Colorful Image Colorization

Richard Zhang, Phillip Isola, Alexei A. Efros
{rich.zhang,isola,efros}@eecs.berkeley.edu

University of California, Berkeley

Abstract. Given a grayscale photograph as input, this paper attacks the problem of hallucinating a *plausible* color version of the photograph. This problem is clearly underconstrained, so previous approaches have either relied on significant user interaction or resulted in desaturated colorizations. We propose a fully automatic approach that produces vibrant and realistic colorizations. We embrace the underlying uncertainty of the problem by posing it as a classification task and use class-rebalancing at training time to increase the diversity of colors in the result. The system is implemented as a feed-forward pass in a CNN at test time and is trained on over a million color images. We evaluate our algorithm using a "colorization Turing test," asking human participants to choose between a generated and ground truth color image. Our method successfully fools humans on 32% of the trials, significantly higher than previous methods. Moreover, we show that colorization can be a powerful pretext task for self-supervised feature learning, acting as a *cross-channel encoder*. This approach results in state-of-the-art performance on several feature learning benchmarks.

Keywords: Colorization, Vision for Graphics, CNNs, Self-supervised learning

1 Introduction

Consider the grayscale photographs in Figure 1. At first glance, hallucinating their colors seems daunting, since so much of the information (two out of the three dimensions) has been lost. Looking more closely, however, one notices that in many cases, the semantics of the scene and its surface texture provide ample cues for many regions in each image: the grass is typically green, the sky is typically blue, and the ladybug is most definitely red. Of course, these kinds of semantic priors do not work for everything, e.g., the croquet balls on the grass might not, in reality, be red, yellow, and purple (though it's a pretty good guess). However, for this paper, our goal is not necessarily to recover the actual ground truth color, but rather to produce a *plausible* colorization that could potentially fool a human observer. Therefore, our task becomes much more achievable: to model enough of the statistical dependencies between the semantics and the textures of grayscale images and their color versions in order to produce visually compelling results.

Given the lightness channel L, our system predicts the corresponding a and b color channels of the image in the CIE Lab colorspace. To solve this problem,

Fig. 1. Example input grayscale photos and output colorizations from our algorithm. These examples are cases where our model works especially well. Please visit `http://richzhang.github.io/colorization/` to see the full range of results and to try our model and code. Best viewed in color (obviously).

we leverage large-scale data. Predicting color has the nice property that training data is practically free: any color photo can be used as a training example, simply by taking the image's L channel as input and its ab channels as the supervisory signal. Others have noted the easy availability of training data, and previous works have trained convolutional neural networks (CNNs) to predict color on large datasets [1,2]. However, the results from these previous attempts tend to look desaturated. One explanation is that [1,2] use loss functions that encourage conservative predictions. These losses are inherited from standard regression problems, where the goal is to minimize Euclidean error between an estimate and the ground truth.

We instead utilize a loss tailored to the colorization problem. As pointed out by [3], color prediction is inherently multimodal – many objects can take on several plausible colorizations. For example, an apple is typically red, green, or yellow, but unlikely to be blue or orange. To appropriately model the multimodal nature of the problem, we predict a distribution of possible colors for each pixel. Furthermore, we re-weight the loss at training time to emphasize rare colors. This encourages our model to exploit the full diversity of the large-scale data on which it is trained. Lastly, we produce a final colorization by taking the *annealed-mean* of the distribution. The end result is colorizations that are more vibrant and perceptually realistic than those of previous approaches.

Evaluating synthesized images is notoriously difficult [4]. Since our ultimate goal is to make results that are compelling to a human observer, we introduce a novel way of evaluating colorization results, directly testing their perceptual realism. We set up a "colorization Turing test," in which we show participants real and synthesized colors for an image, and ask them to identify the fake. In this quite difficult paradigm, we are able to fool participants on 32% of the instances (ground truth colorizations would achieve 50% on this metric), significantly higher than prior work [2]. This test demonstrates that in many cases,

our algorithm is producing nearly photorealistic results (see Figure 1 for selected successful examples from our algorithm). We also show that our system's colorizations are realistic enough to be useful for downstream tasks, in particular object classification, using an off-the-shelf VGG network [5].

We additionally explore colorization as a form of self-supervised representation learning, where raw data is used as its own source of supervision. The idea of learning feature representations in this way goes back at least to autoencoders [6]. More recent works have explored feature learning via data imputation, where a held-out subset of the complete data is predicted (e.g., [7,8,9,10,11,12,13]). Our method follows in this line, and can be termed a *cross-channel encoder*. We test how well our model performs in generalization tasks, compared to previous [14,8,15,10] and concurrent [16] self-supervision algorithms, and find that our method performs surprisingly well, achieving state-of-the-art performance on several metrics.

Our contributions in this paper are in two areas. First, we make progress on the graphics problem of automatic image colorization by (a) designing an appropriate objective function that handles the multimodal uncertainty of the colorization problem and captures a wide diversity of colors, (b) introducing a novel framework for testing colorization algorithms, potentially applicable to other image synthesis tasks, and (c) setting a new high-water mark on the task by training on a million color photos. Secondly, we introduce the colorization task as a competitive and straightforward method for self-supervised representation learning, achieving state-of-the-art results on several benchmarks.

Prior work on colorization Colorization algorithms mostly differ in the ways they obtain and treat the data for modeling the correspondence between grayscale and color. Non-parametric methods, given an input grayscale image, first define one or more color reference images (provided by a user or retrieved automatically) to be used as source data. Then, following the Image Analogies framework [17], color is transferred onto the input image from analogous regions of the reference image(s) [18,19,20,21]. Parametric methods, on the other hand, learn prediction functions from large datasets of color images at training time, posing the problem as either regression onto continuous color space [22,1,2] or classification of quantized color values [3]. Our method also learns to classify colors, but does so with a larger model, trained on more data, and with several innovations in the loss function and mapping to a final continuous output.

Concurrent work on colorization Concurrently with our paper, Larsson et al. [23] and Iizuka et al. [24] have developed similar systems, which leverage large-scale data and CNNs. The methods differ in their CNN architectures and loss functions. While we use a classification loss, with rebalanced rare classes, Larsson et al. use an un-rebalanced classification loss, and Iizuka et al. use a regression loss. In Section 3.1, we compare the effect of each of these types of loss function in conjunction with our architecture. The CNN architectures are also somewhat different: Larsson et al. use hypercolumns [25] on a VGG network [5], Iizuka et al. use a two-stream architecture in which they fuse global and local features, and we use a single-stream, VGG-styled network with added depth and dilated convolutions [26,27]. In addition, while we and Larsson et al. train our models on ImageNet [28], Iizuka et al. train their model on Places

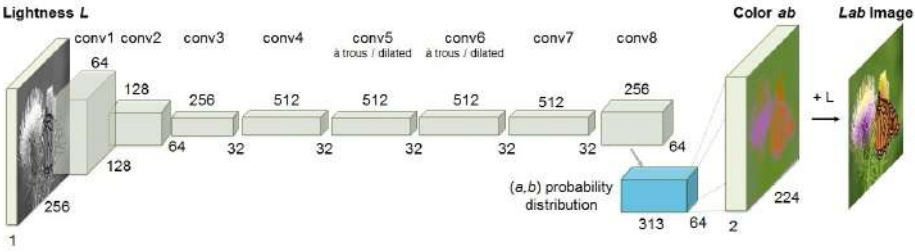

Fig. 2. Our network architecture. Each `conv` layer refers to a block of 2 or 3 repeated `conv` and `ReLU` layers, followed by a `BatchNorm` [30] layer. The net has no `pool` layers. All changes in resolution are achieved through spatial downsampling or upsampling between `conv` blocks.

[29]. In Section 3.1, we provide quantitative comparisons to Larsson et al., and encourage interested readers to investigate both concurrent papers.

2 Approach

We train a CNN to map from a grayscale input to a distribution over quantized color value outputs using the architecture shown in Figure 2. Architectural details are described in the supplementary materials on our project webpage[1], and the model is publicly available. In the following, we focus on the design of the objective function, and our technique for inferring point estimates of color from the predicted color distribution.

2.1 Objective Function

Given an input lightness channel $\mathbf{X} \in \mathbb{R}^{H \times W \times 1}$, our objective is to learn a mapping $\widehat{\mathbf{Y}} = \mathcal{F}(\mathbf{X})$ to the two associated color channels $\mathbf{Y} \in \mathbb{R}^{H \times W \times 2}$, where H, W are image dimensions.

(We denote predictions with a $\widehat{\cdot}$ symbol and ground truth without.) We perform this task in CIE *Lab* color space. Because distances in this space model perceptual distance, a natural objective function, as used in [1,2], is the Euclidean loss $\mathrm{L}_2(\cdot, \cdot)$ between predicted and ground truth colors:

$$\mathrm{L}_2(\widehat{\mathbf{Y}}, \mathbf{Y}) = \frac{1}{2} \sum_{h,w} \|\mathbf{Y}_{h,w} - \widehat{\mathbf{Y}}_{h,w}\|_2^2 \tag{1}$$

However, this loss is not robust to the inherent ambiguity and multimodal nature of the colorization problem. If an object can take on a set of distinct *ab* values, the optimal solution to the Euclidean loss will be the mean of the set. In color prediction, this averaging effect favors grayish, desaturated results. Additionally, if the set of plausible colorizations is non-convex, the solution will in fact be out of the set, giving implausible results.

[1] `http://richzhang.github.io/colorization/`

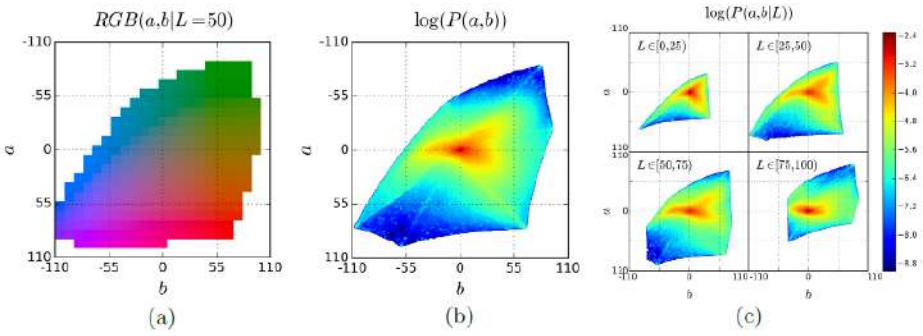

Fig. 3. (a) Quantized ab color space with a grid size of 10. A total of 313 ab pairs are in gamut. (b) Empirical probability distribution of ab values, shown in log scale. (c) Empirical probability distribution of ab values, conditioned on L, shown in log scale.

