successfully applied to many different tasks in which the out-

put is an image, such as optical flow [Fischer et al. 2015], super-resolution [Dong et al. 2016], contour detection [Shen et al. 2015], and semantic segmentation [Long et al. 2015]. These are based on convolutional neural networks [Fukushima 1988; LeCun et al. 1998] and can process images of any resolution. While most approaches tackle single tasks, networks that jointly handle two tasks such as the one we propose in this work have been used for depth estimation [Eigen and Fergus 2015], where depth, surface normals, and semantic labels are predicted jointly, and for learning feature embeddings [Bell and Bala 2015]. Multi-scale fusion has also been used [Eigen and Fergus 2015; Wang et al. 2015c], however, these approaches, in general, fuse images at different scales by scaling the features and concatenating them at a common resolution before further processing. In contrast, our fusion approach generates a global image feature vector which is fused with the local features. By using a single global feature vector for the entire image, it is possible to exploit the class labels for more performance. As far as we know, we are the first to fuse a global feature with local features in a single framework where everything can be learned end-to-end.

3 Joint Global and Local Model

Our approach is based on deep Convolutional Neural Networks [Krizhevsky et al. 2012] that have been proven able to learn complex mappings from large amounts of training data. Our network is formed by several subcomponents that form a Directed Acyclic Graph (DAG) and contain important discrepancies with widely-used standard models. In particular, our model:

- can process images of any resolution,
- incorporates global image priors for local predictions, and
- can directly transfer the style of an image into the colorization of another.

An overview of the model and its subcomponents can be seen in Fig. 2. It consists of four main components: a low-level features network, a mid-level features network, a global features network, and a colorization network. The components are all tightly coupled and trained in an end-to-end fashion. The output of our model is the chrominance of the image which is fused with the luminance to form the output image.

3.1 Deep Networks

Deep networks are neural networks that are formed by many layers. These networks serve to predict continuous values from a given input. They consist of layers that realize a fuction of the form:

$$\mathbf{y} = \tilde{\sigma}\left(\mathbf{b} + W\mathbf{x}\right) \ , \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{y} \in \mathbb{R}^m$ are the input and output of the layer, respectively, W is an m-by-n matrix of weights, $\mathbf{b} \in \mathbb{R}^m$ is a bias vector, and $\tilde{\sigma} : \mathbb{R}^m \to \mathbb{R}^m$ is a non-linear transfer function $\sigma : \mathbb{R} \to \mathbb{R}$ applied componentwise. Both the weights and the bias are learnt through back-propagation [Rumelhart et al. 1986], which consists of using the chain rule to propagate a loss to update the parameters. The loss consists in the error between the prediction of the network and the training data ground truth.

Convolutional Neural Networks are special cases in which weights are "shared" spatially across an image. This has the effect of reducing the number of parameters needed for a layer and gaining a certain robustness to translation in the image. Typically, a layer is organized as a 2D image with multiple channels so that the components of the vectors $\mathbf{x}, \mathbf{y}$ are pixels indexed by a cartesian coordinate and a channel number. The matrix W above is defined as a convolution of the image with a bank of filters, which, in the neural network view, means that the weights are "shared". If one 2D layer is a C-channel $h \times w$ image and the next layer is a C'-channel $h' \times w'$ image, then $n = h \cdot w \cdot C$ and $m = h' \cdot w' \cdot C'$. In this case, the function (1) can be written for each pixel as:

$$\mathbf{y}_{u,v} = \bar{\sigma}\left(\mathbf{b} + \sum_{i=-k'_{\mathrm{h}}}^{k'_{\mathrm{h}}} \sum_{j=-k'_{\mathrm{w}}}^{k'_{\mathrm{w}}} W_{k'_{\mathrm{h}}+i,k'_{\mathrm{w}}+j}\ \mathbf{x}_{u+i,v+j}\right),$$
$$k'_{\mathrm{h}} = \frac{k_{\mathrm{h}}-1}{2}, \qquad k'_{\mathrm{w}} = \frac{k_{\mathrm{w}}-1}{2}, \tag{2}$$

where k_{w} and k_{h} are the kernel width and height (odd numbers), respectively, $\mathbf{x}_{u,v} \in \mathbb{R}^C$ and $\mathbf{y}_{u,v} \in \mathbb{R}^{C'}$ are the pixel component of the input and the output of the layer, $\bar{\sigma}(\cdot)$ is a componentwise non-linear transfer function, $W_{s,t}$ are C'-by-C matrices of the kernel, and $\mathbf{b} \in \mathbb{R}^{C'}$ is the layer bias vector. Note that the fact that the weights $W_{s,t}$ do not depend on the spatial coordinates u, v corresponds to the fact that the weights are shared between the "neurons" that can be parallelly translated to each other.

The most common non-linearity for neural networks is the Rectified Linear Unit (ReLU):

$$\sigma_{\mathrm{ReLU}}(x) = \max\left(0, x\right) \ . \tag{3}$$

We will also employ the Sigmoid transfer function for the color output layer which is defined as:

$$\sigma_{\mathrm{Sigmoid}}(x) = \frac{1}{1+e^{-x}} \ . \tag{4}$$

A model is built by concatenating many of these layers consecutively. Most classification networks [Krizhevsky et al. 2012; Simonyan and Zisserman 2015] initially use convolutional layers. At the end they use regular fully-connected layers, which forces the output to be a vector of a specific size, however, this also fixes the input size of the network. Thus, these networks are only able to process images of a fixed size. As we will next explain, our method does not share this limitation. (However, we do use a similar subnetwork with fully-connected layers in the global features network. See §3.2.2.)

3.2 Fusing Global and Local Features for Colorization

We use a novel approach to fuse both global and local features together. The global features act as an image prior on the local features to indicate what type of image the input is. For example, if the global features indicate that it is an indoor image, the local features will be biased to not attempt to add sky colors or grass colors to the image, but instead will add colors suitable for furnitures. We intertwine both a global image feature network, similar to those that compete in image classification tasks, with a fully convolutional neural network that colorizes the image. In order to improve the model efficiency, both networks use a number of common shared low-level features.

