www.editada.org

International Journal of Combinatorial Optimization Problems and Informatics, 16(3), May-Aug 2025, 118-132. ISSN: 2007-1558.
https://doi.org/10.61467/2007.1558.2025.v16i3.514

Web-based clothing searcher utilizing convolutional neural networks and dissimilarity rules for color classification in the HSL color space and ORB algorithm for garment characterization

Luciano Martinez[1], Martín Montes Rivera [1], Alberto Ochoa Zezzatti [2], Julio Ponce [3], Eder Guzmán [1]
[1] Research and Postgraduate Studies, Universidad Politécnica de Aguascalientes, Calle Paseo San Gerardo #201, Fracc. San Gerardo, 20342 Aguascalientes, Mexico.
[2] Universidad Autónoma de Ciudad Juárez, México.
[3] Universidad Autónoma de Aguascalientes, México.
E-mails: lucianomrtnz@gmail.com, martin.montes@upa.edu.mx, alberto.ochoa@uacj.mx, julk_cpg@hotmail.com, jose.guzman@upa.edu.mx.

Abstract. This study proposes a methodology for searching for clothing on the web using convolutional neural networks (CNN) and a color obtaining module. The system is based on the creation of a color base in the HSL (Hue, Saturation, Luminosity) space. It determines the color of an image based on dissimilarity rules. The base image, previously processed by a CNN, along with the result of obtaining the color, generates a string of characters. This string is entered into a search engine, resulting in garments that resemble the base image. These garments are processed with the ORB algorithm to extract their features and compare them with those of the base image. In this way, it can be determined which of the garments is most like the base image. This work offers new perspectives and useful techniques for obtaining colors, significantly improving the accuracy and efficiency of object search on the web.

Keywords: Image searching, Convolutional neural networks, color classification, HSL space, ORB.

Article Info
Received Aug 26, 2024
Accepted Jul 1, 2025

1 Introduction

In the digital age, the Internet is a vast universe teeming with objects and information. Finding specific objects within this vast expanse can be a daunting task. Efficient sorting and searching mechanisms are crucial in navigating this digital landscape. In this context, we present a novel search system that leverages Convolutional Neural Networks (CNNs) to classify clothing items based on user provided images.

Our CNN model is trained to identify various types of clothing, such as trousers, shirts, dresses, among others. The outcome of this classification process is a character string that describes the type of clothing. In addition to clothing classification, our system is also capable of detecting the predominant color in an image. This is achieved by creating a color database in the HSL (Hue, Saturation, Luminosity) space, which classifies 12 colors with different color centers.

Utilizing dissimilarity rules based on Euclidean distance, our system can determine the dominant color in an image. The result is a concatenated string that combines the clothing classification and the predominant color. For instance, if an image of blue trousers is inputted, the system will generate the string "blue trousers". This string can then be inputted into a prebuilt web search engine to find similar garments available online.

Furthermore, we have developed an application using Tkinter that serves as an interface for implementing this methodology. The application allows users to select and predict images, predict colors, search for similar images on the web, and determine the best image based on the downloaded images and the base image.

Moreover, the use of the ORB technique for feature extraction and matching enhances the accuracy of image comparisons. This technique extracts features from the images and calculates the difference in the sum of RGB values, providing a more detailed comparison between the base image and other images.

This work presents a fresh perspective on searching and classifying specific objects on the Internet. By integrating object classification with color recognition, and leveraging advanced techniques like CNN and ORB, our system significantly enhances the accuracy and efficiency of online searches. This research underscores the potential of machine learning and image processing techniques in improving web search methodologies.

2 Relater Word

Object classification using CNN has been an active area of study in recent years. This article (Chen et al., 2021) reviews CNN based image classification algorithms. The authors cover the development of CNNs, from their predecessors to recent state-of the art network architectures.

Color detection in images using the HSL color space has been the subject of several studies. For example, this paper (Su et al., 2019) introduces the HSL color space for better color recognition and processing instead of the CIELAB color space. In addition, they add flexible combinations of the weight coefficient for HSL to achieve different results.

In another study, new face detection methods based on HSL and HSI2 color spaces were presented. In this work (Elaw et al., 2019), methods based on two main steps were proposed: first, skin like regions are detected using the gradient values of the proposed color space. Then, the desired faces are determined from the recommended regions according to the main facial features, such as eyes, mouth, and nose.

On the Euclidean distance side, this other research focuses on face detection and recognition using texture features and Euclidean distance (J. Yu & Li, 2013).

The use of the ORB algorithm has been instrumental in various research studies for feature detection. Building on this, the research introduces an algorithm to optimize the stitching of low altitude aerial images captured by UAVs in dynamic environments. It employs semantic segmentation and the ORB algorithm to differentiate the foreground and background of the image. By comparing feature point information with the semantic information of the foreground, it eliminates foreground feature points for feature point matching. This approach enhances the quality of stitching low-altitude aerial images of UAV (Zhang et al., 2022).

This study aims to improve the ORB algorithm for image feature extraction. The proposed ALGD ORB algorithm addresses issues such as dense and overlapping feature points, and unbalanced distribution. It introduces an adaptive threshold to enhance feature point detection and uses an improved quadtree method to homogenize their distribution. It combines feature descriptors generated from both gray scale and gray difference to enhance the distinctiveness of the descriptor. Experimental results demonstrate that it significantly improves the uniformity of feature point distribution (Chu et al., 2023).

The related literature demonstrates the potential and effectiveness of Convolutional Neural Networks (CNN) for object classification and color detection in images, as well as the use of Euclidean distance as a dissimilarity rule. Additionally, the ORB algorithm has proven useful for feature extraction in images, providing a solid foundation for feature detection and matching. However, our work goes a step further and proposes a system that combines all these techniques. This innovative approach aims to improve the accuracy and efficiency of online clothing searches, providing more relevant and precise results for users.