Instead, we treat the problem as multinomial classification. We quantize the ab output space into bins with grid size 10 and keep the $Q = 313$ values which are in-gamut, as shown in Figure 3(a). For a given input $\mathbf{X}$, we learn a mapping $\widehat{\mathbf{Z}} = \mathcal{G}(\mathbf{X})$ to a probability distribution over possible colors $\widehat{\mathbf{Z}} \in [0, 1]^{H \times W \times Q}$, where Q is the number of quantized ab values.

To compare predicted $\widehat{\mathbf{Z}}$ against ground truth, we define function $\mathbf{Z} = \mathcal{H}_{gt}^{-1}(\mathbf{Y})$, which converts ground truth color $\mathbf{Y}$ to vector $\mathbf{Z}$, using a soft-encoding scheme[2]. We then use multinomial cross entropy loss $\mathrm{L}_{cl}(\cdot, \cdot)$, defined as:

$$\mathrm{L}_{cl}(\widehat{\mathbf{Z}}, \mathbf{Z}) = -\sum_{h,w} v(\mathbf{Z}_{h,w}) \sum_{q} \mathbf{Z}_{h,w,q} \log(\widehat{\mathbf{Z}}_{h,w,q}) \tag{2}$$

where $v(\cdot)$ is a weighting term that can be used to rebalance the loss based on color-class rarity, as defined in Section 2.2 below. Finally, we map probability distribution $\widehat{\mathbf{Z}}$ to color values $\widehat{\mathbf{Y}}$ with function $\widehat{\mathbf{Y}} = \mathcal{H}(\widehat{\mathbf{Z}})$, which will be further discussed in Section 2.3.

2.2 Class rebalancing

The distribution of ab values in natural images is strongly biased towards values with low ab values, due to the appearance of backgrounds such as clouds, pavement, dirt, and walls. Figure 3(b) shows the empirical distribution of pixels in ab space, gathered from 1.3M training images in ImageNet [28]. Observe that the number of pixels in natural images at desaturated values are orders of magnitude higher than for saturated values. Without accounting for this, the

[2] Each ground truth value $\mathbf{Y}_{h,w}$ can be encoded as a 1-hot vector $\mathbf{Z}_{h,w}$ by searching for the nearest quantized ab bin. However, we found that *soft*-encoding worked well for training, and allowed the network to quickly learn the relationship between elements in the output space [31]. We find the 5-nearest neighbors to $\mathbf{Y}_{h,w}$ in the output space and weight them proportionally to their distance from the ground truth using a Gaussian kernel with $\sigma = 5$.

loss function is dominated by desaturated *ab* values. We account for the class-imbalance problem by reweighting the loss of each pixel at train time based on the pixel color rarity. This is asymptotically equivalent to the typical approach of resampling the training space [32]. Each pixel is weighed by factor $\mathbf{w} \in \mathbb{R}^Q$, based on its closest *ab* bin.

$$v(\mathbf{Z}_{h,w}) = \mathbf{w}_{q^*}, \text{ where } q^* = \arg\max_q \mathbf{Z}_{h,w,q} \tag{3}$$

$$\mathbf{w} \propto \left((1-\lambda)\widetilde{\mathbf{p}} + \frac{\lambda}{Q}\right)^{-1}, \quad \mathbb{E}[\mathbf{w}] = \sum_q \widetilde{\mathbf{p}}_q \mathbf{w}_q = 1 \tag{4}$$

To obtain smoothed empirical distribution $\widetilde{\mathbf{p}} \in \Delta^Q$, we estimate the empirical probability of colors in the quantized *ab* space $\mathbf{p} \in \Delta^Q$ from the full ImageNet training set and smooth the distribution with a Gaussian kernel $\mathbf{G}_\sigma$. We then mix the distribution with a uniform distribution with weight $\lambda \in [0, 1]$, take the reciprocal, and normalize so the weighting factor is 1 on expectation. We found that values of $\lambda = \frac{1}{2}$ and $\sigma = 5$ worked well. We compare results with and without class rebalancing in Section 3.1.

2.3 Class Probabilities to Point Estimates

Finally, we define $\mathcal{H}$, which maps the predicted distribution $\widehat{\mathbf{Z}}$ to point estimate $\widehat{\mathbf{Y}}$ in *ab* space. One choice is to take the mode of the predicted distribution for each pixel, as shown in the right-most column of Figure 4 for two example images. This provides a vibrant but sometimes spatially inconsistent result, e.g., the red splotches on the bus. On the other hand, taking the mean of the predicted distribution produces spatially consistent but desaturated results (left-most column of Figure 4), exhibiting an unnatural sepia tone. This is unsurprising, as taking the mean after performing classification suffers from some of the same issues as optimizing for a Euclidean loss in a regression framework. To try to get the best of both worlds, we *interpolate* by re-adjusting the temperature T of the softmax distribution, and taking the mean of the result. We draw inspiration from the simulated annealing technique [33], and thus refer to the operation as taking the *annealed-mean* of the distribution:

$$\mathcal{H}(\mathbf{Z}_{h,w}) = \mathbb{E}\big[f_T(\mathbf{Z}_{h,w})\big], \quad f_T(\mathbf{z}) = \frac{\exp(\log(\mathbf{z})/T)}{\sum_q \exp(\log(\mathbf{z}_q)/T)} \tag{5}$$

Setting $T = 1$ leaves the distribution unchanged, lowering the temperature T produces a more strongly peaked distribution, and setting $T \to 0$ results in a 1-hot encoding at the distribution mode. We found that temperature $T = 0.38$, shown in the middle column of Figure 4, captures the vibrancy of the mode while maintaining the spatial coherence of the mean.

Our final system $\mathcal{F}$ is the composition of CNN $\mathcal{G}$, which produces a predicted distribution over all pixels, and the annealed-mean operation $\mathcal{H}$, which produces a final prediction. The system is not quite end-to-end trainable, but note that the mapping $\mathcal{H}$ operates on each pixel independently, with a single parameter, and can be implemented as part of a feed-forward pass of the CNN.

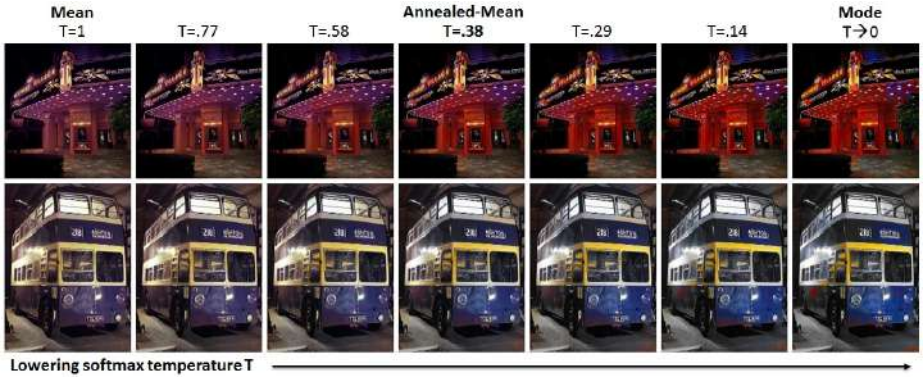

Fig. 4. The effect of temperature parameter T on the *annealed-mean* output (Equation 5). The left-most images show the means of the predicted color distributions and the right-most show the modes. We use $T = 0.38$ in our system.

3 Experiments

In Section 3.1, we assess the graphics aspect of our algorithm, evaluating the perceptual realism of our colorizations, along with other measures of accuracy. We compare our full algorithm to several variants, along with recent [2] and concurrent work [23]. In Section 3.2, we test colorization as a method for self-supervised representation learning. Finally, in Section 10.1, we show qualitative examples on legacy black and white images.

3.1 Evaluating colorization quality

We train our network on the 1.3M images from the ImageNet training set [28], validate on the first 10k images in the ImageNet validation set, and test on a separate 10k images in the validation set, same as in [23]. We show quantitative results in Table 1 on three metrics. A qualitative comparison for selected success and failure cases is shown in Figure 5. For a comparison on a full selection of random images, please see our project webpage.

To specifically test the effect of different loss functions, we train our CNN with various losses. We also compare to previous [2] and concurrent methods [23], which both use CNNs trained on ImageNet, along with naive baselines:

1. **Ours (full)** Our full method, with classification loss, defined in Equation 2, and class rebalancing, as described in Section 2.2. The network was trained from scratch with k-means initialization [36], using the ADAM solver for approximately 450k iterations[3].
2. **Ours (class)** Our network on classification loss but no class rebalancing ($\lambda = 1$ in Equation 4).

[3] $\beta_1 = .9$, $\beta_2 = .99$, and weight decay $= 10^{-3}$. Initial learning rate was 3×10^{-5} and dropped to 10^{-5} and 3×10^{-6} when loss plateaued, at 200k and 375k iterations, respectively. Other models trained from scratch followed similar training protocol.

