CoRe: Color Regression for Multicolor Fashion Garments

Alexandre Râmé\textsuperscript{1,2}, Arthur Douillard\textsuperscript{1,2}, Charles Ollion\textsuperscript{2}
\textsuperscript{1}Sorbonne Université, \textsuperscript{2}Heuritech,

Abstract

Developing deep networks that analyze fashion garments has many real-world applications. Among all fashion attributes, color is one of the most important yet challenging to detect. Existing approaches are classification-based and thus cannot go beyond the list of discrete predefined color names. In this paper, we handle color detection as a regression problem to predict the exact RGB values. That’s why in addition to a first color classifier, we include a second regression stage for refinement in our newly proposed architecture. This second step combines two attention models: the first depends on the type of clothing, the second depends on the color previously detected by the classifier. Our final prediction is the weighted spatial pooling over the image pixels RGB values, where the illumination has been corrected. This architecture is modular and easily expanded to detect the RGBs of all colors in a multicolor garment. In our experiments, we show the benefits of each component of our architecture.

1. Introduction

Convolutional Neural Networks (CNNs) \cite{20, 26} generated a lot of interest in the fashion industry. Recent datasets of fashion images \cite{17, 28, 51} encouraged various approaches for attributes classification \cite{7, 41}, visual search \cite{13, 19, 30, 45} and object detection \cite{23, 34}.

One of the key attributes to describe a fashion item is its color. However, colors are subjective properties of garments as not all humans recognize colors the same way \cite{32}: their automatic estimation is therefore challenging. So far, most approaches consider the problem through the angle of discrete color naming: a classifier chooses amongst 11 colors of the English language \cite{4}, based on features from histograms \cite{2, 9, 40} or from CNNs \cite{10, 21, 24, 33, 43, 46, 47}. Recent approaches increased the number of color names up to 28 \cite{48} or 313 \cite{50}. We take a step forward and tackle the problem through the angle of continuous color regression. Rather than only predicting an approximate discrete color name, we aim at predicting the exact continuous color RGB. This refined information is necessary for many industrial applications such as precise visual search and fine-grained trend detection to handle color inherent ambiguity. Moreover, the regression paradigm is more versatile and adaptable; this is especially important for real-world applications, where product requirements can evolve. In contrast, adding a new color name in classification approaches would require a dataset relabelling and the training of a new network.

To understand the challenge of this regression task, let’s consider an unsupervised approach, that would naively average the different pixel RGBs of the image. That would fail for three reasons. First, it would suffer from varying illuminations, sparking discrepancies between the raw pixels and what a human would perceive in ideal white light. Second, it would consider ‘parasite’ pixels from either the background or other garments. Third, in case of multicolor garments, it would average pixels belonging to different colors and would predict neutral RGBs.

In this paper, we propose a new supervised two-stage architecture for color regression, inspired by the object detection literature \cite{35}. The first stage is a standard color classifier — similar to previous works \cite{10, 33, 43, 46, 47} — which predicts the main colors of the considered garment. Our contributions lie in the second regression-based stage, refining the previous discrete prediction by weighting the different pixels of the illumination-corrected image.

First, we developed an illumination module that auto-
matically corrects the contrast of the image — and this without supervision. This is inspired by the color constancy literature [3,5,6,15,22,29,37,38]. Second, in order to reduce the impact of complex backgrounds or complex garments structure, we leverage a semantic segmenter [10,44] pre-trained to detect fashion garments. Third, we detect all pixels in this garment that are close to the colors predicted by the first stage. This enables the handling of multicolor garments by removing pixels from conflicting colors. These last two components are combined to find the appropriate weighting of each pixel in the image — enabling to focus on the appropriate regions of the image.

Empirically, we validate the effectiveness of our approach in our new fashion dataset collected from real-world images. More importantly, we show that all our components contribute.

2. Model

The network in Figure 2 is trained end-to-end on the task of continuous color regression, with annotated RGB values, but with neither illuminations nor clothing segmentation ground-truth annotations.

2.1. Pixels correction

Our model needs to correct the bias in the pixels of the image, for example the quality or the lighting of the photo. To do so, first, all images are contrast normalized in a pre-processing step by global histogram stretching in order to be robust across various lighting conditions. Note that this does not change the relative contributions of the three RGB channels. Second, to remove illumination color casts, we need to detect the initial illumination of the image (a scalar value per channel); this is a regression task, achieved easily by a deep neural network. Note that following [29], we choose explicitly the VGG16 [39] neural architecture because of its capacity to extract low-level features [16]. Finally, each pixel in the image is corrected using the Von Kries method [42], i.e., scaling each channel by the detected illumination value.

2.2. Spatial pooling over pixels for color regression

To sample only the relevant regions of the image, we apply a spatial pooling over the pixels of the image weighted by two complementary attention module. They modulate the importance of each pixel in the spatial pooling. The first object-attention focuses on the clothing category, the second colorname-attention focuses on the color name predicted. These attention masks are then multiplied pixelwise to create the combined-attention.

