ColorNN Book: A Recurrent-Inspired Deep Learning Approach to Consistent Video Colorization

Divyahans Gupta
Stanford University
Stanford, California 94305
dgupta2@stanford.edu

Sanjay Kannan
Stanford University
Stanford, California 94305
skalon@stanford.edu

ABSTRACT

Can computers add meaningful color to black-and-white videos? While deep neural networks have achieved state-of-the-art performance on coloring still images, naive extensions to video have proven unsatisfactory. The frames of a video are neither independent of one another, nor should they be colored in this way. In this paper, we use the temporal context of video frames to enhance the subjective visual quality of their colorings.

Author Keywords

Video Colorization, Transfer Learning, Convolutional Neural Networks, Recurrent Neural Networks

1. INTRODUCTION

The image colorization problem has been the subject of much interest in the research community. Specifically, we define the problem as follows: Given a grayscale representation of a digital image, what is the most intelligent assignment of colors to grayscale pixels? (See Figure 1 for an instance of this task.) Absent a reference coloring, there is clearly no answer that is objectively the most intelligent.

Figure 1: The original and colorized versions of Dorothea Lange's *Migrant Mother* (1936), taken in California during the Great Depression.

However, some colorings (green trees and blue skies, for instance) carry a degree of inherent plausibility due to the consistency of their texture or structure. We intuit this judgment based on our experiences seeing color in the real world, suggesting that computers can do the same. We can imagine that a computer, given a sufficiently large corpus of colored images, would learn an internal concept of plausible colorings.

In fact, recent advances in state-of-the-art deep neural networks (accompanied by the relevant gains in computing power) are making this approach to visual intelligence not only possible, but commonplace. Convolutional neural networks (CNNs, or ConvNets) have contributed in large part to this phenomenon; their sparse, localized structure presents benefits both for increasing computational feasibility as well as capturing the latent geometric structures in an image.

The yearly ImageNet Large Scale Visual Recognition Challenge (ILSVRC) has provided a perennial high-water mark on the related task of image classification. ConvNets have become a staple of winning submissions, following the record-shattering work of Krizhevsky et al. [1] in 2012.

Our work begins by surveying the landscape of ConvNets as they pertain to the image colorization problem. We then draw on the ideas of a few other papers in developing our own neural network for this purpose. Finally, we turn our attention to a natural extension of the image problem: consistent video colorization.

Thus far, efforts to colorize video have been marked by a per-frame approach, independently colorizing the still shots in a filmed sequence. While this approach is theoretically valid, current image colorization networks are far from perfect, and even minor deviations in color between frames appear jarring in tandem to viewers.

Furthermore, many scenes have a number of plausible colorings. A T-Shirt might be red, green, blue, or any number of other colors. In a filmed sequence, however, a T-Shirt in one frame is probably the same T-Shirt in the next dozen frames.

To this end, we note that each frame of video does not exist in a vacuum, nor should its colorization. Given a plausible frame coloring at some time t, any coloring of frame $t+1$ should ideally maintain visual consistency with its predecessor. Informally, the same objects should have the same colors between two frames, even if these objects were somehow displaced or transformed in between.

This temporal dependency suggests the use of recurrent neural network (RNN) architectures as a modeling tool. In recent times, Hochreiter and Schmidhuber's long short-term memory model (LSTM) [2] has emerged as a robust way to implement recurrent connections in a neural network. As its name suggests, it is sensitive to both short-term correlations in sequential data (e.g. frame t and frame $t+1$) as well as longer-term properties.

Even without an explicitly recurrent network architecture, we can still design simpler coloring networks to require a running video context as input. This approach results in models with many fewer trainable parameters, making development and iteration computationally tractable.

Our contribution to this area of research is progress on the problem of automated video coloring. Here, we present an empirical analysis of a recurrent-inspired convolutional neural network, and assess its performance on a selected corpus of media.

2. RELATED WORK

In practice, it is difficult to build a sufficiently complex network architecture for image-to-image transformation. Such a network would need to first understand the diversity of macro-level concepts in an image, and then on top of that propagate per-pixel inputs to generate a locally-sensitive output.

Rather than approach this task directly, then, several papers on the topic have employed *transfer learning*. Intuitively, well-studied tasks like image classification often share overlapping sub-problems with less-popular tasks like image coloring. These include forming a coherent internal representation of an image that is high in information and low in redundancy.