3 Materials and Methos

The methodology of our research is detailed below.

3.1 CNN

We use a sequential CNN model for garment image classification (Sharma et al., 2018). Our model consists of several layers, including Conv2D layers for feature extraction (*Conv2D Layer*, 2022 , MaxPooling2D for dimensionality reduction (*Capa*

MaxPooling2D, 2022) and Dense layers for final classification (*Dense Layer*, 2022). As can be seen in Figure 1, the sequential model of our CNN is presented.

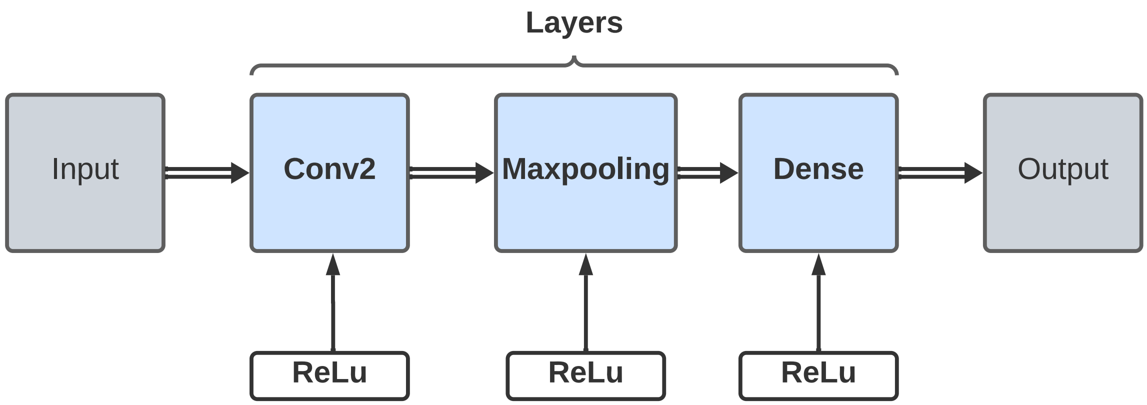


Fig. 1. Sequence model of the convolutional neural network.

The hyperparameters and configurations used in our model include:

- Learning rate: We use the Adam optimizer, which adaptively adjusts the learning rate for each parameter.
- Batch size: 32
- Number of epochs: 15
- In addition, our CNN model has the following configurations:
- Hidden layers: The model has three convolutional layers (Conv2D), three maximum pooling layers (MaxPooling2D), a flattening layer (Flatten), a dropout layer (Dropout), and two dense layers (Dense).
- Kernel size: 3x3
- Padding: We use the 'same' padding type in all convolutional layers. This means that zeros are added around the input image to allow the convolutional kernel to be applied at the edges and corners of the image.
- Strides: The default value for Conv2D and MaxPooling2D layers in Keras is (1,1), which means that the kernel moves one pixel at a time.

To improve model performance and avoid overfitting (Bejani & Ghatee, 2021), we implement data augmentation techniques (Filipi Gonçalves & Santos, 2022). This involves creating new modified images from the original ones, e.g., by rotating, scaling, or cropping the original images. In fig 2, an image is shown that has been processed with various data augmentation techniques. These techniques are rotation, scaling, translation, and cropping. The images are randomly rotated in a range of -10 to +10 degrees, enlarged or reduced in a range of -10% to +10%, and flipped horizontally with a 50% probability. The input images for the model are 180x180 pixels.

Fig. 2. Data augmentation techniques.

In addition to data augmentation, we also use segmentation in the HSV color space to improve the accuracy of the model (Pardede et al., 2019). This involves converting images from the RGB color space to the HSV color space and then segmenting the images based on hue, saturation, and value values.

The model was trained using a dataset consisting of images of different types of clothing. Table 1 shows the types of garments and the number of garments by color used as a database. The dataset is obtained from the Internet, more specifically from the Kaggle site (*Apparel Dataset | Kaggle*, n.d.). The data was divided into training, validation, and test sets. We evaluated the model performance by classification accuracy. We obtained a result of 90%.

Table 1. Clothing data set

Cloth	Amount/Color
Dress	566 Yellow, 818 White, 800 Red, 450 Black, 520 Blue
Pants	246 Pink, 274 White, 308 Red, 870 Black, 798 Blue, 311 Brown, 227 Green
Shirt	230 Green, 332 Red,715 Black, 741 Blue
Shoes	455 Green, 766 Black, 464 Brown, 610 Red, 523 Blue, 600 White, 403 Silver
Shorts	328 Black, 299 Blue, 135 Green, 120 White, 195 Yellow
Hoodies	188 Brown, 347 Pink, 349 Red
Suit	243 Green, 320 Black, 354 White
Skirt	513 Pink, 361 Silver, 409 Yellow

3.2 Determining the Predominant Color of a Base Image

To determine the predominant color in an image, we use a Python program that calculates the percentage of colors in the image using the Euclidean distance in the HSL (Hue, Saturation, Lightness) color space (J. Yu & Li, 2013). The process starts by converting the image from RGB to HSL. Then, a database containing different colors in HSL format is read. This database represents a sample region for each color in HSL space and serves as a color markers (Lin et al., 2015). By comparing the colors in the image with the color markers in the database, we can determine the predominant color in the image.
This method allows us to obtain an accurate representation of the predominant color in an image, which is crucial for our task of searching for similar garments.