2.2.1 Through object-attention

Following the work from [47], we use the popular fully convolutional Deeplabv3 [8]. This attention network is pre-trained on the task of semantic segmentation over the clothing crops from Modanet [51] and therefore has an emphasis on the garment surface: it produces an object-attention different for each clothing category (top, coat, ...). This spatial prior is then fine-tuned: ideally, it would learn to identify the regions which contain the relevant color information given a garment type.

2.2.2 Through colorname-attention

The previous object-attention will fail for multicolor garments. Indeed, if the garment contains a set of distinct and distant RGB values, the mean of this set would be predicted: it leads to unsaturated color predictions. For example, in Figure 2, including the white pixels from the shirt in our spatial pooling would predict a RGB closer to light pink.

Predicting the RGB of the main color

For simplicity, we first describe how to detect only the RGB of the main color of a (potentially multicolor) garment. First, we detect the main color name; specifically, following previous works, the VGG16 features are followed by a fully connected layer with a softmax activation function that predicts a distribution over 72 color names. This is trained by minimizing a categorical cross-entropy loss $L_{CE}$.

Second, our colorname-attention only selects pixels in the image sufficiently close to the color detected by this first color classifier. We map this discrete color to its RGB continuous value: e.g., (134, 71, 71) for velvet red. Now, given a pixel $p$ of RGB value $RGB_p$, the colorname-attention $CA_p$ for pixel $p$ is:

$$CA_p \propto \exp \left( \frac{-1}{127.5+T} (RGB_p - RGB_c)^2 \right).$$  \hspace{1cm} (1)$$

It sums to 1 over all pixels and its peakedness depends on $T$, the temperature of the RGB spatial softmax. $T$ is the only parameter of this colorname-attention module, which can be either a predefined hyperparameter, either learned, either input-dependant and predicted with a fully connected layer from VGG features. This last option works best as shown in our Experiments from Section 3.

Predicting the RGBs of all colors

We can easily generalize our approach for detecting multiple RGBs. To do so, we predict multiple color names with a sigmoid [46] followed by a combination of categorical cross entropy and binary cross entropy as in [34]. Multiple colors share the same object-attention but have different colorname-attentions: therefore we predict different RGBs with our combined-attention.
Figure 2. **Architecture components and training procedure.** *First,* the DeepLabv3 object-attention network selects pixels inside the garment. *Second,* the VGG16 CNN extract features before feeding two heads: the first predicts the image initial illumination (to be corrected with the Von Kries method), and the second head detects the main discrete color name amongst 72 available colors (trained with a categorical cross entropy $L_{CE}$). This main color is used to create a colorname-attention. These attentions are used for weighted spatial pooling over the pixels of the image for color regression. Best results are obtained when colorname-attention and object-attention are combined, as this would select only pixels that are simultaneously inside the clothing and that have a RGB value close to the RGB of the predicted discrete color name. All components are trained so that the final RGB matches the RGB annotation with an Euclidean distance in the LAB space ($L_{L2}$). This architecture is easily extended to detect the RGBs of multiple colors (red and white here).

The challenge is then to know how many colors should be predicted. Following the work from [46], we explicitly *learn the number of colors* in each garment (up to 3 different colors). Note that we apply class weights for handling unbalanced classes.

Selecting the color names with maximum scores would often predict several times the same major color: for example a light red and a dark red would converge towards a medium red. Thus, we apply a non-maximum-suppression algorithm to delete predicted RGBs too close to each other. In Figure 2, the network’s confusion between two different kinds of reds would have prevented from predicting white.

### 2.3. Training overview

The LAB color space best models perceptual distance in colorization [11, 50]: thus, the chosen regression objective is $L_{L2}$, the Euclidean distance between predicted and ground truth colors converted to LAB rather than RGB. Our two losses — color naming $L_{CE}$ and color regression $L_{L2}$ — are summed before backpropagation; this enables the learning of all architecture components end-to-end.

### 2.4. Discussion: analogy with Faster R-CNN

Our two-stage approach is highly inspired by the Faster R-CNN [35] architecture. The anchors of the Region Proposal Network are replaced by color names. In both cases, the first classifier gives a rough estimate, refined by the second regression stage. Specifically, given a selected anchor (resp. color name), the final regression adds a small continuous offset leading to a more precise box (resp. RGB values). Non-maximum-suppression algorithms are also standard to handle overlapping crops in object detection.

### 3. Experiments

#### 3.1. Setup

**Dataset** As far as we know, there is no dataset for the task of continuous colors regression. Thus, we have collected a new dataset of 30,269 fashion garments: 5,363 coats, 8,166 dresses, 3,991 pants, 6,871 shoes and 5,878 tops. The validation dataset and the test dataset are composed of 2000 images: the other images are used for training. Each garment was labeled by a single operator with its exact color RGB: not like it appears in the image, but like the operator thought it would appear in ideal white light. The annotation process
Table 1. **Results for RGB regressions**: percentage of predictions closer to the ground truths RGBs than several thresholds (10, 20, 30, 40), according to the *deltaE ciede2000* distance [14,31]. **Bold** highlights best score.