Since leading image classification networks have weights and architectures for public download, it is possible to co-opt pre-trained layers from these networks for external use. In a blog post from January 2016, Dahl [3] extracts *hypercolumns* from the VGG-16 network, and uses their concatenated outputs as inputs to his coloring network. (Hypercolumns are simply the activations above each input pixel in the layers of a CNN; see [4] for more details.) VGG-16 [5], a top contender in ILSVRC 2014, was chosen in this case for its powerful yet simple representation of images.

However, the most successful colorization algorithms to date have not used transfer learning, instead opting to conduct end-to-end training on large image repositories (the most common of which is ImageNet). With total control of their architectures, these approaches design each layer and train each parameter to be coloring-specific.

Most recently, Zhang et al. [6] paired their CNN with a class rebalancing scheme that weights rarer and more vibrant pixel colors more heavily in an effort to avoid desaturation. Meanwhile, Iizuka et al. [7] recently demonstrated state-of-the-art results with a CNN that fuses local as well as global image data for coloring. Both networks were trained from scratch: the former on ImageNet, and the latter on MIT's Places2. Notably, Zhang et al. computed classification by quantizing the color space, while Iizuka et al. computed a regression loss.

Despite the recent progress in image colorization, video coloring remains an open problem. Zhang et al. used their network to color several videos with a frame-by-frame procedure, and these in fact suffer from the color inconsistencies we describe. While applying lessons from image colorization attempts, we endeavor to solve these inconsistencies using a custom recurrent-inspired architecture.

3. METHOD

We first describe a *color model*, which operates exclusively on still images with no temporal context. While this model is wholly self-contained, we subsequently address a superset of this model, which conditions a frame's coloring on the colorings of predecessor frames (the *consistency model*).

3.1 Color Model

Many of the models described above are not true image-to-image networks, in the sense that they do not take a grayscale image and output an RGB-colorized image in their last layer. Rather, they typically use a perceptual color space like YUV [8], which separates color into a brightness Y (or *luma*) channel, and two (U, V) color channels.

Conceptually, this choice has two advantages. First, when plotted in the (U, V) color plane, the Euclidean distance between any two colors approximates their perceptual similarity. As a result, color prediction can be reasonably framed as a regression problem.

Figure 2: Taken from our test set, the top image is the ground truth for the given scene. The output of the color model is displayed below. While our result is not quite as vivid as the original, we manage to find a pretty plausible coloring. We even preserve the mountain's reflection in the lake.

Figure 3: We also experimented with training on larger data sets, including substantial amounts of foliage. The top image is the ground truth for the given scene, while the bottom image is our color model's output. This example demonstrates the ambiguities inherent in coloring images. Both red and green are valid colors for foliage textures when no additional context is present. In this case, our model has probably seen more instances of green foliage.

Second, the separation of lightness and color channels suggests a model's interface. The Y channel becomes a model's input layer, while the model aims to predict U and V. We frame our problem in precisely this way. Final reconstruction of an image happens outside of the model, by concatenating predicted color channels with known brightness values.

Next, we also chose to employ transfer learning. Our resources made it computationally infeasible to train end-to-end networks on large datasets, as in Zhang et al. and Iizuka et al. Specifically, we constructed an architecture (inspired by Dahl) that inherits layers from a VGG-16 model trained on ImageNet (see Figure 4). Like Dahl, we chose VGG-16 due to its simplicity and success in ILSVRC 2014.

Even though VGG-16 was trained on color images, we hypothesized that its convolutional layers would maintain most of their activation behavior on grayscale images. The discriminatory properties of VGG-16's layers are utilized by our added layers to apply appropriate colorings more quickly and accurately than if we trained a model from scratch. For example, a convolutional layer in VGG-16 that discriminates foliage would help our added layers learn to color the image green. We hypothesized that VGG-16 might drastically decrease the training time needed to obtain realistic colorings.

As in Dahl's case, our architecture forms residual connections between the convolutional layers from VGG-16, and the convolutional layers we added above the model. A notable discrepancy between the architectures lies in our final layers. While Dahl's model outputs a $224 \times 224 \times 2$ tensor and computes a Euclidean distance loss in the U and V color axes, we compute per-pixel Softmax probabilities over n color bins per axis. We then track the summed categorical cross-entropy loss based on the correct bin for each pixel.

