Automatic Colorization of Grayscale Images

Rasoul Kabirzadeh, Austin Sousa, Patrick Blaes

Introduction:

There exists a wealth of photographic images, from antique photography to low-resolution video, which lack color information. Assigning color to a black-and-white image is a highly ill-posed problem; given a certain image, there is often no "correct" attainable color. For example, a red balloon is indistinguishable from a blue balloon when photographed in black-and-white. Due to the indeterminate nature of the problem, image colorization techniques currently rely heavily on human interaction. Several vintage movies have been colorized entirely by hand, requiring every single frame to be inspected at great financial and time cost.

Here we present a method, using machine learning techniques, to assign aesthetically-believable colors to black-and-white photographs. The applications of such a method allow for a new appreciation of old, black and white photographs and cinema, along with allowing better interpretation of modern grayscale images such as those from CCTV cameras, astronomical photography, or electron microscopy.

Description of Method:

Current literature suggests two classes of colorization methods -- one in which a color is assigned to an individual pixel based on its intensity, and another in which the image is segmented into regions, each of which are then assigned a single hue. Our method uses the former approach: we analyze the color content of a provided training image, and then attempt to predict colorization for a single target grayscale image on a per-pixel basis. While segmentation techniques have intrinsic appeal, and could result in more-consistent colorization, these methods rely on accurate image segmentation, which can be thrown off by shadows, texture, or color gradients.

Figure 1 shows block diagrams of the training and colorizing processes. Training consists of (a) discretizing the color space, (b) selecting a subset of pixels to train on, c) extracting a set of features from each pixel using the luminescence channel, and (d) training an array of binary classifiers -- one per each color. Similarly, colorizing a grayscale image consists of (a) extracting the same set of features from each pixel, (b) predicting whether or not each pixel is a certain color, and c) selecting the color with the highest likelihood.

Our method is implemented in Python 2.7, using the *numpy*, *sklearn*, and *openCV* libraries. Each of the processing steps is described below.

Color space discretization:

The colorizer is trained via the *train(files)* method. Each image in the set of training files is converted to the L*a*b* (Luminance, a, b) color space, which reduces the output color channels from three to two ({RGB} -> {ab}). The Luminance channel becomes the grayscale image from which features are extracted. The A and B channels are then discretized into N uniformly-spaced bins, which results in a set of N^2 possible output colors.

Feature extraction:

Our system assigns a feature vector to a uniformly-spaced subset of pixels from the training image. At each training pixel, several features are computed -- the extended 128 SURF (Speeded-Up Robust Features) parameters taken over a 20x20 grid surrounding the training pixel; the mean and variance of the training pixel; and the training pixel's relative (x,y) location within the image. The pixel's grayscale luminescence, however, is not used as a feature, as there appears to be little correlation between brightness and color. The result is a 132-entry feature vector. At present we do not perform any reduction of features.

Classification system:

Our classification system presently consists of a set of support vector machines; one per each bin of the discretized color space. This array of SVMs performs a binary classification for each color bin, predicting whether a pixel is or is not the corresponding color. This classification style is known as *One vs. All* multilabel classification.

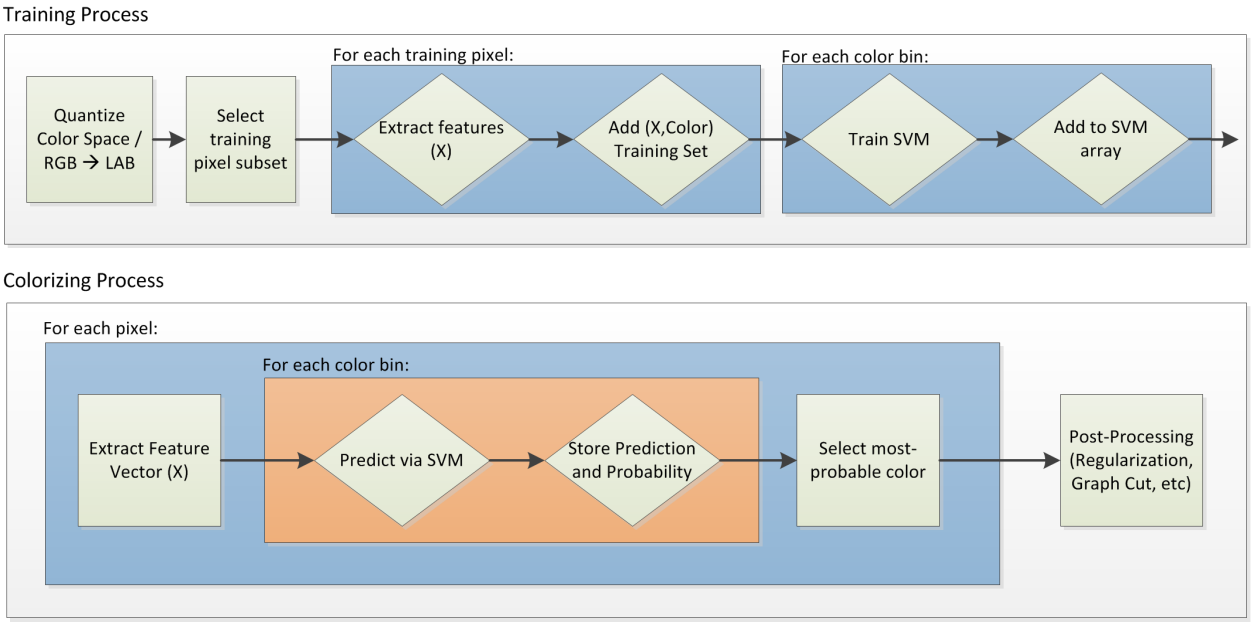

Figure 1: Block diagram of training and colorizing processes

Colorizing Process:

