Bottle Color Recognition (CV Approach) Report

Suprateem Banerjee
Machine Learning Engineer - Intern
Nimblebox.ai

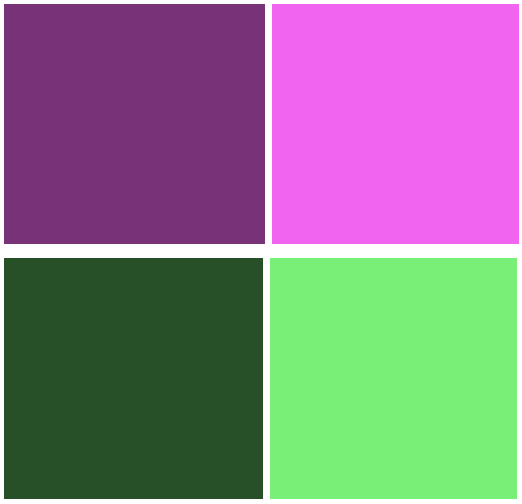
Problem: To identify whether two provided bottles are of similar colour tones.
Github Code

I start this by stating what I feel is the main hurdle at this stage: lack of data. I don't mean any data (there are some we can scrape off Google Images, of course), but the issue is they all are pictured **under different lighting scenarios**. That is to say, the main issue remains the lack of **coherent** data to experiment with. This issue is more prominent when we look at bottles that are transparent, and the background essentially determines their color signature.

However, given the limited incoherent data we could scrape off Google, I base my experiment and observations on these alone.

The approach that seems very intuitive to me is one that uses **normalization of the RGB** distribution in an image. Normalization gives us a ratio of R, G and B, instead of values. This helps us generalize the hue across different (but similarly toned) lighting conditions.

For example, these colors may seem different, but are simply darker or lighter versions of themselves (the ratio is equal)

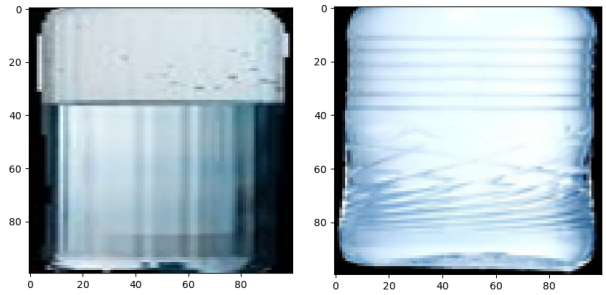


Then, it was intuitive to **clip off the top 20% of the image** so as to eliminate the cap of the bottle, which is often colored differently. This number can be tweaked based on use case.

One more step would be to handle **too bright or too dark pixels**. This is necessary because every camera has a dynamic range, and when we exceed the range too much, we get distorted colors. To resolve this we consider only pixels in the range 5 - 250 (total range being 0-255). This also helps us escape deep shadows and reflecting light on the surface of the bottles.

Using this method, I find two similarly colored bottles offer very little deviance, so much so that 0.1 seems to be a plausible error threshold for R, G and B channels combined.

Example:

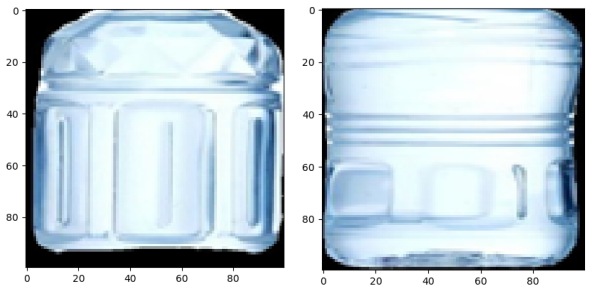


Source R: 0.30, G:0.34, B:0.36
Sample R: 0.29, G:0.34, B:0.38

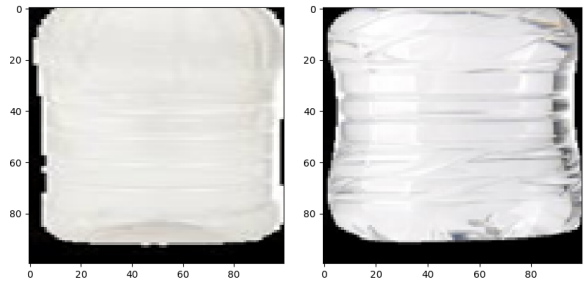
Deviation is calculated per channel as **abs(source_value - sample_value) / source_value**

Deviation of R: 0.04
Deviation of G: 0.01
Deviation of B: 0.05

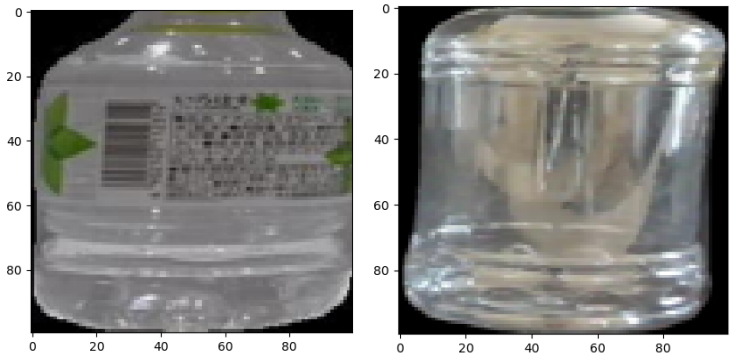
We can confirm this observation across most of the samples we have scraped off the internet.