3.2.1 Shared Low-Level Features

A 6-layer Convolutional Neural Network obtains low-level features directly from the input image. The convolution filter bank the network represents are shared to feed both the global features network and the mid-level features network. This is similar to the weight sharing in Siamese networks [Bromley et al. 1994], however, in our approach only a subset of the full network is shared. Instead of using max-pooling layers to reduce the size of the feature maps, we use convolution layers with increased strides. This is also important for increasing the spatial support of each layer. A stride of 2

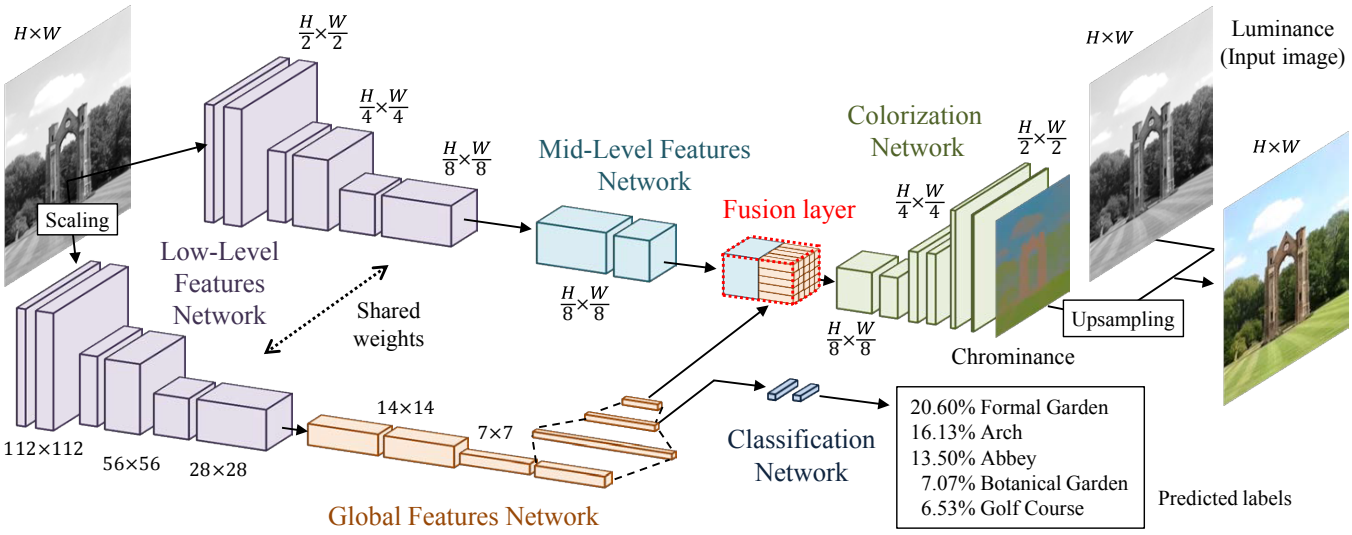


Figure 2: *Overview of our model for automatic colorization of grayscale images.*

Table 1: *Architectures of the different networks used in our model. Fully-Connected (FC) layers refer to the standard neural network layers from Eq.* (1)*, convolution layers use Eq.* (2)*. The output layer consists of a convolutional layer with a Sigmoid transfer layer instead of a ReLU transfer layer. Outputs refers to the number of output channels for the output of the layer. Upsample layers consist of using the nearest neighbour approach to increase the resolution of the output by a factor of* 2*.*

(a) Low-Level Features network

Type	Kernel	Stride	Outputs
conv.	3×3	2×2	64
conv.	3×3	1×1	128
conv.	3×3	2×2	128
conv.	3×3	1×1	256
conv.	3×3	2×2	256
conv.	3×3	1×1	512

(b) Global Features network

Type	Kernel	Stride	Outputs
conv.	3×3	2×2	512
conv.	3×3	1×1	512
conv.	3×3	2×2	512
conv.	3×3	1×1	512
FC	-	-	1024
FC	-	-	512
FC	-	-	256

(c) Mid-Level features network

Type	Kernel	Stride	Outputs
conv.	3×3	1×1	512
conv.	3×3	1×1	256

(d) Colorization network

Type	Kernel	Stride	Outputs
fusion	-	-	256
conv.	3×3	1×1	128
upsample	-	-	128
conv.	3×3	1×1	64
conv.	3×3	1×1	64
upsample	-	-	64
conv.	3×3	1×1	32
output	3×3	1×1	2

indicates that instead of computing Eq. (2) for consecutive pixels, every other pixel is computed. If padding is added to the layer, the output is effectively half the size of the input layer. This can be used to replace the max-pooling layers while maintaining performance [Springenberg et al. 2015]. We use 3×3 convolution kernels exclusively and a padding of 1×1 to ensure the output is the same size (or half if using a stride of 2) as the input. An overview of the architecture of the shared features is shown in Table 1-(a).

3.2.2 Global Image Features

The global image features are obtained by further processing the low-level features with four convolutional layers followed by three fully-connected layers. This results in a 256-dimensional vector representation of the image. The full details of the global image features network can be seen in Table 1-(b). Note that due to the nature of the linear layers in this network, it requires the input of the low-level features network to be of fixed size of 224×224 pixels. However, this limitation does not affect the full approach as we will discuss later.

3.2.3 Mid-Level Features

The mid-level features are obtained by processing the low-level features further with two convolutional layers. The output is bottlenecked from the original 512-channel low-level features to 256-channel mid-level features. Note that unlike the global image features, the low-level and mid-level features networks are fully convolutional networks, such that the output is a scaled version of the input. In particular the output of the mid-level features network is a volume of size $H/8 \times W/8 \times 256$, where H and W are the original image height and width respectively. The architecture used is shown in Table 1-(c).