<table>
<thead>
<tr>
<th>Method</th>
<th>Color-attention</th>
<th>Object-attention</th>
<th>Illumination</th>
<th>(\leq 10)</th>
<th>(\leq 20)</th>
<th>(\leq 30)</th>
<th>(\leq 40)</th>
<th>(\geq 10)</th>
<th>(\geq 20)</th>
<th>(\geq 30)</th>
<th>(\geq 40)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unsupervised K-means Clustering [27]</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>47</td>
<td>72</td>
<td>83</td>
<td>89</td>
<td>19</td>
<td>31</td>
<td>37</td>
<td>42</td>
</tr>
<tr>
<td>Colorname RGB [46]</td>
<td></td>
<td></td>
<td>✓</td>
<td>51</td>
<td>72</td>
<td>87</td>
<td>92</td>
<td>34</td>
<td>52</td>
<td>64</td>
<td>68</td>
</tr>
<tr>
<td>Ours</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>50</td>
<td>78</td>
<td>90</td>
<td>95</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Ours</td>
<td>✓</td>
<td>✓</td>
<td>Set to 1</td>
<td>48</td>
<td>76</td>
<td>88</td>
<td>93</td>
<td>35</td>
<td>56</td>
<td>65</td>
<td>68</td>
</tr>
<tr>
<td>Ours</td>
<td>✓</td>
<td>✓</td>
<td>Trainable</td>
<td>59</td>
<td>87</td>
<td>93</td>
<td>95</td>
<td>44</td>
<td>64</td>
<td>69</td>
<td>71</td>
</tr>
<tr>
<td>Ours</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>67</td>
<td>89</td>
<td>94</td>
<td>96</td>
<td>46</td>
<td>66</td>
<td>70</td>
<td>72</td>
</tr>
<tr>
<td>Our best</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>73</td>
<td>90</td>
<td>93</td>
<td>95</td>
<td>54</td>
<td>68</td>
<td>71</td>
<td>73</td>
</tr>
</tbody>
</table>

is therefore more complex and time-consuming than classical classification. The color names can automatically be derived by nearest neighbors in the *LAB* space. In case of multicolor items, the operators were asked to tag them in decreasing order of importance.

**Implementation** Our code is in Tensorflow [1] and Keras [12]. We chose an Adam [25] optimizer with a learning rate of 0.0001 and a batch of 16 images during 50 epochs. We applied standard data augmentation methods: random cropping and translation.

### 3.2. Results

Table 1 summarizes our experiments. To compare approaches, we count the percentage of predictions that are closer to the color annotation than a given threshold, leveraging the *deltaE ciede2000* distance [14,31] that is arguably “the best metric for understanding how the human eye perceives color difference” [36]. With our best network that incorporates all our components, 73% of all predictions have a distance to the main color smaller than 10; in practice, these differences are small to the human eye. We also report similar metrics when the goal is to detect all colors in the garment. We showcase in Figure 3 multiple predictions of our model.

**Baselines** We compare our model against two existing baselines. First we train an unsupervised *K-means* pixels clustering [27] directly on the pixel space, after a background removal done by an external semantic segmentation model trained on Modanet [51]. Colors are ranked according to the number of pixels in each cluster. The second baseline, *Colorname RGB*, is the direct extension of the multitagger approach from [46]: it detects the color names (*e.g.* velvet red, dark purple, etc.) and then produces the associated RGB values. This baseline is depicted in Figure 2.

**Attention** First, fine-tuning the semantic segmentation model on the regression task (object-attention) already surpasses previous approaches when predicting only the main color. The colorname-attention improves results in the multi-color setup. These two attentions, when combined, are mutually reinforcing and complementary.

**Temperature** We show that the temperature \(T\) value is important: bigger \(T\) leads to sharp distribution and takes into account fewer pixels from the initial images that with a lower \(T\). Rather than grid-searching its optimal value, it can be learned for improved results. Moreover, the optimal \(T\) depends on the image: therefore, best results are obtained with \(T\) predicted from VGG features. This analysis is consistent with recent insights for calibration via input-dependent temperature scaling [49].

**Illumination** Finally, including the illumination module improves performances. Future work could further improve results by pretraining on the *Color Checker Dataset* [18].

### 4. Conclusion

In this paper we addressed the color regression problem for fashion garments. By collecting a unique dataset of 30,269 images, we empirically show the benefits of our newly proposed two-stage architecture. These performance gains would have a potential impact on many real-world usages, notably to better detect fashion trends and for visual search. Finally, we hope to shed light on properties of fashion garments (perhaps surprisingly) complex to detect.
References


