# Color Palettes: Pattern Recognition and Classification of Images

## Introduction

The introduction is a crucial part of your report and sets the stage as well as motivates the work you have completed. It briefly introduces the problem, emphasizes its importance and shows to the reader what you did in your work. That is, you show the reader that your problem is fundamental and that it needs to be solved as it fills a gap in the known literature and prior work.

The introduction starts by addressing the problem in general and points the reader in the direction how you want to solve your problem / research question. As a metaphor, John Swales developed a three-stage model for research introductions:

[Move 1:] Establish a territory (claim centrality of topic)

[Move 2:] Establish a niche (indicate gap)

[move 3:] Occupy the niche (outline purpose and indicate research structure/methods).

You may follow that model while writing the introduction. In the first part of the introduction or in a separate section, you are expected to mention and organize supporting literature where you outline the {\em state of the art} of your research question.

At the end of the introduction you may include a short summary of the structure of the rest of the paper. The introduction together with the conclusion should give a complete short version of your work and what you have achieved.

## Related work

After the introduction there follows the related work section. Try to classify the related methods in the literature with respect to the problems and solutions in your work. Establish the position of your work with respect to those competing on the same turf. Summarize the core contributions of the most important related prior solutions within the context of your work: if different from yours, clearly indicate the limitations that your solution improves upon; if similar to yours, state its advantage over other methods and thus indirectly justify your method. At the end, it should be clear why a new solution is required to solve the same problem and how your method is different from other related work.

You should know your literature well in order to find the missing link, or the most important unsolved problem, understand your contribution to the field, place your contributions in the midst of other works, and to sell your work.

Select references carefully and organize them into related groups. Within one topic you can organize references from old (first seminal) paper to newest achievements, and if too many related solutions exist, restrict to the last few most important papers if necessary.

## Problem statement

The technical sections introduce and motivate the proposed solution in the light of the problem, which requires a precise problem statement together with any assumptions and requirements. Where algorithmic or mathematical descriptions are not appropriate, other technical and implementation problems can be stated that are to be solved in this work.

## Technical solution

One or more sections should be directed towards the detailed description of the proposed solution, including technical details about the used data structures, algorithms and mathematical methods. The technical description should allow the resourceful and interested reader to reproduce and verify your work, together with the implementation information given in a later section.

This is the most important part of the report that should answer every little technical question that arises in the reader's mind. Your algorithm might be a puzzle with many pieces which are described in a linear order in the report -- and many such orders may be possible. A good order to describe the different components of your solution is one that allows to clearly explain one component after another based exclusively on what the reader has already seen in any previous sections, thus minimizing any forward references.

This core part of the report should be organized into coherent subsections, giving an overview and introducing formalism first. Following an overview of the necessary steps of the entire method, each step can then be described elaborately in each subsection.

Note that implementation details should be avoided as much as possible, and the focus should be on the formal and algorithmic solution; the implementation section is specifically targeted to explain programming details.

## Implementation

The implementation section focuses on the programming problems and details such as the organization of the source code, the dependencies of the different modules etc.

## Experimental results

The evaluation discusses the proposed and competing solutions in the light of the initially stated problem requirements and limitations. This typically involves some sort of experimental evaluation which leads to some type of qualitative or quantitative results.

Quantitative results include observed numbers indicating performance timings (speed) or accuracy measures of the given implementation and test datasets. If possible, statistical tests and analysis should be given, or where applicable formal proofs. Meaningful and informative numerical results must be complete and unambiguous. Explain in detail how the evaluation has been designed, as well as the experimental setup and test cases. This includes accurate description of the test data (type, properties, size etc.) as well as the test setup (e.g. view settings, screen resolution etc.) and the measured variables (frame rate, throughput, accuracy etc.).

Qualitative results may be reported if clear quantitative measures are not feasible or applicable. Qualitative results clearly show the features and functionality of the completed work, indicating if and how they are novel or different from prior work. Qualitative results are especially suitable if something {\em new} has been achieved that no-one has done before in the same way.

Essentially, the goal of the experimental results is to convince the reader by numbers, tests and images (and maybe user studies), giving some sort of proof why the proposed solution is good, different and/or better than other solutions.

## Discussion

Puts the results in perspective, discussing it in relationship to other related work. Indicate possible (side-)effects and eventual limitations due to the evaluation. State the {\em take home message} of the paper that the reader should remember and provide an outlook on possible future work that extends the given solution or fixes specific limitations. Close with a brief description (that is different from the Abstract) of the proposed solution.

Summarize your main findings in one or maximum two pages. Try to keep yourself short and clear. Give a short discussion about your results where you focus on what your findings mean. E.g., show how your results and interpretations agree with the original question and with other published work or if there are any possible practical applications for your work. At the end, give hints on further improvements or development directions / areas.

## Conclusion

##################################################################################

Themes: Aesthetics, Color, Colorization, Color science, Color space, Color palette, Palette, Color scheme, Color contrast, Color Analysis, Color search, Search, Psychological aspects of Color, Technique, Colors, Psychological aspects, Painting, Color in art; Machine Learning, Aritificial Intelligence, Data-Driven Approach to Color,

“Roses are red, violets are blue…” Are they red, are they really blue?

Glossary

(index with links to fachbegriffe in text)

1. Achromatic Colors: Neutral colors defined as having neither hue nor chroma
2. Color Chart: color matrix exhibiting color scales where any two dimensions vary while the third dimension stays constant
3. Colorimetry:
4. Color Scale: a series of colors with gradation in one dimension while the second and third dimensions stay constant
5. Color Scheme:
6. Color Wheel: the visible spectrum of color placed on a wheel
7. Color Gamut: mathematical definition of color in a color space[[1]](#footnote-1)
8. Color Palette:
9. Color Sphere: a spherical color space containing a universe of colors along each color dimension
10. Color Tree: a tree-shaped color space with a trunk and branches of varying length representing color dimensions and containing a universe of colors
11. Cool Colors: greens, blues and violets
12. Chroma: strength of a color
13. Chromatic Colors: All colors other than neutral colors, i.e. having hue and chroma
14. Secondary Colors: Composed of mixing two primary colors
15. Tertiary Colors: Composed by mixing one primary color with an adjacent secondary color
16. Primary Colors: also Component Colors, Original Colors which are needed to derive all other colors
17. Hue: a color (without saturation and value)
18. Intermediary Colors: color obtained from the mixing of primary colors, when secondary they are determined visually by the mid-points between two primary colors
19. Warm Colors: reds, oranges and yellow
20. Value: light of a color
21. Visible Spectrum: all light beams of color that the eye can see

Abstract

This research project lies at the cross-section of computer science (particularly artificial intelligence) and the arts (particularly film and color theory). My contributions have been to design efficient mechanisms for color palette search and comparison (image processing) and to leverage machine learning and search methods in application to web tool design. I developed innovative computer vision / machine learning systems in the area of visual search. I’m involved in data analysis, prototyping, testing and deployment.

Related Work

Most Important Color Theorists

Theorist of color have created color spaces (spheres, solids or trees) from which to locate and quantify every color.

In color science, **Isaac Newton** (1642-1726) fundamentally shaped our modern understanding of light and color. He used a prism to refract white light to different degrees to revolve it into seven primary colors, red, orange, yellow, green, blue, indigo and violet-purple (ROY G BIV), that explains the colors of the rainbow (Figure).[[2]](#footnote-2) He observes that these original colors can be compounded to a variety of intermediate shades.[[3]](#footnote-3) He discovers the visible spectrum of light (400-700 millimicron wave lengths) – also called “gamut” - in a series of scientific experiments.[[4]](#footnote-4)

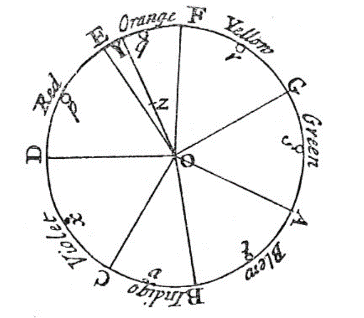


Figure 1: Newton's Seven Primary Colors

**Johannes Wolfgang von Goethe** (1749-1832) builds on top of Isaac Newton’s understanding of color as a physical phenomenon to include the perceptual, psychological and subjective way of viewing colors. He makes contributions to color theory by deriving color contrasts, such as the effect of light and dark on the eye (white to black), the laws of color harmony, such as the three complementary colors (yellow-red to blue, green to red and blue-red to yellow), the achromatism and hyperchromatism of object-glasses, and the augmentation of color.[[5]](#footnote-5) He uses six primary colors to form based on using a prism to elaborate a color star (Figure) – red, blue and yellow are the original colors while orange, violet and green are their complements. He assigns the yellow-blue contrast in color to the plus-minus polarity.[[6]](#footnote-6) He describes a scale of 44 colors attained by the coloring of metals in a physical experiment.[[7]](#footnote-7)

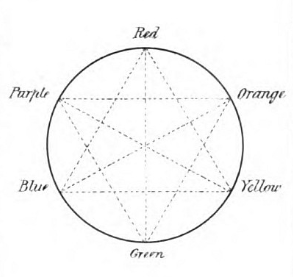
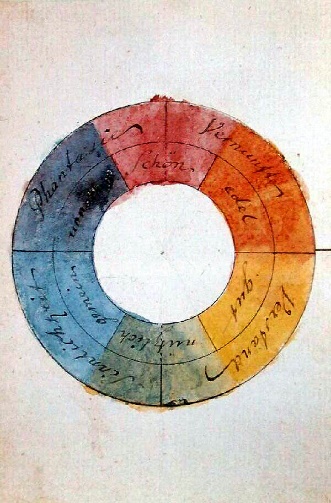
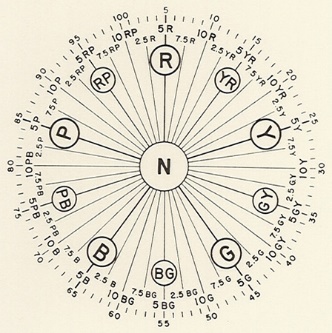


Figure 3: Goethe's Color Wheel

Figure 2: Goethe's Color Star

**Albert Munsell** (1858-1918) develops a color system based on a three-dimensional color space along the axes of hue, value and chroma (similar to saturation).[[8]](#footnote-8) First, he introduces five basic colors: red, yellow, green, blue and purple which are visually equidistant in hue (peeling an orange into sections) and adds another five colors between them (Figure 4).[[9]](#footnote-9) Then, he divides the hue wheel into 100 equidistant steps, sections the value axis into 10 units and suggests and open chromatic scale (Figure 5, Figure 6).[[10]](#footnote-10) He devised this color system to communicate color more easily along a color tree.[[11]](#footnote-11) In the end, he has combined the art and science of color into a single color theory for matching colors more effectively.[[12]](#footnote-12)

Ein Bild, das Text, Karte enthält.

Automatisch generierte BeschreibungEin Bild, das Text enthält.

Automatisch generierte Beschreibung

Figure 5: Munsell's Color Tree – Hue and Value

Figure 4: Munsell's 10 Hues

Figure 2: Munsell's Color Tree - Open-Scale Chroma

**Johannes Itten** (1888-1967) uses subjective feelings and objective color principles to describe his theory of color.[[13]](#footnote-13) His famous “color star” of 12 color hues is used to develop different methods of color contrast.[[14]](#footnote-14) He has a remarkable impact on present-day color theory through his development of color contrasts. He derived many contrasts such as contrast by hue, contrast by value (light-and-dark contrast), contrast by temperature (cold-warm contrast), contrast by saturation, contrast by extension (based on Goethe), contrast by complements (neutralization) and simultaneous contrast.

Ein Bild, das Objekt, Windrad, Zeichnung, Drachen enthält.

Automatisch generierte Beschreibung

Figure 3: Johannes Ittens' Color Star “Farbstern”

Figure 4: Johannes Itten's Color Circle “Farbkreis”

**Josef Albers** (1888-1976) systematically demonstrated color’s elasticity by making one color look like two, two like one and three like two. Studies on simultaneous contrast in color deception. His thought on color relativity: he distinguishes factual colors from actual colors (physical fact and psychic effect). Factual colors are how we name, measure or locate color (theory, rule-based, fixed, absolute, isolated). Actual color refers to how color appears to us when it exists in a context (practice, trial-and-error, fluid, contingent, integrated).

Theresa-Marie Rhyine, Digital Visualization

Benchmark Color Palette Search Tools

1. Coolors, <https://coolors.co/palettes/trending>

Descriptions: Adobe’s Coolors has two main features : generating color palettes and exploring trending color palettes. In searching trending color palettes, a search bar pops up where basic colors can be searched. These basic colors are red, orange, brown, yellow, green, turquoise, blue, violet, pink and grey – a total of 10 basic colors. The searched color name will be added as a tag to the filters. The filters can be extended to more color names in the search query. The searched basic colors are mapped to a set of corresponding HEX colors for fetching corresponding color schemes. It suffices to have one color in this set of HEX colors included in the color scheme for the color scheme to display in the search results. The color scheme has the HEX color shown when hovering over a color patch in the color scheme. The resulting color schemes contain five color patches. These color schemes are made by coolors’ users. The color schemes can be viewed, saved or exported. Also, there exists the option to open the chosen palette in the Coolors color palette generator or copying the URL of the color scheme.

Evaluation: It is also possible to search color schemes by HEX code, but the possibility to search color palettes by HEX code is not made obvious in the search bar. There are no names for the color schemes being displayed.

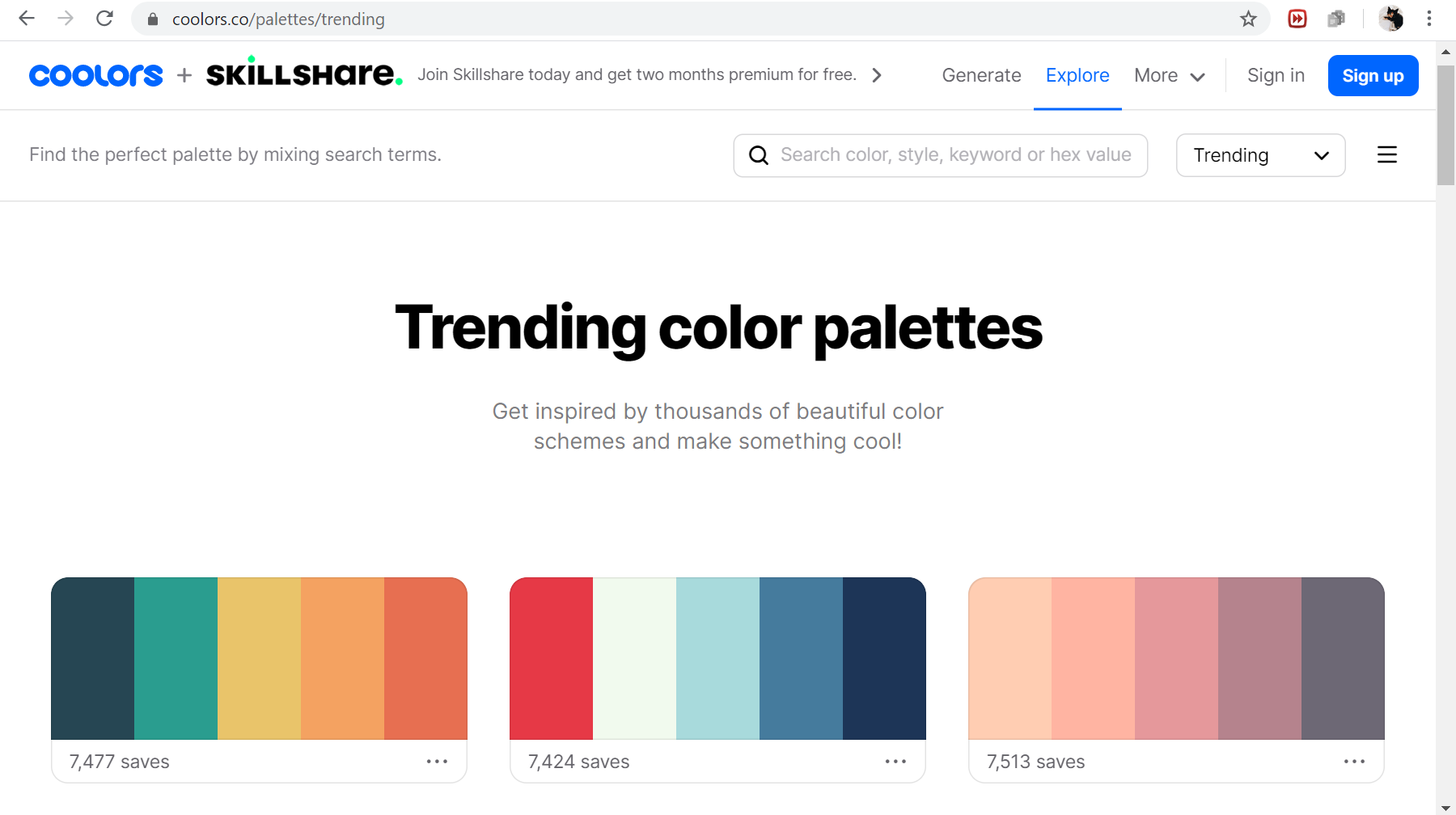


Figure 5: Coolors' Search Palette Webpage

1. Muzli – Designers’ Secret Source, <https://colors.muz.li/color/ccffcc>

Description: Searching for color names, such as “aliceblue”, “dark” or “antiquewhite”, or code (HEX code) in the search bar. Color name is converted to a HEX code. Color name is mapped to one HEX code only. For example, “brown” is “#A52A2A”. Color scheme of five color patches are given in the result. Color schemes are analogic, mono, triade, complementary, tetrade and random.

Evaluation: “Azure” as “#F0FFFF” is not how the general public sees the color “azure”. “Dark” is mapped to “#222222” only, but conceptually “dark” could be any color. In the search bar some color names are recommended in a list after typing the first few letters, however the colors are mostly neutrals, very light or dark only.