3.3 Color database

This study uses 12 specific colors: purple, red, blue, cyan, green, yellow, orange, pink, white, brown, black, and gray. These colors have been selected to cover the most vibrant and saturated tones of the color spectrum. In addition, the inclusion of colors such as white, brown, black, and gray allows us to represent the lightest and darkest tones of the spectrum. Although this article mentions the use of 6 colors to represent different areas of the color spectrum in the HSL (Hue, Saturation, Lightness) model, we have expanded this approach to include 12 colors (Lin et al., 2015). The HSL model is useful for representing colors in terms of their hue (Hue), saturation (Saturation), and lightness (Lightness). By using 12 colors instead of 6, we can cover a wider range of tones in the HSL spectrum and represent a wider variety of colors. Each of the 12 selected colors has several shades. Each of these shades is defined in the HSL format, which allows for an accurate and consistent representation of color.
In Table 2, we present an extract from our database showing a selection of 12 colors that are used in our database. Each color is presented with its name and its value in the HSL (Hue, Saturation, Lightness) format.

Table 2. Color centers in HSL space

COLOR	HSL VALUE
Purple	hsl(285, 37%, 34%)
Red	hsl(355, 96%, 63%)
Blue	hsl(239, 97%, 24%)
Cyan	hsl(196, 46%, 71%)
Green	hsl(89, 22%, 31%)
Yellow	hsl(50, 84%, 53%)
Orange	hsl(25, 75%, 65%)
Pink	hsl(355, 93%, 74%)
White	hsl(192, 22%, 95%)
Brown	hsl(17, 33%, 39%)
Black	hsl(131, 54%, 8%)
Gray	hsl(66, 13%, 69%)

3.4 Color Segmentation

In addition, a segmentation of the color regions in the image is carried out. This involves dividing the image into different areas, each of which contains pixels of a similar color. Figure 3 shows the base image we are working with.

Fig. 3. Base Image

Various filters are also applied to enhance the contrast of the image. These filters can include techniques such as brightness and contrast adjustment, histogram equalization, and edge enhancement. The application of these filters can improve the visual quality of the image and make details easier to see.

Figures 4a and 4b illustrate the results after segmentation and filter application. Figure 4a shows the different zones segmented by color tones, and Figure 4b shows the image with filters applied to enhance contrast.

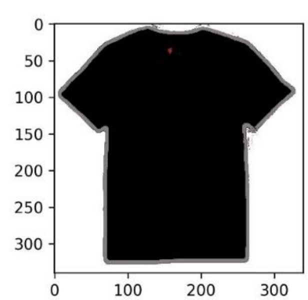


Fig. 4a. Segmentation

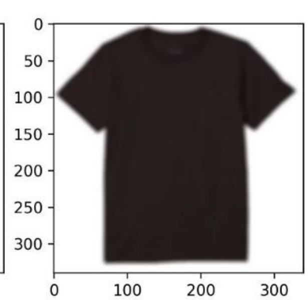


Fig. 4b. Filtering

3.5 Image Processing and Color Classification

Each pixel of the image is classified by calculating the Euclidean distance between that pixel and each color marker. The smallest distance indicates that the pixel is more like that color. In this way, each pixel is classified and the percentage of each color in the image is determined.

The resulting color will be represented as a string of characters. For example, if the predominant color in the image is black, these bits will be represented as characters and will be concatenated with the result of the classification of the CNN. The combination of these two strings will form a keyword that will be entered into the search engine to obtain a corresponding image, in this case, several blue shirts.

3.6 Search engineer

For practical research purposes, a keyword based search engine was developed. This engine, implemented with SerpApi, analyzes the keywords entered by the user and compares them with the content of the indexed web pages (*SerpApi: Google Search API*, n.d.).

An API request is made, and the results are returned in a Python dictionary. The code iterates through the returned images and adds each unique link to a list of results. Additionally, the images are extracted and placed in a desired folder. The images are named in an increasing order starting from 'image_0', incrementing with each iteration.

As shown in Figure 5, the search engine first prompts the user to enter a keyword to search for. Once the keyword is entered, the search engine performs its function and provides a series of links as a result. These links correspond to images that match the keyword entered by the user. This process illustrates how the keyword-based search engine can facilitate the retrieval of relevant information in an efficient manner. The retrieved images, now stored and organized in a local folder, are ready for further processing or analysis.

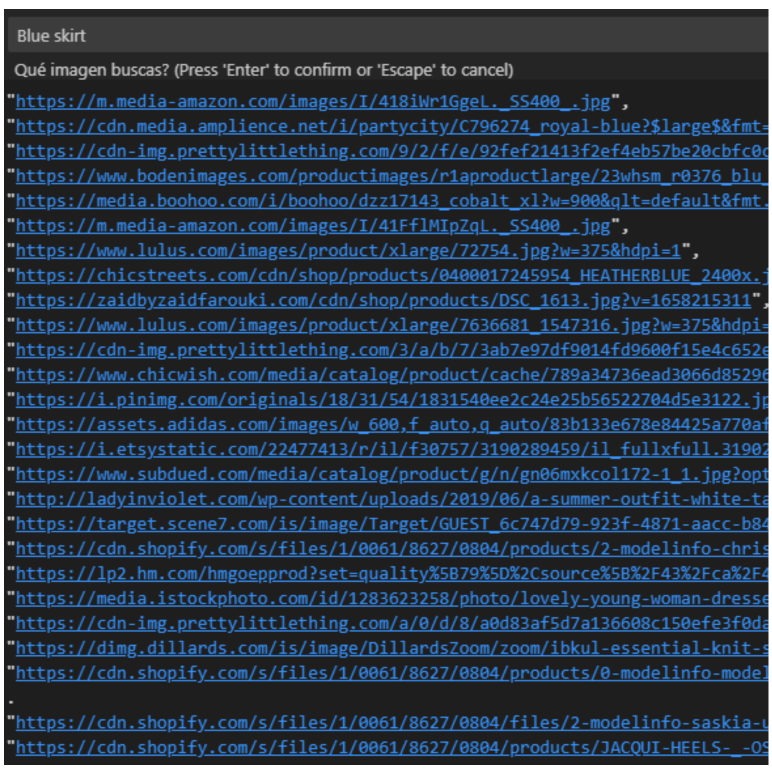


Fig. 5. Search engineer

The purpose of the search engine is to aggregate the concatenation of the strings obtained from the garment classification results using the CNN and the base image color retrieval.

