COLOR REPRESENTATION IN DEEP NEURAL NETWORKS

Martin Engilberge *Edo Collins* *Sabine Süsstrunk*

School of Computer and Communication Sciences, EPFL, Switzerland

ABSTRACT

Convolutional neural networks are top-performers on image classification tasks. Understanding how they make use of color information in images may be useful for various tasks. In this paper we analyze the representation learned by a popular CNN to detect and characterize color-related features. We confirm the existence of some object- and color-specific units, as well as the effect of layer-depth on color-sensitivity and class-invariance.

Index Terms— Convolutional Neural Networks, Color information, Color sensitivity, Latent features, Deep learning

1. INTRODUCTION

Over the past decade, deep neural networks have achieved state-of-the-art performance on many and various benchmarks. In particular, Convolutional Neural Networks (CNNs) have shown top performance on image classification tasks. However, in spite of their ubiquity, the representation learned by such models remains not fully understood. For instance, in [1] the authors show that a deep CNN may learn a feature corresponding to the presence of human and animal faces, even when the classification task does not involve a 'face' class.

Similarly, we seek to gain a better understanding of how a deep CNN might utilize color information, by detecting spontaneous features that make use of such information.

The identification of units corresponding to *color-sensitive* features can be useful for various tasks. For instance, secondary color-based classification, i.e. further classification of the object by its color (e.g. red car, blue bird). While this could be accomplished in various ways, we seek to make use of what the network has already learned, while avoiding the effort of manually annotating images with color labels.

The activation profile of color-sensitive units can also be used to enrich the semantic descriptor derived at the final layers of the CNN. These unit activations could also be used for manipulation of the input image via optimization methods.

We establish three unit properties we believe are useful to that end:

- Color-sensitivity – how affected is the activation of a unit by the presence or absence of color.
- Hue-specificity – how affected is the activation of a color-sensitive unit by the presence or absence of a specific hue.
- Class-invariance – how affected is the activation of a color-sensitive unit by the presence of a specific object class.

In this paper we present methods to measure these properties. Our findings provide interesting insights. For instance, we confirm the existence of class-specific and hue-specific features, e.g. 'yellow bus', and justify the use of deep pre-trained models as general feature extractors by showing features to be class-invariant through most of the CNN forward-pass pipeline.

In section 4 we show a simple method for detecting units that are sensitive to the presence or absence of color in images. In section 5 we continue to precisely characterize the hue specificity of these units by studying their response to monochrome images. Finally, in section 6 we present an analysis based on the co-activation matrix of color-sensitive units and object class labels to test for class specificity or invariance.

The model and data on which we base our investigation are presented in section 3.

2. RELATED WORK

We are not aware of any study focusing specifically on analyzing how CNNs leverage color information. In [2] the authors present anecdotal results demonstrating the impact of object color on classification performance. In particular, they show that manipulating the color of an object in a way that deviates from the training data has a negative impact on classification performance.

General approaches for visualizing deep features have been proposed [3, 1]. However, we have found the visualizations produced by these methods correlate only spuriously with the color-profiles that trigger unit activations in practice, such as presented in section 5. We therefore excluded these methods from the current analysis.

To analyze the class invariance (or specificity) of a unit, we utilize a method similar to that of [4], where we use the co-activation matrix instead of the covariance matrix.

3. NETWORK AND DATASET

We chose to use the popular VGG-19 network [5]. This model consists of five convolutional blocks, followed by three fully connected layers. Each convolutional block consists of multiple convolutional layers with the same filter size followed by a max-pooling operation. Convolutional layers are named according to the index of their convolutional block and their position within it, e.g. 'conv4_3' refers to the third convolutional layer in the fourth block. Fully connected layers are indexed from 6 to 8, e.g. 'fc6'.

We also used the PASCAL VOC dataset [6]. In particular, we used a subset of the dataset which is annotated with precise localization maps, i.e. a segmentation mask. In total

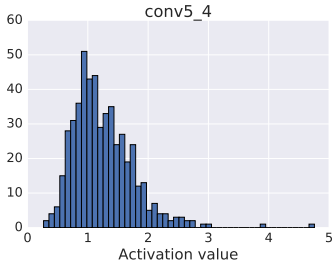
our image dataset consists of 3K images distributed over 20 object classes.