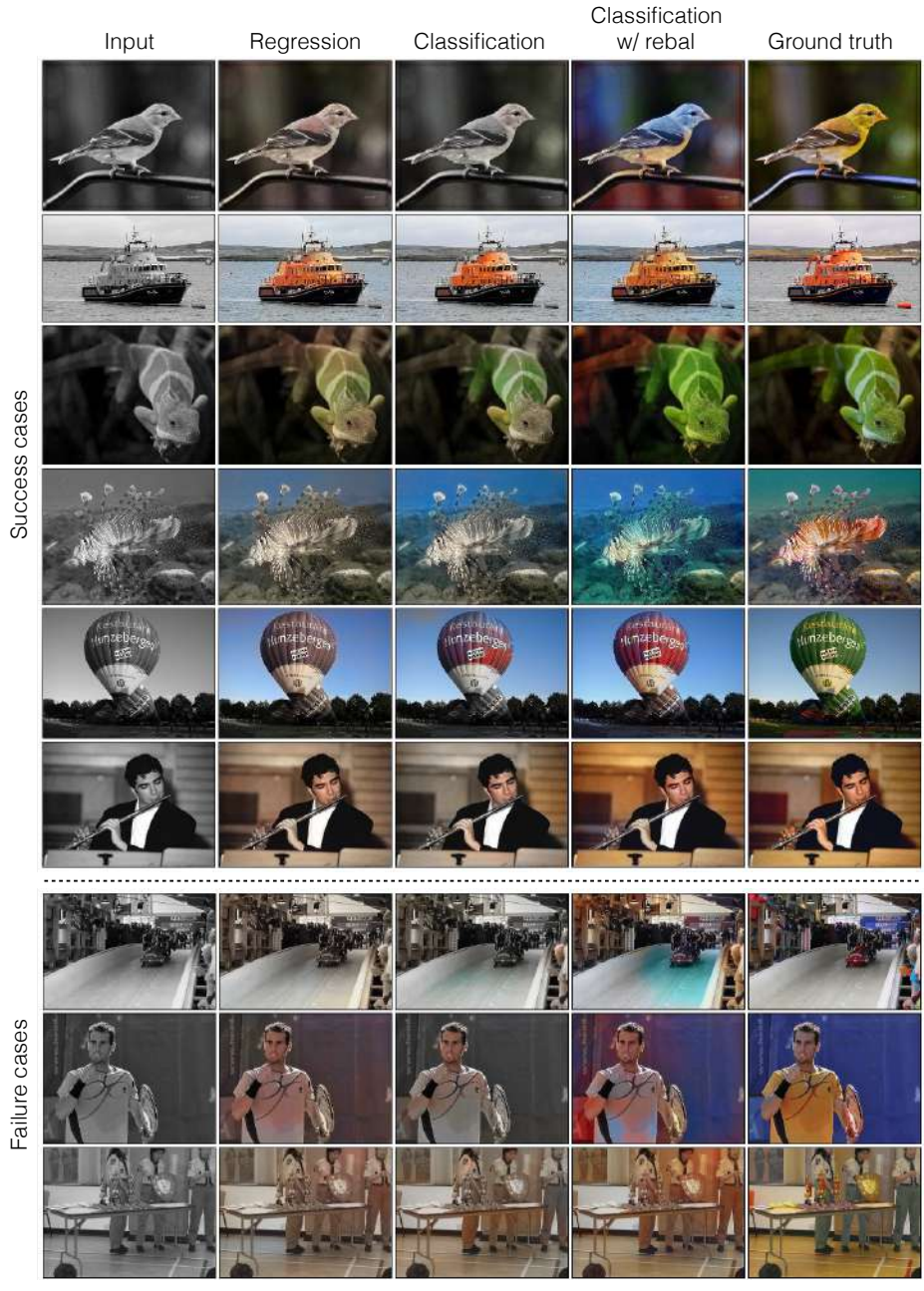

Fig. 5. Example results from our ImageNet test set. Our classification loss with rebalancing produces more accurate and vibrant results than a regression loss or a classification loss without rebalancing. Successful colorizations are above the dotted line. Common failures are below. These include failure to capture long-range consistency, frequent confusions between red and blue, and a default sepia tone on complex indoor scenes. Please visit `http://richzhang.github.io/colorization/` to see the full range of results.

Colorization Results on ImageNet							
	Model			AuC		VGG Top-1	AMT
Method	Params (MB)	Feats (MB)	Runtime (ms)	non-rebal (%)	rebal (%)	Class Acc (%)	Labeled Real (%)
Ground Truth	–	–	–	100	100	68.3	50
Gray	–	–	–	89.1	58.0	52.7	–
Random	–	–	–	84.2	57.3	41.0	13.0±4.4
Dahl [2]	–	–	–	90.4	58.9	48.7	18.3±2.8
Larsson et al. [23]	588	495	122.1	**91.7**	65.9	**59.4**	**27.2±2.7**
Ours (L2)	129	127	17.8	91.2	64.4	54.9	21.2±2.5
Ours (L2, ft)	129	127	17.8	91.5	66.2	56.5	23.9±2.8
Ours (class)	129	142	22.1	91.6	65.1	56.6	25.2±2.7
Ours (full)	129	142	22.1	89.5	**67.3**	56.0	**32.3±2.2**

Table 1. Colorization results on 10k images in the ImageNet validation set [28], as used in [23]. AuC refers to the area under the curve of the cumulative error distribution over *ab* space [22]. Results column 2 shows the class-balanced variant of this metric. Column 3 is the classification accuracy after colorization using the VGG-16 [5] network. Column 4 shows results from our AMT *real vs. fake* test (with mean and standard error reported, estimated by bootstrap [34]). Note that an algorithm that produces ground truth images would achieve 50% performance in expectation. Higher is better for all metrics. Rows refer to different algorithms; see text for a description of each. Parameter and feature memory, and runtime, were measured on a Titan X GPU using *Caffe* [35].

3. **Ours (L2)** Our network trained from scratch, with L2 regression loss, described in Equation 1, following the same training protocol.
4. **Ours (L2, ft)** Our network trained with L2 regression loss, fine-tuned from our full classification with rebalancing network.
5. **Larsson et al. [23]** A CNN method that also appears in these proceedings.
6. **Dahl [2]** A previous model using a Laplacian pyramid on VGG features, trained with L2 regression loss.
7. **Gray** Colors every pixel gray, with $(a, b) = 0$.
8. **Random** Copies the colors from a random image from the training set.

Evaluating the quality of synthesized images is well-known to be a difficult task, as simple quantitative metrics, like RMS error on pixel values, often fail to capture visual realism. To address the shortcomings of any individual evaluation, we test three that measure different senses of quality, shown in Table 1.

1. Perceptual realism (AMT): For many applications, such as those in graphics, the ultimate test of colorization is how compelling the colors look to a human observer. To test this, we ran a *real vs. fake* two-alternative forced choice experiment on Amazon Mechanical Turk (AMT). Participants in the experiment were shown a series of pairs of images. Each pair consisted of a color photo next to a re-colorized version, produced by either our algorithm or a baseline. Participants were asked to click on the photo they believed contained *fake* colors generated by a computer program. Individual images of resolution 256×256 were shown for one second each, and after each pair, participants were given unlimited time to respond. Each experimental session consisted of 10 practice trials

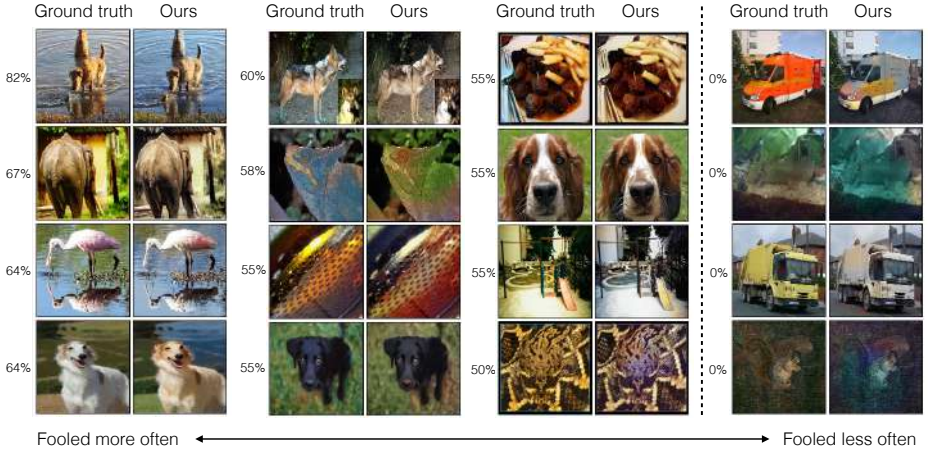

Fig. 6. Images sorted by how often AMT participants chose our algorithm's colorization over the ground truth. In all pairs to the left of the dotted line, participants believed our colorizations to be more real than the ground truth on $\geq 50\%$ of the trials. In some cases, this may be due to poor white balancing in the ground truth image, corrected by our algorithm, which predicts a more prototypical appearance. Right of the dotted line are examples where participants were never fooled.

(excluded from subsequent analysis), followed by 40 test pairs. On the practice trials, participants were given feedback as to whether or not their answer was correct. No feedback was given during the 40 test pairs. Each session tested only a single algorithm at a time, and participants were only allowed to complete at most one session. A total of 40 participants evaluated each algorithm. To ensure that all algorithms were tested in equivalent conditions (i.e. time of day, demographics, etc.), all experiment sessions were posted simultaneously and distributed to Turkers in an i.i.d. fashion.

To check that participants were competent at this task, 10% of the trials pitted the ground truth image against the Random baseline described above. Participants successfully identified these random colorizations as fake 87% of the time, indicating that they understood the task and were paying attention.

Figure 6 gives a better sense of the participants' competency at detecting subtle errors made by our algorithm. The far right column shows example pairs where participants identified the fake image successfully in 100% of the trials. Each of these pairs was scored by at least 10 participants. Close inspection reveals that on these images, our colorizations tend to have giveaway artifacts, such as the yellow blotches on the two trucks, which ruin otherwise decent results.

Nonetheless, our full algorithm fooled participants on 32% of trials, as shown in Table 1. This number is significantly higher than all compared algorithms ($p < 0.05$ in each case) except for Larsson et al., against which the difference was not significant ($p = 0.10$; all statistics estimated by bootstrap [34]). These results validate the effectiveness of using both a classification loss and class-rebalancing.

Note that if our algorithm exactly reproduced the ground truth colors, the forced choice would be between two identical images, and participants would be fooled 50% of the time on expectation. Interestingly, we can identify cases where participants were fooled *more* often than 50% of the time, indicating our results were deemed more realistic than the ground truth. Some examples are shown in the first three columns of Figure 6. In many case, the ground truth image is poorly white balanced or has unusual colors, whereas our system produces a more prototypical appearance.

2. Semantic interpretability (VGG classification): Does our method produce realistic enough colorizations to be interpretable to an off-the-shelf object classifier? We tested this by feeding our *fake* colorized images to a VGG network [5] that was trained to predict ImageNet classes from *real* color photos. If the classifier performs well, that means the colorizations are accurate enough to be informative about object class. Using an off-the-shelf classifier to assess the realism of synthesized data has been previously suggested by [12].

The results are shown in the second column from the right of Table 1. Classifier performance drops from 68.3% to 52.7% after ablating colors from the input. After re-colorizing using our full method, the performance is improved to 56.0% (other variants of our method achieve slightly higher results). The Larsson et al. [23] method achieves the highest performance on this metric, reaching 59.4%. For reference, a VGG classification network fine-tuned on grayscale inputs reaches a performance of 63.5%.

In addition to serving as a perceptual metric, this analysis demonstrates a practical use for our algorithm: without any additional training or fine-tuning, we can improve performance on grayscale image classification, simply by colorizing images with our algorithm and passing them to an off-the-shelf classifier.

3. Raw accuracy (AuC): As a low-level test, we compute the percentage of predicted pixel colors within a thresholded L2 distance of the ground truth in *ab* color space. We then sweep across thresholds from 0 to 150 to produce a cumulative mass function, as introduced in [22], integrate the area under the curve (AuC), and normalize. Note that this AuC metric measures *raw prediction accuracy*, whereas our method aims for *plausibility*.