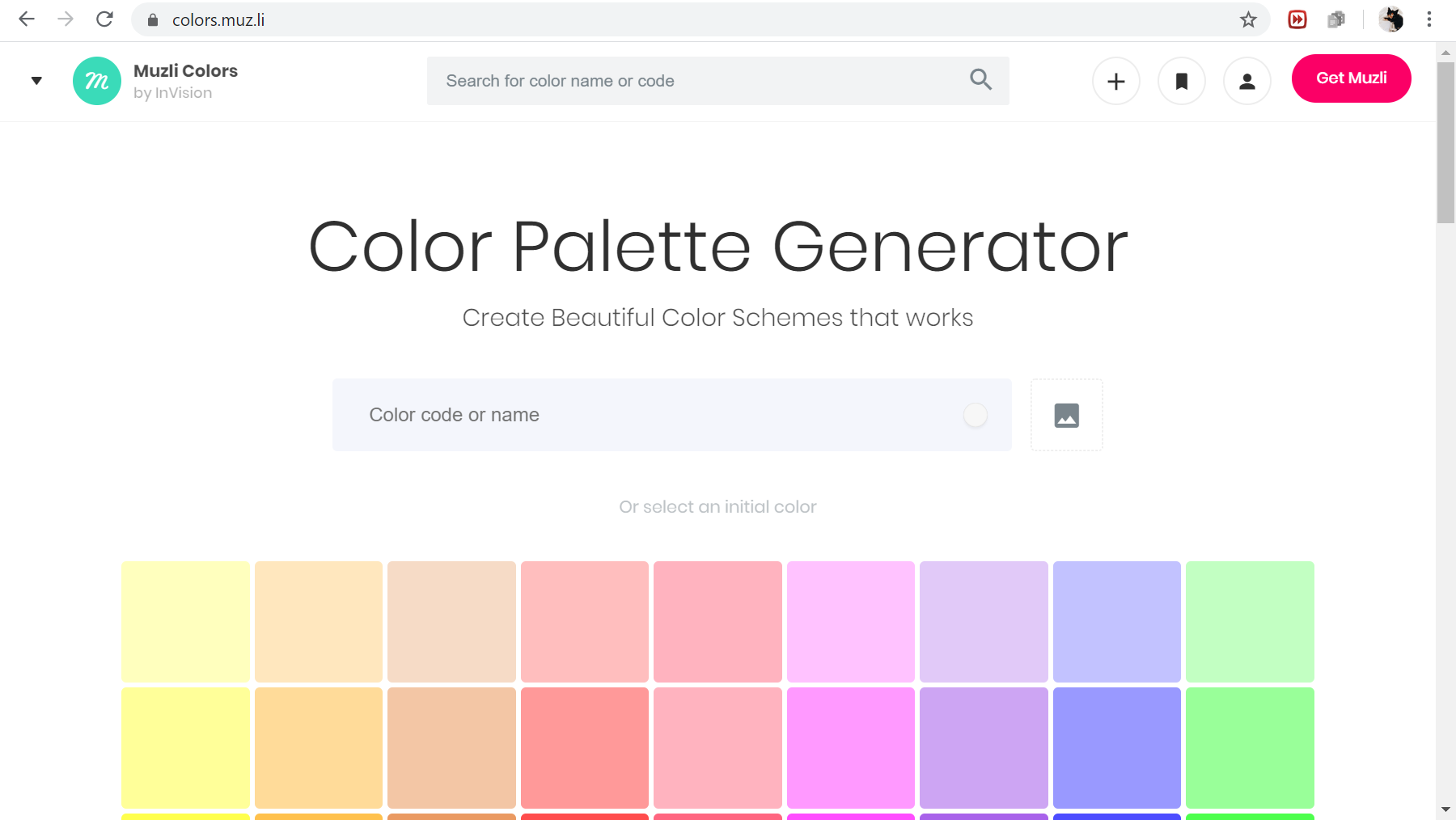
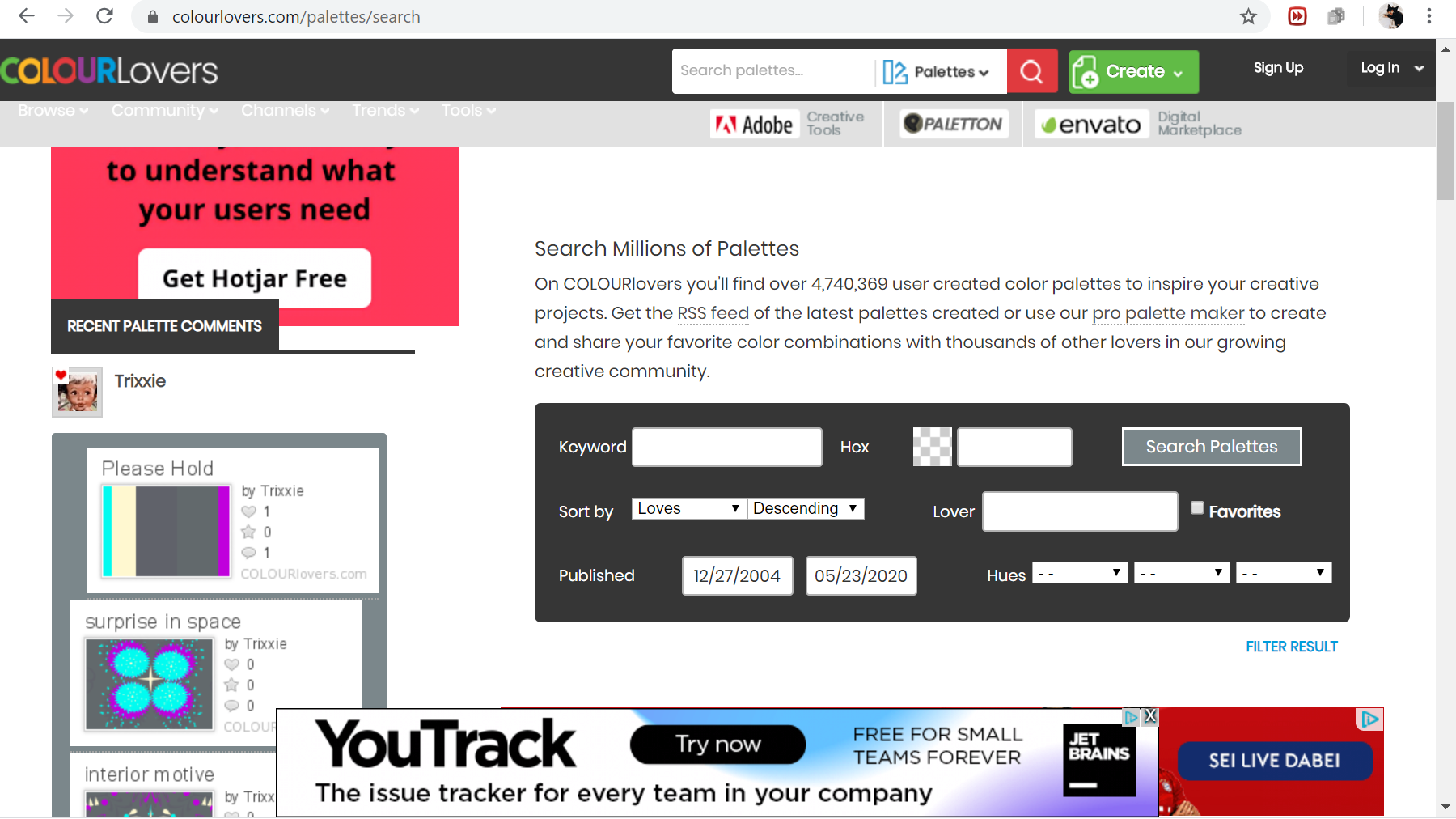


Figure 6: Muzli's Search Palette Webpage

1. ColourLovers, <https://www.colourlovers.com/palettes>

Description: The site has a color palette maker and search tool for exploring user-created color palettes. The searchbar for color palettes is more sophisticated: not only is it possible to search color palettes by keyboard (color names such as “wine”, “chocolate” and “teal”) and HEX code, it is also possible to search colors by RGB and HSV values. The HSV axes are displayed as colorbars next to the color picker window square. The search results are sortable by loves, name, date added, views and comments in descending or ascending order. Also, they can be sorted by hues. The search results can be filtered by publication date. The palettes in the search result are named, the user naming the palette is displayed too. In addition, each color palette comes with information about the number of comments, favorites, view and loves it had attained so far. There are five color swatches in a color scheme.

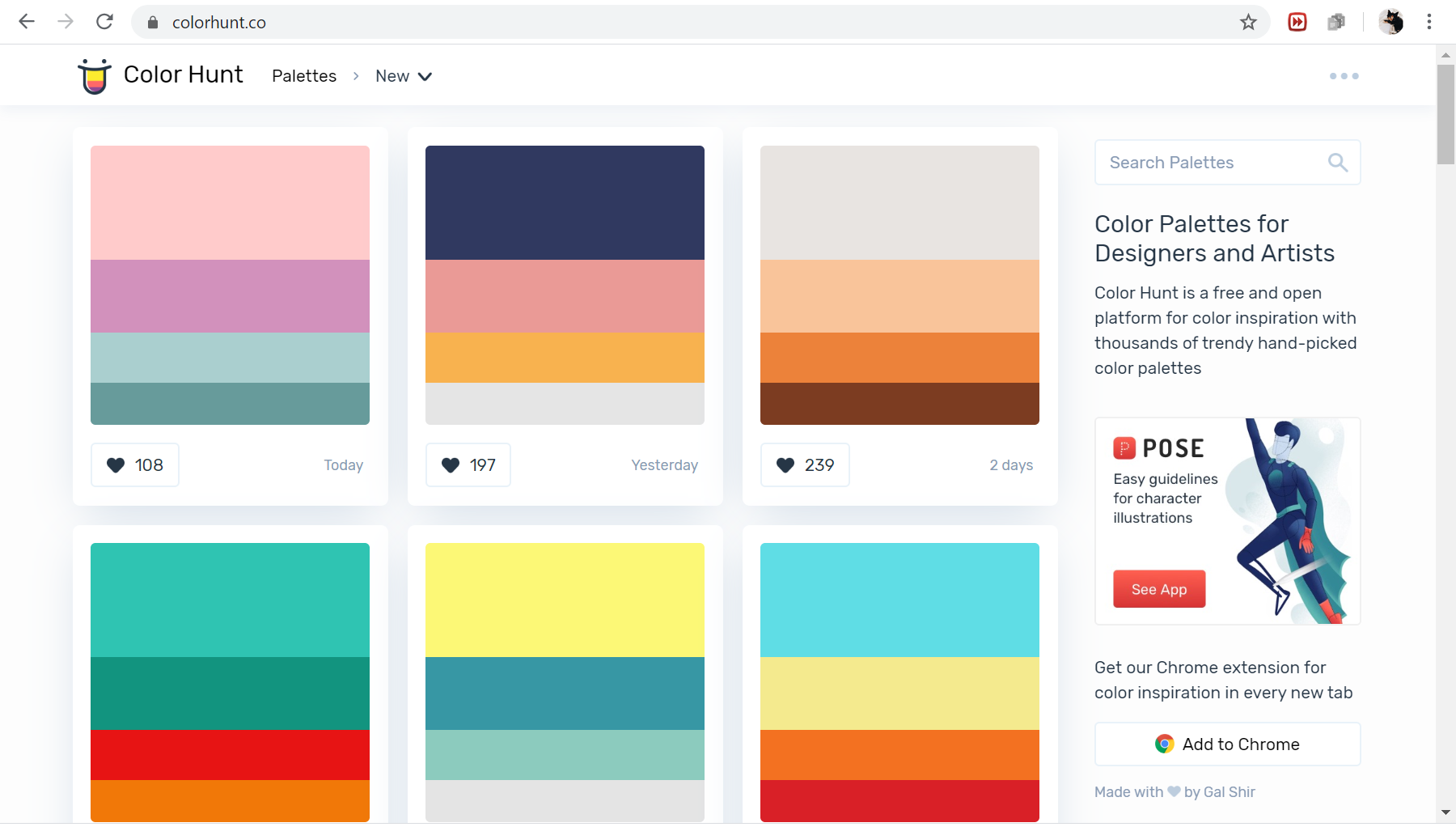
Evaluation: Although this site’s search functionality is the most sophisticated compared to other similar service providers, the mappings of “teal” to the color palettes containing the color is more fallacious than not. However, for “goldenrod” it is surprisingly accurate. In between are searches such as “crimson” with some false results among correct search results. Hence, it is not clear how the color names are mapped to the colors in the color palette. It is not clear why the color swatches vary in proportion in some color schemes and in others they are equally spaced.

Figure 7: ColourLovers' Search Palette Webpage

1. Color Hunt, <https://colorhunt.co/>

Description: The searchbar for color palettes suggest the color hues that are searchable which are red, orange, brown, yellow, green, turquoise, blue, purple, pink, grey, black and white – twelve hues in total. The color palettes are also searchable by contrast such as warm, cold, bright and dark. In addition, the color palettes can be searched by texture such as gold, neon, pastel, skin or by themes such as vintage, retro, wedding, Christmas, Halloween. Searching color palettes by climate such as sunset, summer, autumn, winter and spring is possible, too. The search results contain four color patches and information about the number of likes and the elapsed period since the color scheme has been published online. When hovering over the color swatches the HEX code of the color displays.

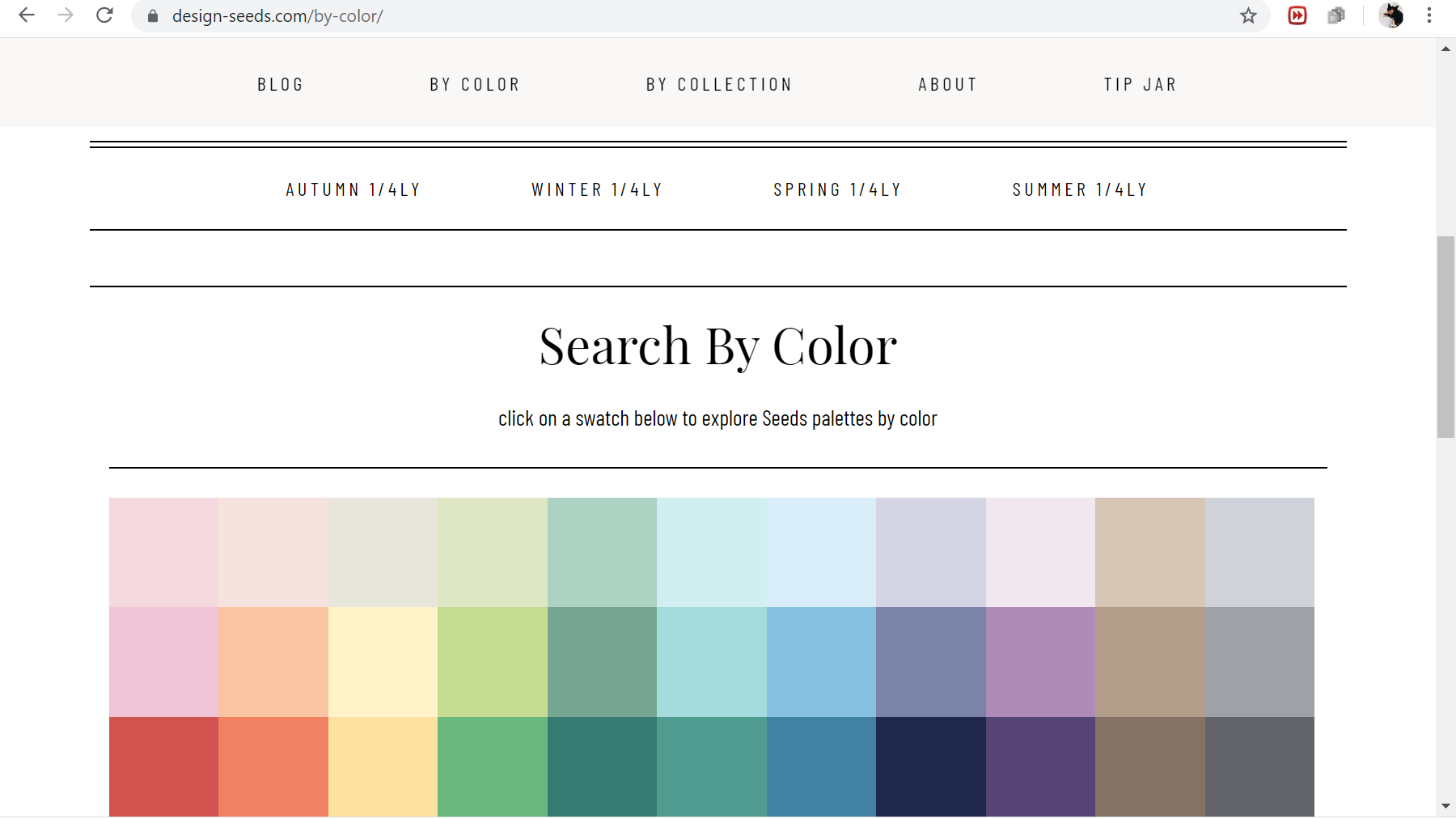
Evaluation: The color palettes are not named. While all color swatches’ proportional size remains equal across the color schemes, it is not clear why the colors occupy varying proportionate spaces within the color scheme.

Figure 8: Color Hunt's Search Palette Webpage

1. Design Seeds {for all who love color}, <https://www.design-seeds.com/>

Description: The search functionality for color palettes takes place by clicking on a color swatch in a 4x11 matrix of color swatches. The color swatches correspond to color names such as “blush”, “ice” or “metal” which are linked to a set of HEX codes.[[15]](#footnote-15) If a color palette has one of this HEX code colors in them, it figures among the search results. The search results contain all the color palettes by name, date of publication, color patches and user name. There are always six color patches within a color scheme. The resulting color of a color palettes are displayed with their HEX code.