3.7 Image Feature Detection and Comparison Using the ORB Algorithm

While we already have a technique for recognizing and classifying a garment, as well as a method for determining the predominant color, and not to mention the search engine, these three sections or techniques could, in theory, make the application work and obtain the desired objects. However, a crucial component is missing: the ability to detect if the image to be compared is the same as the base image.

That is, if you are using a base garment style, what is desired is that the garments obtained are of the same style. To ensure that the garments obtained from the search engine are of the same style as the base garment, we implemented in Python an approach based on the comparison of the morphologies of the images. This approach is based on the use of the ORB computer vision algorithm for feature extraction from images.

This algorithm extracts features from images and generates descriptors that can be compared to determine if two images are similar. In our case, we use this algorithm to compare the morphology of a base image with that of other images obtained from the internet, thus ensuring that the garments obtained are of the same style as the base garment.

In Figure 6, a black round-neck style t-shirt with short sleeves is observed. The key points, which are generated by the ORB algorithm, are represented by small circles of varied colors such as pink, blue, green, and yellow.

The program will display a list in descending order of the images that have more features in common with the original image, placing first the one that most resembles the original. Table 6 offers a clearer representation of this list.

Fig. 6. Feature extraction using ORB.

In this section, we not only extract features but also enhance the image comparison technique. Specifically, we convert the base image and the image to be compared to RGB to obtain a scalar number of their colors. For both images, we add up the R, G, B components, which gives us a single number, and this is compared between the base image and the image to be compared, the smaller result will be the closest to the base image.

Table 7 shows the results of the color comparison between the base image and the images to be compared. Here I explain in detail:

Sum_RGB_base: This is the scalar number obtained from the sum of the R, G, B components of the base image. In this case, it is 509.37398 for all comparisons.

Sum_RGB_current: This is the scalar number obtained from the sum of the R, G, B components of the current image that is being compared with the base image. You can see that this value varies for each compared image.

Diff_Sum_RGB: This is the absolute difference between Sum_RGB_base and Sum_RGB_current. It represents the color difference between the base image and the image to be compared. The smaller this value, the closer the image to compare will be to the color of the base image.

Looking at the table, you can see that the first image to be compared has the same color as the base image, as Diff_Sum_RGB is 0. The other images have colors that differ to a greater extent from the base image, as indicated by their higher Diff_Sum_RGB values.

Table 3. RGB differences

SUM_RGB_BASE	SUM_RGB_CURRENT	DIFF_SUM_RGB
509.37398	509.37398	0
509.37398	618.036702	108.662721
509.37398	363.630554	145.743426
509.37398	646.963172	137.589192

If you look closely, we have two parts. The first is the number of features of the base image compared to the image to be compared, and the second part is the scalar number of the color of both figures. To determine which image is most like the base image, the following formula was used:

$$Cal = \omega_1 P_1 + \frac{P_2}{\omega_2} \tag{1}$$

Where:

$\boldsymbol{Cal}$ it is the total score to determine which image is the best.

$\boldsymbol{\omega_1, \omega_2}$ are the weights that determine the relative importance or contribution of the variables.

P_1 It is the scalar number obtained from the difference of the R, G, B components of the base image against the obtained image.

$\boldsymbol{P_2}$It is the number of features obtained by the ORB algorithm.

The results obtained will be the product of this equation, which is a measure of similarity based on features and colors of the images. These results will be presented as a list ordered from smallest to largest. In this list, the smallest value will correspond to the image that most closely resembles the original image.

4 Results

4.1 Convolutional Neural Network Performance and Results in Image Classification

To evaluate the accuracy of our model, we tested the CNN using an image that was not part of the training database. The image, which represents a blue shirt (see Figure 3), was processed by our model. The results indicated that our model correctly classified the garment with an accuracy of 99.53%. This high level of accuracy suggests that our model is capable of successfully identifying and classifying garments based on their type and color. In addition, you can see the result of the classification of the blue shirt in Figure 6. This visual result provides a clear representation of how our CNN model was able to successfully classify the garment.

Fig. 6. Shirt Classification

4.2 Image Processing and Color Classification Using Euclidean Distance

As mentioned in the 'Image Processing and Color Classification' section, we take as an example obtaining the color of an image of a black shirt. This section shows the result obtained by determining the color of an image using Euclidean distance and a set of color data with different spectrum centers in relation to their color.

A base image, in this case Figure 3, is processed with code that determines the predominant color of the image. The code generates a data frame with 12 colors and the percentage of each color present in the image. According to the data from Table 2, the predominant color in the image is black, which represents approximately 56.56% of the image. The second most predominant color is white, which represents approximately 38.78% of the image. Gray is also present but in a much smaller proportion. The colors purple, pink, red, blue, cyan, green, yellow, orange, and brown are not present in the image.

The detailed results are presented in Table 3. From this table, we can see that the image is primarily composed of two large sections: the white background and the black color, which is of interest to us. The latter will be the string that will be sent to the result of the image prediction and will be concatenated to search for images on the internet. The minority colors in the image can be appreciated in the image segmentation, which is visualized in Figure 4a. Although these colors represent a small proportion of the total image, they can be important for understanding the complete composition of the image. The image segmentation can help highlight these details that might go unnoticed in a regular visual inspection of the image.