3.2.4 Fusing Global and Local Features

In order to be able to combine the global image features, a 256-dimensional vector, with the (mid-level) local image features, a $H/8 \times W/8 \times 256$-dimensional volume, we introduce a fusion layer. This layer serves to incorporate the global features into local features. We write the output of the fusion layer for mid-level coordinates (u, v) as:

$$\mathbf{y}_{u,v}^{\text{fusion}} = \sigma\left(\mathbf{b} + W\begin{bmatrix}\mathbf{y}^{\text{global}} \\ \mathbf{y}_{u,v}^{\text{mid}}\end{bmatrix}\right), \tag{5}$$

where $\mathbf{y}_{u,v}^{\text{fusion}} \in \mathbb{R}^{256}$ is the fused feature at (u, v), $\mathbf{y}^{\text{global}} \in \mathbb{R}^{256}$ is the global feature vector, $\mathbf{y}_{u,v}^{\text{mid}} \in \mathbb{R}^{256}$ is the mid-level feature at (u, v), W is a 256-by-512 weight matrix, and $\mathbf{b} \in \mathbb{R}^{256}$ is a bias. Here, both W and b are learnable part of the network.

This can be thought of as concatenating the global features with the local features at each spatial location and processing them through a small one-layer network. This effectively combines the global feature and the local features to obtain a new feature map that is, as the mid-level features, a 3D volume. Therefore, the resulting features are independent of any resolution constraints that the global image features might have.

3.2.5 Colorization Network

Once the features are fused, they are processed by a set of convolutions and upsampling layers, the latter which consist of simply upsampling the input by using the nearest neighbour technique so that the output is twice as wide and twice as tall. These layers are alternated until the output is half the size of the original input. The output layer of the colorization network consists of a convolutional layer with a Sigmoid transfer function that outputs the chrominance of the input grayscale image. The architecture can be seen in Table 1-(d). Finally, the computed chrominance is combined with the input intensity image to produce the resulting color image.

In order to train the network, we use the Mean Square Error (MSE) criterion. Given a color image for training, we convert the image to grayscale and CIE L*a*b* colorspace. The input of the model is the grayscale image while the target output is the a*b* components of the CIE L*a*b* colorspace. The a*b* components are globally normalized so they lie in the $[0, 1]$ range of the Sigmoid transfer function. We then scale the target output to the size of the output of the colorization network and compute the MSE between the output and target output as the loss. This loss is then back-propagated through all the networks (global features, mid-level features and low-level features) to update all the parameters of the model.

3.3 Colorization with Classification

While training with only color images using the MSE criterion does give good performance, it makes obvious mistakes due to not properly learning the global context of the image, *e.g.*, whether it is indoors or outdoors. As learning these networks is an non-convex problem, we facilitate the optimization by also training for classification jointly with the colorization. As we train the model using a large-scale dataset for classification of N classes, we have classification labels available for training. These labels correspond to a global image tag and thus can be used to guide the training of the global image features. We do this by introducing another very small neural network that consists of two fully-connected layers: a hidden layer with 256 outputs and an output layer with as many outputs as the number of classes in the dataset, which is $N = 205$ in our case. The input of this network is the second to last layer of the global features network with 512 outputs. We train this network using the cross-entropy loss, jointly with the MSE loss for the colorization network. Thus, the global loss of our network becomes:

$$L(y^{\text{color}}, y^{\text{class}}) = \|y^{\text{color}} - y^{\text{color},*}\|^2_{\text{FRO}} - \alpha \left(y^{\text{class}}_{l^{\text{class}}} - \log \left(\sum_{i=0}^{N} \exp \left(y^{\text{class}}_i \right) \right) \right) , \quad (6)$$

where y^{color} and $y^{\text{color},*}$ is the output of the colorization network and the ground truth color, respectively, $\|\cdot\|_{\text{FRO}}$ is the Frobenius norm, y^{class} is the output of the classification network, l^{class} is the true class label for the image, and α is a parameter that weighs the relative importance of the cross-entropy classification loss. Setting $\alpha = 0$ would disable the classification loss and only use the colorization loss.

Note that when performing back-propagation, the color loss affects the entire network, while the classification loss only affects the classification network, global features network, and the shared low-level features network; it doesn't affect neither the colorization network nor the mid-level features network. For a visualization of the full architecture including the classification network, refer to Fig. 2.

3.4 Optimization and Learning

While our model is able to process images of any size, it is most efficient when the input images are 224×224 pixels, as the shared low-level features layers can share outputs. Note that when the input image size is of a different resolution, while the low-level feature weights are shared, a rescaled image of size 224×224 must be used for the global features network. This requires processing both the original image and the rescaled image through the low-level features network, increasing both memory consumption and computation time. While for evaluation this in not a problem, since processing time is generally under a second, the learning procedure has to process millions of images many times. It is therefore critical to be as efficient as possible when training. For this reason, we train the model exclusively with images of size 224×224 pixels. These images are obtained by first scaling the training images to 256×256 pixels and performing random crops to the final size. This cropping allows the model to generalize better to other images. We also randomly flip the images horizontally with 50% probability for further robustness.

Learning very deep networks such as the one proposed directly from a random initialization is a very challenging task. One of the recent improvements that have made this possible is batch normalization [Ioffe and Szegedy 2015]. Batch normalization consists of normalizing the output before the transfer function at each layer. The normalization is done by keeping a running mean and standard deviation of the input to the transfer function, and using this mean and standard deviation to normalize the input so that it is roughly mean-centered with a standard deviation of one. This has shown to speed up the learning greatly and allow learning of very deep networks from random initializations. We use batch normalization throughout the entire network during training. Once the network is trained, the batch normalization mean and standard deviation can be folded into the weights and the bias of the layer. This allows the network to not perform unnecessary computations during inference.

It is common to use Stochastic Gradient Descent (SGD) to optimize deep networks. However, this relies on setting a global learning rate which becomes critical to learning. Furthermore, this learning rate must also be gradually decreased as the model is taught. While there are standard practices for classification networks [Krizhevsky et al. 2012], there is no standard for tackling colorization with a novel architecture like the one proposed in this work. Therefore, instead of having to experiment and heuristically determine a good learning rate scheduler, we sidestep the problem by using the ADADELTA [Zeiler 2012] optimizer. This approach adaptively sets the learning rates for all the network parameters without requiring us to set a global learning rate. We follow the optimization procedure until convergence of the loss.