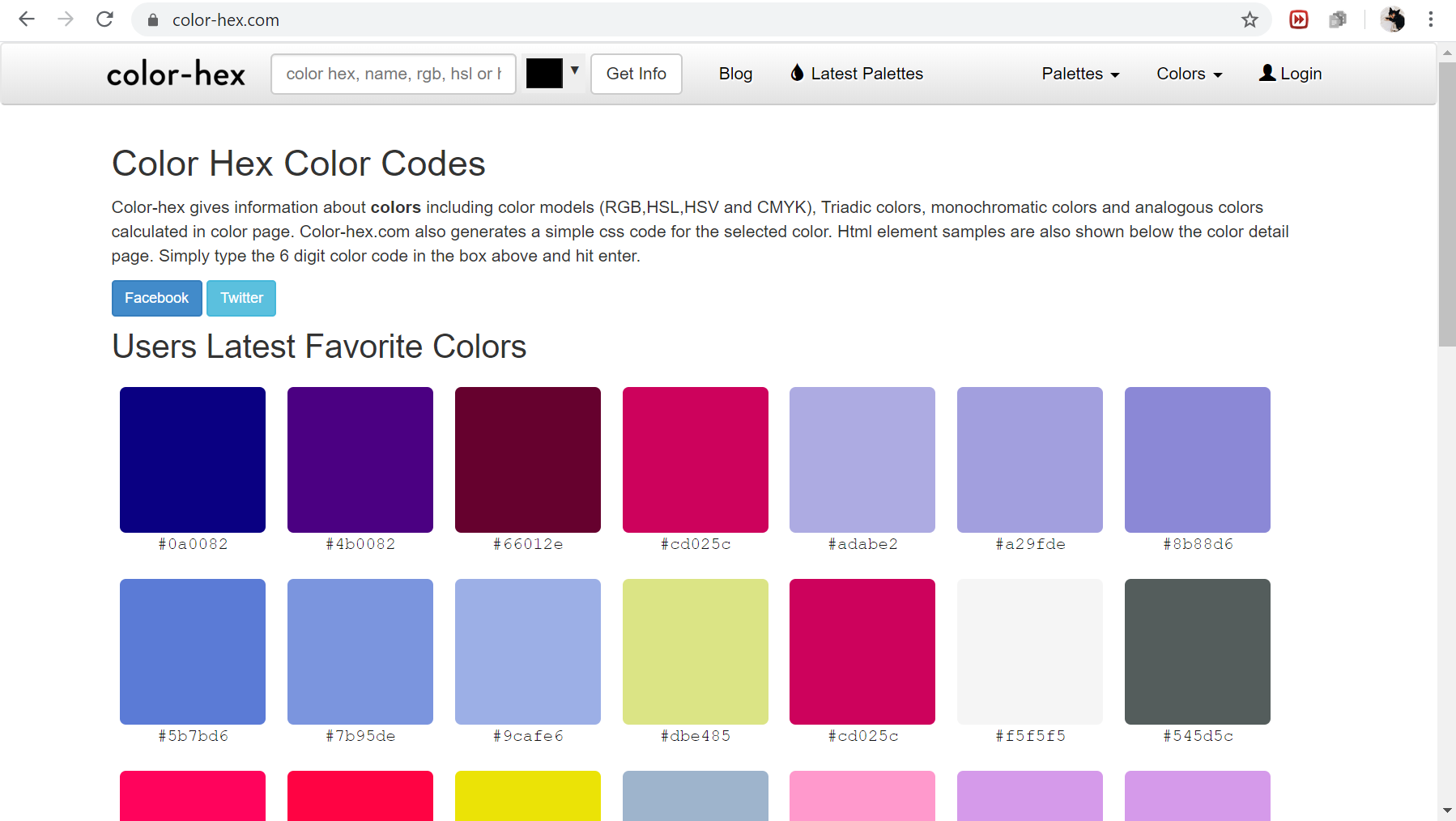
Evaluation: The color names of the searchable colors are not displayed on the searchable color chart. The image from which the color scheme is derived is viewable as well, but one color scheme might take up too much space because of the clean and informative exhibition of the color schemes.

Figure 9: Design Seeds' Search Palette Webpage

1. Color-hex, <https://www.color-hex.com/color-palettes/>

Description: The searchbar takes color names such as “chartreuse”, “babyblue” or “sunflower“, but also proper names such as “Arabica”, “Paris” or “Alps” and adjectives “frenzy”, “playful” or “bitter”. The search returns color schemes (called color palettes) containing five color patches and the name of the color scheme that is matched to the searched keyword. The name of the color scheme usually follows the pattern “left-most-color-name to right-most-color-name”, but descriptive names are also possible. The search results are clickable. On click, the color scheme is shown more prominently, with its title name, the username of the person who created the color scheme, and the color patches of the color scheme together with a list of its HEX code (on hovering as well) and RGB values. The color scheme is downloadable as PNG image. The HEX codes are clickable links that redirects to a page where the HEX code color and RGB values are showcased together with information of the colors values in many additional color spaces (HSL, HSV, CMYK called “process color”, XYZ, Yxy, Hunter Lab, CIE-Lab, web-safe HEX color code). Also the HEX color’s dominant color content (red, green or blue) is described. More information about the HEX color’s base numbers (binary, octal, decimal and hex) are provided in a table. The HEX color is recast to color schemes of shades and tints with eleven color patches each. Also, there are two bar charts that show the relative percentage of the RGB and CMYK channels for the HEX color. The HEX color is recast into color schemes of triadic (3), analogous (3), monochromatic (7) and complementary color (2) with the number of colors for the color scheme given in brackets. The HEX color can be previewed against a black and white background. There is information about the CSS embeddings of the HEX color and the HEX codes of related colors.

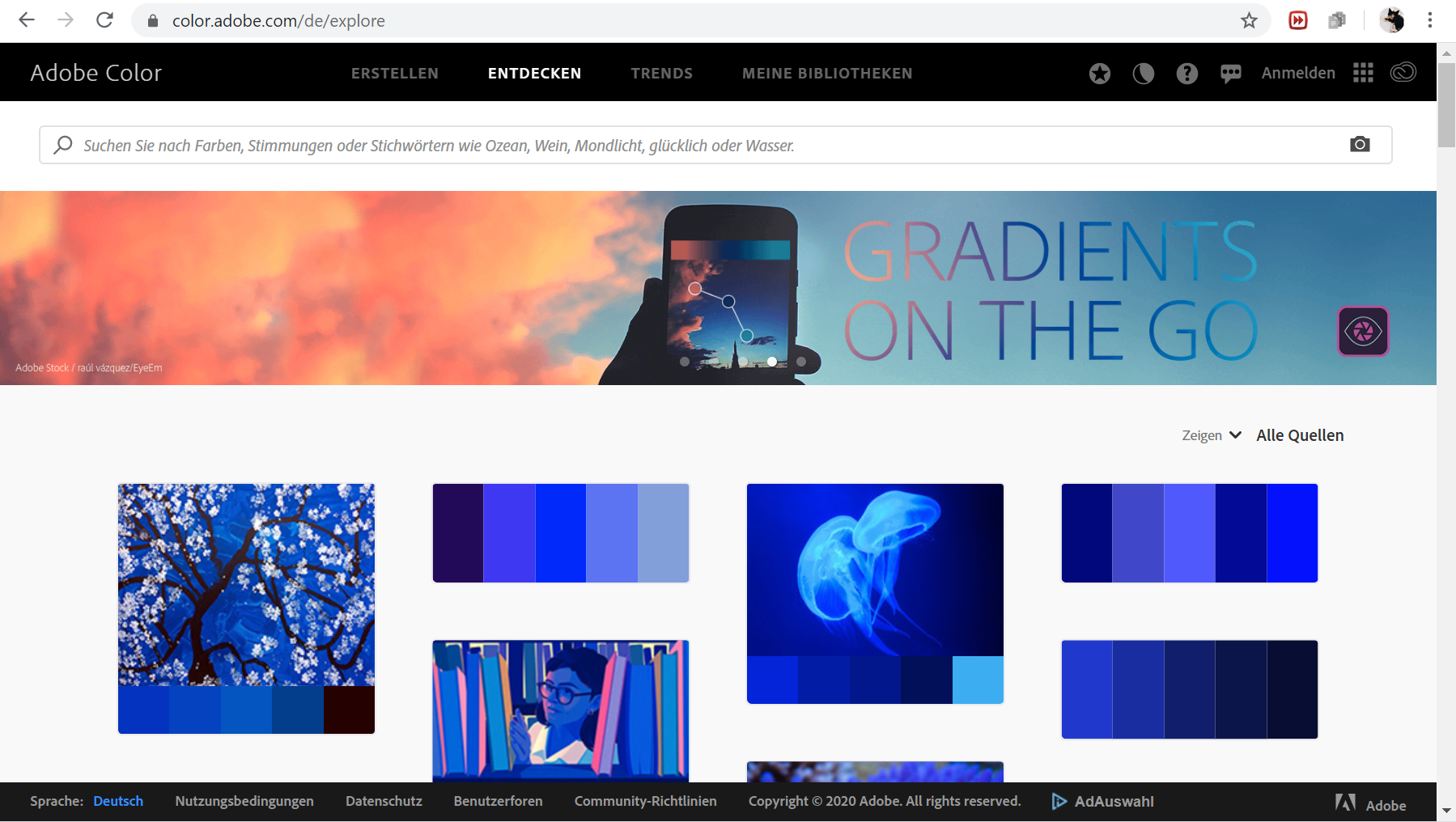
Evaluation: The information given in the redirected pages is exhaustive, but at the same time the search results could have more information on first view for example about the HEX codes of each color in the color scheme and the download functionality. A feature for an easier handling of the color space values such as the immediate copying onto the click board could help the user when working with color schemes. The color schemes in the search result are not sortable.

Figure 10: Color-hex' Search Palette Webpage

1. Adobe Color CC Explore, <https://color.adobe.com/de/explore>

Description: The searchbar prompts the user to enter a keyword based on colors, atmospheres or concrete names such as ocean, wine, moonlight, happy or water. If a search for “sunflower” is carried out, most of the images have sunflowers in them or a sunflower text line. Thus, the keywords do not correspond to the search result’s color schemes, but rather to the objects in the images. First the images are searched in the Adobe Stock of images, then the images’ color schemes are rendered in a succeeding step. The camera icon in the searchbar makes it possible to upload an image on click and search similar images that have most of the same color patterns in common. The images in the search result all have their own color scheme of five color patches displayed at the bottom of the image. Sometimes the color scheme is returned without an image. When clicking on a color scheme, the HEX codes are shown for each color in the color scheme. Functionalities such as downloading or including the color schemes to a library requires an Adobe Creative Suite account.

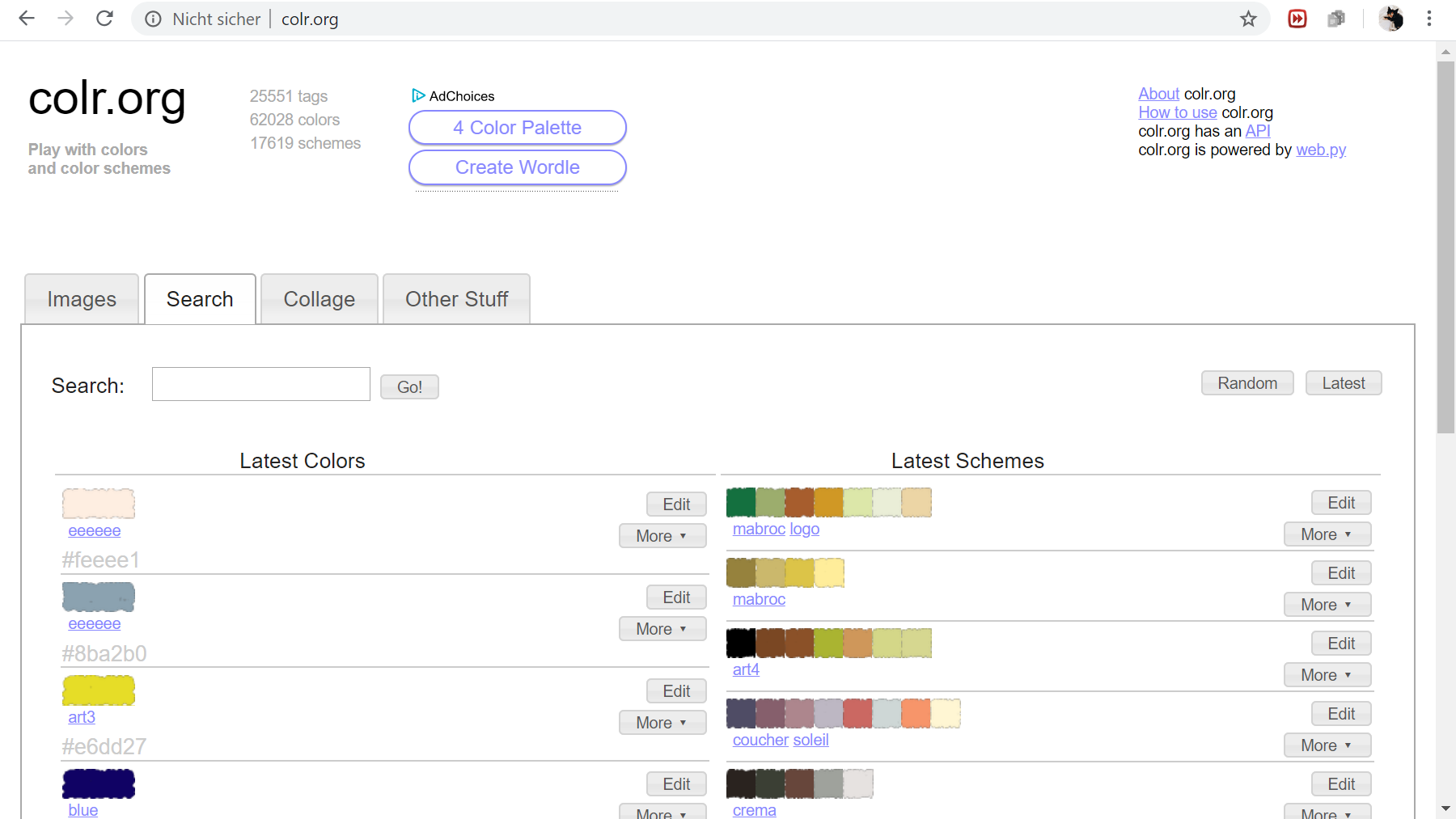
Evaluation: It confuses the user to see these different keyword prompts in the searchbar, because this series of prompted words do not follow a logical sequence. The user cannot infer from the prompts which keywords are searchable and which are not. But, usually such a distinction is the purpose of making keyword prompts in the searchbar. In addition, returning both images with color schemes and color schemes in the search result seems to be too eclectic. On hovering over the image, the color scheme overrides the whole image. This, too, adds too much dynamism to the webpage.

Figure 11: Adobe Color CC Explore's Search Palette WEbpage

1. Colr, <http://www.colr.org/>

Description: The searchbar takes words as keywords and matches them to the tags attached to each color scheme. The user can create their own color scheme. The number of color patches in a color scheme is user-defined as well as the tags assigned to the color scheme. If there is only one color patch, it is called color and it is returned in the search result on the left. If there are more than one color patches, it is called a color scheme and these are returned in the search result on the right. The color scheme can be edited, expanded, similar schemes can be searched and as well as matching images. The color scheme is downloadable as .aco file and there is tag history for each color scheme. The colors of a color scheme are chosen using either a HEX code or a color picker. Matching paints and their color names are given in a separate window.

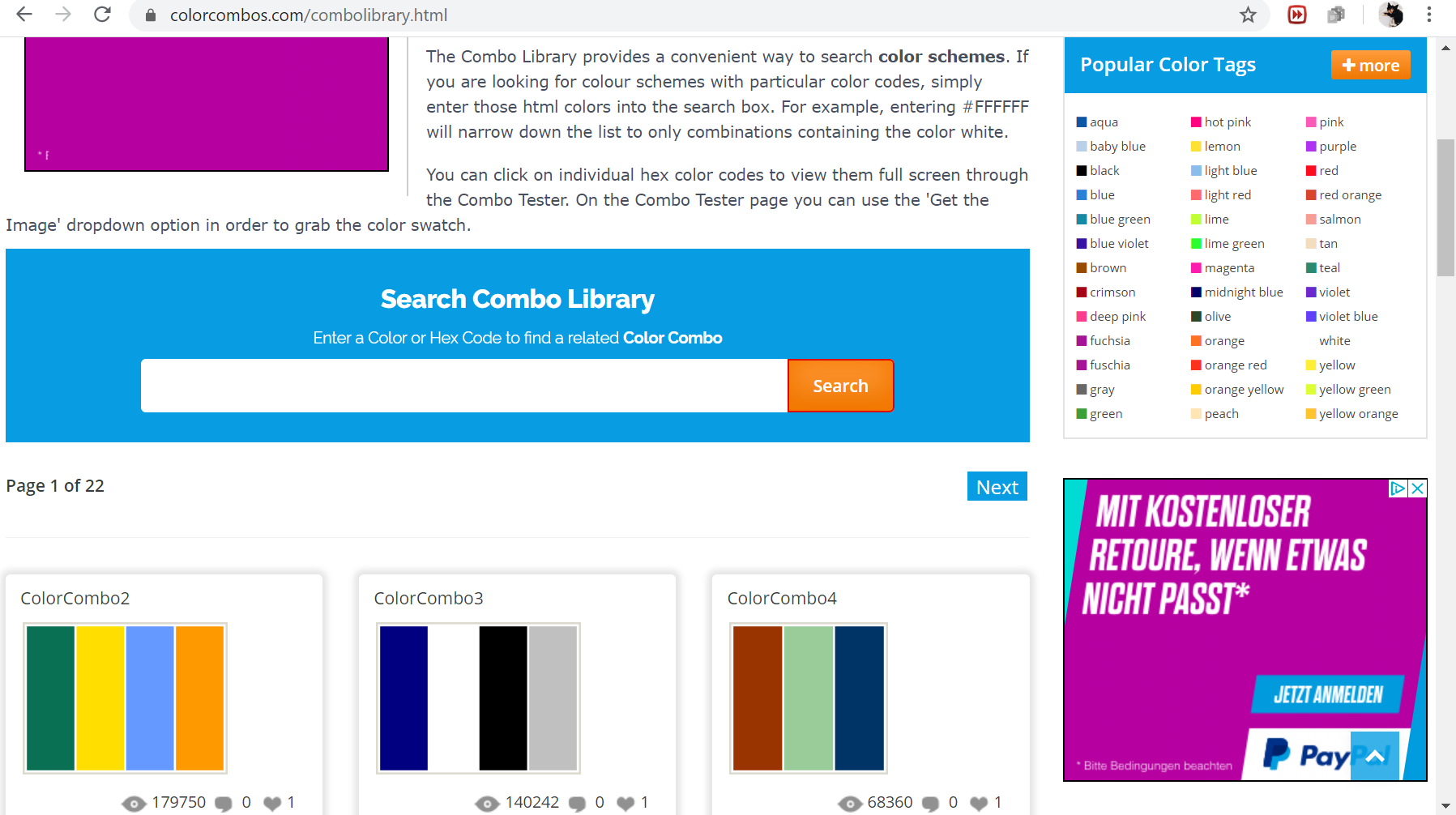
Evaluation: Tagging color schemes involves a lot of time and effort. Since tags are user-defined, any user can assign tags to colors and color schemes. If the matching is not done over a predefined list of color names and their corresponding colors, color name definitions will fluctuate over time and can be arbitrary at worst.

Figure 12: Colr's Search Palette Webpage

1. Color Combos, <https://www.colorcombos.com/color-tags.html>

Description: This search bar prompts the user to enter a color name or hex code to find related color schemes of varying color patches called “color combo”. There is a list of all color names possible in the “popular color tags” section. The search result includes all color schemes, their names “ColorCombo” + a digit, the number of views, the number of comments and the number of likes. The color name is matched to different hex code colors. Each unique hex code color is assigned multiple tags. For example, color code #097054 is assigned the tags 097054, “blue green” and “watercourse”. The word tags can be assigned to a different color code as well. The tag description are defined by the administrator of the website. When clicking on a color scheme, the HEX, RGB, CMYK and HSV color codes are displayed together with the color image and the tags. There are functionalities for testing (editing) and copying the color scheme.

