Colorful Image Colorization

Richard Zhang, Phillip Isola(✉), and Alexei A. Efros

University of California, Berkeley, USA
{rich.zhang,isola,efros}@eecs.berkeley.edu

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 Fig. 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.

B. Leibe et al. (Eds.): ECCV 2016, Part III, LNCS 9907, pp. 649–666, 2016.
DOI: 10.1007/978-3-319-46487-9_40

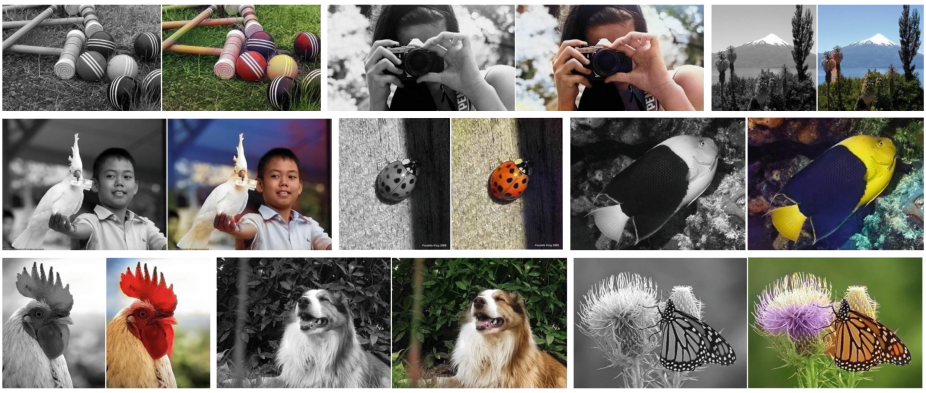

Fig. 1. Example input grayscale photos and output colorizations from our algorithm. These examples are cases where our model works especially well. Please visit http://richzhang.github.io/colorization/ to see the full range of results and to try our model and code. Best viewed in color (obviously). (Color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

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, 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 Fig. 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., [7–13]). 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 [8,10,14,15] 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) [18–21]. 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 [1,2,22] 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 Sect. 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

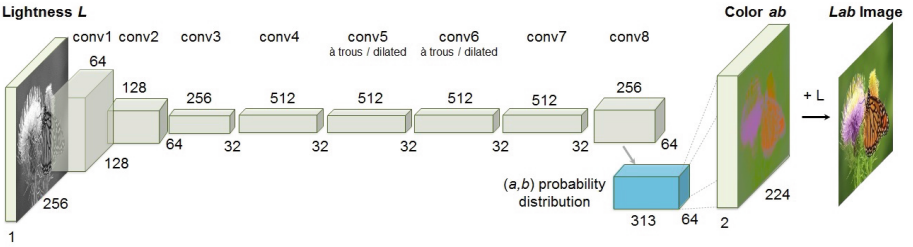

Fig. 2. Our network architecture. Each `conv` layer refers to a block of 2 or 3 repeated `conv` and `ReLU` layers, followed by a `BatchNorm` [30] layer. The net has no `pool` layers. All changes in resolution are achieved through spatial downsampling or upsampling between `conv` blocks. (Color figure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

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 [29]. In Sect. 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 Fig. 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.

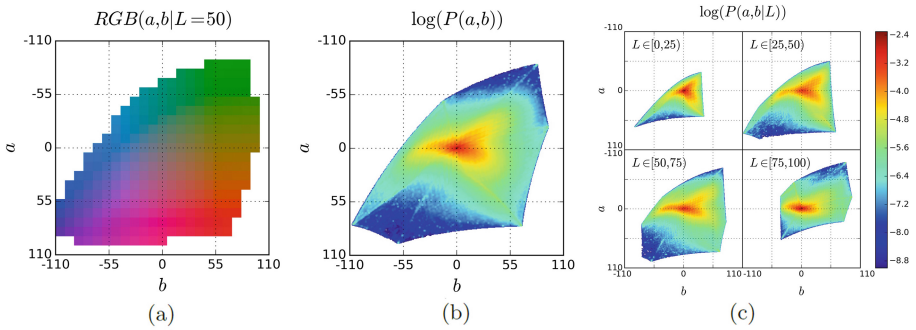

Fig. 3. (a) Quantized ab color space with a grid size of 10. A total of 313 ab pairs are in gamut. (b) Empirical probability distribution of ab values, shown in log scale. (c) Empirical probability distribution of ab values, conditioned on L, shown in log scale. (Color figure online)

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 Fig. 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 $\widehat{\mathbf{Z}} \in [0, 1]^{HxWxQ}$, 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[1]. 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 Sect. 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 Sect. 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

[1] 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 $\sigma = 5$.

number of pixels in natural images at desaturated values are orders of magnitude higher than for saturated values. Without accounting for this, the 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 Sect. 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 Fig. 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 Fig. 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 Fig. 4, captures the vibrancy of the mode while maintaining the spatial coherence of the mean.