Table 3. Color Distribution in the Base Image

COLOR	PERCENTAGE
Black	0.565631488
White	0.387811419
Gray	0.045847751
Pink	0.000198962
Brown	0.000510381
Purple	0
Red	0
Blue	0
Cyan	0
Green	0
Yellow	0
Orange	0

4.3 Similarity measurement result based on image features and colors.

In the section ‘Detection and comparison of image features using the ORB algorithm’, we explore how to obtain the main features of the base image in relation to the images to be compared. In addition, we introduced another component: the comparison of RGB colors between the same images. For this, we used the image from Figure 6 and compared it with the images extracted from the search engine, as shown in Figure 7. It should be clarified that only 20 images were used due to the time it takes to download. The results are presented in Table 4.

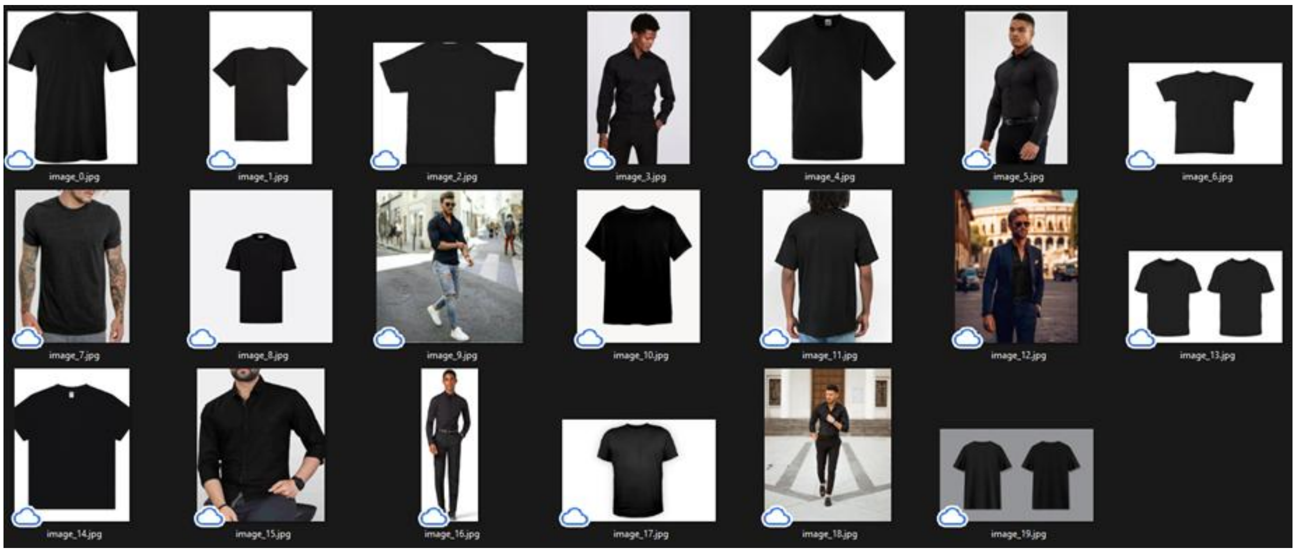


Fig.7. Images downloaded by search engine.

Table 4 presents the results obtained by applying the ORB technique to extract features from images and calculate the difference in the sum of RGB values. We used the image from Figure 17 as the base image and compared it with other images extracted from the search engine (see Figure 7). The table shows several columns of data:

- ‘image_name’: The name of each compared image.
- ‘num_matches’: The number of feature matches between the base image and the compared image.
- ‘Sum_RGB_base’: The sum of the RGB values of the base image.
- ‘Sum_RGB_current’: The sum of the RGB values of the compared image.
- ‘Diff_Sum_RGB’: The difference between the sum of the RGB values of the base image and the compared image.
- ‘Rating’: A measure of the similarity between the images based on the difference in the sum of the RGB values.

For example, the image ‘image_0.jpg’ has 8 feature matches with the base image, and the difference between the sum of the RGB values of the base image and ‘im-age_0.jpg’ is 15.63, resulting in a similarity rating of 7.88. According to these results, ‘image_0.jpg’ is the most similar image to the base image among the compared im-ages. It is important to mention that the images in Table 4 are ordered from the most like the least like the base image, according to the similarity rating.

Table 4. Results similarity measure based on image features and colors.

NAME_IMAGE	NO_MATCHES	SUM_RGB_BASE	SUM_RGB_CURRENT	DIFF_SUM_RGB	CALIFICACION
image_0.jpg	8	353.06866	337.438017	15.630643	7.877821
image_10.jpg	9	353.06866	378.565633	25.496974	12.804042
image_2.jpg	4	353.06866	326.813965	26.254695	13.252347
image_3.jpg	22	353.06866	387.04931	33.98065	17.013052
image_19.jpg	17	353.06866	305.576688	47.491972	23.775398
image_4.jpg	15	353.06866	400.855078	47.786418	23.926542
image_9.jpg	28	353.06866	409.863417	56.794758	28.415236
image_11.jpg	19	353.06866	419.155459	66.086799	33.069716
image_5.jpg	6	353.06866	273.261144	79.807516	39.987091
image_8.jpg	8	353.06866	440.319097	87.250437	43.687718
image_15.jpg	24	353.06866	259.291541	93.777118	46.909393
image_18.jpg	9	353.06866	457.663824	104.595164	52.353137
image_12.jpg	23	353.06866	493.50621	140.437551	70.240514
image_14.jpg	7	353.06866	508.318127	155.249467	77.696162
image_1.jpg	9	353.06866	512.797278	159.728618	79.919865
image_17.jpg	19	353.06866	185.10523	167.96343	84.008031
image_6.jpg	11	353.06866	527.887247	174.818587	87.454748
image_16.jpg	9	353.06866	529.192188	176.123529	88.11732
image_13.jpg	11	353.06866	146.091089	206.97757	103.53424
image_7.jpg	7	353.06866	596.213935	243.145275	121.644066

4.4 Search for the best image.

To facilitate the linking of the above parts and for research purposes, we chose to create an application using Tkinter (Roseman, 2020). The interface consists of five buttons: “Select Image”, “Predict Image”, “Predict Color”, “Search Image”, and “Best Image”, plus an image result field.