4. ANALYZING THE COLOR SENSITIVITY OF THE NETWORK

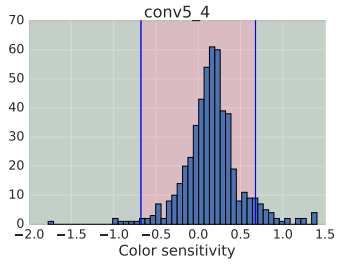
In this section we characterize the propagation of color information through the network by searching for general color sensitivity in unit activation.

4.1. Color-Sensitive Units

A unit is called color-sensitive if its activation value is affected by the absence or presence of color. This effect can be measured directly by observing unit activations in response to a set of colored images against a grayscale version of the same images. Our assumption is that units whose activation changes significantly across the two image sets are those that captures the presence or absence of color.



(a) Histogram of the unit activation for color images.



(b) Histogram of color-sensitivity. Color-sensitive units in green.

Fig. 1: Mean activation and color-sensitivity of units in the 4th layer of the 5th convolutional block (Conv5_4).

For units in fully-connected layers, we simply consider the activation value in response to an input image. For units in convolutional layers, we compute the mean activation across the spatial dimension, resulting in a single scalar value. For unit j and image i we denote the activation as $a_j^{(i)}$. Since VGG-19 uses the ReLU activation function, we have that $a_j^{(i)} \in [0, \infty)$.

In Figure 1(a) we show the histogram of unit activations, i.e. $\mathbb{E}_i[a_j^{(i)}]$.

For every image with index i we denote its grayscale version with the index i'. Of interest therefore is the difference in activation $a_j^{(i)} - a_j^{(i')}$. We measure the color sensitivity of a unit, Col_j^{sens}, as follows:

$$Col_j^{sens} = \frac{\sum_{i \in I} a_j^{(i)}}{N} - \frac{\sum_{i \in I} a_j^{(i')}}{N} \quad (1)$$

Where N is the number of images in dataset I. The histogram of these residuals is shown in Figure 1(b). The resulting distribution in centered close to zero, indicating that the majority of units have a similar activation regardless of whether the network is processing color or grayscale images.

Taking the absolute value of equation (1) allows us to easily identify units that are sensitive to color: the larger the absolute value is the more color sensitive a unit is.

To automatically select a threshold value, above which we consider a unit to be color-sensitive, we use the standard deviation across a layer. We define the threshold T as

$$T = 2 * Std(Col_{LayerL}^{sens})$$

where Col_{LayerL}^{sens} is the distribution of Col_j^{sens} of all units in layer L, e.g. Figure 1(b).

In Figure 1(b) the two vertical lines represent the threshold over which we consider the units to be color-sensitive. There are two lines because our threshold is defined with respect to the absolute value of the color sensitivity. Color-sensitive units make up 7.1% of units in every layer, with small variations. Figure 2 shows their distribution across network layers.

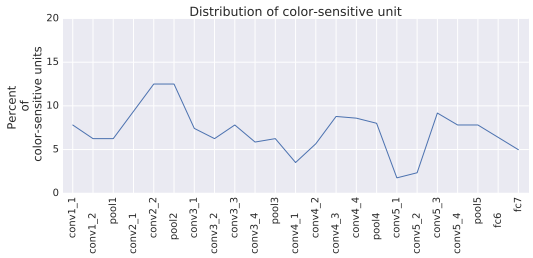


Fig. 2: Percentage of color-sensitive units, out of all units, for each layer of the network

5. HUE SPECIFICITY

With color-sensitive units identified, we proceed to more precisely characterize their hue-specificity. We wish to know for each unit whether its sensitivity is limited to specific colors.

We start this analysis with a set of 100 monochrome images created using the HSV color space. We set the saturation and value to 1, and vary the hue between 0 and 1 with increments of 0.01.

These images are fed to the network and the resulting activation is recorded for all color-sensitive units. Figure 3 shows the activation of six units that were chosen as to show the different types of units we identified, and how unit behavior varies across network layers.