Our network, trained on classification without rebalancing, outperforms our L2 variant (when trained from scratch). When the L2 net is instead fine-tuned from a color classification network, it matches the performance of the classification network. This indicates that the L2 metric can achieve accurate colorizations, but has difficulty in optimization from scratch. The Larsson et al. [23] method achieves slightly higher accuracy. Note that this metric is dominated by desaturated pixels, due to the distribution of *ab* values in natural images (Figure 3(b)). As a result, even predicting gray for every pixel does quite well, and our full method with class rebalancing achieves approximately the same score.

Perceptually interesting regions of images, on the other hand, tend to have a distribution of *ab* values with higher values of saturation. As such, we compute a class-balanced variant of the AuC metric by re-weighting the pixels inversely by color class probability (Equation 4, setting $\lambda = 0$). Under this metric, our full method outperforms all variants and compared algorithms, indicating that class-rebalancing in the training objective achieved its desired effect.

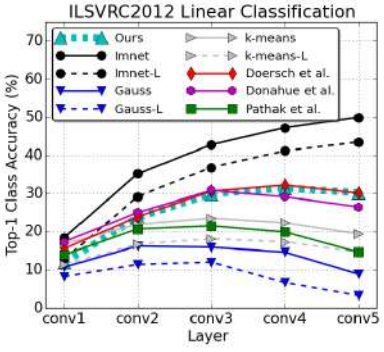

Fig. 7. ImageNet Linear Classification

Dataset and Task Generalization on PASCAL [37]

	Class. (%mAP)				Det. (%mAP)		Seg. (%mIU)	
fine-tune layers	**[Ref]**	**fc8**	**fc6-8**	**all**	**[Ref]**	**all**	**[Ref]**	**all**
ImageNet [38]	-	76.8	78.9	79.9	[36]	56.8	[42]	48.0
Gaussian	[10]	–	–	53.3	[10]	43.4	[10]	19.8
Autoencoder	[16]	24.8	16.0	53.8	[10]	41.9	[10]	25.2
k-means [36]	[16]	32.0	39.2	56.6	[36]	45.6	[16]	32.6
Agrawal et al. [8]	[16]	31.2	31.0	54.2	[36]	43.9	–	–
Wang & Gupta [15]	–	28.1	52.2	58.7	[36]	47.4	–	–
*Doersch et al. [14]	[16]	44.7	55.1	**65.3**	[36]	**51.1**	–	–
*Pathak et al. [10]	[10]	–	–	56.5	[10]	44.5	[10]	29.7
*Donahue et al. [16]	–	38.2	50.2	58.6	[16]	46.2	[16]	34.9
Ours (gray)	–	**52.4**	**61.5**	**65.9**	–	46.1	–	35.0
Ours (color)	–	**52.4**	**61.5**	**65.6**	–	46.9	–	**35.6**

Table 2. PASCAL Tests

Fig. 7. Task Generalization on ImageNet We freeze pre-trained networks and learn linear classifiers on internal layers for ImageNet [28] classification. Features are average-pooled, with equal kernel and stride sizes, until feature dimensionality is below 10k. ImageNet [38], k-means [36], and Gaussian initializations were run with grayscale inputs, shown with dotted lines, as well as color inputs, shown with solid lines. Previous [14,10] and concurrent [16] self-supervision methods are shown.
Tab. 2. Task and Dataset Generalization on PASCAL Classification and detection on PASCAL VOC 2007 [39] and segmentation on PASCAL VOC 2012 [40], using standard mean average precision (mAP) and mean intersection over union (mIU) metrics for each task. We fine-tune our network with grayscale inputs (gray) and color inputs (color). Methods noted with a * only pre-trained a subset of the AlexNet layers. The remaining layers were initialized with [36]. Column **Ref** indicates the source for a value obtained from a previous paper.

3.2 Cross-Channel Encoding as Self-Supervised Feature Learning

In addition to making progress on the graphics task of colorization, we evaluate how colorization can serve as a pretext task for representation learning. Our model is akin to an autoencoder, except that the input and output are different image channels, suggesting the term *cross-channel encoder*.

To evaluate the feature representation learned through this kind of cross-channel encoding, we run two sets of tests on our network. First, we test the *task generalization* capability of the features by fixing the learned representation and training linear classifiers to perform object classification on already seen data (Figure 7). Second, we fine-tune the network on the PASCAL dataset [37] for the tasks of classification, detection, and segmentation. Here, in addition to testing on held-out *tasks*, this group of experiments tests the learned representation on *dataset generalization*. To fairly compare to previous feature learning algorithms, we retrain an AlexNet [38] network on the colorization task, using our full method, for 450k iterations. We find that the resulting learned representation achieves higher performance on object classification and segmentation tasks relative to previous methods tested (Table 2).

ImageNet classification The network was pre-trained to colorize images from the ImageNet dataset, without semantic label information. We test how well

the learned features represent the object-level semantics. To do this, we freeze the weights of the network, provide semantic labels, and train linear classifiers on each convolutional layer. The results are shown in Figure 7.

AlexNet directly trained on ImageNet classification achieves the highest performance, and serves as the ceiling for this test. Random initialization, with Gaussian weights or the k-means scheme implemented in [36], peak in the middle layers. Because our representation is learned on grayscale images, the network is handicapped at the input. To quantify the effect of this loss of information, we fine-tune AlexNet on grayscale image classification, and also run the random initialization schemes on grayscale images. Interestingly, for all three methods, there is a 6% performance gap between color and grayscale inputs, which remains approximately constant throughout the network.

We compare our model to other recent self-supervised methods pre-trained on ImageNet [14,10,16]. To begin, our `conv1` representation results in worse linear classification performance than competing methods [14,16], but is comparable to other methods which have a grayscale input. However, this performance gap is immediately bridged at `conv2`, and our network achieves competitive performance to [14,16] throughout the remainder of the network. This indicates that despite the input handicap, solving the colorization task encourages representations that linearly separate semantic classes in the trained data distribution.

PASCAL classification, detection, and segmentation We test our model on the commonly used self-supervision benchmarks on PASCAL classification, detection, and segmentation, introduced in [14,36,10]. Results are shown in Table 2. Our network achieves strong performance across all three tasks, and state-of-the-art numbers in classification and segmentation. We use the method from [36], which rescales the layers so they "learn" at the same rate. We test our model in two modes: (1) keeping the input grayscale by disregarding color information (Ours (gray)) and (2) modifying `conv1` to receive a full 3-channel *Lab* input, initializing the weights on the *ab* channels to be zero (Ours (color)).

We first test the network on PASCAL VOC 2007 [39] classification, following the protocol in [16]. The network is trained by freezing the representation up to certain points, and fine-tuning the remainder. Note that when `conv1` is frozen, the network is effectively only able to interpret grayscale images. Across all three classification tests, we achieve state-of-the-art accuracy.

We also test detection on PASCAL VOC 2007, using Fast R-CNN [41], following the procedure in [36]. Doersch et al. [14] achieves 51.1%, while we reach 46.9% and 47.9% with grayscale and color inputs, respectively. Our method is well above the strong k-means [36] baseline of 45.6%, but all self-supervised methods still fall short of pre-training with ImageNet semantic supervision, which reaches 56.8%.

Finally, we test semantic segmentation on PASCAL VOC 2012 [40], using the FCN architecture of [42], following the protocol in [10]. Our colorization task shares similarities to the semantic segmentation task, as both are per-pixel classification problems. Our grayscale fine-tuned network achieves performance of 35.0%, approximately equal to Donahue et al. [16], and adding in color information increases performance to 35.6%, above other tested algorithms.

Fig. 8. Applying our method to legacy black and white photos. Left to right: photo by David Fleay of a Thylacine, now extinct, 1936; photo by Ansel Adams of Yosemite; amateur family photo from 1956; *Migrant Mother* by Dorothea Lange, 1936.

3.3 Legacy Black and White Photos

Since our model was trained using "fake" grayscale images generated by stripping *ab* channels from color photos, we also ran our method on real legacy black and white photographs, as shown in Figure 8 (additional results can be viewed on our project webpage). One can see that our model is still able to produce good colorizations, even though the low-level image statistics of the legacy photographs are quite different from those of the modern-day photos on which it was trained.

4 Conclusion

While image colorization is a boutique computer graphics task, it is also an instance of a difficult pixel prediction problem in computer vision. Here we have shown that colorization with a deep CNN and a well-chosen objective function can come closer to producing results indistinguishable from real color photos. Our method not only provides a useful graphics output, but can also be viewed as a pretext task for representation learning. Although only trained to color, our network learns a representation that is surprisingly useful for object classification, detection, and segmentation, performing strongly compared to other self-supervised pre-training methods.

Acknowledgements

This research was supported, in part, by ONR MURI N000141010934, NSF SMA-1514512, an Intel research grant, and a hardware donation by NVIDIA Corp. We thank members of the Berkeley Vision Lab and Aditya Deshpande for helpful discussions, Philipp Krähenbühl and Jeff Donahue for help with self-supervision experiments, and Gustav Larsson for providing images for comparison to [23].

References

1. Cheng, Z., Yang, Q., Sheng, B.: Deep colorization. In: Proceedings of the IEEE International Conference on Computer Vision. (2015) 415–423
2. Dahl, R.: Automatic colorization. In: http://tinyclouds.org/colorize/. (2016)
3. Charpiat, G., Hofmann, M., Schölkopf, B.: Automatic image colorization via multimodal predictions. In: Computer Vision–ECCV 2008. Springer (2008) 126–139
4. Ramanarayanan, G., Ferwerda, J., Walter, B., Bala, K.: Visual equivalence: towards a new standard for image fidelity. ACM Transactions on Graphics (TOG) **26**(3) (2007) 76
5. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
6. Bengio, Y., Courville, A., Vincent, P.: Representation learning: A review and new perspectives. IEEE transactions on pattern analysis and machine intelligence **35**(8) (2013) 1798–1828
7. Ngiam, J., Khosla, A., Kim, M., Nam, J., Lee, H., Ng, A.Y.: Multimodal deep learning. In: Proceedings of the 28th international conference on machine learning (ICML-11). (2011) 689–696
8. Agrawal, P., Carreira, J., Malik, J.: Learning to see by moving. In: Proceedings of the IEEE International Conference on Computer Vision. (2015) 37–45
9. Jayaraman, D., Grauman, K.: Learning image representations tied to ego-motion. In: Proceedings of the IEEE International Conference on Computer Vision. (2015) 1413–1421
10. Pathak, D., Krähenbühl, P., Donahue, J., Darrell, T., Efros, A.: Context encoders: Feature learning by inpainting. In: CVPR. (2016)
11. Lotter, W., Kreiman, G., Cox, D.: Deep predictive coding networks for video prediction and unsupervised learning. arXiv preprint arXiv:1605.08104 (2016)
12. Owens, A., Isola, P., McDermott, J., Torralba, A., Adelson, E.H., Freeman, W.T.: Visually indicated sounds. CVPR (2016)
13. Owens, A., Wu, J., McDermott, J.H., Freeman, W.T., Torralba, A.: Ambient sound provides supervision for visual learning. In: ECCV. (2016)
14. Doersch, C., Gupta, A., Efros, A.A.: Unsupervised visual representation learning by context prediction. In: Proceedings of the IEEE International Conference on Computer Vision. (2015) 1422–1430
15. Wang, X., Gupta, A.: Unsupervised learning of visual representations using videos. In: Proceedings of the IEEE International Conference on Computer Vision. (2015) 2794–2802
16. Donahue, J., Krähenbühl, P., Darrell, T.: Adversarial feature learning. arXiv preprint arXiv:1605.09782 (2016)
17. Hertzmann, A., Jacobs, C.E., Oliver, N., Curless, B., Salesin, D.H.: Image analogies. In: Proceedings of the 28th annual conference on Computer graphics and interactive techniques, ACM (2001) 327–340
18. Welsh, T., Ashikhmin, M., Mueller, K.: Transferring color to greyscale images. ACM Transactions on Graphics (TOG) **21**(3) (2002) 277–280
19. Gupta, R.K., Chia, A.Y.S., Rajan, D., Ng, E.S., Zhiyong, H.: Image colorization using similar images. In: Proceedings of the 20th ACM international conference on Multimedia, ACM (2012) 369–378
20. Liu, X., Wan, L., Qu, Y., Wong, T.T., Lin, S., Leung, C.S., Heng, P.A.: Intrinsic colorization. In: ACM Transactions on Graphics (TOG). Volume 27., ACM (2008) 152