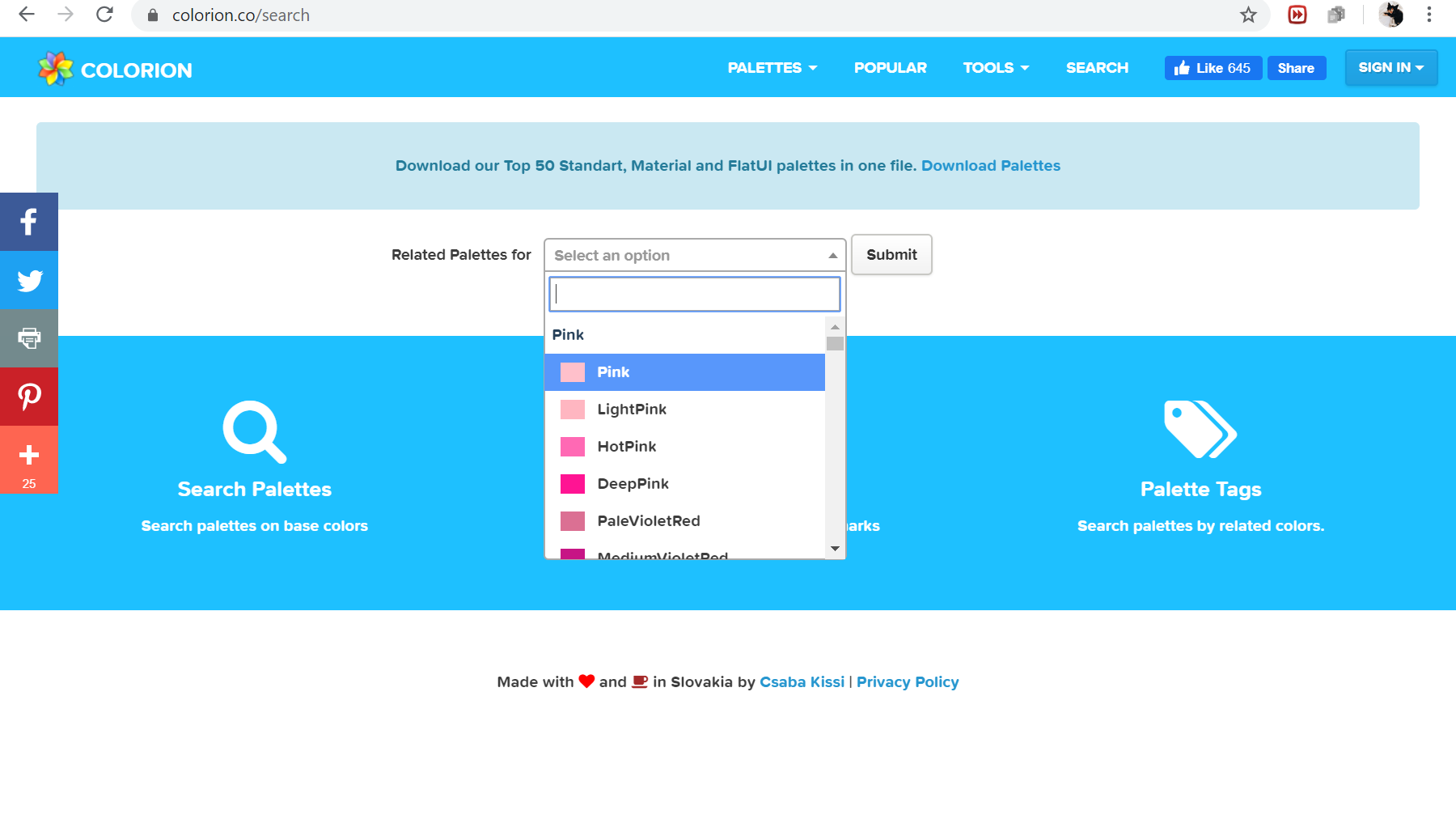
Evaluation: The tooltip could be improved to display only the hex code color of the selected color in the color scheme instead of listing the hex codes all at once. The color schemes rely on hex odes mostly. It is not possible to choose from the menu all possible color names in one go.

Figure 13: Color Combo's Search Palette Webpage

1. Colorion, <https://www.colorion.co/search>

Description: Related Palettes can be searche for a given color. The colors can be chosen from a drop-down menu in the searchbar. The color names are lister under color categories such as pink, purple, red, orange, yellow, green, cyan, blue, brown, white and grey. For example, for pink six discrete color names together with the color image are displayed. The search result contains color schemes of varying length. Below the color schemes different buttons exist for sharing, saving, liking, obtaining more information and downloading the color scheme. When hovering over a color in a color scheme its hex code displays and by clicking the hex code is copied to the clipboard. In the information section, the hex code, the related color, the related hex code and the related color name is given for each color. The relation binds all similar hex code colors to a single color name category. There is color representative for all similar hex codes colors with the same color name. In addition, the RGB and HSL values are listed with type (dark/light), and the colors obtained from darkening or lightening the original color. Finally, the original color’s complementary color is shown. While the hex code has an exact match, the color names are matched to many different hex codes.

Evaluation: The color schemes are not named. The color patches look like location icons. The RGB and HSL values are formatted in a way that is not easily transferrable.

Figure 14: Colorion's Search Palette Webpage

TODO: Alexandra on flowchart on google, Overview of diagram

TODO: video on testtool with histogram of images

Color Space

There are many color spaces. For an extended list please consult the appendix. In this project, only the most important color spaces, RGB (+L), CMYK, HSV, LAB and HEX, will be considered.

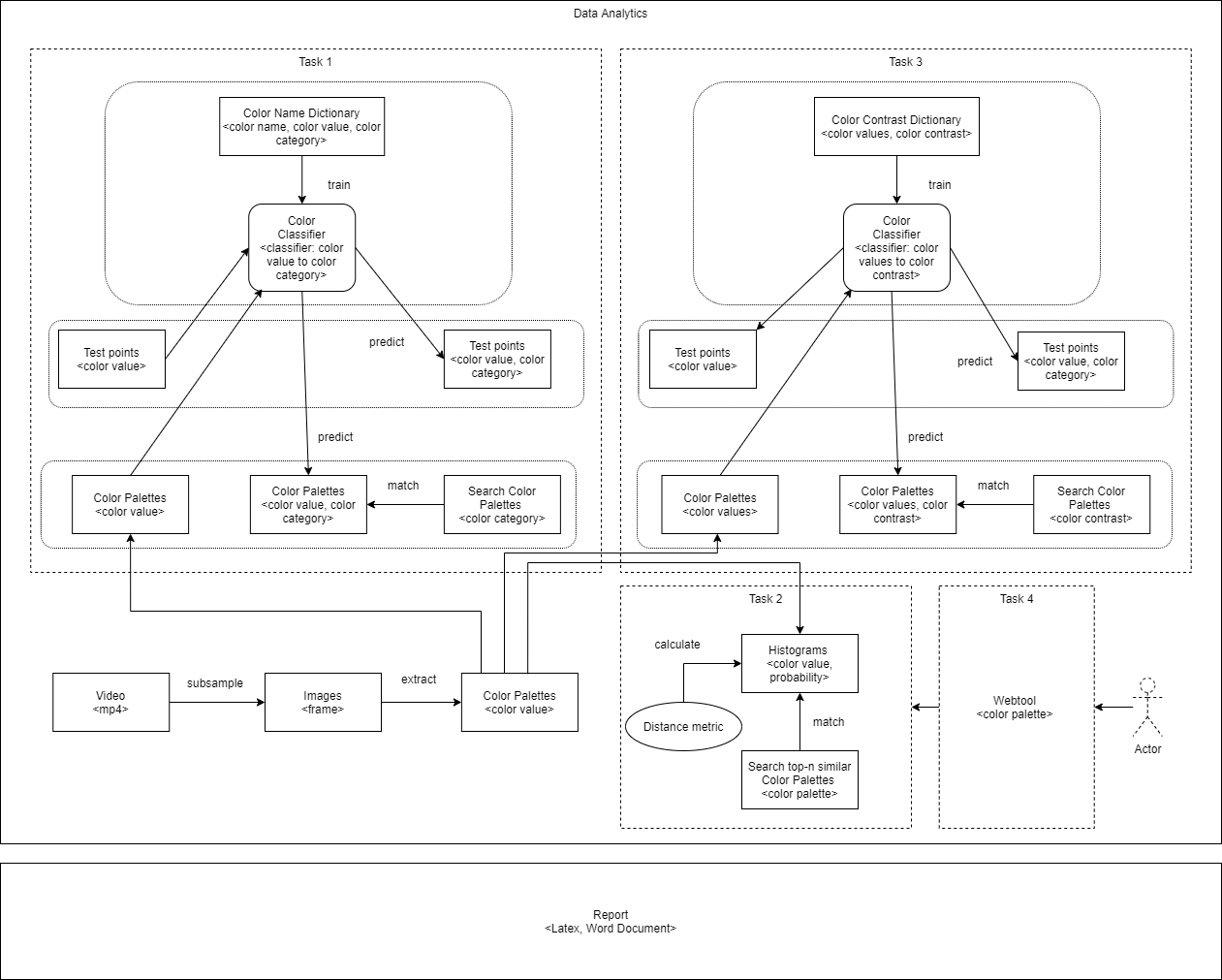


Figure 15: Overview of Tasks

The project consists of the following main tasks:

Task 1: Given a hierarchically structured color palette, classify the contained colors such that a palette is searchable using categorical names.

The FilmColors project’s Visual Annotation webtool (VIAN) predefines categorical color names such as “orange”, “copper” and “mustard” that can be used for the search query. A function in VIAN is able to extract a color palette from an image in hierarchical order, i.e. get n-most salient color patches at the lowest image level and hierarchically merge them together in a tree until the highest level is reached with only one color that is the average color of the image. The task is to make the colors in these color palettes searchable based on the VIAN color name options in a search query.

When defining a set of colors for the search query, color blindness could be a matter of concern in viewing the colors. On Color-hexa ([www.colorhexa.com](http://www.colorhexa.com)), there is a color blindness simulator that displays for a given color how the color is seen by different kinds of color blind users.

Find hidden connections between images: match 1 same color, 2 same colors, … 5 same colors. These images follow the same color pattern.

Color names such as “orange”, “copper” or “mustard” can be used to search color palettes. Such color also present in color palettes which makes them searchable. Color names are categorical, meaning they can be mapped to discrete color values in so-called color name dictionaries. So, these color names will have at least one of the RGB, HSV or LAB color value conversions already present in a color name dictionary, depending on the color space (RGB, HSV or LAB) chosen by the builder of the dictionary. The color spaces mentioned are the most prominent color spaces. If one of the color spaces is missing, the others can be easily derived by computation. For example, the hexadecimal code (HEX code) or the CMYK values are also important color values to have for web-visualization and printing. The color names can be very diverse ranging from colors such as “crimson” or “chartreuse” to “azure” or “mint”. Visualizing the color’s image gives a visual sense of what the colors look like. A color’s hue can be represented by the h-channel in HSV or as categorical name (basic color) for the color that is represented on a color wheel.

Example of different color modi for “sunflower yellow”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Color name** | **Color image** | **Color hue** |  |  |
| “sunflower yellow” | Ein Bild, das Blume, Zeichnung, Vogel enthält.  Automatisch generierte Beschreibung | “yellow” |  |  |
| **RGB** | **HSV** | **LAB** | **HEX code** | **CMYK** |
| (255, 218, 3) | (51, 98.8, 100.0) | (88.22, 1.48, 86.80) | #ffda03 | (0, 15, 99, 0) |

Beyond the color names and their color values, the color name dictionaries can be extended to include classifications of all color names to basic colors such as “red”, “green” or “blue” which by themselves are color names with their own discrete color value in the original color name dictionary.

Color names and their values can vary across languages because of cultural differences. This project will only concern itself with English color names. While the English color world culture is preferred here, any other dictionary, Japanese, Korean or Russian, given the color name and color value should be parsable, too, by replacing the dictionary now in use. Thus, the user is able to select an own color name dictionary.

Below is a table that summarizes the three blocks that incorporate colors. The block’s optional entities are listed in order of importance or procedure.

|  |  |  |
| --- | --- | --- |
| **Color Name Dictionary** | **Basic Color Categories** | **Color Palettes** |
| * EPFL Color Thesaurus[[16]](#footnote-16) (720 colors) | * Itten’s (6 colors) | * Row 20 (101 colors) |
| * XKCD Color Survey (954 colors) | * Boynton's (11 colors) | * Row 1 (1 color) |
| * Goethe Metal Colors[[17]](#footnote-17) (44 colors) | * RGB space corners (6 colors) |  |
| * ColorHexa List of Colors By Name[[18]](#footnote-18) (746 colors) | * LAB space corners (9 colors) |  |
| * ColorCombos Tags Index (1561 colors)[[19]](#footnote-19) | * VIAN (28 colors) |  |
| * Werner’s Nomenclature of Colors (110 colors)[[20]](#footnote-20) |  |  |
| Extension | Extension | Extension |
| * Google Image Search (100 colors averaged to 1 new color) | * Provided Classification: XKCD Color Survey, Werner’s Nomenclature of Colors, |  |
| * Pinterest Image Search (10 colors averaged to 1 new color) | * Last-Word Classification |  |
|  | * Manual Classification |  |
|  | * Machine Learning Classification |  |

A classification of all color names to their respective basic color category can mean that all color names are mapped to one basic color category. For example, color name “blue-green” is classified to “green” because the emphasis on the second word “green” makes the color more “green” than “blue”. Alternatively, more leeway is given to colors at the border between two basic colors. Whether it is a border or rather a gray zone between two basic colors is left to the user to decide. Color name “blue-green”, for example, can be classified to “blue” and “green”. In that case a hard border makes way for a soft border. In machine learning words, a multi-class classification approach turns into a multi-label approach. The user can decide which approach to take.

Turning to color palettes, they too are made up of colors. These colors are present as BGR color values and convertible to any other color space. Using the color name dictionary, these color values can be classified into their color names and basic colors.

Choosing Color Spaces

TODO: justification for LAB

For classification, the colors will be converted into LAB values. This is because it is a perceptually uniform color space. Because the LAB space is larger than all other color spaces, users can search more specifically about a color. This is also the reason why CIE-Lab color values can contain more information about color. For visualization purposes, the colors’ RGB values are displayed. The displayed colors will vary across devices.

Querying Color Names

First, to query a basic color which is matched to a corresponding color name found in one of the colors of a color palette. For example, the user could query “red”, a basic VIAN color. Then, the user gets all color palettes containing this category’s corresponding color names such as “red”, “ocra”, and “blood red” etc.

Second, to query a color names which is matched to the corresponding color name found in one of the colors of a color palette.

TOASK: For example, the user could query “red”, a color name from the color name dictionary. Then, the user gets all color palettes containing color name “red” only. The user could query “ocra” as well, a color name from the color name dictionary. Then, the user gets all color palettes containing color name “ocra” only.

RESULT: not possible because for each color name there needs to be at least three basic colors for the classifier to learn, the more the better.

DESIRABLE:

The user is able to pick a color on a color wheel (color wheel picker), to choose a color value that is then passed to the search query.

The decision boundaries of each basic color are visualized in space using a 3d Voronoi tessellation (knn classifier, k=1) approach. Using triangulation, two VIAN colors such as “cyan” and “blue” are taken, the perpendicular at the middle of the distance between these two points defines the decision boundary. This 3D Voronoi tessellation creates a decision boundary for each VIAN color.

1. A Dictionary of Color Names

A Dictionary of Color Names is a list of unique color names such as “cream”, “seafoam” and “midnight blue” with a color value assigned to it. While the color value given online is typically a HEX code, the preferred color space for sampling and visualizing the colors is RGB (or BGR when taking the inverse of RGB). Since each RGB channel falls into the range of 0-255[[21]](#footnote-21), a Color Name Dictionary (CND) can consist of a maximum of 256x256x256 = 16’777’216 color names. These are all possible colors of an RGB color cube. Other colors outside this range are possible in other color spaces but cannot be visualized for the human visual system. Most color name dictionaries are much shorter in length due to the relative imprecise way of language in dealing with seeable colors.

Color name dictionaries can be found in books or on websites dealing with color conversion. A good source of color name dictionaries are webtools that use them to generate colors. In the following are three examples of sources for retrieving a color name dictionary.

Data Acquisition

1. Werner’s Nomenclature of Colors

Abraham Werner and Patrick Syme (1821) developed the world’s first encyclopedia of colors[[22]](#footnote-22). While Werner described the color and gave examples from nature on where to find them, Syme added color patches to Werner’s color names and published the nomenclature of colors subsequently. This color name dictionary contains 110 color names.

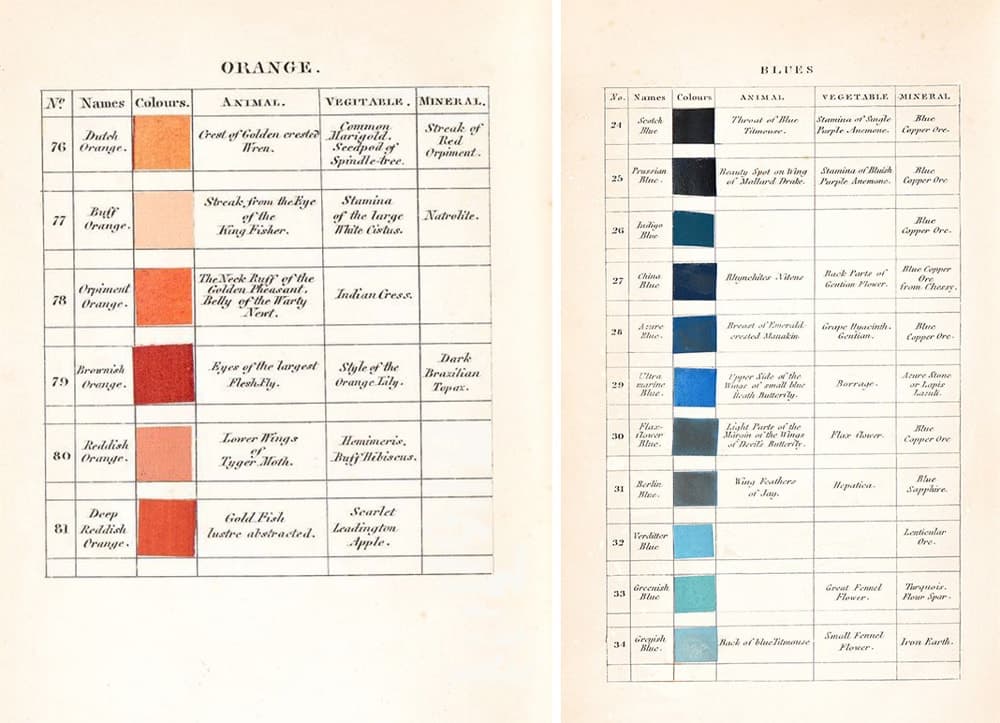


Figure 16: Werner's Nomenclature of Colors

1. The EPFL Color Thesaurus

Albrecht Lindner et al. (2012) at EPFL have created a Color Thesaurus <https://colorthesaurus.epfl.ch/> which contains a color value for color names in a certain language[[23]](#footnote-23) . The dictionary was built based on the first 100 image results on Google Image Search for “color” and “colorname” to derive a color names’s color value. HSV channels as well as sRGB, HEX and CIE-Lab color value are included. The dictionary can be used in conjunction with a color wheel picker.