First, we press the ‘Select Image’ button to select the base image to predict. Subsequently, this image will be displayed in the interface, as can be seen in Figure 8.

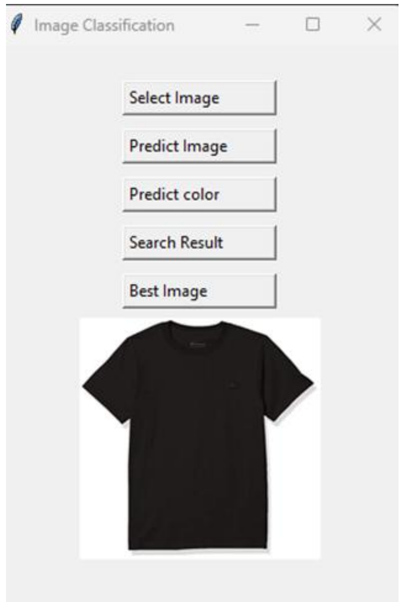


Fig. 8. Select Image button.

For the application to identify the type of garment, we select the "Predict image" but-ton. This button feeds the selected image into the artificial neural network model trained to classify the garment into one of the labels of the CNN model. Then, it will show us a message with the name of the garment and the percentage of accuracy of the prediction. The result can be seen in Figure 9.

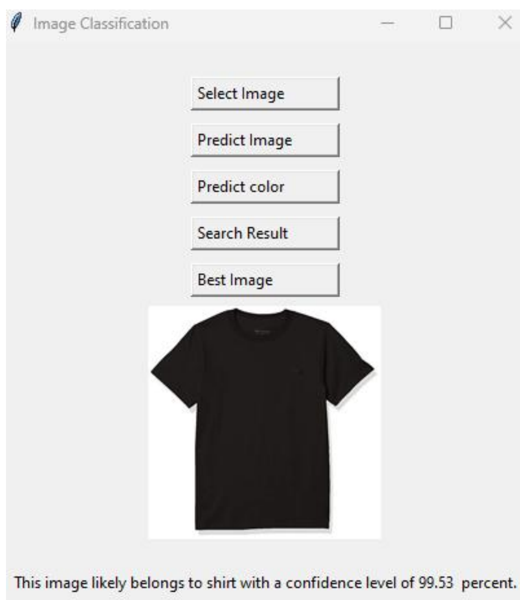


Fig. 9. Predict Image button.

The “Predict Color” button passes the image to the Gaussian function to obtain the predominant color. This action will provide us with a message with the main color of the image. We can see an example of this in Figure 10.

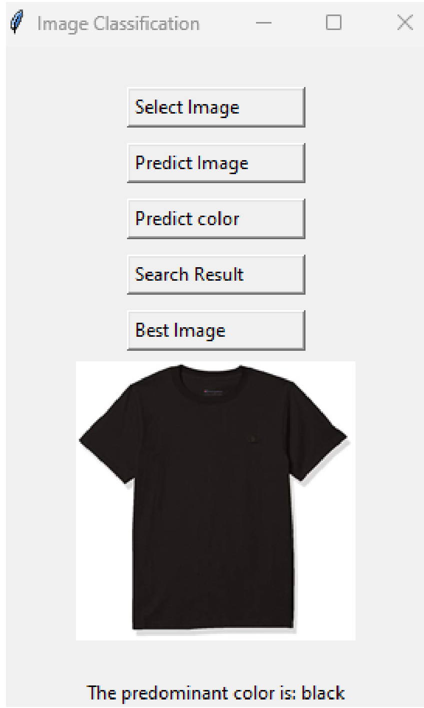


Fig. 10. Predict Color Image.

Once we have identified the garment and its color, the next step is to search for similar images on the web. To do this, we use the 'Search image' button, which will use the two words obtained previously, 'black' and 'shirt', as search terms. This process will provide us in a pop-up window with all the links containing images related to these terms, which will be automatically downloaded. An example of this process is shown in Figure 11.