21. Chia, A.Y.S., Zhuo, S., Gupta, R.K., Tai, Y.W., Cho, S.Y., Tan, P., Lin, S.: Semantic colorization with internet images. In: ACM Transactions on Graphics (TOG). Volume 30., ACM (2011) 156
22. Deshpande, A., Rock, J., Forsyth, D.: Learning large-scale automatic image colorization. In: Proceedings of the IEEE International Conference on Computer Vision. (2015) 567–575
23. Larsson, G., Maire, M., Shakhnarovich, G.: Learning representations for automatic colorization. European Conference on Computer Vision (2016)
24. Iizuka, S., Simo-Serra, E., Ishikawa, H.: Let there be Color!: Joint End-to-end Learning of Global and Local Image Priors for Automatic Image Colorization with Simultaneous Classification. ACM Transactions on Graphics (Proc. of SIGGRAPH 2016) **35**(4) (2016)
25. Hariharan, B., Arbeláez, P., Girshick, R., Malik, J.: Hypercolumns for object segmentation and fine-grained localization. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2015) 447–456
26. Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. arXiv preprint arXiv:1606.00915 (2016)
27. Yu, F., Koltun, V.: Multi-scale context aggregation by dilated convolutions. International Conference on Learning Representations (2016)
28. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: Imagenet large scale visual recognition challenge. International Journal of Computer Vision **115**(3) (2015) 211–252
29. Zhou, B., Lapedriza, A., Xiao, J., Torralba, A., Oliva, A.: Learning deep features for scene recognition using places database. In: Advances in neural information processing systems. (2014) 487–495
30. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167 (2015)
31. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531 (2015)
32. Farabet, C., Couprie, C., Najman, L., LeCun, Y.: Learning hierarchical features for scene labeling. Pattern Analysis and Machine Intelligence, IEEE Transactions on **35**(8) (2013) 1915–1929
33. Kirkpatrick, S., Vecchi, M.P., et al.: Optimization by simmulated annealing. science **220**(4598) (1983) 671–680
34. Efron, B.: Bootstrap methods: another look at the jackknife. In: Breakthroughs in Statistics. Springer (1992) 569–593
35. Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., Darrell, T.: Caffe: Convolutional architecture for fast feature embedding. In: Proceedings of the 22nd ACM international conference on Multimedia, ACM (2014) 675–678
36. Krähenbühl, P., Doersch, C., Donahue, J., Darrell, T.: Data-dependent initializations of convolutional neural networks. International Conference on Learning Representations (2016)
37. Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes (voc) challenge. International journal of computer vision **88**(2) (2010) 303–338
38. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems. (2012) 1097–1105
39. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html

40. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html
41. Girshick, R.: Fast r-cnn. In: Proceedings of the IEEE International Conference on Computer Vision. (2015) 1440–1448
42. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2015) 3431–3440
43. Noroozi, M., Favaro, P.: Unsupervised learning of visual representations by solving jigsaw puzzles. arXiv preprint arXiv:1603.09246 (2016)
44. Chakrabarti, A.: Color constancy by learning to predict chromaticity from luminance. In: Advances in Neural Information Processing Systems. (2015) 163–171
45. Xiao, J., Hays, J., Ehinger, K.A., Oliva, A., Torralba, A.: Sun database: Large-scale scene recognition from abbey to zoo. In: Computer vision and pattern recognition (CVPR), 2010 IEEE conference on, IEEE (2010) 3485–3492
46. Ratliff, N.D., Silver, D., Bagnell, J.A.: Learning to search: Functional gradient techniques for imitation learning. Autonomous Robots **27**(1) (2009) 25–53
47. Patterson, G., Hays, J.: Sun attribute database: Discovering, annotating, and recognizing scene attributes. In: Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on, IEEE (2012) 2751–2758

Appendix

The main paper is our ECCV 2016 camera ready submission. All networks were retrained from scratch, and are referred to as the **v2** model. Due to space constraints, we were unable to include many of the analyses presented in our original **arXiv v1** paper. We include these analyses in this Appendix, which were generated from a previous **v1** version of the model. All models are publicly available on our website.

Section 5 contains additional representation learning experiments. Section 6 investigates additional analysis on the VGG semantic interpretability test. In Section 7, we explore how low-level queues affect the output. Section 8 examines the multi-modality learned in the network. Section 9 defines the network architecture used. In Section 10, we compare our algorithm to previous approaches [22] and [1], and show additional examples on legacy grayscale images.

5 Cross-Channel Encoding as Self-Supervised Feature Learning (continued)

In Section 3.2, we discussed using colorization as a pretext task for representation learning. In addition to learning *linear* classifiers on internal layers for ImageNet classifiers, we run the additional experiment of learning *non-linear* classifiers, as proposed in [43]. Each internal layer is frozen, along with all preceding layers, and the layers on top are randomly reinitialized and trained for classification. Performance is summarized in Table 3. Of the unsupervised models, Noroozi et al. [43] have the highest performance across all layers. The architectural modifications result in 5.6× feature map size and 7.35× model run-time, up to the `pool5` layer, relative to an unmodified Alexnet. Of the remaining methods, Donahue et al. [16] performs best at `conv2` and Doersch et al. performs best at `conv3` and `conv4`. Our method performs strongly throughout, and best across methods at the `conv5` layer.

Author	Training	Input	Model			[Ref]	Layers			
			Params	Feats	Runtime		conv2	conv3	conv4	conv5
Krizhevsky et al. [38]	labels	rgb	1.00	1.00	1.00	–	56.5	56.5	56.5	56.5
Krizhevsky et al. [38]	labels	L	0.99	1.00	0.92	–	50.5	50.5	50.5	50.5
Noroozi & Favaro [43]	imagenet	rgb	1.00	5.60	7.35	[43]	56.0	52.4	48.3	38.1
Gaussian	imagenet	rgb	1.00	1.00	1.00	[43]	41.0	34.8	27.1	12.0
Doersch et al. [14]	imagenet	rgb	1.61	1.00	2.82	[43]	47.6	**48.7**	**45.6**	30.4
Wang & Gupta [15]	videos	rgb	1.00	1.00	1.00	[43]	46.9	42.8	38.8	29.8
Donahue et al. [16]	imagenet	rgb	1.00	0.87	0.96	[16]	**51.9**	47.3	41.9	31.1
Ours	imagenet	L	0.99	0.87	0.84	–	46.6	43.5	40.7	**35.2**

Table 3. ImageNet classification with nonlinear layers, as proposed in [43]. Note that some models have architectural differences. We note the effect of these modifications by the number of model parameters, number of features per image, and run-time, as a multiple of Alexnet [38] without modifications, up to the `pool5` layer. Noroozi et al. [43] performs best on all layers, with denser feature maps due to smaller stride in `conv1` layer, along with `LRN` and `pool` ordering switched. Doersch et al. [14] remove groups in `conv` layers. Donahue et al. [16] remove `LRN` layers and change `ReLU` to `leakyReLU` units. Ours removes `LRN` and uses a single channel input. We also note the source of performance numbers. Column **Ref** indicates the source for a value obtained from a previous paper.

6 Semantic Interpretability of Colorizations

In Section 3.1, we investigated using the VGG classifier to evaluate the semantic interpretability of our colorization results. In Section 6.1, we show the categories which perform well, and the ones which perform poorly, using this metric. In Section 6.2, we show commonly confused categories after recolorization.

6.1 Category Performance

In Figure 9, we show a selection of classes that have the most improvement in VGG classification with respect to grayscale, along with the classes for which our colorizations hurt the most. Interestingly, many of the top classes actually have a color in their name, such as the green snake, orange, and goldfinch. The bottom classes show some common errors of our system, such as coloring clothing incorrectly and inconsistently and coloring an animal with a plausible but incorrect color. This analysis was performed using 48k images from the ImageNet validation set, and images in the top and bottom 10 classes are provided on the website.

Our process for sorting categories and images is described below. For each category, we compute the top-5 classification performance on grayscale and recolorized images, $\mathbf{a}_{gray}, \mathbf{a}_{recolor} \in [0,1]^C$, where $C = 1000$ categories. We sort the categories by $\mathbf{a}_{recolor} - \mathbf{a}_{gray}$. The re-colored vs grayscale performance per category is shown in Figure 11(a), with top and bottom 50 categories highlighted. For the top example categories, the individual images are sorted by ascending rank of the correct classification of the recolorizeed image, with tiebreakers on descending rank of the correct classification of the grayscale image. For the bottom example categories, the images are sorted in reverse, in order to highlight the instances when recolorization results in an errant classification relative to the grayscale image.

Fig. 9. Images colorized by our algorithm from selected categories. Categories are sorted by VGG object classification accuracy of our colorized images relative to accuracy on gracyscale images. Top: example categories where our colorization *helps* the most. Bottom: example categories where our colorization *hurts* the most. Number in parentheses indicates category rank amongst all 1000. Notice that the categories most affected by colorization are those for which color information is highly diagnostic, such as birds and fruits. The bottom examples show several kinds of failures: 1) artificial objects such as modems and clothes have ambiguous colors; color is not very informative for classification, and moreover, our algorithm tends to predict an incoherent distribution of red and blue, 2) for certain categories, like the gray fox, our algorithm systematically predicts the wrong color, confusing the species.