4 Experimental Results and Discussion

We evaluate different variants of our model as well as comparing against the state of the art. Evaluation is done on a large set of diverse images that includes historical black-and-white images and close-up images. We also evaluate our model in a user study and find that the output of our model is considered "natural" 92.6% of the time. Furthermore, we also show how it was possible to do style

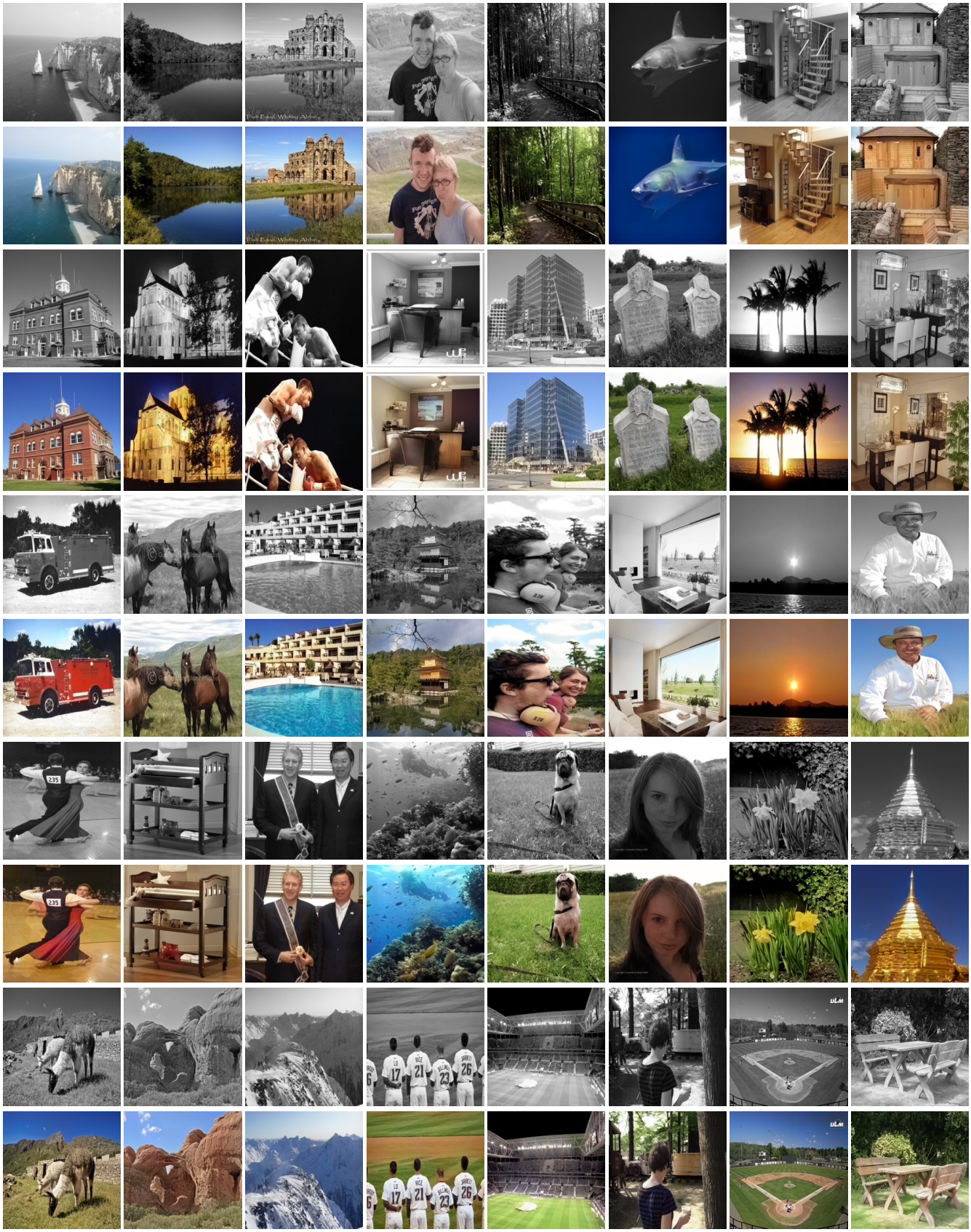


Figure 3: *We show the results of our approach on some of the images from the validation set of the Places dataset. Note the diversity of scene types and images, including indoor scenes, outdoor scenes, and close-ups of objects. Note that all these results were obtained automatically without any user intervention.*