Figure 3(a) and 3(b) show units sensitive to a single color. An interesting point to note is that while unit (b) displays more degrees of freedom - allowed for by its position in a deeper layer - it still retains a simple response curve similar to that of unit (a).

Figure 3(c) depicts a curve with high activation for hue values 0 and 1, consistent with the cyclic nature of the hue spectrum. In Figure 3(e) we once more see that even at deep layers, some unit exhibit simple activation curves responding to a single hue in this case the color yellow. Figure 3(f) show a unit can be sensitive to two different colors, e.g. red and green.

Lastly, while some units seem to be sensitive to a limited number of specific colors, some seem to have the opposite behavior: sensitive to almost everything *except* a specific color, as shown in Figure 3(d). Thus, a low sensitivity to a specific

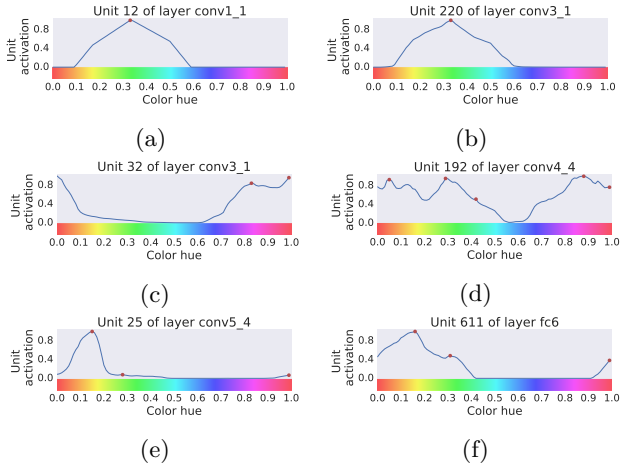


Fig. 3: Activation curves for various units in response to monochrome images of varying hues.

color results in a negative impact on activation when this precise color is detected.

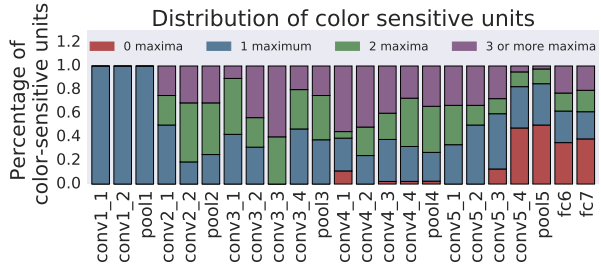


Fig. 4: Distribution of the local maxima detected in the activation curve of color-sensitive units for every layer.

The red dots in Figure 3 indicate local maxima of the activation curves. The number of maxima on a curve can be used as a rough measure of how "well-behaved" a unit is, i.e. a single maximum indicates a simple sensitivity to a single hue, with more maxima indicating more complex behaviors. The distribution of the number of maxima is shown in Figure 4. We see that in spite of increasing degrees of freedom, most units have a simple activation curve with one to two maxima.

We also notice color-sensitive units with no maxima. These are color-sensitive units that have no activation at all in response to monochrome images, presumably due to lack of any shapes and texture.

6. CLASS INVARIANCE

So far we have identified color-sensitive units and characterized their hue-specificity. In this section we study how these units are used to perform object recognition, for which VGG-19 was trained. This will answer whether a single color-sensitive unit contributes to the classification of multiple object classes in an invariant way, or if it is class-specific.

6.1. Unit Activation

To get a qualitative impression of the relation between color-sensitive units and object classes, we return to the set of units used in Figure 3. For each unit, we find the images in the dataset which maximize the unit's activation (see Figure 5). Unit (e) In the fifth row seems to be both color-specific and class specific, while units in other rows show a degree of class-invariance while being consistent with their color preference.