6.2 Common Confusions

To further investigate the biases in our system, we look at the common classification confusions that often occur after image recolorization, but not with the original ground truth image. Examples for some top confusions are shown in Figure 10. An image of a "minibus" is often colored yellow, leading to a misclassification as "school bus". Animal classes are sometimes colored differently than ground truth, leading to misclassification to related species. Note that the colorizations are often visually realistic, even though they lead to a misclassification.

To find common confusions, we compute the rate of top-5 confusion $\mathbf{C}_{orig}, \mathbf{C}_{recolor} \in [0,1]^{C \times C}$, with ground truth colors and after recolorization. A value of $\mathbf{C}_{c,d} = 1$ means that every image in category c was classified as category d in the top-5. We find the class-confusion added after recolorization by computing $\mathbf{A} = \mathbf{C}_{recolor} - \mathbf{C}_{orig}$, and sort the off-diagonal entries. Figure 11(b) shows all $C \times (C-1)$ off-diagonal entries of $\mathbf{C}_{recolor}$ vs $\mathbf{C}_{orig}$, with the top 100 entries from $\mathbf{A}$ highlighted. For each category pair (c, d), we extract the images that contained the confusion after recolorization, but

not with the original colorization. We then sort the images in descending order of the classification score of the confused category.

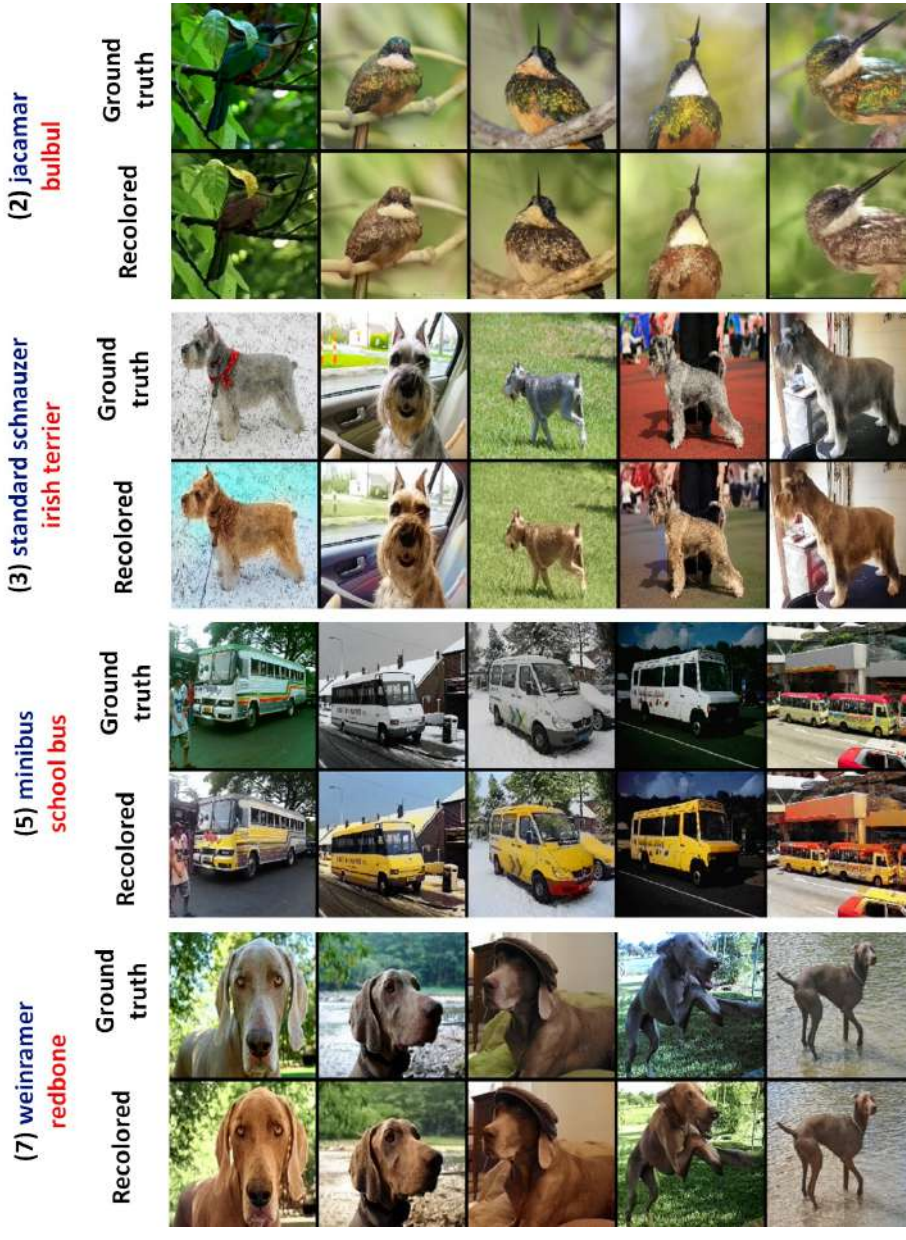

Fig. 10. Examples of some most-confused categories. Top rows show ground truth image. Bottom rows show recolorized images. Rank of common confusion in parentheses. **Ground truth** and **confused** categories after recolorization are labeled.

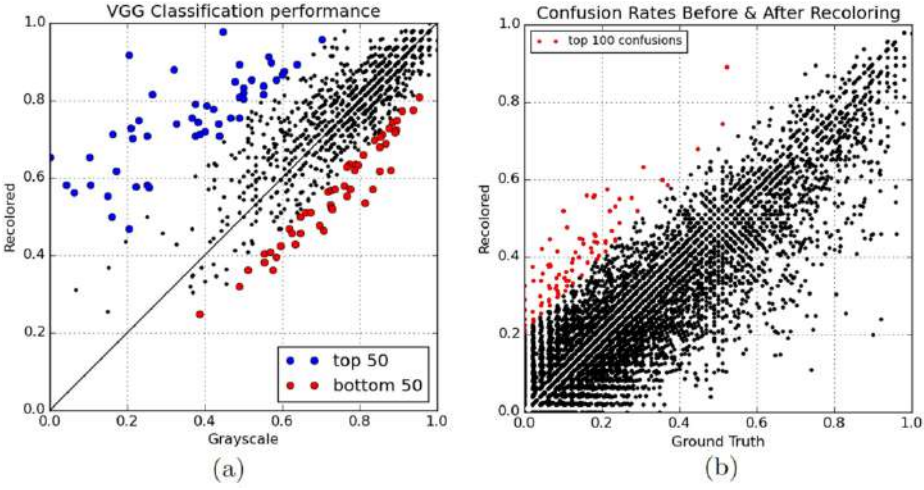

Fig. 11. (a) Performance of VGG top-5 classification on recolorized images vs grayscale images per category (b) Top-5 confusion rates with recolorizations and original colors. Test was done on last 48,000 images in ImageNet validation set.

7 Is the network exploiting low-level cues?

Unlike many computer vision tasks that can be roughly categorized as low, mid or high-level vision, color prediction requires understanding an image at both the pixel and the semantic-level. We have investigated how colorization generalizes to high-level semantic tasks in Section 3.2. Studies of natural image statistics have shown that the lightness value of a single pixel can highly constrain the likely color of that pixel: darker lightness values tend to be correlated with more saturated colors [44].

Could our network be exploiting a simple, low-level relationship like this, in order to predict color?[4] We tested this hypothesis with the simple demonstration in Figure 12. Given a grayscale Macbeth color chart as input, our network was unable to recover its colors. This is true, despite the fact that the lightness values vary considerably for the different color patches in this image. On the other hand, given two recognizable vegetables that are roughly isoluminant, the system is able to recover their color.

In Figure 12, we also demonstrate that the prediction is somewhat stable with respect to low-level lightness and contrast changes. Blurring, on the other hand, has a bigger effect on the predictions in this example, possibly because the operation removes the diagnostic texture pattern of the zucchini.

8 Does our model learn multimodal color distributions?

As discussed in Section 2.1, formulating color prediction as a multinomial classification problem allows the system to predict multimodal distributions, and can capture the inherent ambiguity in the color of natural objects. In Figure 13, we illustrate the

[4] E.g., previous work showed that CNNs can learn to use chromatic aberration cues to predict, given an image patch, its (x,y) location within an image [14].

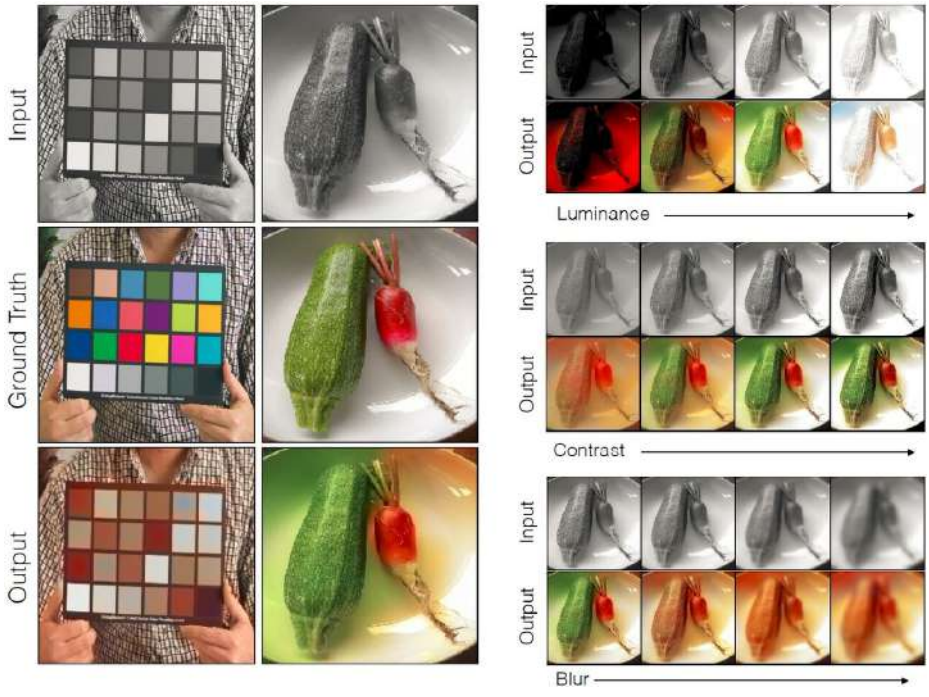

Fig. 12. Left: pixel lightness on its own does not reveal color, as shown by the color chart. In contrast, two vegetables that are nearly isoluminant are recognized as having different colors. Right: stability of the network predictions with respect to low-level image transformations.

probability outputs $\widehat{\mathbf{Z}}$ and demonstrate that the network does indeed learn multimodal distributions. The system output $\widehat{\mathbf{Y}}$ is shown in the top-left of Figure 13. Each block illustrates the probability map $\widehat{\mathbf{Z}}_q \in [0,1]^{H,W}$ given ab bin q in the output space. For clarity, we show a subsampling of the Q total output bins and coarsely quantize the probability values. In Figure 13(a), the system clearly predicts a different distribution for the background vegetation and the foreground bird. The background is predicted to be green, yellow, or brown, while the foreground bird is predicted to be red or blue. Figure 13(b) shows that oranges can be predicted to be different colors. Lastly, in Figure 13(c), the man's sarong is predicted to be either red, pink, or purple, while his shirt is classified as turquoise, cyan or light orange. Note that despite the multimodality of the prediction, taking the annealed-mean of the distribution produces a spatially consistent prediction.