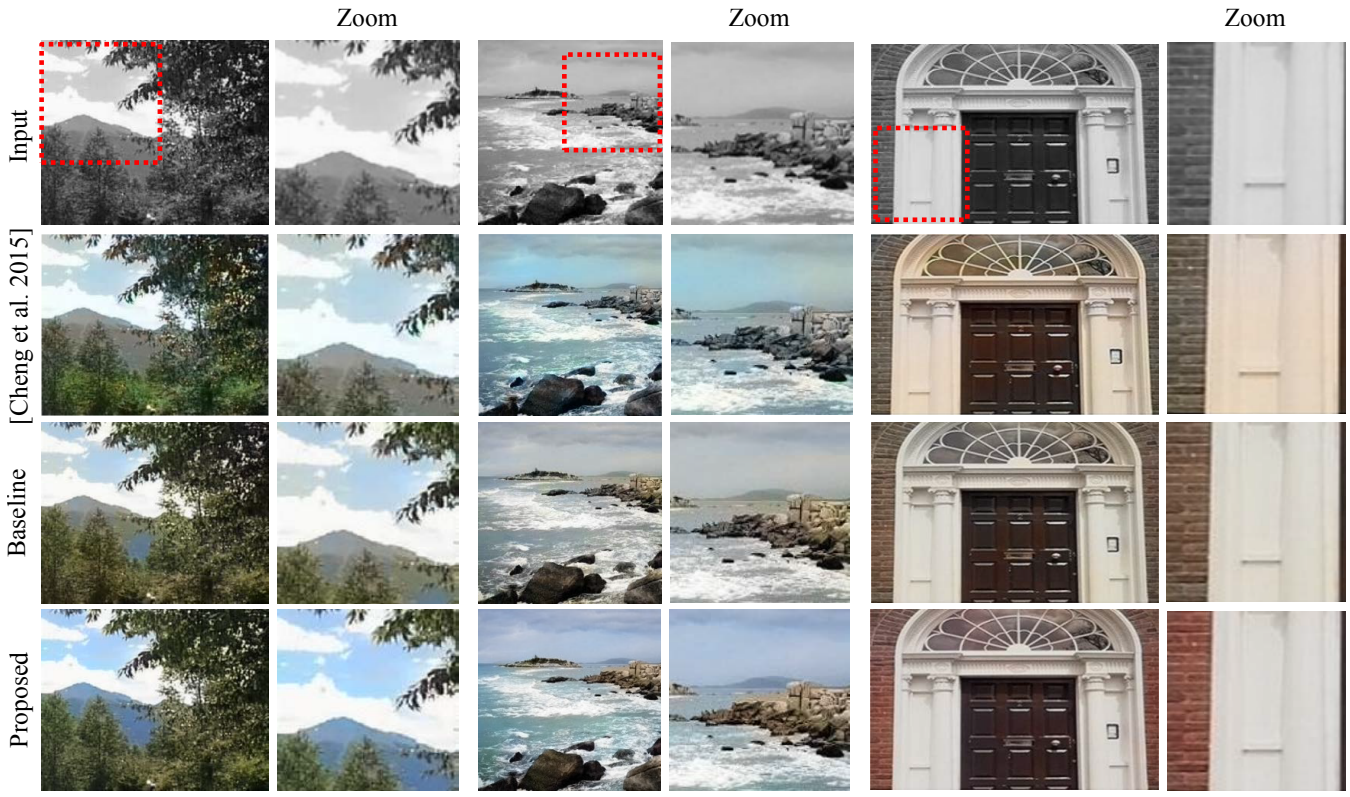


Figure 4: *We compare against the state of the art [Cheng et al. 2015]. First row contains the input image, the second row is the result by [Cheng et al. 2015], the third row is the result by our baseline approach without the global features, while the fourth row is the results by our approach. The area marked with a red dotted line is zoomed in. We can see that while [Cheng et al. 2015] fail to color important portions of the image (e.g., mountain, rocks, brick wall), our approach realistically colors the entire image.*

transfer directly using our architecture by exploiting global features computed on a different image.

Unless otherwise mentioned, we use $\alpha = {}^{1}/_{300}$ for training all models so that the classification loss and the colorization loss are similar in magnitude. We train our model on the Places scene dataset [Zhou et al. 2014], which consists of 2,448,872 training images and 20,500 validation images. In total, there are 205 classes corresponding to the types of the scene such as abbey, conference center, or volcano. We filter the images by removing grayscale images and those that have little color variance with a small automated script. This results in 2,327,958 training images and 19,546 validation images. We randomly sort the training images and used them for optimizing our model parameters. We train using a batch size of 128 for 200,000 iterations corresponding to roughly 11 epochs. This takes roughly 3 weeks on one core of a NVIDIA® Tesla® K80 GPU . All results shown are taken from the validation set and not the training set.

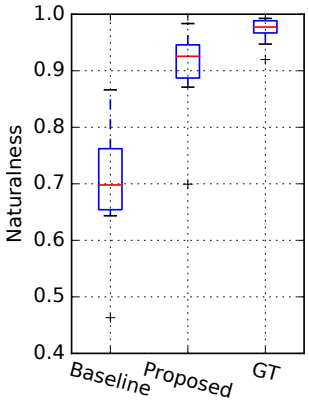
As a baseline, we also test our model without the global features, consisting of the low-level feature network, the mid-level feature network, and the colorization network with a 3×3 convolution in place of the fusion layer, all concatenated together. The baseline is trained in the same way as the main model, but with $\alpha = 0$ as there is no classification network. As we will show, this strong baseline already outperforms the state of the art, although it performs much worse than our full model. This baseline can be thought of as more straightforward deep learning approach to colorization. However, unlike standard approaches, it already contains a large set of non-standard improvements, such as not using fully-connected layers, using upscaling layers, and using a Sigmoid transfer function for the final layer.

4.1 Colorization Results

We show colorization results on the validation set of the Places dataset in Fig. 3. Note that these are all unconstrained and very challenging images to colorize. Our model exploits the semantic context of each image with the global features, allowing it to properly colorize baseball fields, underwater scenes, and fire trucks. Note that all these results are generated automatically without any human intervention.

4.2 Comparison with State of the Art

We compare against the state of the art [Cheng et al. 2015] in Fig. 4. We note that, in all cases the approach of [Cheng et al. 2015] failed to color important portions of the image, our approach could color all images in a coherent manner, as shown in the zoomed areas. In *forest*, neither [Cheng et al. 2015] nor our baseline could properly color the far away mountain nor the sky, while our full model could realistically color both. In *shore*, our approach accurately colorized the ocean, rocks and sky, while the other two approaches biased tones towards either the rocks or the ocean. Finally, in *doorway*, the red brick walls could be colorized accurately only by our approach, despite being a small portion of the original image. Notice also that our approach clearly distinguished between the colors of the



Approach	Naturalness (median)
Ground Truth	97.7%
Proposed	92.6%
Baseline	69.8%

Figure 5: *Results of our user study evaluating the **naturalness** of the colorizations. We evaluated the Ground Truth (GT), the baseline model, and our full model on the validation images for 10 different users. We can see that our model greatly outperforms the baseline and gets close to the ground truth.*

wall, the arch, and the door itself. All this was done by exploiting global context and training with a very large set of real images, which allowed our model to generate very realistic colorizations. On the other hand, [Cheng et al. 2015] are limited to images in which the semantic features work well: sky, sea, etc... This limits their application to outdoor scenes in which there are no people. Our approach can be applied to all sorts of images, *e.g.*, indoor scenes, portraits, and historical images with heavy artefacts as we will show in the next subsections.