1. The Color Survey

The XKCD Color Survey (2010) is a dataset with 954 most common RGB monitor colors, as defined by several hundred thousand participants in the color name survey. The participants – including color blind people - were made to name colors. [[24]](#footnote-24) This color name dictionary contains the color names and their corresponding HEX color codes.

Processing

From one of these data sources, a raw color name dictionary is downloaded and saved to the local machine. To process the raw color name dictionary, only English color names are retained (a limitation of this project). For example, for the EPFL Color Thesaurus data set, the raw color name dictionary has 6’167 rows of colors in 9 different languages, their English names, sRGB and CIE-Lab color values. For example, for “baby blue” a discrete color value in sRGB or CIE-Lab is returned. Thus, we process this csv-data set in Excel by removing all non-English color names. There remain 720 colors. This is what is called an original color name dictionary.

The original color name dictionary (CND) needs at least a list of different unique color names and a corresponding discrete color value in any color space (for example RGB, HEX, CMYK, LAB, HSV). Typically, the builder of a color name dictionary provides a HEX color code for each color name. Starting from the original color name dictionary, a fully-fledged color name dictionary (FFCND) can be developed.

A FFCND contains an index of all unique color names, the language of the color name (in this project only English color names are considered, but this can be changed to include other languages as well), the color names and color values (RGB, HSV, LAB, HEX) in it. If one of these attributes is missing, it can be derived by inference or conversion (see ColorNameDict00ColorConversion.py). The following attributes need to be present in the FFCND: id, lang, name, srgb, srgb\_r, srgb\_g, srgb\_b, hsv, hsv\_h, hsv\_s, hsv\_v, lab, lab\_l, lab\_a, lab\_b, hex. Other color spaces can be added such as hsl and lch, but this is not necessary. The color’s image can be plotted by filling up an array with the same srgb value per pixel.

Fully-Fledged Color Name Dictionary: Header, Entries and Type

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Id | lang | name | srgb | srgb\_r | … | hsv | … | lab | … | Lab\_b | hex |
| 1 | eng | Adobe | (249, 168, 27) | 249 |  |  |  | (70, 20, 75) |  | 75 |  |
| 2 | eng | Algae | (81, 90, 50) | 81 |  |  |  | (37, -11, 22) |  | 22 |  |
| 3 | eng | Amber | (184, 97, 25) | 184 |  |  |  | (51, 31, 52) |  | 52 |  |
| … |  |  |  |  |  |  |  |  |  |  |  |
| Int | Str | Str | Tuple | Int | … | Tuple | … | Tuple | … | Int | Str |

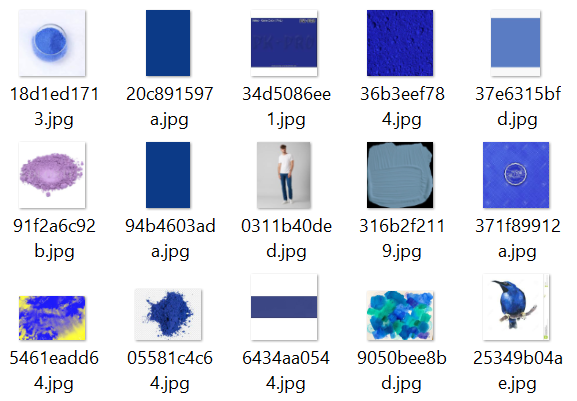
Extending the Dictionary

Usually a new color name needs to be added to the dictionary because none of the previous color names were mappable to one or several basic color categories. For these, corresponding color names need to exist. For example, except for the basic color category “ultramarine” (one of VIAN’s color name search keys) all color names in a color name dictionary were classifiable into one of the basic color categories. One-to-many color names can be mapped to a basic color category.

If a basic color is not found in the color names in the color name dictionary, an additional approach is needed to incorporate a new color name inside the color name dictionary that would be classified to a new basic color.

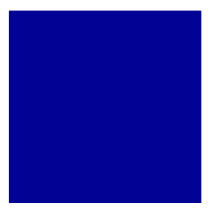
Finding a Color Name’s Value

For example, the color name dictionary from the Color Thesaurus project was built using the first hundred Google Image Search results for a given color name. Using the same approach, a home-bread script to login to Google Image Search to fetch the first one hundred images for color name “ultramarine” in English (see ColorNameDict000ImageDownloadGoogle.py). The search query as described by Lindner et al.[[25]](#footnote-25) was the “color name” and the word “color” in English.[[26]](#footnote-26) For “ultramarine” this would mean to insert “ultramarine color” in the search bar. Below are sample images for “ultramarine”. It can be adapted for any other color, if the set of VIAN colors should ever be adapted to include new color names.

First 15 out of 100 first Google Image Search Results for Color “ultramarine”

We crop the image where possible to extract the center image – this makes sense, because most of the images have white surroundings. When calculating the average color across all images, the white would tint the ultramarine into a diluted color value – a bias that needs to be avoided. As a result, we get a very unbiased, saturated ultramarine color with RGB color value (2,3,150).

Ultramarine – RGB(2,3,150)



Instead of Google Image Search, images can be downloaded from other search engine image search hosts such as Bing Image Search, Yahoo Image Search or Getty Images Image Search Engine. Besides search engines, image posts can be leveraged from social media platforms. For instance, images can be sourced from Pinterest, Flickr or Instagram (see ColorNameDict000ImageDownloadInstagram.py). as well.

There are webtools that take a color name as input and return a collection of color images and their HEX codes for it. An example of such a webtool is Picular, <https://picular.co/>. Picular matches a given color name to a set of corresponding HEX codes. The user can decide on which HEX code color for “hot pink” to take or the average of all HEX code colors could be computed. For colors that are not typically associated with “hot pink” averaging the HEX colors would result in a biased “hot pink” color.

Once the new color name’s color value is determined in all color spaces, the FFCND is extended by an additional row (see ColorNameDict00ColorConversion.py). Note that color names in the FFCND have a unique color value, while on Google Image Search a color name can have different color values.

The issue with adding only one new color name for a basic color is that the basic color will have only one single color name representative in the data set, namely basic color “ultramarine” is represented by color name “ultramarine”. In machine learning, every basic color category needs to have at least three (KNN) to five values in a class depending on the classifier. Hence, “ultramarine” needs to have more color name representatives.

Finding Color Names

Color names are not derivable by use of mathematics such as color values. Instead, color names are more vague; they leave room for associations about colors, often making use of poetic imagery or scenes from nature.

Finding new color names are made easy using webtools such as chir.ag’s “Name that Color”, <http://chir.ag/projects/name-that-color/>. It takes a HEX code value of a color and returns the corresponding color name. Hence, the HEX codes for the most salient ultramarine color images can be entered for finding related color names.

Another tool is Color-Blindness’ “Color Name & Hue”, <https://www.color-blindness.com/color-name-hue/>. It comprises 1640 different color names. It is possible to enter RGB, HSB and HEX codes for finding related color names.

1. Basic Color Categories

The color names in a color name dictionary need to be classified into a color category. This color category forms a set of basic colors to which all color names can be categorized. The purpose of having basic color categories is to group color names together. Hence, each color name can be related closer or further apart from another color name, depending on its basic color category.

Definition

Basic colors, or original or component or primary colors, are categories that group other color names together. For instance, “olive”, “seafoam” and “lime” can all be classified into the color “green” which makes “green” a basic color. Typically, there should not be more than a dozen basic colors. How many and which of these basic colors should make it up to the list depends on different schools of thought in color theory.

Systems

In his book, color theorist **Itten** derives six basic colors “red”, “orange”, “yellow”, “green”, “blue” and “violet” by looking at a prism infused with light. Another way to define basic colors is by looking at different color spaces and their corners. For example, in **RGB space**, we find the following six colors in each corner of the cube: “red”, “green”, “yellow”, “blue”, “violet” and “cyan”. In the same vein**, LAB space** corners contain nine colors “blue”, “cyan”, “yellow”, “brown”, “pink”, “orange”, “red”, “green” and “magenta”. On the other hand, **Boynton** proposes eleven colors “that are almost never confused”[[27]](#footnote-27). They are “orange”, “brown”, “gray”, “pink”, “magenta”, “yellow”, “green”, “red”, “blue”, “white” and “black”. Since “black”, “white” and “gray” have no chroma, they are all positioned at the same location on the color wheel.

In the FilmColors project, however, there are 28 basic color categories which are searchable on VIAN. Most of the color names are warm colors and some of them are too similar to form color categories of their own. For example, “peach” and “cream” are basic colors, even if their color image is very alike.

|  |  |  |
| --- | --- | --- |
| **Origin** | **Basic Colors\*** | **Frequency** |
| Johannes Itten | “red”, “orange”, “yellow”, “green”, “blue”, “violet” | 6 |
| RGB space | “red”, “green”, “yellow”, “blue”, “violet”, “cyan” | 6 |
| LAB space | “blue”, “cyan”, “yellow”, “brown”, “pink”, “orange”, “red”, “green”, “magenta” | 9 |
| Werner’s Nomenclature of Colors | White, gray, black, blue, purple and violet, green, yellow, orange, red, brown | 10 |
| Robert Boynton | “orange”, “brown”, “gray”, “pink”, “magenta”, “yellow”, “green”, “red”, “blue”, “white”, “black” | 11 |
| VIAN (FilmColors) [[28]](#footnote-28) | amber, apricot, beige, black, blue, bronze, brown, copper, coral, cream, cyan, gold, green, grey, lavender, magenta, mustard, orange, peach, pink, purple, red, rust, sepia, silver, ultramarine, white, and yellow | 28 |

\*Colors are sorted in order of appearance or alphabetically

Methods

There are three techniques by which color names can be classified:

1. Provided classification
2. Last-word classification
3. Manual classification

There are two methods by which color names can be classified:

1. Multi-class classification: every color name has exactly one-color category
2. Multi-label classification: some color names can have two color categories

Processing

The fully-fledged color name dictionary (FFCND) consists of color names that need to be categorized into one-to-many basic colors. Sometimes the color name dictionary has already provided groupings of colors into basic color categories. In that case the classification is provided. For example, Werner’s Nomenclature of Colors clusters color names together into ten different basic colors. If this does not exist, last-word classification can speed the categorization process up: for two-named colors such as “purple blue”, the color is categorized using the second word “blue”. For example, for “apple green” the corresponding basic color is “green”. All colors that are not derivable by a second word match or identical match by last-word classification (ColorNameDict01LastWordClassification.py) were manually classified into basic colors (ColorNameDict02ManualClassification.py).

For multi-class classification, the first column of a basic color category suffices, while for multi-label classification both columns containing basic color categories need to be considered. The second column, from a multi-class perspective, is a basic color category that is the second most preferred color category for a border-line color. In multi-label classification, it is assumed that both color categories have equal weight, however. Technically, a third or fourth, etc. basic color categorization could be created, which could mean that this color would be situated at the border of three or four basic colors. This can happen for a color at the end of a border line, for example, dividing two basic colors into two. For simplicity, we assume that all colors are on a border between two basic colors only.

Any two basic colors which have border colors in-between are not derivable by computing all possible combinations of basic colors when given the set of basic colors. For instance, there exists two basic colors “orange” and “blue” that do not share a border due to their complementarity. Orange is at the opposite extreme of blue. “They cannot blend into each other, just like you cannot simultaneously take the directions right and left at an intersection.”[[29]](#footnote-29) Calling something orange-blue or blue-orange is equivalent to saying gray. Color borders can only be made out by using the color wheel. When plotting all basic colors on a color wheel the nearest neighboring basic color is the color with which border colors can be formed.

The FFCND is extended to an EFFCND with two more columns for basic colors: cat1 and cat2. In total a EFFCND needs to have the following attributes for a given system of basic colors: id, lang, name, srgb, srgb\_r, srgb\_g, srgb\_b, hsv, hsv\_h, hsv\_s, hsv\_v, lab, lab\_l, lab\_a, lab\_b, hex, cat1, cat2. (HSL and LCH color spaces can be added optionally.) The color’s image can be plotted by filling up an array with the same srgb value per pixel.

Extended Fully-Fledged Color Name Dictionary: Header, Entries and Type

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Id | lang | name | srgb | … | hex | cat1 | cat2 |
| 1 | Eng | Adobe | (249, 168, 27) |  |  | “orange” |  |
| 2 | Eng | Algae | (81, 90, 50) |  |  | “green” |  |
| 3 | Eng | Amber | (184, 97, 25) |  |  | “orange” |  |
| … |  |  |  |  |  |  |  |
| Int | Str | Str | Tuple | … | Str | Str | str |

Just like the color names have color space values computed in the EFFCND, the cat1 and cat2 color names’ values can be found by averaging all related color names’ color values (or codes) for a given color space. An alternative method is to search the color name in the name column and visualize the color with the corresponding color values, for example by filling a patch with srgb values. These values are not included in the EFFCND to save space and memory, but instead are evaluated on-the-fly in LAB space.

Multi-labels

The colors at the border between two basic colors are difficult to categorize. Such border colors can have many basic color categorizations. When classifying border colors into basic colors, the second basic color category is already provided in the original CND with color names such as “bluey green”, “orange red” or “yellow-green” – these are all two-base-word color names that can be directly extracted (see ColorNameDict03MultiLabelClassification.py). Otherwise, based on manual classification the colors are scanned by hand, finding border colors and appending a second basic color to the first basic color category is another viable option. Yet another technique is to use machine learning classification for each color in the CND and if the prediction differs from the first basic color, the prediction is appended to that color in the second basic color column.

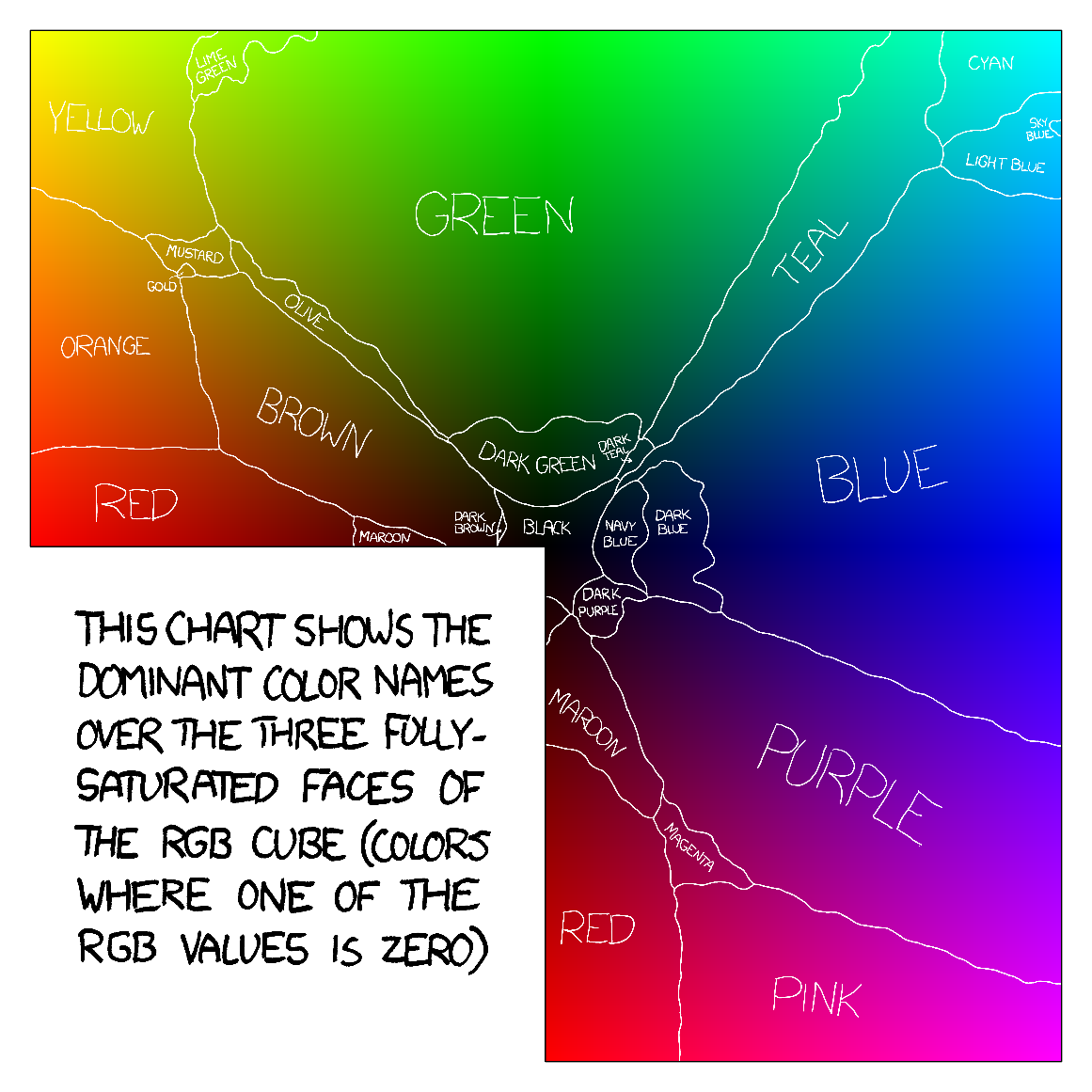


Figure 17: Color Survey Color Borders

Recommendations

Based on these differing systems of defining basic colors, a set of basic color categories is proposed which takes into account the preceding considerations about basic colors to make color categorization as workable as possible. These are “pink”, “magenta”, “red”, “orange”, “yellow”, “brown”, “green”, “cyan”, “blue” and “violet” – ten basic colors. It is assumed that “white”, “gray” and “black” are not colors because they occupy the same location on a color wheel.