Colorization via a trained set of classifiers is implemented in the *colorize(image)* method. For each pixel in the target grayscale image, a feature vector is generated. True/false values for each color are predicted using the array of trained SVMs. Presently, in the event of multiple colors being predicted as true, our algorithm selects the last true color; future implementations will either use a confidence value from the classifiers, or define a cost function incorporating surrounding pixels to make a more-logical selection. In preliminary tests, however, it is more-common for a pixel to return zero predicted colors. In this case, our algorithm leaves the pixel gray. Again, incorporating probabilistic metrics or a cost function will assure that a color is always selected.

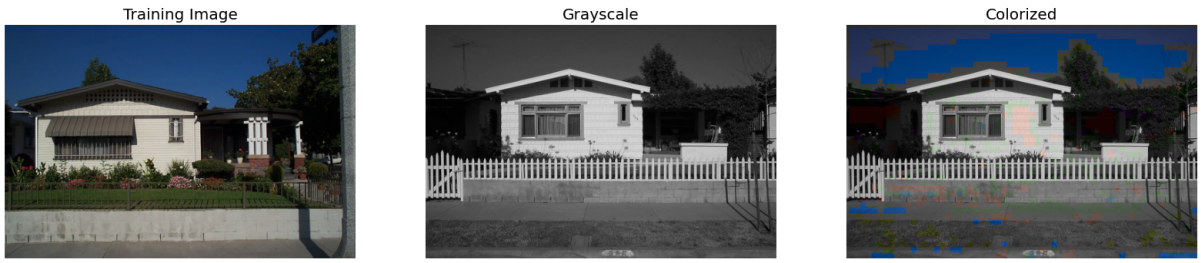

Figure 2: Example using the Caltech Houses dataset

Figure 3: Example using a nature scene

Preliminary Results:

Despite only minimal optimization, our initial algorithm performs reasonably well. Figure 2 shows an example run, using two images from the Caltech Houses image library. Detailed colors are missed, and many pixels returned no predicted color. However, the general colors of sky, grass, and brick are all present, and shows promise that realistic results are achievable. Figure 3 shows a similar trial using a nature scene. This training / colorizing pair performs better than the previous example, most likely due to the small set of distinct color regions -- blue sky, tan mountains, and green foliage.

Further work:

While our initial implementation shows promise, numerous improvements can be made.

Improvements in colorspace discretization:

-Rather than uniformly quantizing the color space, a set of ideal colors could be selected, using a clustering algorithm such as K-means.

Improvement in feature selection / reduction of feature complexity:

-No attempt has been made to reduce the feature vectors, which are currently fairly large (132 elements). PCA / SVD could be used to select a subset of features with the highest variance across the image.

-Additional features which have not been experimented with include Fourier / Cosine components, or a combination of Fourier and Wavelet components using a basis pursuit method.

-The SURF function allows for several parameters to be adjusted, including scale and cell size.

-Global / regional features have not been experimented with. This method would first attempt to segment the image into broad regions -- possibly foreground and background -- and use this assignment as an additional feature.

Improvement in classification system:

Currently we use an array of binary classifiers, which return a yes/no answer if a pixel is predicted to be a certain color. However there are numerous cases in which no color returns true, in which case the pixel remains grayscale. Furthermore, in the event of multiple SVMs returning true, we simply select the last true color. However, some SVM implementations can return a confidence value along with a prediction, which could be used to select a most-probable color, instead of a simple yes/no.

Other classification techniques (such as Naive Bayes or Logistic Regression) may outperform SVMs, and warrant some experimentation.

Post-Processing:

Currently no post-processing is implemented. However, several techniques could be easily applied to regularize the selected colors, to fill in uncolored patches at edges and boundaries, and to smooth out sharp jumps over gradients. These techniques include simple Gaussian blurs on the color channel, nearest-neighbor interpolation at the edges of uncolored regions, or a graph cut process as outlined in [5].

References

[1] A. Bugeau and V. T. Ta, “Patch-based Image Colorization,” Pattern Recognition (ICPR), 2012.

[2] S. Liu and X. Zhang, “Automatic grayscale image colorization using histogram regression,” Pattern Recognition Letters, 2012.

[3] X.-H. Wang, J. Jia, H.-Y. Liao, and L.-H. Cai, “Affective Image Colorization,” J. Comput. Sci. Technol., vol. 27, no. 6, pp. 1119–1128, Nov. 2012.

[4] Chapter Machine Learning Methods for Automatic Image Colorization, with Ilja Bezrukov, Yasemin Altun, Matthias Hofmann and Bernhard Schölkopf, chapter of the book Computational Photography: Methods and Applications, R. Lukac Editor, CRC Press