4.3 User Study

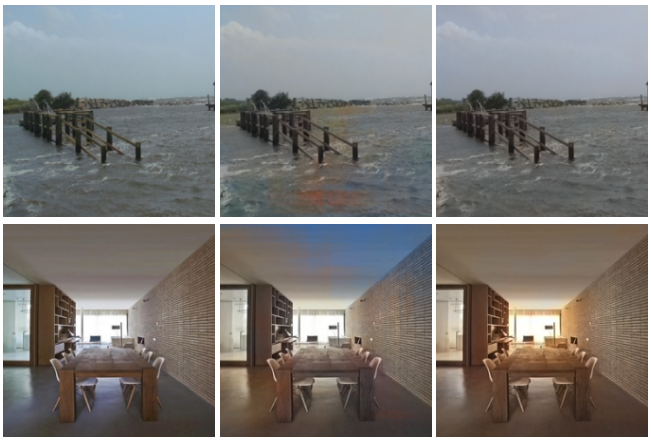
We performed a user study asking the question *"Does this image look natural to you?"* to evaluate the **naturalness** of the ground-truth validation images, the results of our baseline, and the results of our model. Images are randomly chosen and shown to the users one-by-one. The study was done with 10 different users, each shown roughly 500 images of each type for a total of 1,500 images each. Indications were given to the users to use their gut feeling and not try to spend too much time looking at the details of the images. All the images were shown at a resolution of 224×224 pixels. We show the results in Fig. 5. We can see that, while the baseline performs fairly poorly with roughly 70% of the images being considered natural, our approach has a median of 92.6%, which is close to the 97.7% of the ground truth. This strongly indicates that our model is able to generalize well and create realistic colorizations.

4.4 Importance of Global Features

The global features we propose in this work play a fundamental role in establishing the context of the scene. Just looking at local image patches leaves a lot of ambiguities which are hard to solve. For example, our baseline that does not include the global features sometimes makes critical errors such as coloring the sky in indoor images, or coloring the ground in images of oceans and lakes as shown in Fig. 6. This is further corroborated by our user study in which the results of our proposed model are deemed "natural" 92.6% of the time while the baseline model is only considered "natural" roughly 70% of the time as shown in Fig. 5.

4.5 Style Transfer through Global Features

One of the more interesting things our model can do is adapting the colorization of one image to the style of another. This is straightforward to do with our model due to the decorrelation between the global features and the mid-level features. In order to colorize an image A using the style taken from an image B, we compute the mid-level local features of image A and the global features from image B. We then fuse these features and process them with our colorization network. Note that we compute both the local and the global features from grayscale images; we do not use any color information at all. We show example of this style transfer in Fig. 7 in which we both change the season and time of day using the global features.



Ground truth Baseline Proposed

Figure 6: *Comparisons of our proposed architecture with and the baseline without global features. We can see that, without global features, the baseline sometimes makes the mistake of colorizing the sky in indoor images or the ground in the middle of a lake. In contrast, our proposed model uses the global features to establish the scene context in order to not make such trivial mistakes.*

Table 2: *We compare the results of our classification with the state of the art on the Places dataset validation split converted to grayscale. We use the standard approach of evaluating Top-1 Accuracy (whether or not the top predicted label is correct) and Top-5 Accuracy (whether or not the ground truth label is in the top-5 predictions). Despite using a much smaller network for classification than the competing networks, we can obtain comparable results for grayscale images. Furthermore, being able to classify is a side-product of training our classification model.*

Approach	Top-1 Acc.	Top-5 Acc.
Ours	50.6%	80.6%
VGG16 [Wang et al. 2015a]	52.8%	82.3%
CNDS [Wang et al. 2015b]	45.0%	75.6%
AlexNet [Zhou et al. 2014]	37.4%	68.5%

4.6 Colorizing the Past

We also test our model on historic black-and-white images. Note that, due to the age and the type of film used, these images are significantly different from modern images and the dataset used to train our model. Furthermore, as they are true black-and-white images: no ground truth exists. Despite these images having significant artifacts and a rough border around the edges, our model can obtain good results as shown in Fig. 8.

Figure 7: *We analyze the effect of using global features computed on different images when performing colorization. This can be seen as a form of style transfer in which the global priors of an image are used on another image. We show the result of using different combinations of local and global features. The first two rows show how it is possible to change the season from spring (yellow flowers blossoming) to fall (dried plants) by changing the global features. In the next two rows we show how it is possible to change the time of day using the global features. By using both local and global images computed from the same image, we get an image at dawn. However, if we change the global feature to dusk, we get the colorization shown in the middle, which appears to be taken at dusk. If we use the global features from a daytime image, we get the result on the right. Note that both the global and the local features were computed on grayscale images.*

4.7 Classification Results

Although our network is designed for colorization and not classification, we compare our classification network against the state of the art in Table 2 to verify to what extent it is able to perform classification. For the sake of comparison, we evaluate the other methods using grayscale input images. However, for our purposes and given the limitation of grayscale images, we believe it is an excellent result, as our network performs nearly as well as the best performing network that is exclusively trained and optimized for classification.

4.8 Comparison of Color Spaces

We compare the effect of using various color spaces. In particular, we compare RGB, YUV and L*a*b* color spaces. In the case of RGB, the output of the model is 3 instead of 2 corresponding to the red, green and blue channels. We train directly using the RGB values; however, for testing, we convert the RGB image to YUV and substitute the input grayscale image as the Y channel of the image. This ensures that the output images of all the models have the same luminance. In the case of YUV and L*a*b*, we output the chrominance and use the luminance of the input image to create the final colorized image. For all different color spaces, we normalize the values to lie in the $[0, 1]$ range of the Sigmoid transfer function of the output layer.

Results are shown in Fig. 9. In general, results are very similar. However, we do find some cases in which the L*a*b* gives the most perceptionally reasonable approach in comparison with RGB and YUV. For this reason, we use L*a*b* in our approach.