When deciding on which colors to incorporate in the search request, another interesting idea would be to include the Pantone® Color Institute’s Color of the Year.[[30]](#footnote-30) The Color of the Year 2020, for example, is “classic blue”. The user could obtain all color palettes (and by extension all images) for which “classic blue” constitutes a distinct part.

Visualizing Basic Colors

The colors in a color name dictionary can be visualized together with their basic colors. The color names of EFFCND can be plotted one-by-one or in a color cloud depending on the color space chosen. The same can be done with the color names’ color categories, the basic colors. The class center average of each basic color category is computed using the average of color values of all color names that have the same basic color category.

In this example, the EFFCND “EPFL Thesaurus – VIAN” is used. The color names are based on the EPFL Color Thesaurus and the basic colors are taken from the VIAN webtool. Note that this EFFCND is interchangeable with other color name dictionaries.

Below, the three first color names are visualized together with their basic color category.

Top 3 Colors and their basic colors in the EFFCND “EPFL Thesaurus – VIAN”

|  |  |  |  |
| --- | --- | --- | --- |
| **Color name** | **Color image** | **Basic color (VIAN)** | **Color image** |
| ‘adobe’ |  | ‘orange’\*  \*basic color’s values found as color name |  |
| ‘algae’ |  | ‘green’\*  \*average of all color names’ CIE-lab values with that basic color |  |
| ‘amber’ |  | ‘orange’ |  |

When plotting all colors in an EFFCND into different color spaces, the colors’ basic colors are used for coloring in their data point dots. This is done for each color space: RGB, HSV and CIE-lab. The EFFCND uses as source the EPFL Color Thesaurus and as system VIAN’s 28 basic colors. Plotting all colors into RGB, HSV and Lab space yields the following 3-D plots:

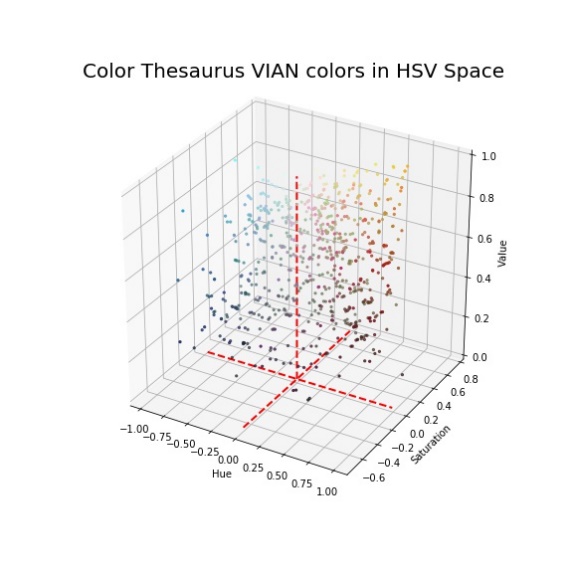
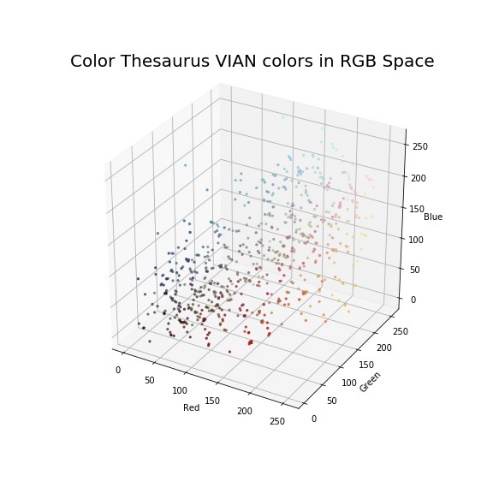
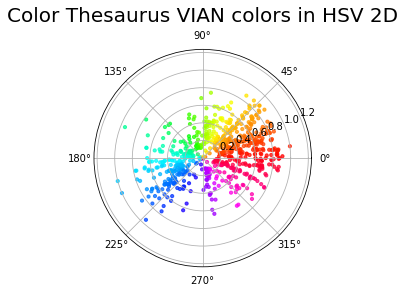
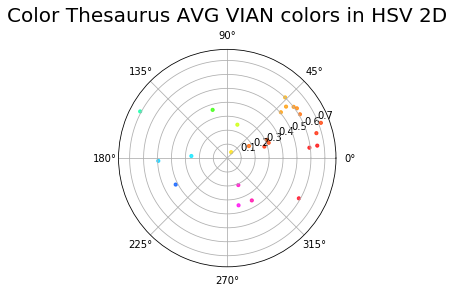


Figure 18: Color Name Dictionary Values in RGB, HSV and LAB space

In HSV space, the colors are situated on a hue circle because the HSV cone has a circular form at the top. Plotting them in 2D can give a better overview of the color hue distribution in an EFFCND. To the left all colors plotted while to the right the basic colors are shown by taking the average of all corresponding colors in the EFFCND. (A less exhaustive method would be to plot the basic color found inside the color names, but some basic colors in VIAN such as ‘green’ are not included in the EPFL Color Thesaurus.)



The VIAN colors are biased towards orange and red color tones on the color wheel. These are predominantly warm colors made available for the user to search. If we take the averages of all CIE-Lab VIAN color values for a VIAN color category and plot the resulting color class centers, we find mostly earthy warm color hues in the color scheme.

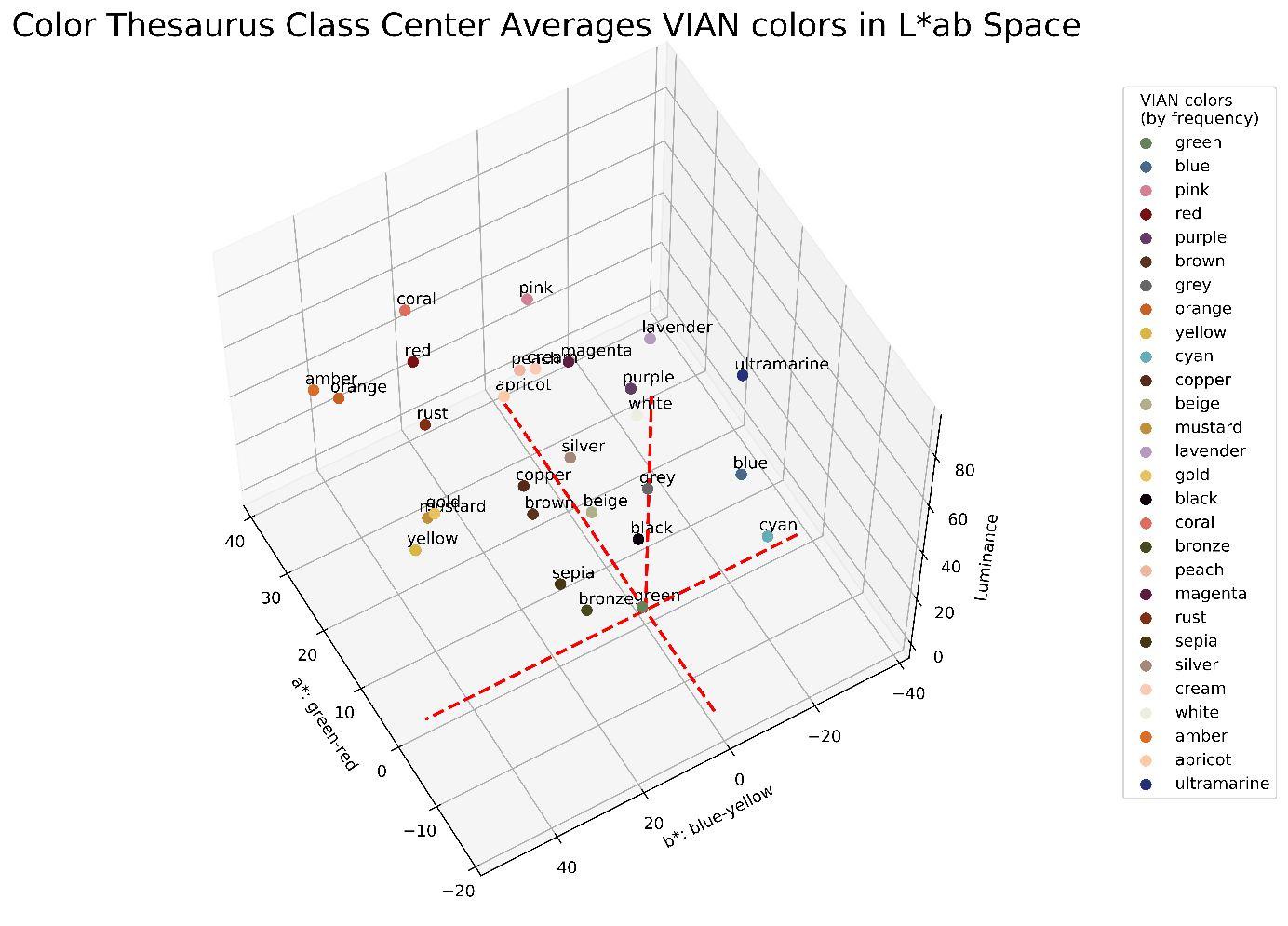
Taking the CIE-Lab average of all color thesaurus values for a VIAN color category, we get a unique value for all 28 VIAN colors. We sort them by their luminance value from dark to light.

Color Thesaurus Class Center Averages VIAN Colors – Color Palette



Some colors like sepia, bronze or copper and brown or yellow and gold or peach, apricot and cream are very close to each other – almost indistinguishable to the human eye. Thus, it is advised to extend the VIAN colors to another set of colors (TODO: WHICH? Include basic colors) so that the user has a larger variety of colors to choose from for the search query.

We plot these VIAN colors in CIE-Lab space as well, to get a better idea of their position in 3D.



TODO: increase font size

If we project these VIAN color points to a constant chroma of 100, that is the highest saturation for a color, convert the lab colors to lch, and round the lch values off to the nearest 10, colors orange, amber and silver become identical in value. Since orange seems to be a good basic color, orange was left in the data. The lch color space is divided into 10 luminance levels ranging from 0 (dark) to 100 (light) and 36 hue angles of 10°-steps around the color wheel.

The VIAN colors in lch space are heavily clustered around warm, earthy color tones, i.e. browns, reds, oranges, and yellows. This is best seen in peach, cream and apricot – VIAN colors that follow each other closely. Only nine out of 26 different VIAN colors (34.6%) include cold, clear color tones, i.e. greens, blues, purples, and magentas. Thus, it is strongly advised to balance off the predetermined set of VIAN colors to incorporate a more equal distribution across the lch color space.

1. Predicting Color Categories for Colors

Based on the EFFCND with both color names and basic color categories in the data set, a machine learning classifier can be trained using supervised learning. During training, the classifier learns to draw decision boundaries between basic colors categories. It learns what color names are attributed to which basic color categories for later inference. The classifier could then take a new color name and predict its most likely basic color category based on what it has learned.

Classification Methods

The task is to classify color names to basic color categories (and not to perform a regression task). Since there are more than two classes for basic colors, the following classification methods[[31]](#footnote-31) can be considered:

1. Multi-class classification: every color name has exactly one-color category

Multi-class classification is a functionality every classifier can do. In such an approach, each sample from the dataset can only have one class label. A sample must have one label.

1. Multi-label classification: some color names can have two color categories

Multi-label classification is a task where each sample can have up to the same number of classes as labels. Hence, a sample’s labels are not mutually exclusive. A sample must have one to many labels where many is equal to the number of possible classes. Positive classes are assigned a 1 while negative classes are assigned a 0 (or -1). KNN supports multi-label classification.

1. Multi-output-multiclass classification

Multi-output-multiclass classification has each sample’s labels attributed to non-binary properties. Is requires that the number of properties (“artist” and “color”) and the number of classes per property (“Van Gogh” and “Rembrandt”; “green” and “red”) are greater than 2. This type of classification is a generalization of A. where only one property is considered and B. where only binary attributes are considered. KNN supports multi-output-multiclass classification

Decision Boundaries

Color names can be mapped to their basic color category. In other words, the LAB space needs to be partitioned into regions where each basic color is at the “center” of all its related color names. For a given new color value it should be possible to determine which basic color it belongs to by looking at the region it is situated in. The decision boundaries dividing each basic color from another helps to delineate these regions.

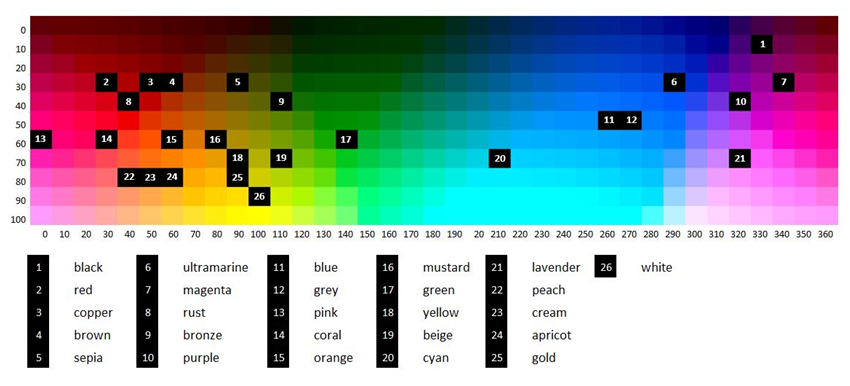


Figure 19: Color Thesaurus Class Center Averages VIAN colors in LCH 2D Space

The machine learning classifier will classify a color name to a basic color based on the Euclidean distance between the color name and the nearest basic colors. Other metrics for calculating distances exist such as the Manhattan, Cosine or Jaccard distance exist, but will not be explored to stay within the limitations of this project’s scope.

A short word about decision boundaries: at the border between yellow and green, it is typically difficult to decide to which the yellow-green color belongs. Either, the user is forced to make a decision where the border is situated between the most greenish yellow color and the most yellowish green color. This hard decision boundary needs to be fixed. But in the process of fixing, the border itself is a color calculated of these two colors, which leave the user to decide about the color class again to infinite precision. Another approach would be to allow for single colors at the border to be classified into both yellow and green. This could be extended to a bandwidth of colors at the border, classified to both yellow and green. This multi-label approach would require a multi-label classification during the supervised learning process of classification.

Building a Machine Learning Classifier

For building a machine learning classifier, a data set with features and labels is needed. This data set will be a color name dictionary where the features are the three-dimensional LAB color values and the labels are the basic color categories which are categorical color names. For each color name there needs to be at least three basic colors for the classifier to learn, the more the better. We choose the CIE-Lab space for doing distance calculations, because colors there are represented as perceptually uniform. The original data set will be an EFFCND. Since the labels are given, a supervised machine learning approach can be adopted. A model learns the correct basic colors for color names in the color name dictionary. This is done by partitioning the LAB space into regions, forming LAB color clusters belonging to the same basic color category and learning decision boundaries among them. Once the model has learned enough and can accurately classify a color name to a basic color, the model is used to predict other color names’ basic color category where no prior categorization exists.

For training data, the entire EFFCND is used. It is split into a training and validation data set by a ratio of 90:10. A baseline machine learning model is trained using the training data set. The machine learning model has a variety of hyperparameters depending on the model for which different values yield different scores (accuracy, precision, recall, F-1 scores). To assess all possible values of the hyperparameters for a model’s performance on the training set, a grid search method was applied. The scores are computed using 5-fold cross-validation on the training data set. The model with the best score from the training data set is applied to the validation set. The validation set’s score is computed by measuring the difference between the predicted labels from the model and the true labels from the validation set. This time, both error scores from the training and validation set are plotted to see how the error score develops when increasing model complexity. The best model is located at the minimum of the error score on the validation set.

For test data, the RGB color space is sampled. Test points are derived for each n-interval in each of the three color channels. These test points are then converted into LAB and classified in LAB using the best machine learning model. A performance score for the model cannot be computed, unless the test points are manually labelled to their true basic color category. Determining the error between the predicted and true basic color categories could then be made possible. This is, however, not part of a canonical machine learning model building process, because improving the model only happens in the model selection phase. After that, the model should be deployable to any new test set as part of an completed up-and-running model.

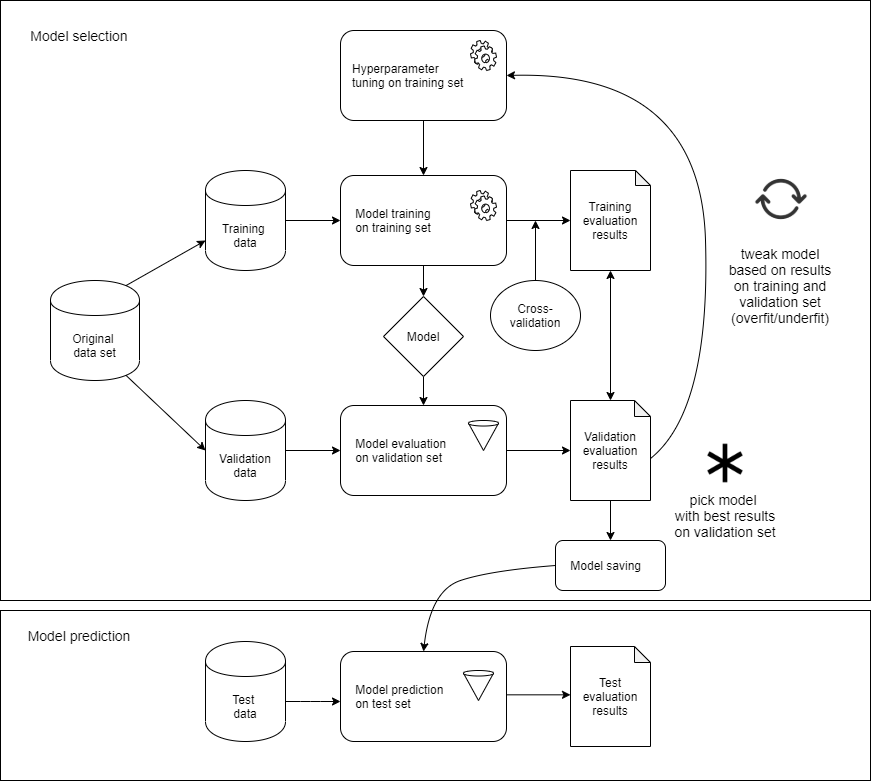


Figure 20: Building and Deploying a Machine Learning Model

When evaluating a model, there are two things that are worth considering:

* Performance: how well does the model generalize to unseen data such as the test data (generalization error) ? This is partly accounted for when evaluating the validation data during the model building process
* Efficiency: how fast is the classifier? If the user disposes of the required computational resources the model is able to scale up to other EFFCNDs and integrate them into a well-rounded user-experience, an efficient classifier is paramount

There are several ways to measure a classifier’s generalization quality:

* [Hamming loss](http://scikit-learn.org/stable/modules/generated/sklearn.metrics.hamming_loss.html#sklearn.metrics.hamming_loss) measures how well the classifier predicts each of the labels, averaged over samples, then over labels
* [accuracy score](http://scikit-learn.org/stable/modules/generated/sklearn.metrics.accuracy_score.html#sklearn.metrics.accuracy_score) measures how well the classifier predicts label combinations, averaged over samples
* [jaccard similarity](http://scikit-learn.org/stable/modules/generated/sklearn.metrics.jaccard_similarity_score.html#sklearn.metrics.jaccard_similarity_score) measures the proportion of predicted labels for a sample to its correct assignment, averaged over samples
* [precision](http://scikit-learn.org/stable/modules/generated/sklearn.metrics.precision_score.html#sklearn.metrics.precision_score) measures how many samples with ,
* [recall](http://scikit-learn.org/stable/modules/generated/sklearn.metrics.recall_score.html#sklearn.metrics.recall_score) measures how many samples ,
* [F1 score](http://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html#sklearn.metrics.f1_score) measures a weighted average of precision and recall, where both have the same impact on the score

Example: KNN and SVC on Thesaurus-VIAN

As an example of building a machine learning classifier, the following color name dictionary is used as the original data set. The machine learning models stated below can best capture the dimensionality and spatial locus of the colors in LAB space and compute the Euclidean distance between them.