9 Network architecture

Figure 2 showed a diagram of our network architecture. Table 4 in this document thoroughly lists the layers used in our architecture during training time. During testing, the temperature adjustment, softmax, mean, and bilinear upsampling are all implemented as subsequent layers in a feed-forward network. Note the column showing the

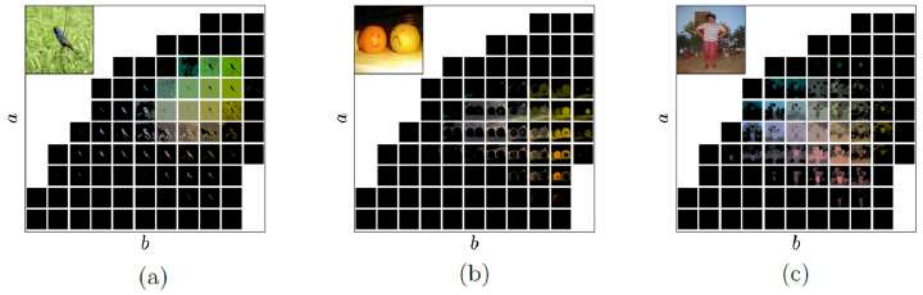

Fig. 13. The output probability distributions per image. The top-left image is final prediction of our system. The black sub-images are quantized blocks of the *ab* gamut. High probabilities are shown as higher luminance and are quantized for clarity. (a) Background of bird is predicted to be green or brown. Foreground bird has distribution across blue and red colors. (b) Oranges are predicted to be different colors. (c) The person's shirt and sarong has uncertainty across turqoise/cyan/orange and red/pink/purple colors, respectively. Note that despite the multimodality of the per-pixel distributions, the results after taking the annealed-mean are typically spatially consistent.

effective dilation. The effective dilation is the spacing at which consecutive elements of the convolutional kernel are evaluated, relative to the input pixels, and is computed by the product of the accumulated stride and the layer dilation. Through each convolutional block from `conv1` to `conv5`, the effective dilation of the convolutional kernel is increased. From `conv6` to `conv8`, the effective dilation is decreased.

10 Colorization comparisons on held-out datasets

10.1 Comparison to LEARCH [22]

Though our model was trained on object-centric ImageNet dataset, we demonstrate that it nonetheless remains effective for photos from the scene-centric SUN dataset [45] selected by Deshpande et al. [22]. Deshpande et al. recently established a benchmark for colorization using a subset of the SUN dataset and reported top results using an algorithm based on LEARCH [46]. Table 5 provides a quantitative comparison of our method to Deshpande et al.. For fair comparison, we use the same grayscale input as [22], which is $\frac{R+G+B}{3}$. Note that this input space is non-linearly related to the L channel on which we trained. Despite differences in grayscale space and training dataset, our method outperforms Deshpande et al. in both the raw accuracy AuC CMF and perceptual realism AMT metrics. Figure 14 shows qualitative comparisons between our method and Deshpande et al., one from each of the six scene categories. A complete comparison on all 240 images are included in the supplementary material. Our results are able to fool participants in the *real vs. fake* task 17.2% of the time, significantly higher than Deshpande et al. at 9.8%.

10.2 Comparison to Deep Colorization [1]

We provide qualitative comparisons to the 23 test images in [1] on the website, which we obtained by manually cropping from the paper. Our results are about the same

Layer	X	C	S	D	Sa	De	BN	L
data	224	3	-	-	-	-	-	-
conv1_1	224	64	1	1	1	1	-	-
conv1_2	112	64	2	1	1	1	✓	-
conv2_1	112	128	1	1	2	2	-	-
conv2_1	56	128	2	1	2	2	✓	-
conv3_1	56	256	1	1	4	4	-	-
conv3_2	56	256	1	1	4	4	-	-
conv3_3	28	256	2	1	4	4	✓	-
conv4_1	28	512	1	1	8	8	-	-
conv4_2	28	512	1	1	8	8	-	-
conv4_3	28	512	1	1	8	8	✓	-
conv5_1	28	512	1	2	8	16	-	-
conv5_2	28	512	1	2	8	16	-	-
conv5_3	28	512	1	2	8	16	✓	-
conv6_1	28	512	1	2	8	16	-	-
conv6_2	28	512	1	2	8	16	-	-
conv6_3	28	512	1	2	8	16	✓	-
conv7_1	28	256	1	1	8	8	-	-
conv7_2	28	256	1	1	8	8	-	-
conv7_3	28	256	1	1	8	8	✓	-
conv8_1	56	128	.5	1	4	4	-	-
conv8_2	56	128	1	1	4	4	-	-
conv8_3	56	128	1	1	4	4	-	✓

Table 4. Our network architecture. **X** spatial resolution of output, **C** number of channels of output; **S** computation stride, values greater than 1 indicate downsampling following convolution, values less than 1 indicate upsampling preceding convolution; **D** kernel dilation; **Sa** accumulated stride across all preceding layers (product over all strides in previous layers); **De** effective dilation of the layer with respect to the input (layer dilation times accumulated stride); **BN** whether `BatchNorm` layer was used after layer; **L** whether a 1x1 `conv` and cross-entropy loss layer was imposed

qualitative level as [1]. Note that Deep Colorization [1] has several advantages in this setting: (1) the test images are from the SUN dataset [47], which we did not train on and (2) the 23 images were hand-selected from 1344 by the authors, and is not necessarily representative of algorithm performance. We were unable to obtain the 1344 test set results through correspondence with the authors.

Additionally, we compare the methods on several important dimensions in Table 6: algorithm pipeline, learning, dataset, and run-time. Our method is faster, straightforward to train and understand, has fewer hand-tuned parameters and components, and has been demonstrated on a broader and more diverse set of test images than Deep Colorization [1].

10.3 Additional Examples on Legacy Grayscale Images

Here, we show additional qualitative examples of applying our model to legacy black and white photographs. Figures 16, 17, and 18 show examples including work of renowned photographers, such as Ansel Adams and Henri Cartier-Bresson, photographs of politicians and celebrities, and old family photos. One can see that our model is often able to produce good colorizations, even though the low-level image statistics of old legacy photographs are quite different from those of modern-day photos.

Results on LEARCH [22] dataset

Algorithm	AuC CMF (%)	AMT Labeled Real (%)
Ours	**90.1**	**17.2±1.9**
Deshpande et al. [22]	88.8	9.8±1.5
Grayscale	89.3	–
Ground Truth	100	50

Table 5. Results on LEARCH [22] test set, containing 240 images from 6 categories *beach, outdoor, castle, bedroom, kitchen,* and *living room.* Results column 1 shows the AuC of thresholded CMF over *ab* space. Results column 2 are from our AMT real vs. fake test.

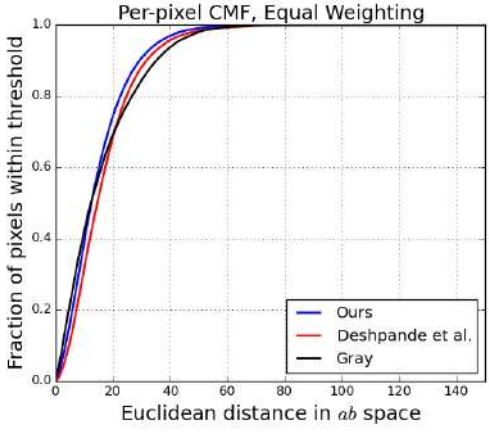

Fig. 14. CMF on the LEARCH [22] test set

	Deep Colorization [1]	**Ours**
Algorithm	(1) Extract feature sets (a) 7x7 patch (b) DAISY (c) FCN on 47 categories (2) 3-layer NN regressor (3) Joint-bilateral filter	Feed-forward CNN
Learning	Extract features. Train FCN [42] on pre-defined categories. Train 3-layer NN regressor.	Train CNN from pixels to color distribution. Tune single parameter on validation.
Dataset	2688/1344 images from SUN [47] for train/test. Limited variety with only scenes.	1.3M/10k images from ImageNet [28] for train/test. Broad and diverse set of objects and scenes.
Run-time	4.9s/image on Matlab implementation	21.1ms/image in *Caffe* on K40 GPU

Table 6. Comparison to Deep Colorization [1]