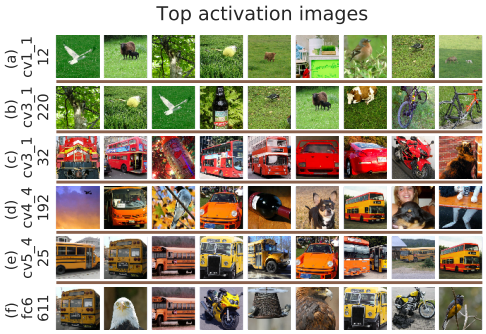


Fig. 5: Top activation images, one unit per line. See Figure 3 for exact hue-specificity of these units.

6.2. Covariance and Co-activation

To analyze the relation between color-sensitive unit and object classes, we start with the covariance. The empirical covariance matrix of some data matrices X and Y is computed as follows:

$$cov(X, Y) = E[(X - E[X]) * (Y - E[Y])]$$

Assuming X and Y have zero mean, this simplifies to:

$$cov(X, Y) = E[X * Y] = \frac{1}{N} X^T Y$$

In the case at hand we are interested in the correspondence between ReLU activations, which are non-negative, and class labels, which are binary. In both cases we refer to a variable as being 'active' if its value is non-zero. Similarly to above, we define the co-activation matrix as:

$$coa(A, B) = A^T B$$

Where A and B are such matrix variables whose values are non-negative. Thus the co-activation is a view of how pairs of variables are active at the same time. We compute the co-activation between every color-sensitive unit of a layer and every class label. Since we have 20 object classes, we obtain a co-activation matrix of dimension $(U \times 20)$ where U is the number of color-sensitive units.

6.3. Co-activation Analysis

The analysis of the co-activation matrix offers insight into how different layers deal with class information. First, since we care about the distribution of units activation across class labels, and not their absolute values, the matrix is normalized

row-wise such that the co-activation values of a single unit sum up to one. With this matrix one can identify which units are skewed towards a set of specific classes, and which lean towards uniformity.

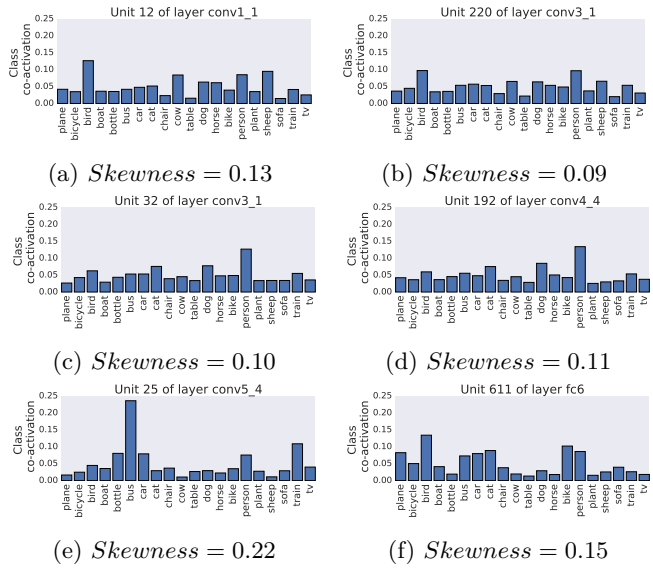
To measure the tendency more precisely, we define a skewness metric which takes a maximum value of 1 indicating the unit corresponds with a single class, and a minimum of 0 indicating the unit activates uniformly across all classes:

$$S(L) = \frac{StD(L)}{Max(StD)} \tag{2}$$

Where L is a row of the co-activation matrix, $0 <= S <= 1$, and StD is the standard deviation. Normalization by $Max(StD)$ makes comparing different units possible.

6.3.1. Unit Level Analysis

Using the skewness metric on the color-sensitive units identified in section 4, we are able to verify the existence of both class-invariant and class-specific color-sensitive units. Once again using the set of units analyzed in Figures 3 and 5, computing the co-activation matrix of these units with class labels yields the distributions in Figure 6.



(a) $Skewness = 0.13$ (b) $Skewness = 0.09$

(c) $Skewness = 0.10$ (d) $Skewness = 0.11$

(e) $Skewness = 0.22$ (f) $Skewness = 0.15$

Fig. 6: Normalized co-activation of example units with every class. See Figures 5 and 3.