**Data set:** The EFFCND used in this example is the Thesaurus-VIAN data set with LAB color values and VIAN basic color categories.

**Machine Learning model:** From ten different machine learning classifiers, only two were yielding good results: K-Nearest Neighbor Classifier and Linear Support Vector Classifier.[[32]](#footnote-32)

A hard-margin SVM finds a hyperplane separating points from different clusters such that no point is misclassified. A soft-margin SVM finds a hyperplane but allows for some points to be misclassified. This is handled by regularization parameter C.

The lower the C parameter the softer the marin

A large value of C basically tells our model that we do not have that much faith in our data’s distribution, and will only consider points close to line of separation.

A small value of C includes more/all the observations, allowing the margins to be calculated using all the data in the area.[[33]](#footnote-33)



For demonstration purposes, color name “sunflower yellow” with index 654 in the EPFL Color Thesaurus is classified using the best SVC classifier (model\_THESAURUS\_VIAN\_Linear SVM\_SVClinear\_C0.011881\_train721\_cat28\_testacc0.671.sav). The classifier returns “yellow” for “sunflower yellow” – a very accurate basic color category for “sunflower yellow”. Second comes “gold” and third “mustard” which make a lot of sense, too.

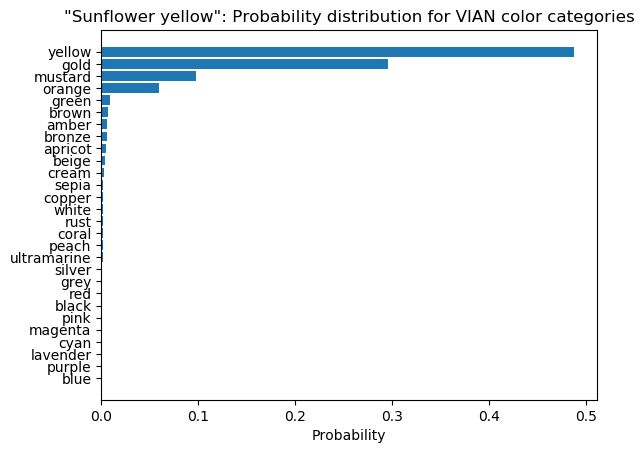


Figure 21: Probability class distribution for "sunflower yellow"

Extending the Dictionary

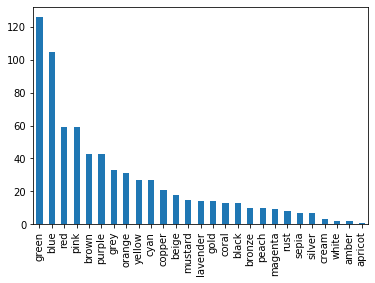
The hand-labelled color categories are an additional set of true basic color labels that can be added to the original color name dictionary to get more discrete values, train a new model with a possibly higher score. However, for these discrete values there exist no color names.

TODO: provide the user with both alternatives, clear-cut border or soft border (Hierarchical – interconnected)

TODO: write about that lab colors can only be seen as rgb, lab colors can also be outside of the visible light, opencv clamps all the lab colors into rgb colors

We trained these two machine learning classifiers on the Color Thesaurus / VIAN dataset taking the CIE-Lab color values as feature and the 27 VIAN color category or all 720 Color Thesaurus color names as label. The VIAN color categories are very imbalanced – green has more than 120 discrete CIE-Lab values while apricot has only 1 value.

|  |  |
| --- | --- |
| VIAN color | Count |
| green | 126 |
| blue | 105 |
| red | 59 |
| pink | 59 |
| … | … |
| cream | 3 |
| white | 2 |
| amber | 2 |
| apricot | 1 |



Making the top and bottom of the list are green, blue, and red – the RGB colors – and white, amber and apricot. The latter colors were specific colors, while the former encompass a wider region of discrete color values.

For training the machine learning classifier, the entire dataset was used. For testing, we sample from the entire CIE-Lab space by taking testpoints from a grid of equal step size for each channel. For example, if 3 steps are chosen, the luminance channel ranging from 0-100 yields testpoints 0, 50 and 100. These CIE-Lab testpoints are visualized as color patches that are compared against the color patch of the classifier-predicted VIAN color label. This allows for a quick manual check whether the classification was correct or not. The resulting accuracy and error was improved over several iterations and parameter combinations. The top 3 results:

TODO: try other parameters

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID | Training Data | ML Classifier | Parameters | Test Data | Accuracy / Error |
| 1 | Rows: 720 original data set + 216 testpoints = 936  Feature: CIE-Lab color values  Label: 27 VIAN colors | KNN | Number of nearest neighbors = 5,  P=2 (Euclidean) | Feature: 27 CIE-Lab color values  Predicted Label: classified 27 VIAN colors  Real Label: manually-labelled (training data) | 100% / 0% |
| 2 | Rows: 720 original data set + 1331 testpoints = 2051  Feature: CIE-Lab color values  Label: 27 VIAN colors | KNN | Number of nearest neighbors = 5,  P=2 (Euclidean) | Feature: 125 CIE-Lab color values  Predicted Label: classified 27 VIAN colors  Real Label: manually-labelled (training data) | 96.8% / 3.2% |
| 3 | Rows: 720 original data set + 1331 testpoints = 2051  Feature: CIE-Lab color values  Label: 27 VIAN colors | KNN | Number of nearest neighbors = 5,  P=2 (Euclidean) | Feature: 27 CIE-Lab color values  Predicted Label: classified 27 VIAN colors  Real Label: manually-labelled (training data) | 96.3% / 3.7% |

1. The **K-Nearest Neighbor classifier** is a lazy learning algorithm that computes for each new test point its classification by taking a simple majority vote of its labeled k nearest neighbors where k is the number of neighbors it checks and p is the distance metric calculated between the new test point and the nearest training point neighbor. The algorithm is robust to noisy training data. But, k and p need to be determined and computation cost could be high compared to other machine learning classifiers.
2. The **Support Vector classifier** separates the training data cloud linearly based on their clusters in space. By computing a hyperplane between the clusters, a decision boundary is created that makes classification of new points into each region possible. To calculate the hyperplane, it maximizes the margin between the two support vectors delineating two adjacent clusters of points. Each cluster belongs to another class.

Confusion matrix:

|  |  |  |
| --- | --- | --- |
| **Real Value** | **Predicted Value** | **Accuracy** |
| Green | Not green | 0 |
| Green | Green | 1 |

TODO: F-1 score /ROC for test set split of training set

TODO: Performance timeit of both approaches

We select the best model using validation results.

TODO: sample rgb color space, then convert them into lab and then classify it there in lab, convert back to rgb to display it

1. Matching a Color to Color Palettes

If user wants to search for images by color, it is usually preferred to search for a color that is then the dominant color in an image in all search results. This method can be extended to include the second most dominant, third most dominant etc. colors of an image that form together the first 101 colors of an image’s color palette. Now, the search result can be fine-tuned to get a broader or more precise pool of images in the search result.

The user searches “sepia”, finds a match of that with color palette’s colors

1. Take all colors inside a color palette, run it through ML model to get classification of colors into VIAN color categories, if one of them is also “sepia”, color space: HSV
2. Take “sepia” color values range, take color values of each color in color palette and see whether it is within the color values range of “sepia”, if yes, display this color palette in, each palette’s colors has a VIAN color tag that is the result of the machine learning prediction, color space = LAB

Extracting Colors from Images

For sample color palettes, we subsample a video footage of “Die Tagesschau” into images from which a color palette per image is extracted for all video frames to get a pool of color palettes (ColorPaletteExtraction.py). The extraction is hierarchical – the lowest row contains the highest amount of different colors for a video image. The user can set which row to fix, we fix it to the lowest row ‘row 20’ for now. In general, it is better to choose a low level of hierarchy for a color palette, because for the highest level only one color will be available for an image. Then, all BGR colors of the lowest row are converted into LAB. Then, they are classified into one out of 28 VIAN colors using above mentioned top-rated machine learning classifier KNN (n\_neighbors = 5, p=2) which was trained on 2051 test data. The user’s VIAN color search key is matched to the predicted VIAN colors of each palette. The number and names of all successfully matched color palettes (gold palettes) are shown as a result of the query. (The gold color palettes first n-to-all color patches can be displayed. )

A different way of extracting a color palette from an image is to use the webtool “Adobe Color” <https://color.adobe.com/de/create/image>. On the webtool, click “create” then “theme extraction”. The user can drag&drop an image for which a color scheme of five color patches will be automatically extracted from data points on the image. There are six different way of placing the data points on the image: colorful, light, muted, strong, dark, without. The color scheme’s HEX color codes are displayed below the color patch. The HEX color codes can be copied to the clipboard using a click functionality. The webtool’s save&use functionality are only open to users with an Adobe Creative Cloud account.

In general, the number of color patches in a color scheme can grow from one to five (Adobe Color) to hundred-one (VIAN) to an infinite amount of colors in a gradient range between any two to infinite many data points on an image (see Adobe Color webtool). Such colors are directly taken from the image, i.e. they are neither an aggregate average of an area in the image nor an aggregate of multiple disparate color points on the image.

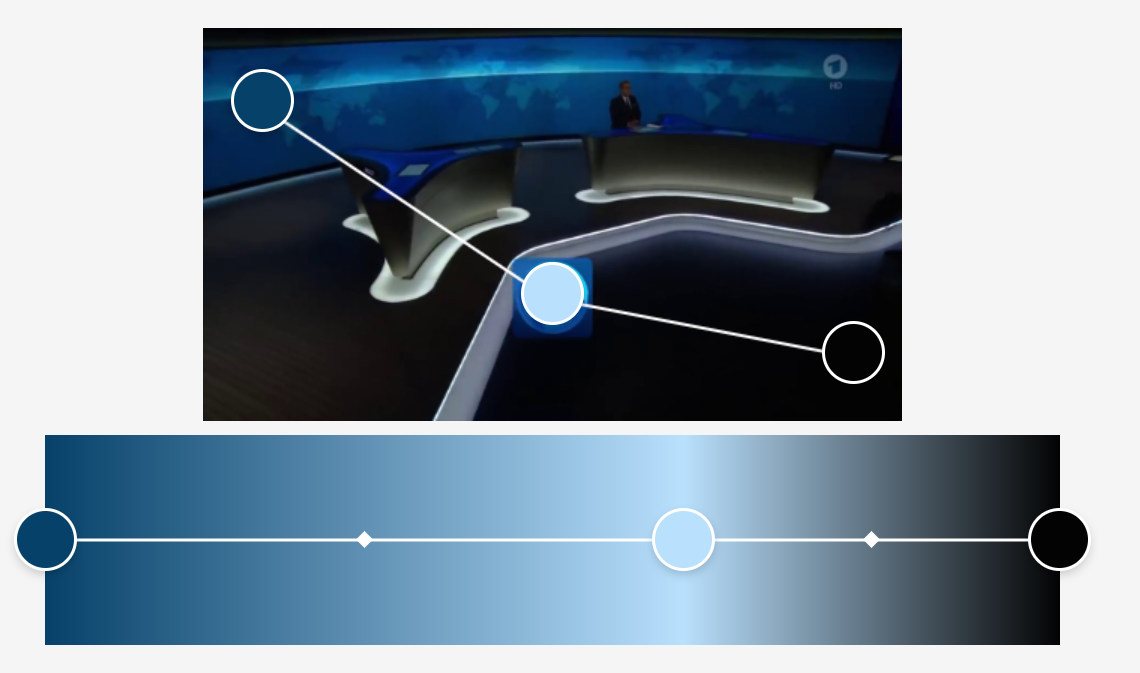


Figure 22: Color Range Extracted from an Image

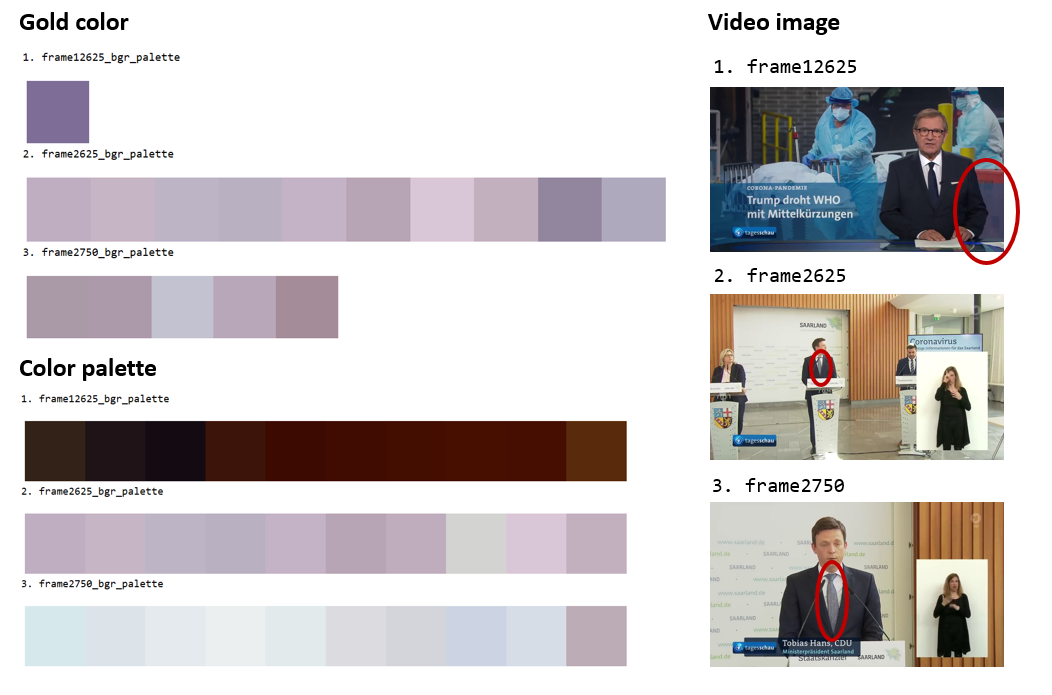
In contrast, color schemes of an image can hold colors that are not direct take-outs, but derived by aggregation. For example, the aggregation could occur by selecting an area in the image and averaging all color pixels in the area to form a superpixel. Another method is to lay a grid on the image to extract direct color pixel take-outs and performing a hierarchical agglomeration on these colors from the lowest level to the highest level. The highest level in the hierarchy consists of a single color that is the average of all colors. This average of all colors extracted from an image is the average color of an image. It is not equal to the dominant color of an image which is the color that occupies the most space in an image.

Methods for Extracting Color Schemes from an Image

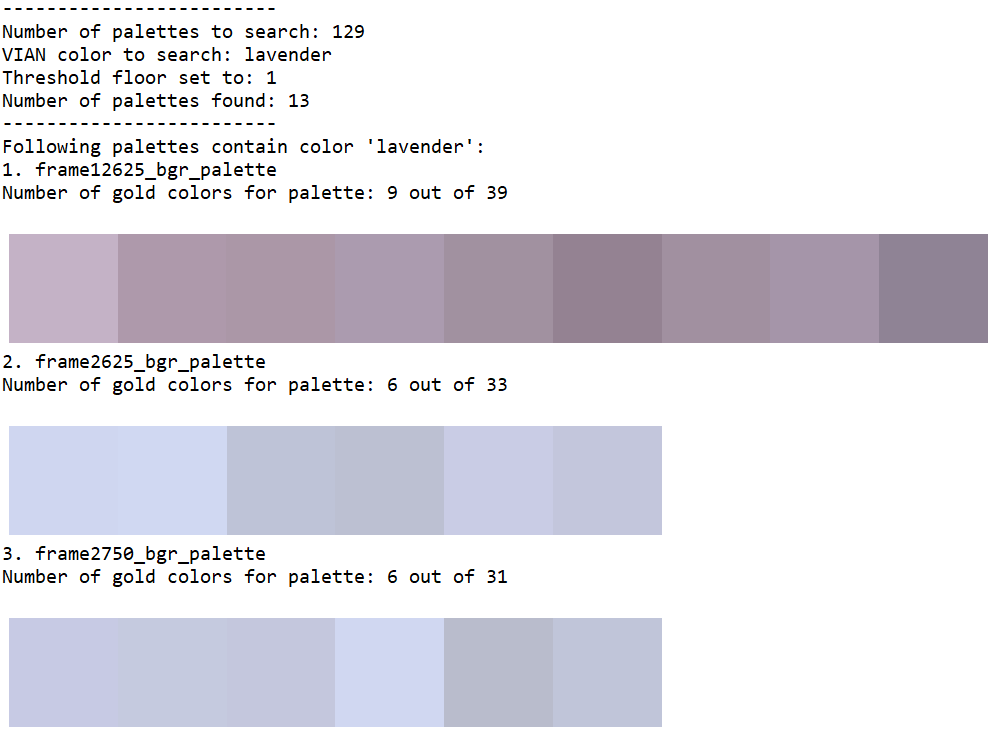
|  |  |  |
| --- | --- | --- |
| **Procedure** | Direct Colors | Derived Colors |
| **Method** | * Color pixels on a grid layed upon an image * Color pixels that follow a certain pattern (darkest, lightest, warmest, coolest etc.) in an image * Random color pixels | * Seed areas in an image and take the average of all color pixels in the area * Average direct color pixels to form superpixels |

Searching Color Palettes by Color Name

For example, the user can search for VIAN color ‘lavender’ in 129 color palettes. If one of these color palettes contain ‘lavender’ the color palette will be shown to the user. The number of palettes that were found containing the color ‘lavender’ are 33. The first three search results are displayed in the following:

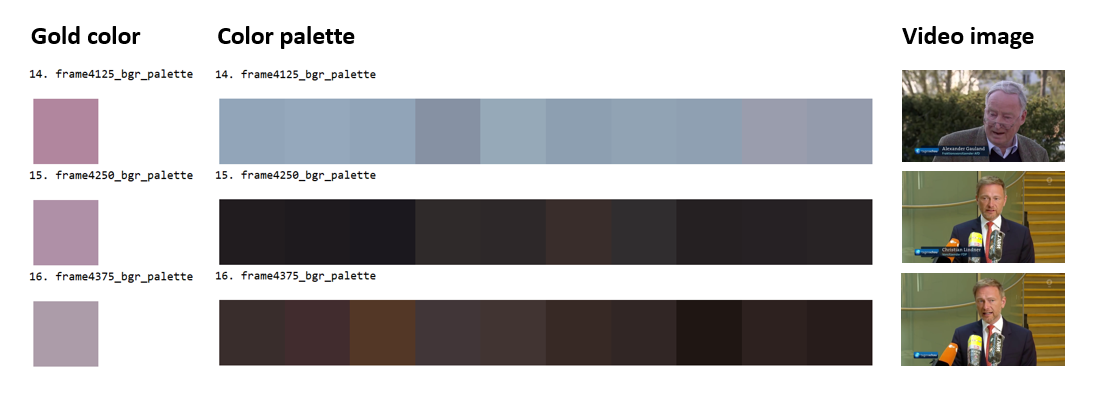
TODO: sort color palette by ratio\_width

Adding to this, the user can set a **threshold** floor for the search key. Because ‘lavender’ in an image takes up only a certain percentage of the total image sometimes it is not useful to the user to get all images where ‘lavender’ makes up only a small portion in the image. Setting a threshold will help the user to sift such images out of the result. The user would have then all color palettes for images where ‘lavender’ is present with some dominance. At the lowest level of an image’s color palette, all 101 colors have a certain ratio width, typically, that fluctuates around 1 percent from the total width. Fixing the threshold to 1%, we get only 13 color palettes back for ‘lavender’.