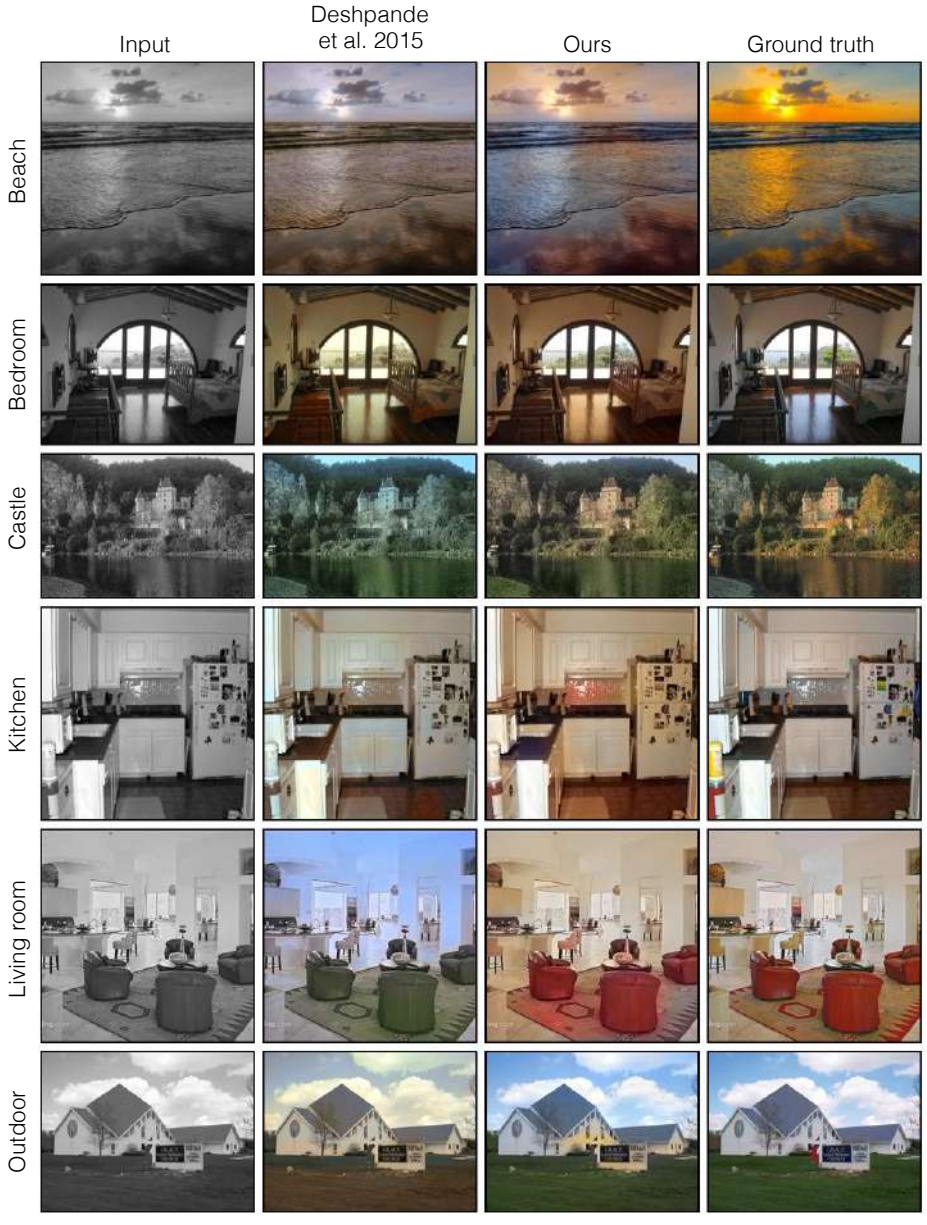

Fig. 15. Our model generalizes well to datasets on which it was not trained. Here we show results on the dataset from [22], which consists of six scene categories from SUN [45]. Compared to the state of the art algorithm on this dataset [22], our method produces more perceptually plausible colorization (see also Table 5 and Figure 14). Please visit `http://richzhang.github.io/colorization/` to see the results on all 240 images.

Fig. 16. Applying our method to black and white photographs by Ansel Adams.

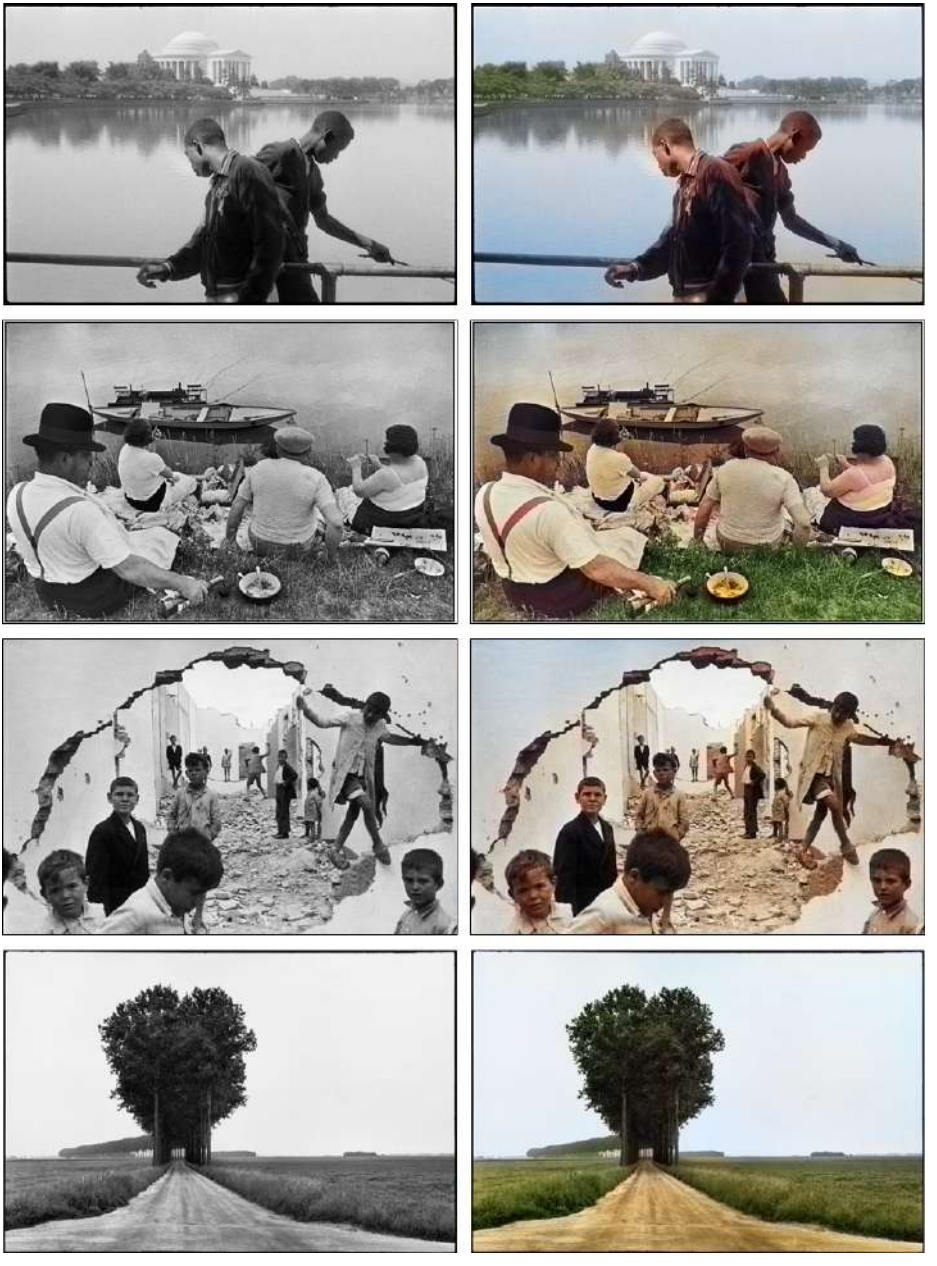

Fig. 17. Applying our method to black and white photographs by Henri Cartier-Bresson.

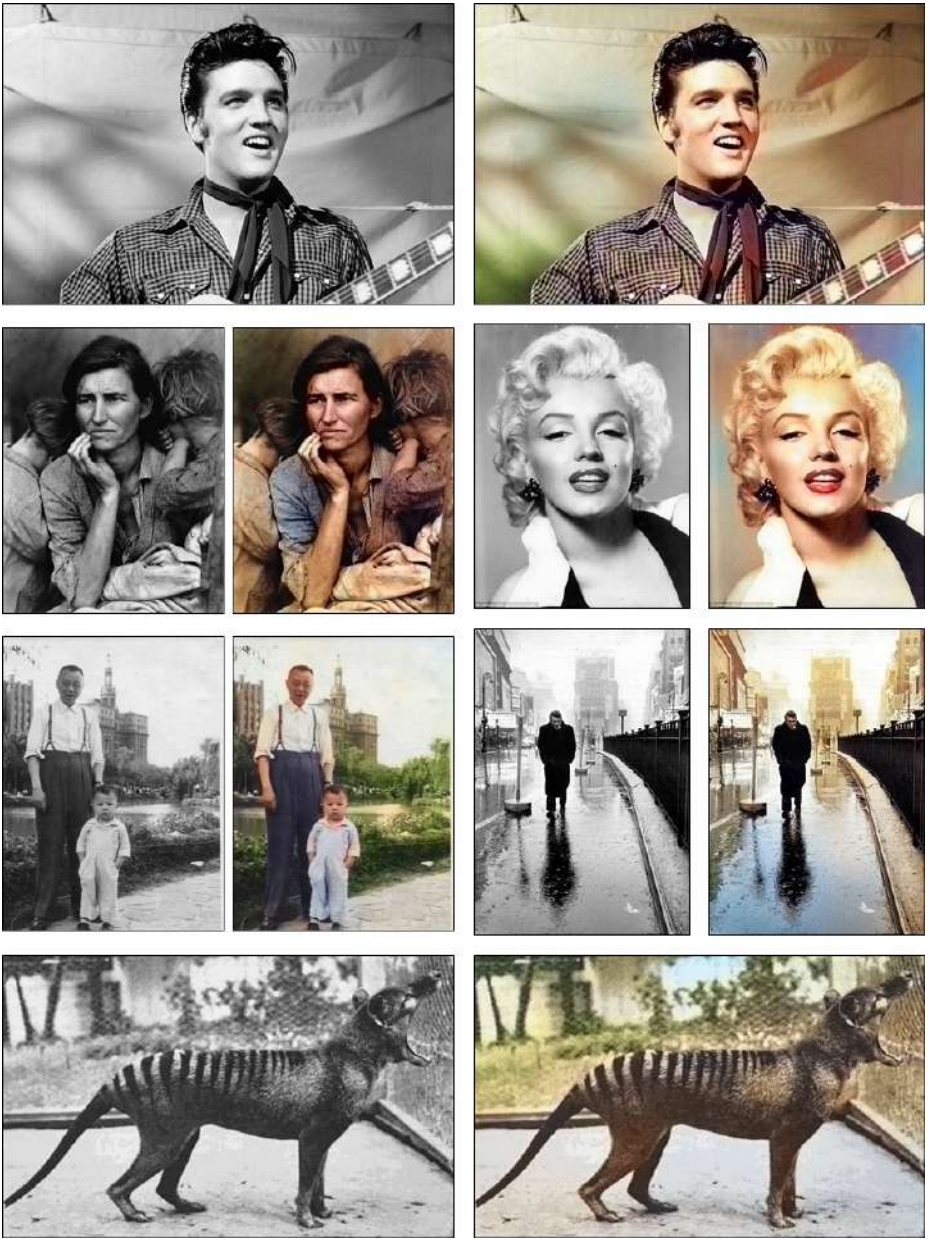

Fig. 18. Applying our method to legacy black and white photographs. Top to bottom, left to right: photo of Elvis Presley, photo of *Migrant Mother* by Dorothea Lange, photo of Marilyn Monroe, an amateur family photo, photo by Henri Cartier-Bresson, photo by Dr. David Fleay of *Benjamin*, the last captive thylacine which went extinct in 1936.