Fig. 4. The effect of temperature parameter T on the *annealed-mean* output (Eq. 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. (Color figure online)

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.

3 Experiments

In Sect. 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 Sect. 3.2, we test colorization as a method for self-supervised representation learning. Finally, in Sect. 3.3, we show qualitative examples on legacy black and white images.

3.1 Evaluating Colorization Quality

We train our network on the 1.3 M images from the ImageNet training set [28], validate on the first 10 k images in the ImageNet validation set, and test on a separate 10 k 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 Fig. 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:

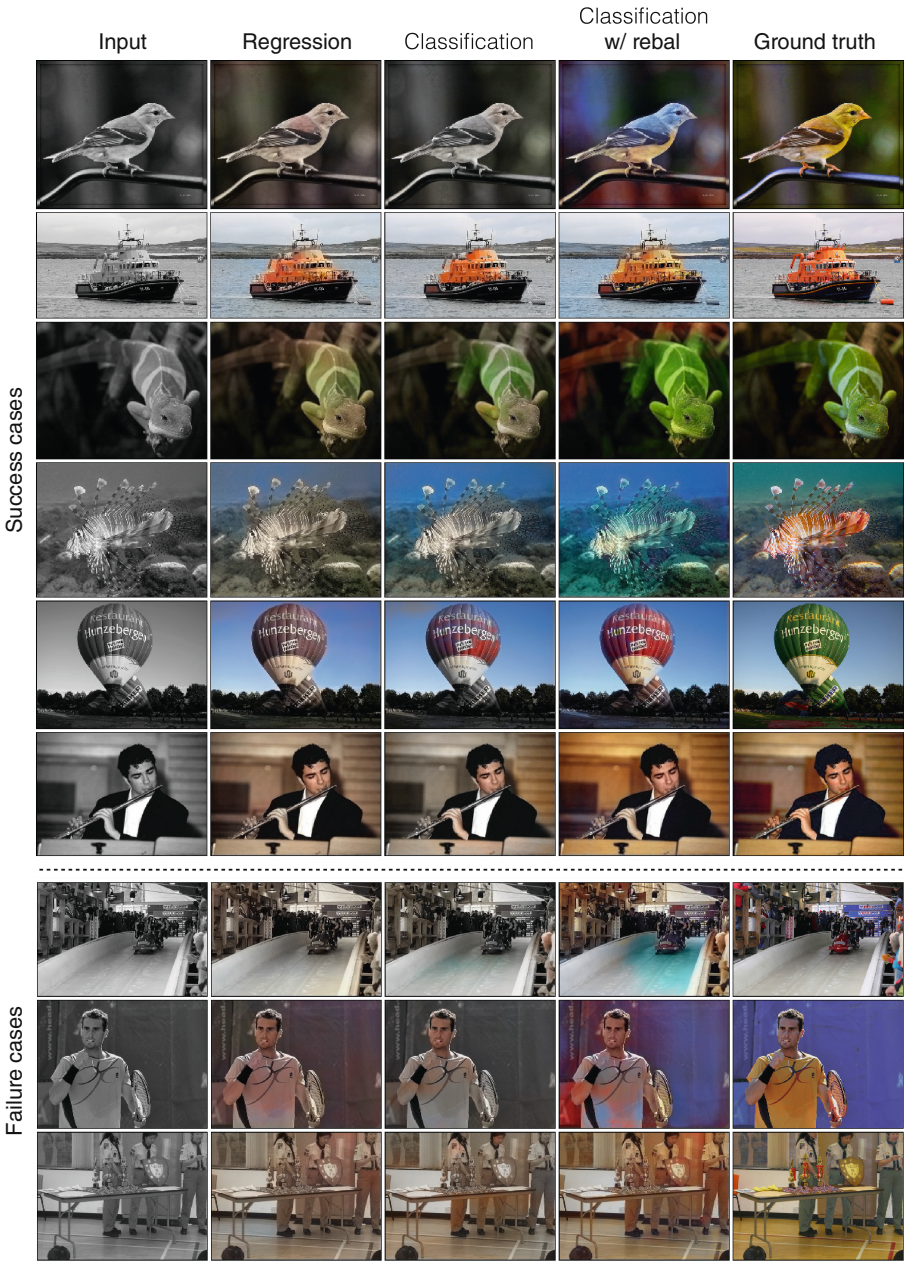

Fig. 5. Example results from our ImageNet test set. Our classification loss with rebalancing produces more accurate and vibrant results than a regression loss or a classification loss without rebalancing. Successful colorizations are above the dotted line. Common failures are below. These include failure 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. (Color figure online)

1. **O**urs (full) Our full method, with classification loss, defined in Eq. 2, and class rebalancing, as described in Sect. 2.2. The network was trained from scratch with k-means initialization [36], using the ADAM solver for approximately 450 k iterations[2].
2. **O**urs (class) Our network on classification loss but no class rebalancing ($\lambda = 1$ in Eq. 4).
3. **O**urs (L2) Our network trained from scratch, with L2 regression loss, described in Eq. 1, following the same training protocol.