Table 3: *We run our model on images of different resolutions and compute the average run times on both CPU and GPU. We can see that, in general, using a GPU gives a $5\times$ speedup gain for our algorithm. Furthermore, we are able to process large images (4MPx) in a few seconds on the GPU.*

Image Size	Pixels	CPU (s)	GPU (s)	Speedup
$224 \times 224^{\dagger}$	50,176	0.399	0.080	$5.0\times$
512×512	262,144	1.676	0.339	$4.9\times$
1024×1024	1,048,576	5.629	1.084	$5.2\times$
2048×2048	4,194,304	20.116	4.218	$4.8\times$

† low-level feature network outputs are shared as done during training.

4.9 Computation Time

We evaluate the time it takes for our model to process images of several resolutions and show results in Table 3. We evaluate both on CPU using an *Intel® Core™i7-5960X CPU @ 3.00 GHz* with 8 cores and on GPU using a *NVIDIA® GeForce® GTX TITAN X*. For each resolution, we compute the mean of 100 computations to get a reliable value. We can see that both GPU and CPU are in the order of less than a few seconds for small images, with GPU giving roughly a $5\times$ speedup. For large images, using a GPU allows the colorization to still be done in a few seconds. Our approach is therefore suitable to near real-time usage.

4.10 Limitations and Discussion

The main limitation of the method lies in the fact that it is data-driven and thus will only be able to colorize images that share common properties with those in the training set. We have trained our model with a very large diverse set of both indoor and outdoor scene images in order to mitigate this. However, this does not contain, for example, human-created images. If we wish to evaluate on significantly different types of images, it would be necessary to train a new model for the new images.

In order to obtain good style transfer results, it is important for both images to have some semantic level of similarity between them, although the image itself can vary drastically as shown in Fig. 7. We found that best results are obtained when using images from the same class, or related classes (*e.g.*, Formal Garden, Botanical Garden, and Golf Course). Transferring the style of semantically unrelated images, such as those of an aquarium to an image of a baseball stadium, do not generally give realistic results, although it is not clear what result to expect in this case. This limitation mainly arises from the fact that our approach only uses the grayscale of both images and does not modify the luminance of the output image, as style transfer is not the main focus of this work.

Colorization is also an inherently ambiguous problem: is a man's shirt red or green? This has no unique solution. By learning from the data, our model will mainly use the dominant colors that it has learned, as shown in Fig. 10. Furthermore, there is no explicit way for the user to control the colors besides manually setting different global features. It is likely that there is a way to handle this by adding an additional optimization on the colorization process itself. However, this possibility is not explored in this paper.

(a) Cranberry Picking, Sep. 1911 (b) Burns Basement, May 1910 (c) Miner, Sep. 1937 (d) Scott's Run, Mar. 1937

Figure 8: *We show results on historical photographs roughly a hundred years old. First row shows the input black-and-white images, while our results are shown in the second row. Despite many artefacts due to the age of the photographies, our model can show an impressive automatic colorization of outdoor photography with people (a), indoor photography with people (b), close-up portraits (c), and can recognize snow (d). All photographies were taken by Lewis Hine and are part of the US National Archives (Public Domain).*

Ground truth RGB YUV L*a*b*

Figure 9: *Comparison of L*a*b*, YUV, and RGB color spaces for our model architecture. We compare a model trained to predict color values using RGB, YUV, and L*a*b* color spaces. While results are generally fairly similar, in some of the more challenging examples, the L*a*b* colorspace gives the most perceptionally reasonable results.*

5 Conclusions

We have presented a novel architecture for the colorization of grayscale images by fusing both global and local information. Our approach is based on convolutional neural networks and is able to perform the colorization without any user intervention. We train our model end-to-end on a large dataset for scene recognition with a joint colorization and classification loss that allows it not only to understand colors, but also adapts the colors to the context of the image, *i.e.*, the sky color in a sunset image is not the same as in a daylight image. Our architecture allows us to process images of any resolution, unlike most deep-learning frameworks. Furthermore, we show that with the same model we can do style transfer, that is, color an image using the context of another. Finally, we evaluated our model on a large diverse set of both indoor and outdoor images and showed that it can produce very credible results. We compared against the state of the art and also performed a user study that corroborates the results. Our approach runs in near real-time and has many potential applications such as automatic colorization of historical photography and movie archives.

Acknowledgements: This work was partially supported by JST CREST.

Input Ground truth Proposed

Figure 10: *Typical failure cases of our method. In the upper row, our model fails to give an autumn color to leaves as our model does not manage to capture the semantic context of the image, i.e., fall. In the bottom row, although a color of the tent is blue in the ground truth, our model colors it orange. This is due to the inherent ambiguity of the colorization problem.*

References

AN, X., AND PELLACINI, F. 2008. AppProp: All-pairs appearance-space edit propagation. *ACM Trans. Graph. 27*, 3 (Aug.), 40:1–40:9.

BELL, S., AND BALA, K. 2015. Learning visual similarity for product design with convolutional neural networks. *ACM Trans. on Graphics (SIGGRAPH) 34*, 4.

BROMLEY, J., GUYON, I., LECUN, Y., SÄCKINGER, E., AND SHAH, R. 1994. Signature verification using a "siamese" time delay neural network. In *NIPS*.

CHARPIAT, G., HOFMANN, M., AND SCHÖLKOPF, B. 2008.

Automatic image colorization via multimodal predictions. In *ECCV*.

CHEN, X., ZOU, D., ZHAO, Q., AND TAN, P. 2012. Manifold preserving edit propagation. *ACM Trans. Graph. 31*, 6 (Nov.), 132:1–132:7.

CHENG, Z., YANG, Q., AND SHENG, B. 2015. Deep colorization. In *Proceedings of ICCV 2015*, 29–43.

CHIA, A. Y.-S., ZHUO, S., GUPTA, R. K., TAI, Y.-W., CHO, S.-Y., TAN, P., AND LIN, S. 2011. Semantic colorization with internet images. *ACM Trans. Graph. 30*, 6, 156:1–156:8.