Deviation of R: 0.01
Deviation of G: 0.0
Deviation of B: 0.01

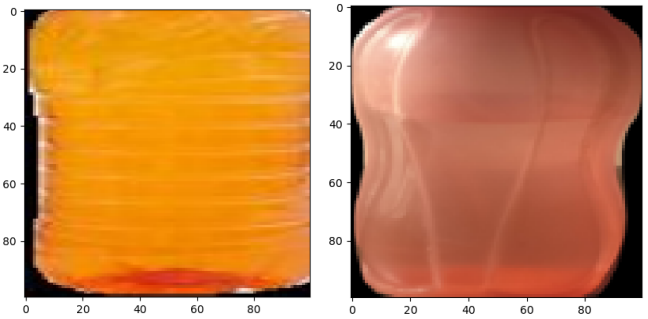


Deviation of R: 0.01
Deviation of G: 0.01
Deviation of B: 0.02



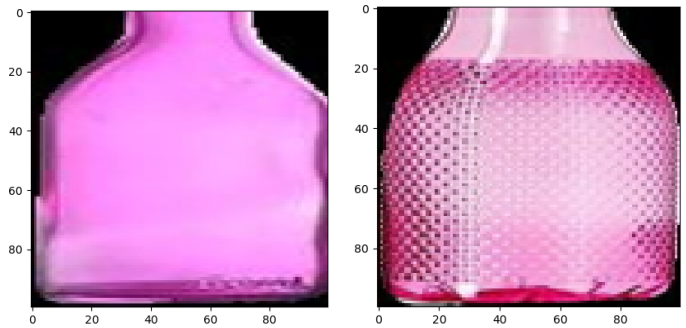
Deviation of R: 0.01
Deviation of G: 0.01
Deviation of B: 0.01

When we try with different colors, such as red and orange, we do get a significant deviation.



Deviation of R: 0.13
Deviation of G: 0.20
Deviation of B: 1.90

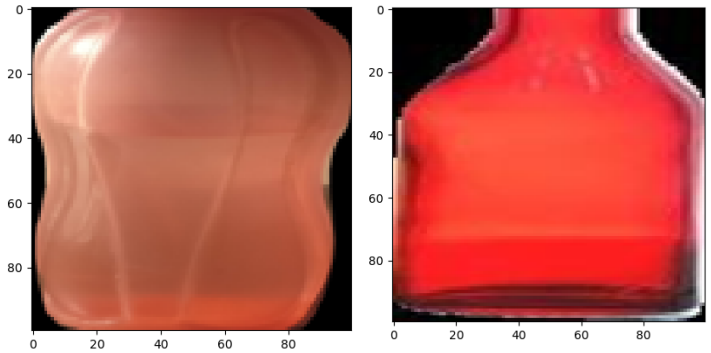
Now that we have established how the algorithm works, let me address one of the problems at hand: different lighting situations for different images of possibly the same color. This time we'll try two pink bottles.



Deviation of R: 0.02
Deviation of G: 0.22
Deviation of B: 0.11

We see the two images have significant deviation in color signatures.This is because the lighting and texture on these images are different. In the present images we see a deviation in hue if we look at the bottom of the bottles. In a production scenario where we match the light and correct the white balance, this is exactly what would help us categorize correctly colored bottles from those that deviate from accurate colors.

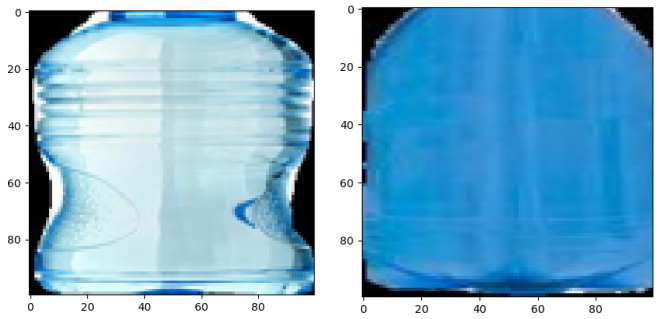
This is again found in two red bottle samples:



Deviation of R: 0.23
Deviation of G: 0.34
Deviation of B: 0.08

The significant deviation in color signatures albeit being of the same color can be attributed to the difference in lighting conditions.

This method can be applied on a conveyor belt set up to make sure bottles are of similar color/consistency/saturation under a certain lighting condition. However, if we wish to classify colors, as in trying to figure out which bottle is of which color, and if two bottles fall under the same color or not, this method does not perform well. Let's look at one such example with two blue bottles.



Deviation of R: 0.65
Deviation of G: 0.05
Deviation of B: 0.44

We notice that even though both bottles are more or less blue, they deviate much in normalized color signature. This is because the lighter blue pixels in the left image have a much less skewed RGB signature than the majority bright blue pixels in the right image. If a light blue pixel is (180, 180, 250), a bright blue pixel would be (20,20,250). These two ratios are vastly different, attributing to the deviation in color signatures as we found.

Personal Observations and Opinion

I feel the problem at hand is one that we are trying to optimize prematurely (a phrase in reference to Knuth's famous quote about this being the root of all evil in programming). Unless we have usable data, I feel this is an effort very much in futility. We cannot explore Deep approaches because we don't have sufficient volume of data. We cannot explore advanced Computer Vision techniques because the data we do have come from various sources and are qualitatively inferior. I recently read in an article that 87% of all ML/Data Science Models never make it into production, and perhaps it is because we keep making models we wouldn't even use. From what I understand, we wish to deploy this model on the cloud, where we can offer it as a service. But, to optimize the error thresholds (which I believe can be reduced to as low as 0.04 under consistent lighting and product textures), we need access to better data to work with.

Summary

To summarize, the method explored in this report can be applied to determine similarity in color signatures in two different objects under consistent lighting conditions with very high confidence. However, this method fails to accurately detect similarity in color if saturations are significantly different.