While we experimented with regression-based models, we ultimately framed the problem as a classification task. Using a regression loss like the Euclidean norm might yield desaturated colorings, as explained in Zhang et al. Furthermore, according to Karpathy's deep learning notes from Stanford CS231N [9], classification losses are easier to optimize for, and are much more stable compared to regression losses. Empirical and visual tests between a regression model and a classification model further validated this decision.

Finally, we limited the scope of our dataset to water features (coasts, lakes, and other seascapes). While we would have preferred to train our model on larger, more diverse datasets such as ImageNet, the computational resources necessary to do so were out of our reach. The images in our dataset were obtained from the McGill Calibrated Colour Image Database, MIT CVCL's Urban and Natural Scene Categories, and MIT CSAIL's SUN data set. It is our hope that this paper will encourage further work with greater computational resources unavailable to us.

3.2 Consistency Model

Having developed a working color model, we needed some way to inform our network of prior context in a sequence of frames. We looked first to the canonical modeling tool for this purpose: recurrent neural networks.

Unlike a traditional feed-forward neural network, the neurons in a recurrent network preserve a selective temporal memory, and output different things based on the input ordering of examples. For instance, knowing that a pixel (x, y) was colored red in the recent past, a correctly-designed recurrent color model would be more likely to color the same pixel red in subsequent frames (or not necessarily, if the model determines that frames are changing quickly).

Intuitively, we should be able to make the pixel neurons in our convolutional layers stateful, and then pass in an ordered sequence of frames for temporal context. Unfortunately for training, making every neuron in a 2D convolutional layer recurrent blows up the number of parameters.

There is a small but growing body of literature on dealing with this issue in the context of image-to-image models, but actually solving this problem was beyond the scope of our research.

Instead of using recurrent nodes in our architecture, we opted to explicitly pass in our knowledge of previous frames when coloring the still image at a given time step. More concretely, our consistency model runs the Y-channel input of a frame t through the color model, and takes in the estimated (U', V') coloring of the previous frame t' as additional input.

Next, we take the final convolutional layers from running the color model on frame t, and concatenate them with a convolution of channels U' and V' from frame t'. We apply two convolutional layers to this concatenation, and finally calculate binned Softmax probabilities as in the color model.

In this way, the consistency model presented here is a sort of truncated RNN. We use the previous frame's color context to inform the coloring of the current frame, rather than a weighted context over several prior frames.

As our results indicate, even a hint of context not only improves consistency between frames, but also, given an accurate coloring of the previous frame, corrects for a weaker color model. Our color model is hardly competitive, but our results improved drastically when we fed the ground truth coloring of the prior frame as context.

We should note that our frames were captured at a rate of 30 frames per second (FPS). At this frequency, the differences between subsequent frames can be minimal, and our model risks learning an identity mapping of the previous frame's coloring. Further work is required to explore this phenomenon, but we hypothesize that a slightly lower frame rate could preserve temporal consistency, while learning more nuanced mappings of time.

4. RESULTS

Here, we present results for the color model and the overall consistency model separately.

4.1 Color Model

Our color model was built in Keras, running on top of a Theano backend. Keras comes with weights for the VGG-16 network trained on ImageNet, so we loaded these into the appropriate layers of our model. At training time, these parameters were marked as frozen, and losses were not back-propagated to the VGG-16 layers.

We used our color model with 4000 images of water features (see Methods for more specific details). Of these, 3000 were assigned to our training set, 500 were assigned to a validation set, and 500 were assigned to a test set.

We quantized the U and V color spaces into $n = 30$ discrete bins. We employed the state-of-the-art *Adam* optimizer and ran our training for over 100 epochs with a batch size of 15. However, our model began to overfit after epoch 16, so we report the data from that epoch.

In the end, our model achieved a total training loss of 3.384. On the training set, we correctly classified 37.362 percent of pixels in the binned U dimension, and we correctly classified 33.908 percent of pixels in the binned V dimension.