4. **O**urs (L2, ft) Our network trained with L2 regression loss, fine-tuned from our full classification with rebalancing network.
5. **L**arsson et al. [23] A CNN method that also appears in these proceedings.
6. **D**ahl[2] A previous model using a Laplacian pyramid on VGG features, trained with L2 regression loss.
7. **G**ray Colors every pixel gray, with $(a, b) = 0$.
8. **R**andom 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 (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.

[2] $\beta_1 = .9$, $\beta_2 = .99$, 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.

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].

Colorization results on ImageNet							
Method	Model			AuC		VGG Top-1 Class Acc (%)	AMT labeled real (%)
	Params (MB)	Feats (MB)	Runtime (ms)	non-rebal (%)	rebal (%)		
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**

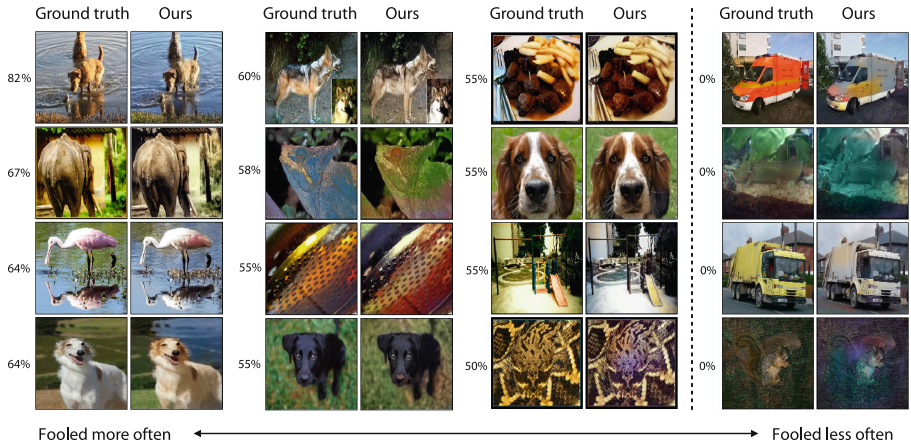

Fig. 6. Images sorted by how often AMT participants chose our algorithm's colorization over the ground truth. In all pairs to the left of the dotted line, participants believed our colorizations to be more real than the ground truth on $\geq 50\,\%$ of the trials. In some cases, this may be due to poor 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.

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 Fig. 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 net-

work [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 (Fig. 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 (Eq. 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.

Figure 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 [10,14] and concurrent [16] self-supervision methods are shown.

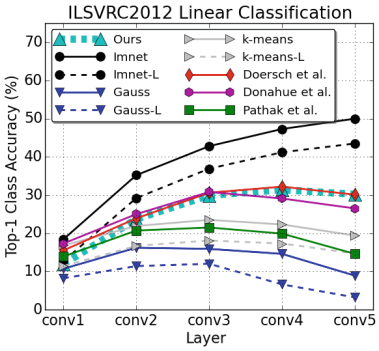

Fig. 7. ImageNet Linear Classification

Table 2. PASCAL Tests.