The first 13 color palettes with a threshold floor of 0 or 1 are the same. The remaining 20 color palettes for a threshold of 0 have significantly less ‘lavender’ in the original video image, because these ‘lavender’ colors make up less than 1 % of the image. In the below image, the next 14 up gold colors for ‘lavender’ are shown with their original color palette (10 out of 101 color patches) and the color palette’s original image.

TODO: sort color palettes (showing 10 colors out of all colors in color palette)



Python Scripts:

1. ColorClassCenter.py
2. ColorClassCenterVisualization.py
3. ColorPaletteSearchColor.py
4. ColorMLClassification.py
5. ColorConversion.py
6. ColorMLClassificationVisualization.py
7. ColorPaletteExtraction.py
8. ColorPaletteInHSVSpace.py
9. ColorPaletteSort.py
10. ColorPaletteVisualization.py
11. ColorSpaceVisualization.py
12. ColorThesaurusPreprocessing.py
13. ColorThesaurus2VIANClassification.py
14. ColorThesaurusVIANInSpace.py
15. ColorVisualization.py
16. ImageDownloadGoogle.py
17. Images2AVGColor.py
18. MovieToImage.py
19. Limitations

The user can search color palettes by color name. The problem with color names is that they evoke different colors for different people of the English-speaking world. Not only do they have different colors in mind, the colors are not defined in a precise and clear way. Rather, color names serve to obscure the meaning of the color that is alluded to.

As early as in 1905, Albert Munsell already remarked: “The terms used for a single hue, such as pea green, sea green, olive green, grass green, sage green, evergreen, invisible green, are not to be trusted in ordering a piece of cloth. They invite mistakes and disappointment. Not only are they inaccurate: they are inappropriate.” Thus early is he cramped by the poverty of color language which more often than not are borrowed from other senses.[[34]](#footnote-34)

He goes on to define color along three dimensions: hue, value and chroma – the HSV color space. Thus, it is suggested to let the user have three sliders corresponding to each dimension of color. Once the color is made concrete, an interval can be defined as an upper and lower bound on each dimension to give more leeway to the search results.

Task 2: Implement a method to compute the distance between hierarchically structured color palettes

The FilmColors project’s Visual Annotation webtool (VIAN) features a function that is able to extract a color palette from an image in hierarchical order. At the lowest level are the most colors extracted for the color palette. At the highest level all colors were merged to form a single color average for the image. The task is to get one such color palette and compare it to other color palettes based on their similarity to the first color palette. For the first color palette, the distance is calculated to all other color palettes to determine the top-k nearest color palettes.

Similarity metric

Distance Euclidean only in LAB space only

Display cluster of points for both cps in LAB space

TODO: Make histogram (color distribution, probability)

Y- axis: ratio width, x-axis : colors (sorted by ratio width) or put it in lab color space histogram a, b, l channels in 3d axis, and bin frequency, no need to see the histogram

Compare multidim histograms (opencv, cv2.comparehist() look for similarity metrics )

Color names in the histogram close to each other are also closer together in another color palette histogram

Preprocessing

Each color palette comes with information about its colors: the bgr values of each color and the ratio width of each color. First, we extend this information about colors by converting the bgr values to lab values for each color and predicting the categorical color names for each color using the top-rated machine learning model features in Task 1.

Method

1. Aggregate Comparison

On color palette level: aggregate individual bgr colors by averaging bgr to a CP metric and compare CP metrics with each other (highest row in hierarchy), summing, taking the closest (Min) or furthers (Max) can distort the results we are trying to get, image gets divided into superpixels and the superpixels are aggregated until only one superpixel is left

One image can only have one color palette of at most 101 colors. At the lowest level, all image’s color palette have at most 101 colors

TODO: Agglomerative clustering to show and skips to show

Display math formula – no!

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CP1 | | CP2 | | CP1-CP2 |
| Color | Ratio\_  width | Color | Ratio\_  width | Distance (eucl, jaccard, manhattan, cosine etc.) |
| Bgr1 to Lab1 or catcol1 | Ratio\_width1 | Bgr1 to Lab1 or catcol1 | Ratio\_width1 | Dist(CP1.Lab1\* Ratio\_width1),(CP2.Lab1\* Ratio\_width1)) |
| Bgr2 to Lab2 or catcol2 | Ratio\_width2 | Bgr2 to Lab2 or catcol2 | Ratio\_width2 | Dist(CP1.Lab2\* Ratio\_width2),(CP2.Lab2\* Ratio\_width2)) |
|  |  |  |  | Argmin(Dist(CP1.LabID\*ratio\_widthID),(CP2.LabID\*ratio\_widthID)))  Min(Distance) |

1. Distributive Comparison

On bgr level: calculate all combinations of pairwise distances between all bgr colors in LAB space (due to perceptual uniformity in that space) and find minimum distance between cp1bgr1 and cp2bgr\*, make metric cp1-cp2, do with other cps, lowest metric pair is closest neighbor (lowest row in hierarchy), include weights for each color by looking at ratio\_width, at highest resolution has 101 colors

All color palettes are loaded into a pool of color palettes. To find the most similar color palettes inside a pool of color palettes, we convert all palette colors to lab values first. The LAB space is chosen because its advantage lies in the perceptual uniformity of color in that space. Then, we build color-palette pairs by calculating all pairwise combinations (with withdrawal) possible between the color palettes in the pool. Each color in one color palette is matched to a color from the other color palette in such a way to get the minimum distance. In this instance, the Euclidean distance metric is chosen. For a color-palette pair, we take the average of all minimum distances that is each color-palette pair’s bondage index. The smaller the bondage, the more similar the color-palette pair.

For 129 color palettes in the pool, the n-closest possible color-palette pair were found (through recursion).

Top-1: Palettes “frame1250”, “frame1375” are the closest to each other.

Top-2: Palettes “frame1125”, “frame1375” are 2. closest to each other.

Top-3: Palettes “frame1125”, “frame1250” are 3. closest to each other.

These color palettes’ original image frame is displayed in pairs to see why they are the closest to each other. While top-1 “frame1250” and “frame1375” are exactly the same, “frame1125”paired up with each of the top-1 image frame are almost the same: the number of deaths 1.861 is not displayed in the former frame image frame. The result is very convincing.

Top 1: Image frames “frame1250”, “frame1375”

Top 2: Image frames “frame1125”, “frame1375”



Top 3: Image frames “frame1125”, “frame1250”



The images were taken from the following video footage sequence:



For finding the top-10 most similar color palettes for given image frame “frame3500\_bgr\_palette” the asymmetry in the pairs needs to be considered when finding the result. Also each color in the color palette has an unequal importance distribution. The weights were set to be equal to the ratio\_width of each color.

The top-3 most similar color palettes are top-1: “frame3625\_bgr\_palette”, top-2: “frame3750\_bgr\_palette”, and top-3: “frame3375\_bgr\_palette”. Had the weights not been calculated, “frame3125\_bgr\_palette” would have figures at the 100th position instead of XX..

Python Scripts:

1. ColorPaletteDistCalc.py
2. Classify the patterns within the color palettes in different types of color contrasts defined by the ERC FilmColors Project

pattern = color contrast Ðiven a color palette (first: 5 rgb-valued patches, then scale it up), classify them into one of the color contrasts categories

1. Implement a web-based user interface that allows the user to select or create a palette and generate a list of similar color palettes

user defines number of patches to fill with colors that the user wants and specifies the proportion of them 0-100%, result: top-k nearest color palettes from a given list of CPs

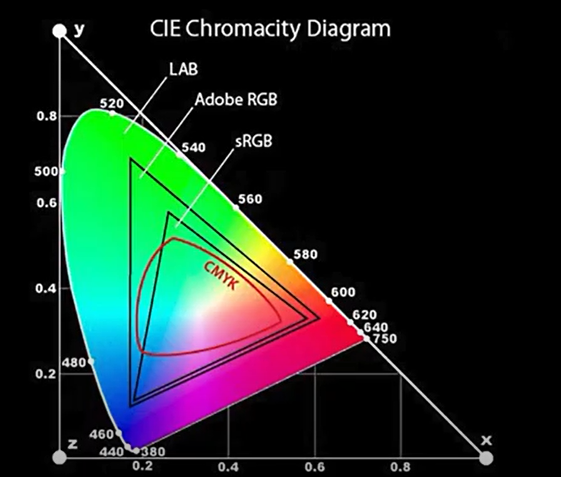
**Appendix**

Color Basics

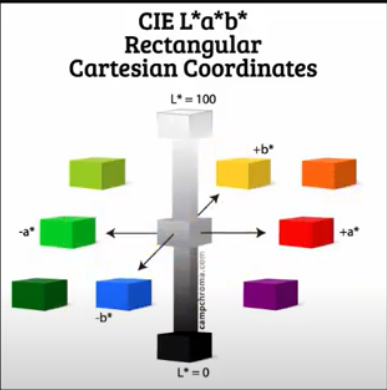
A color is just a reflection of different wavelengths. The visible lights is between 390 and 450 nanometers. We perceive a ray of light through the human visual system.

Color Spaces

1931 CIE chromaticity chart

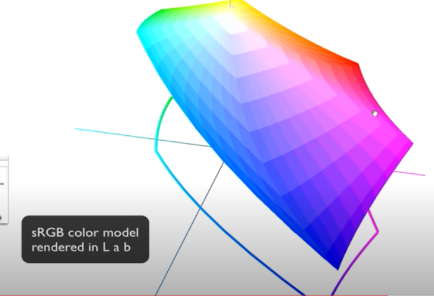
All these different color gamuts – LAB, (English world: Adobe RGB or European world: ICE RGB), sRGB and CMYK have different uses.

CMYK has three primary colors: cyan, magenta and yellow. CMYK is used at your printer.

LAB is much larger than all the other color spaces: It encompasses many more colors and more intensities of colors. LAB has four primary colors, two of which are complementary: green-magenta and blue-yellow. Pulling into minus on the a-channel the image turns into a cold green color, in the other plus direction magenta is added to the image. Same goes for the b-channel, because blue is cold color, the slider needs into the minus and for warm yellow will be shown. In LAB, more information about color is delivered in this color space. LAB colors will be the same across different PCs – it is the only color space that allows for communicating across different devices.

RGB has three primary colors: red, green and blue. RGB is used on the web, your PC monitor and camera. The RGB is interpreted in different ways depending on the PC.

Adobe RGB has a larger color gamut than sRGB by around 35% for blues and greens.



Color Conversion

The preferred websites for converting color values to different color spaces are

1. Nixsensor.org
2. EasyRGB: categorical color names
3. colorizer.org
4. Rapidtables
5. Colorthesaurus

Some of the websites display the color, some only deal with color values.

1. <https://www.december.com/html/spec/colorlinks.html>, <https://people.csail.mit.edu/jaffer/Color/> [↑](#footnote-ref-1)
2. Isaac Newton, Opticks, William Innys, 1704, London. [↑](#footnote-ref-2)
3. Isaac Newton, A New Theory of Light and Colours, The Royal Society, 1671, London. [↑](#footnote-ref-3)
4. Isaac Newton, Opticks, William Innys, 1704, London. [↑](#footnote-ref-4)
5. Goethe, Theory of Colours, John Murray, 1840, London. [↑](#footnote-ref-5)
6. Goethe, Theory of Colours, John Murray, 1840, London. [↑](#footnote-ref-6)
7. Goethe, Theory of Colours, John Murray, 1840, London. [↑](#footnote-ref-7)
8. Albert H. Munsell, A Color Notation, Geo. H. Ellis Co., 1907, Boston. [↑](#footnote-ref-8)
9. Albert H. Munsell, A Color Notation, Geo. H. Ellis Co., 1907, Boston. [↑](#footnote-ref-9)
10. Albert H. Munsell, A Grammar of Color, The Strathmore Paper Company, 1921, Mittineague MA. [↑](#footnote-ref-10)
11. Albert H. Munsell, A Color Notation, Geo. H. Ellis Co., 1907, Boston. [↑](#footnote-ref-11)
12. Albert H. Munsell, Atlas of the Munsell Color System, Wadsworth, Howland & Co., 1915, Malden MA [↑](#footnote-ref-12)
13. Johannes Itten, The Art of Color, Wiley, 1974, New York. [↑](#footnote-ref-13)
14. Johannes Itten, The Elements of Color, Van Nostrand Reinhold Company, 1970, New York. [↑](#footnote-ref-14)
15. The list of all searchable color names are: blush, pink, red, currant, sand, peach, orange, umber, bone, butter, yellow, sage, seed, citrus, leaf, frond, sea-glass, enamel, turq, forest, ice, sky, sea, steel-blue, cloud, chambray, blue-sea, peacock, frost, periwinkle, blue, navy, petal, spring, violet, ore, cocoa, chipmunk, brown, deep-sepia, metal, aluminium, charcoal and black. [↑](#footnote-ref-15)
16. <https://www.epfl.ch/labs/ivrl/research/image-mining/multi-lingual-color-thesaurus/> [↑](#footnote-ref-16)
17. Goethe describes how Cav. Nobili (professor of physical science in Florence) creates a scale of 44 colors attained by the coloring of metals in a physical experiment described in in Goethe, Zur Farbenlehre, John Murray, 1840, London. [↑](#footnote-ref-17)
18. <https://www.colorhexa.com/color-names> [↑](#footnote-ref-18)
19. <https://www.colorcombos.com/color-tags.html> [↑](#footnote-ref-19)
20. Patrick Syme, Werner’s Nomenclature of Colors, William Blackwood, Edinburgh, 1821. [↑](#footnote-ref-20)
21. Each channel is a byte which contains 8 bits. A binary bit can either be 0 or 1. Thus, equals 256 possible ways to realize a byte. [↑](#footnote-ref-21)
22. Patrick Syme, Werner’s Nomenclature of Colors, William Blackwood, Edinburgh, 1821. [↑](#footnote-ref-22)
23. Color names can be searched in 9 different languages – Chinese, French, German, Italian, Japanese, Korean, Portuguese, Russian, and Spanish to yield different color values in one language compared to another for each translation-equivalent color name. [↑](#footnote-ref-23)
24. <https://blog.xkcd.com/2010/05/03/color-survey-results/> [↑](#footnote-ref-24)
25. <https://www.epfl.ch/labs/ivrl/research/image-mining/multi-lingual-color-thesaurus/> [↑](#footnote-ref-25)
26. Albrecht Lindner, Bryan Zhi Li, Nicolas Bonnier, and Sabine Süsstrunk, “A large-scale multi-lingual color thesaurus,” IS&T Color and Imaging Conference, 2012. [↑](#footnote-ref-26)
27. [Eleven Colors That Are Almost Never Confused.](http://spie.org/Publications/Proceedings/Paper/10.1117/12.952730)  [↑](#footnote-ref-27)
28. Found on www.vian.app/keywords, email = [beta-tester@test.uzh.ch](mailto:beta-tester@test.uzh.ch), pw = "vian-beta-tester20", in Global, Hues. All hues except for names that mean a collection of hues such as “desaturated”. [↑](#footnote-ref-28)
29. Frederik A. Gierlinger, Štefan Riegelnik, Wittgenstein on Colour, Walter de Gruyter GmbH, Berlin, 2014, p. 79. [↑](#footnote-ref-29)
30. Pantone® Color of the Year 2020, <https://www.pantone.com/color-intelligence/color-of-the-year/color-of-the-year-2020>. [↑](#footnote-ref-30)
31. <https://scikit-learn.org/stable/modules/multiclass.html> [↑](#footnote-ref-31)
32. The other classifiers which were tested for their usability were Gaussian Process, Decision Tree, Random Forest, AdaBoost, Naïve Bayes, Neural Net and QDA. The other classifiers simply made out a circular region among most basic colors, leaving out the larger space around the circles to a single predominant basic color. As an example, this is the case if kernel methods are implemented for an SVM, non-linear decision boundaries between clusters are made possible. This imbalance was the reason why these classifiers were not considered for this project. Besides, they do not use Euclidean distance, a limitation set forth by this project’s scope. [↑](#footnote-ref-32)
33. <https://scikit-learn.org/stable/auto_examples/svm/plot_svm_margin.html> [↑](#footnote-ref-33)
34. Albert Munsell, A Color Notation, Geo. H. Ellis Co., Boston, 1907. [↑](#footnote-ref-34)