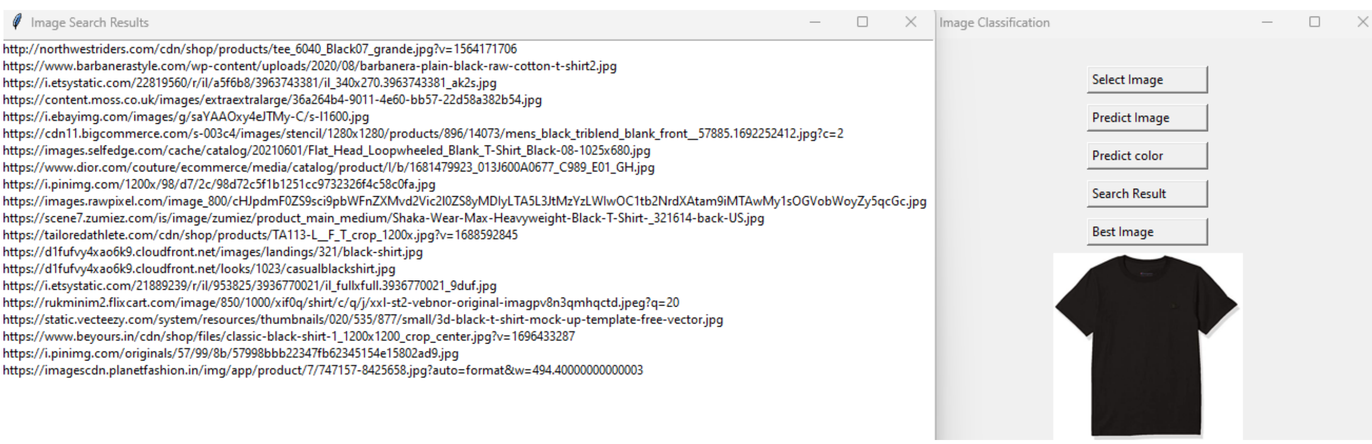
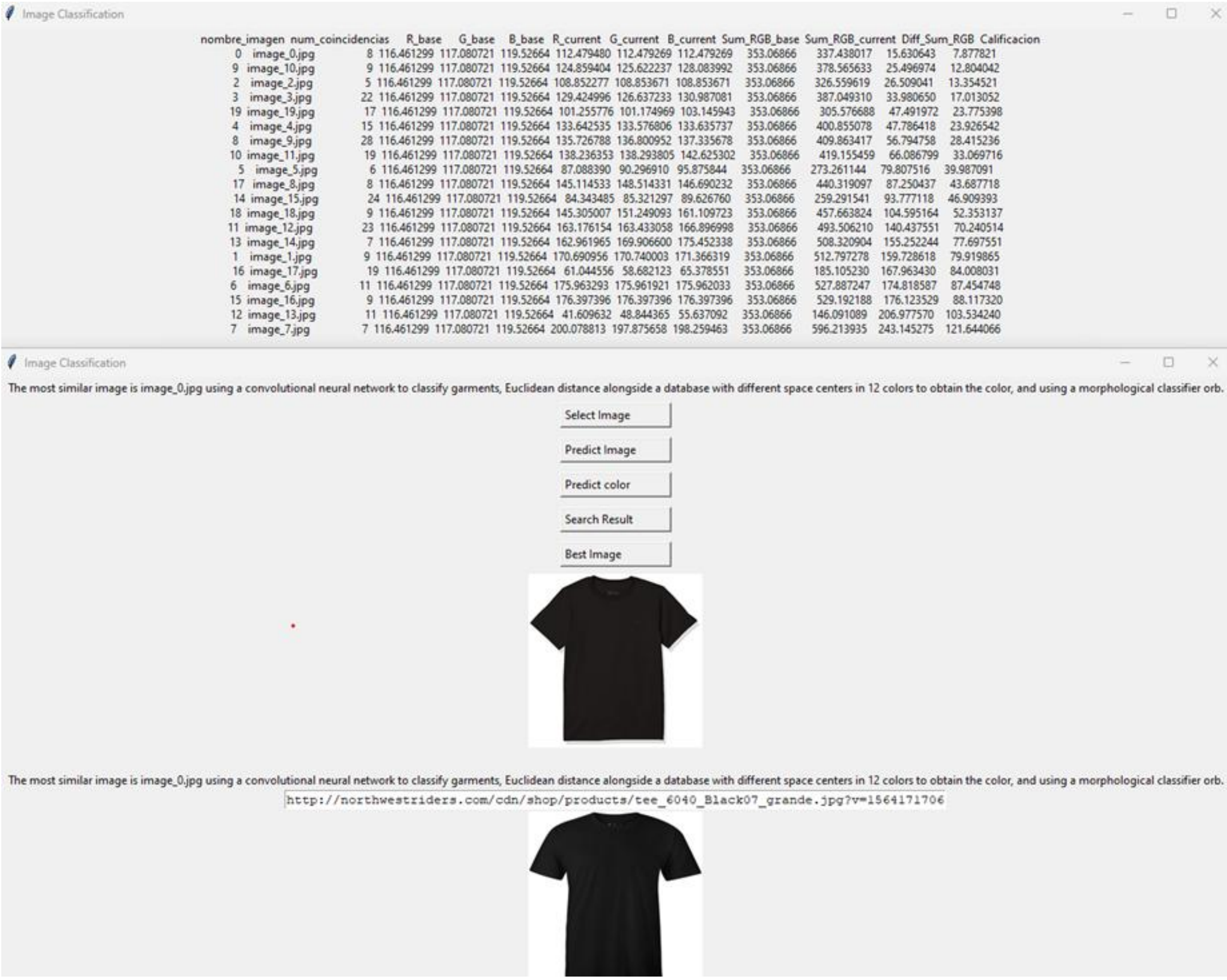


Fig. 11. CNN generated concatenated word search and color retrieval.

The last step is to obtain the best image based on the downloaded images and the base image. To do this, we will use the 'Best Image' button, which will show us the winning image, the name of the image and the download link. In addition, a pop-up window will open showing the data that made this image the winner. In Figure 12, we can see both the base image and the winning image, along with a table showing the calculations that contributed to the selection of the winning image.