Meanwhile, we achieved an final test loss of 3.541. On the test set, we correctly classified 36.198 percent of pixels in the binned U dimension, and we correctly classified 32.474

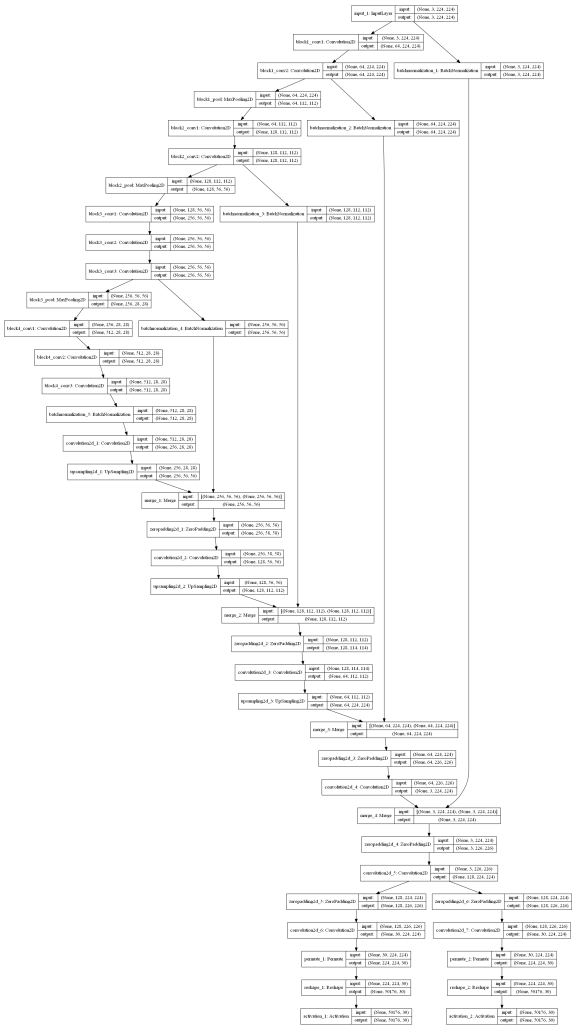

Figure 4: Our color model architecture. The nodes at the top left of the figure represent the VGG-16 network, which is connected at various convolutional layers to our network.

Figure 5: The bottom image is a ground truth frame from the video test set. In order, the top image is its grayscale representation, the second image is the output of our color model (with no temporal context attached), and the third image is the output of our consistency model (with color context from the previous frame). The consistency model preserves the ocean's color between this frame and its predecessor, resulting in a more faithful coloring.

percent of pixels in the binned V dimension. See Figures 2 and 3 for typical outputs of our model.

4.2 Consistency Model

Since the consistency model uses the color model as a subcomponent, we pre-trained a color model and then froze its weights for use in the consistency model. (This is similar to what we did before with VGG-16.)

We used our consistency model with 10 videos of coastal scenes, each with 300 frames (10 seconds at 30 FPS). Of these, 7 videos went to our training set, 3 went to a validation set, and 3 went to a test set.

After only ten epochs, we achieved an final test loss of 3.241. On the test set, we correctly classified 34.203 percent of pixels in the binned U dimension, and we correctly classified 75.345 percent of pixels in the binned V dimension. See Figure 5 for a comparison of model outputs.

5. NEXT STEPS

Future work will focus on improving both components of our project, where practical constraints hampered our progress: creating a more robust color model, and using truly recurrent architectures for temporal consistency. We believe that solutions to the latter component will pave the way for the next generation of autonomous agents, which must understand and react to large amounts of spatial-temporal data.

6. ACKNOWLEDGMENTS

This research was conducted for the final project assignment in Stanford University's CS 229 course. We would like to thank Andrew Ng, John Duchi, and the rest of the teaching staff for their patience and expertise. We would like to especially thank our project TA Nihit for his advice and encouragement.

7. REFERENCES

[1] Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[2] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[3] Ryan Dahl. Automatic colorization, 2016.

[4] Bharath Hariharan, Pablo Arbeláez, Ross Girshick, and Jitendra Malik. Hypercolumns for object segmentation and fine-grained localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 447–456, 2015.

[5] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[6] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. *arXiv preprint arXiv:1603.08511*, 2016.

[7] Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi Ishikawa. 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 (TOG)*, 35(4):110, 2016.

[8] GMRB Black. Yuv color space. *Communications Engineering Desk Reference*, 469, 2009.

[9] Andrej Karpathy. Course notes, 2016.