DONG, C., LOY, C. C., HE, K., AND TANG, X. 2016. Image super-resolution using deep convolutional networks. *PAMI 38*, 2, 295–307.

EIGEN, D., AND FERGUS, R. 2015. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *ICCV*.

FISCHER, P., DOSOVITSKIY, A., ILG, E., HÄUSSER, P., HAZIRBAS, C., GOLKOV, V., VAN DER SMAGT, P., CREMERS, D., AND BROX, T. 2015. Flownet: Learning optical flow with convolutional networks.

FUKUSHIMA, K. 1988. Neocognitron: A hierarchical neural network capable of visual pattern recognition. *Neural networks 1*, 2, 119–130.

GUPTA, R. K., CHIA, A. Y.-S., RAJAN, D., NG, E. S., AND ZHIYONG, H. 2012. Image colorization using similar images. In *ACM International Conference on Multimedia*, 369–378.

HUANG, Y.-C., TUNG, Y.-S., CHEN, J.-C., WANG, S.-W., AND WU, J.-L. 2005. An adaptive edge detection based colorization algorithm and its applications. In *ACM International Conference on Multimedia*, 351–354.

IOFFE, S., AND SZEGEDY, C. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*.

IRONY, R., COHEN-OR, D., AND LISCHINSKI, D. 2005. Colorization by example. In *Eurographics Conference on Rendering Techniques*, 201–210.

KRIZHEVSKY, A., SUTSKEVER, I., AND HINTON, G. E. 2012. Imagenet classification with deep convolutional neural networks. In *NIPS*.

LECUN, Y., BOTTOU, L., BENGIO, Y., AND HAFFNER, P. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE 86*, 11, 2278–2324.

LEVIN, A., LISCHINSKI, D., AND WEISS, Y. 2004. Colorization using optimization. *ACM Transactions on Graphics 23*, 689–694.

LI, Y., JU, T., AND HU, S.-M. 2010. Instant propagation of sparse edits on images and videos. *Computer Graphics Forum 29*, 7, 2049–2054.

LIU, X., WAN, L., QU, Y., WONG, T.-T., LIN, S., LEUNG, C.-S., AND HENG, P.-A. 2008. Intrinsic colorization. *ACM Transactions on Graphics (SIGGRAPH Asia 2008 issue) 27*, 5 (December), 152:1–152:9.

LONG, J., SHELHAMER, E., AND DARRELL, T. 2015. Fully convolutional networks for semantic segmentation. In *CVPR*.

LUAN, Q., WEN, F., COHEN-OR, D., LIANG, L., XU, Y.-Q., AND SHUM, H.-Y. 2007. Natural image colorization. In *Eurographics Conference on Rendering Techniques*, 309–320.

PITIÉ, F., KOKARAM, A. C., AND DAHYOT, R. 2007. Automated colour grading using colour distribution transfer. *Comput. Vis. Image Underst. 107*, 1-2 (July), 123–137.

QU, Y., WONG, T.-T., AND HENG, P.-A. 2006. Manga colorization. *ACM Trans. Graph. 25*, 3 (July), 1214–1220.

REINHARD, E., ASHIKHMIN, M., GOOCH, B., AND SHIRLEY, P. 2001. Color transfer between images. *IEEE Computer Graphics and Applications 21*, 5 (sep), 34–41.

RUMELHART, D., HINTON, G., AND WILLIAMS, R. 1986. Learning representations by back-propagating errors. In *Nature*.

SHEN, W., WANG, X., WANG, Y., BAI, X., AND ZHANG, Z. 2015. Deepcontour: A deep convolutional feature learned by positive-sharing loss for contour detection. In *CVPR*.

SIMONYAN, K., AND ZISSERMAN, A. 2015. Very deep convolutional networks for large-scale image recognition. In *ICLR*.

SPRINGENBERG, J. T., DOSOVITSKIY, A., BROX, T., AND RIEDMILLER, M. A. 2015. Striving for simplicity: The all convolutional net. In *ICLR Workshop Track*.

SÝKORA, D., DINGLIANA, J., AND COLLINS, S. 2009. LazyBrush: Flexible painting tool for hand-drawn cartoons. *Computer Graphics Forum 28*, 2, 599–608.

TAI, Y., JIA, J., AND TANG, C. 2005. Local color transfer via probabilistic segmentation by expectation-maximization. In *CVPR*, 747–754.

WANG, L., GUO, S., HUANG, W., AND QIAO, Y. 2015. Places205-vggnet models for scene recognition. *CoRR abs/1508.01667*.

WANG, L., LEE, C., TU, Z., AND LAZEBNIK, S. 2015. Training deeper convolutional networks with deep supervision. *CoRR abs/1505.02496*.

WANG, X., FOUHEY, D. F., AND GUPTA, A. 2015. Designing deep networks for surface normal estimation. In *CVPR*.

WELSH, T., ASHIKHMIN, M., AND MUELLER, K. 2002. Transferring color to greyscale images. *ACM Trans. Graph. 21*, 3 (July), 277–280.

WU, F., DONG, W., KONG, Y., MEI, X., PAUL, J.-C., AND ZHANG, X. 2013. Content-based colour transfer. *Computer Graphics Forum 32*, 1, 190–203.

XU, K., LI, Y., JU, T., HU, S.-M., AND LIU, T.-Q. 2009. Efficient affinity-based edit propagation using k-d tree. *ACM Trans. Graph. 28*, 5 (Dec.), 118:1–118:6.

XU, L., YAN, Q., AND JIA, J. 2013. A sparse control model for image and video editing. *ACM Trans. Graph. 32*, 6 (Nov.), 197:1–197:10.

YATZIV, L., YATZIV, L., SAPIRO, G., AND SAPIRO, G. 2004. Fast image and video colorization using chrominance blending. *IEEE Transaction on Image Processing 15*, 2006.

ZEILER, M. D. 2012. ADADELTA: an adaptive learning rate method. *CoRR abs/1212.5701*.

ZHOU, B., LAPEDRIZA, A., XIAO, J., TORRALBA, A., AND OLIVA, A. 2014. Learning deep features for scene recognition using places database. In *NIPS*.