In Figure 6 we show how each unit is skewed towards the different classes. Among these eight units, we find unit (e) is highly skewed, in accordance with Figure 5 where top activation images of this unit all belong to a single class. This is an example of a unit being color specific and class specific, as it seems to correspond mainly with yellow buses. The other units of the figure are mostly class invariant with a skewness factor around 0.12.

Figure 7 shows for every class, how units are skewed towards it. The average co-activation is similar for all classes, and all classes have color-sensitive units skewed toward them, (shown as outliers in the figure).

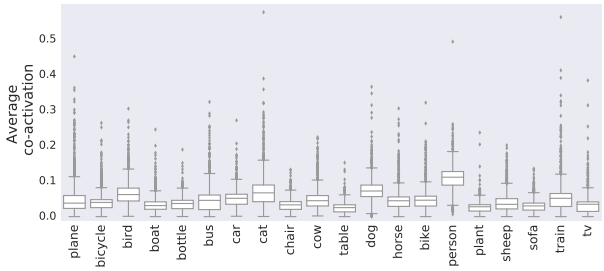


Fig. 7: Average co-activation per class for all color-sensitive units

6.3.2. Layer Level Analysis

Finally, we investigate the evolution of color-sensitivity (equation 1) and class-specificity (equation 2) across the layers of the network. This is shown in Figure 8. We can see that the deeper the layer, the more units become class-specific and less color-sensitive. This is expected since the network is trained for the task of object recognition, making it invariant to other aspects of the input.

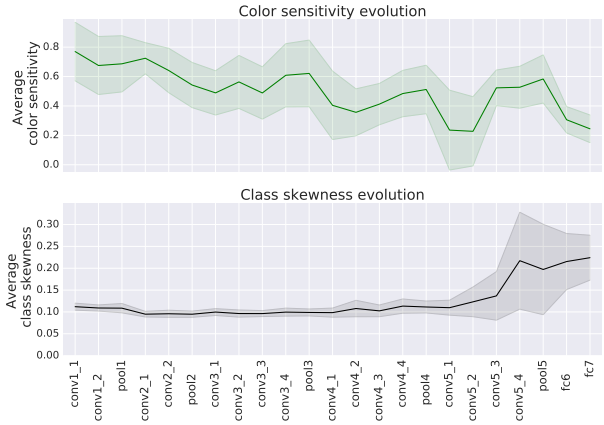


Fig. 8: Average color-sensitivity (above in green) and average skewness toward classes (below in black), for every layer. Standard deviation in light color.

7. CONCLUSION AND FUTURE WORK

We have introduced and operationalized the concepts of unit color-sensitivity, hue-specificity and class-invariance, which we will be useful in picking out a subset of network activation relevant to color-related tasks, such as secondary color-based classification and color enhancement.

Our statistical analysis has led to interesting findings . We showed that in the case of VGG-19 color information is processed by specific units that are color-sensitive. Further analysis showed these units possess different hue-specific characteristics, depending on which layer of the network they belonged to. We find that units in earlier layers being more sensitive to color and less so to class, while units in deeper layers displaying the opposite trend, justifying the use of pre-trained models as general feature extractors.

8. REFERENCES

[1] Jason Yosinski, Jeff Clune, Anh Nguyen, Thomas Fuchs, and Hod Lipson, "Understanding neural networks through deep visualization," *arXiv preprint arXiv:1506.06579*, 2015.

[2] Alexey Dosovitskiy and Thomas Brox, "Inverting visual representations with convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4829–4837.

[3] Matthew D. Zeiler and Rob Fergus, "Visualizing and understanding convolutional networks," *Computer Vision - ECCV 2014*, Jan. 2014.

[4] Yani Ioannou, Duncan Robertson, Darko Zikic, Peter Kontschieder, Jamie Shotton, Matthew Brown, and Antonio Criminisi, "Decision forests, convolutional networks and the models in-between," *arXiv preprint arXiv:1603.01250*, 2016.

[5] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2014.

[6] Mark Everingham, SM Ali Eslami, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman, "The pascal visual object classes challenge: A retrospective," *International Journal of Computer Vision*, vol. 111, no. 1, pp. 98–136, 2015.