Dataset and task generalization on PASCAL [37]					
	Classification			Detection	Segmentation
	(% mAP)			(% mAP)	(% mIU)
Fine-tuned layers	fc8	fc6-fc8	All	All	All
ImageNet [38]	76.8	78.9	79.9	56.8	48.0
Gaussian	–	–	53.3	43.4	19.8
Autoencoder	24.8	16.0	53.8	41.9	25.2
k-means [38]	32.0	39.2	56.6	45.6	32.6
Agrawal et al. [8]	31.2	31.0	54.2	43.9	–
Wang & Gupta [15]	28.1	52.2	58.7	44.0	–
*Doersch et al. [14]	44.7	55.1	**65.3**	**51.1**	–
*Pathak et al. [10]	–	–	56.5	44.5	29.7
*Donahue et al. [16]	38.2	50.2	58.6	45.1	34.9
Ours (gray)	**52.4**	**61.5**	**65.9**	46.9	35.0
Ours (color)	**52.4**	**61.5**	**65.6**	47.9	**35.6**

Table 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].

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 (Fig. 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 450 k 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 Fig. 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 [10,14,16]. To begin, our `conv1` representation results in worse linear classification performance than competiting 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 [10,14,36]. 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 51.5 %, 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 Fig. 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, pp. 415–423 (2015)
2. Dahl, R.: Automatic colorization (2016). http://tinyclouds.org/colorize/.
3. Charpiat, G., Hofmann, M., Schölkopf, B.: Automatic image colorization via multimodal predictions. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part III. LNCS, vol. 5304, pp. 126–139. Springer, Heidelberg (2008)
4. Ramanarayanan, G., Ferwerda, J., Walter, B., Bala, K.: Visual equivalence: towards a new standard for image fidelity. ACM Trans. Graph. (TOG) **26**(3), 76 (2007)
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 Trans. Pattern Anal. Mach. Intell. **35**(8), 1798–1828 (2013)
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 2011), pp. 689–696 (2011)
8. Agrawal, P., Carreira, J., Malik, J.: Learning to see by moving. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 37–45 (2015)
9. Jayaraman, D., Grauman, K.: Learning image representations tied to ego-motion. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1413–1421 (2015)
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. In: 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, pp. 1422–1430 (2015)
15. Wang, X., Gupta, A.: Unsupervised learning of visual representations using videos. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2794–2802 (2015)
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, pp. 327–340. ACM (2001)
18. Welsh, T., Ashikhmin, M., Mueller, K.: Transferring color to greyscale images. ACM Trans. Graph. (TOG) **21**(3), 277–280 (2002)
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, pp. 369–378. ACM (2012)
20. Liu, X., Wan, L., Qu, Y., Wong, T.T., Lin, S., Leung, C.S., Heng, P.A.: Intrinsic colorization. ACM Trans. Graph. (TOG) **27**, 152 (2008). ACM

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. ACM Trans. Graph. (TOG) **30**, 156 (2011). ACM
22. Deshpande, A., Rock, J., Forsyth, D.: Learning large-scale automatic image colorization. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 567–575 (2015)
23. Larsson, G., Maire, M., Shakhnarovich, G.: Learning representations for automatic colorization. In: 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 Trans. Graph. (Proc. SIGGRAPH 2016) **35**(4), 110 (2016). Kindly check and confirm if the inserted page range is correct for Ref. [24]
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, pp. 447–456 (2015)
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. In: 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. Int. J. Comput. Vis. **115**(3), 211–252 (2015)
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, pp. 487–495 (2014)
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. IEEE Trans. Pattern Anal. Mach. Intell. **35**(8), 1915–1929 (2013)
33. Kirkpatrick, S., Vecchi, M.P., et al.: Optimization by simmulated annealing. Science **220**(4598), 671–680 (1983)
34. Efron, B.: Bootstrap methods: another look at the jackknife. In: Kotz, S., Johnson, N.L. (eds.) Breakthroughs in Statistics, pp. 569–593. Springer, Heidelberg (1992)
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, pp. 675–678. ACM (2014)
36. Krähenbühl, P., Doersch, C., Donahue, J., Darrell, T.: Data-dependent initializations of convolutional neural networks. In: 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. Int. J. Comput. Vis. **88**(2), 303–338 (2010)
38. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, pp. 1097–1105 (2012)
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, pp. 1440–1448 (2015)
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, pp. 3431–3440 (2015)