	nombre_imagen	num_coincidencias	R_base	G_base	B_base	R_current	G_current	B_current	Sum_RGB_base	Sum_RGB_current	Diff_Sum_RGB	Calificacion
0	image_0.jpg	8	116.461299	117.080721	119.52664	112.479480	112.479269	112.479269	353.06866	337.438017	15.630643	7.877821
9	image_10.jpg	9	116.461299	117.080721	119.52664	124.859404	125.622237	128.083992	353.06866	378.565633	25.496974	12.804042
2	image_2.jpg	5	116.461299	117.080721	119.52664	108.852277	108.853671	108.853671	353.06866	326.559619	26.509041	13.354521
3	image_3.jpg	22	116.461299	117.080721	119.52664	129.424996	126.637233	130.987081	353.06866	387.049310	33.980650	17.013052
19	image_19.jpg	17	116.461299	117.080721	119.52664	101.255776	101.174969	103.145943	353.06866	305.576688	47.491972	23.775398
4	image_4.jpg	15	116.461299	117.080721	119.52664	133.642535	133.576806	133.635737	353.06866	400.855078	47.786418	23.926542
8	image_9.jpg	28	116.461299	117.080721	119.52664	135.726788	136.800952	137.335678	353.06866	409.863417	56.794758	28.415236
10	image_11.jpg	19	116.461299	117.080721	119.52664	138.236353	138.293805	142.625302	353.06866	419.155459	66.086799	33.069716
5	image_5.jpg	6	116.461299	117.080721	119.52664	87.088390	90.296910	95.875844	353.06866	273.261144	79.807516	39.987091
17	image_8.jpg	8	116.461299	117.080721	119.52664	145.114533	148.514331	146.690232	353.06866	440.319097	87.250437	43.687718
14	image_15.jpg	24	116.461299	117.080721	119.52664	84.343485	85.321297	89.626760	353.06866	259.291541	93.777118	46.909393
18	image_18.jpg	9	116.461299	117.080721	119.52664	145.305007	151.249093	161.109723	353.06866	457.663824	104.595164	52.353137
11	image_12.jpg	23	116.461299	117.080721	119.52664	163.176154	163.433058	166.896998	353.06866	493.506210	140.437551	70.240514
13	image_14.jpg	7	116.461299	117.080721	119.52664	162.961965	169.906600	175.452338	353.06866	508.320904	155.252244	77.697551
1	image_1.jpg	9	116.461299	117.080721	119.52664	170.690956	170.740003	171.366319	353.06866	512.797278	159.728618	79.919865
16	image_17.jpg	19	116.461299	117.080721	119.52664	61.044556	58.682123	65.378551	353.06866	185.105230	167.963430	84.008031
6	image_6.jpg	11	116.461299	117.080721	119.52664	175.963293	175.961921	175.962033	353.06866	527.887247	174.818587	87.454748
15	image_16.jpg	9	116.461299	117.080721	119.52664	176.397396	176.397396	176.397396	353.06866	529.192188	176.123529	88.117320
12	image_13.jpg	11	116.461299	117.080721	119.52664	41.609632	48.844365	55.637092	353.06866	146.091089	206.977570	103.534240
7	image_7.jpg	7	116.461299	117.080721	119.52664	200.078813	197.875658	198.259463	353.06866	596.213935	243.145275	121.644066

Fig. 12. Best image

5 Conclusion

In addition to the methodology presented, the use of the ORB (Oriented FAST and Rotated BRIEF) technique played a significant role in this research. ORB is a robust and efficient method for feature extraction and matching, which is crucial in comparing images. It was used to extract features from the images and calculate the difference in the sum of RGB values. This provided a more detailed comparison between the base image and the other images, enhancing the accuracy of the results.

The application developed using Tkinter served as an effective tool for implementing and demonstrating the methodology. It provided a user-friendly interface for selecting and predicting images, predicting colors, searching for similar images on the web, and determining the best image based on the downloaded images and the base image. The application streamlined the process and made it more accessible and understandable.

The combination of these techniques and tools, along with the unique approach of using a database with 12 colors in HSL space, resulted in a highly efficient and accurate methodology for obtaining relevant objects on the web. This research demonstrates the potential of integrating various techniques and approaches in image processing and computer vision tasks. It opens up new possibilities for further research and development in this field.

In conclusion, this research has successfully demonstrated a comprehensive and effective methodology for image classification and retrieval. The use of CNN for garment classification, Euclidean distance for color retrieval, ORB for feature extraction, and the developed application for implementation, all contributed to the successful realization of this research. The results obtained validate the efficiency and accuracy of the methodology, paving the way for future research in this area.

6 References

Apparel Dataset | Kaggle. (n.d.). Retrieved September 27, 2022, from https://www.kaggle.com/datasets/kaiska/apparel-dataset/code

Bejani, M. M., & Ghatee, M. (2021). A systematic review on overfitting control in shallow and deep neural networks. *Artificial Intelligence Review*, *54*(8), 6391–6438. https://doi.org/10.1007/s10462-021-09975-1

Capa MaxPooling2D. (2022). Retrieved September 27, 2022, from https://keras.io/api/layers/pooling_layers/max_pooling2d/

Chen, L., Li, S., Bai, Q., Yang, J., Jiang, S., & Miao, Y. (2021). Review of Image Classification Algorithms Based on Convolutional Neural Networks. *Remote Sensing*, *13*(22). https://doi.org/10.3390/rs13224712

Chu, G., Peng, Y., & Luo, X. (2023). ALGD-ORB: An improved image feature extraction algorithm with adaptive threshold and local gray difference. *PloS One*, *18*(10), e0293111. https://doi.org/10.1371/journal.pone.0293111

Conv2D layer. (2022). Retrieved September 27, 2022, from https://keras.io/api/layers/convolution_layers/convolution2d/

Dense layer. (2022). Retrieved September 27, 2022, from https://keras.io/api/layers/core_layers/dense/

Elaw, S., Abd-Elhafiez, W. M., & Heshmat, M. (2019). Comparison of Video Face Detection methods Using HSV, HSL and HSI Color Spaces. *2019 14th International Conference on Computer Engineering and Systems (ICCES)*, 180–188. https://doi.org/10.1109/ICCES48960.2019.9068182

Filipi Gonçalves, C., & Santos, D. (2022). *Avoiding Overfitting: A Survey on Regularization Methods for Convolutional Neural Networks*. https://doi.org/10.1145/3510413

Lin, T., Liao, B.-H., Hsu, S.-L., & Wang, J. (2015). Experimental investigation of HSL color model in error diffusion. *2015 8th International Conference on Ubi-Media Computing (UMEDIA)*, 268–272. https://doi.org/10.1109/UMEDIA.2015.7297467

Pardede, J., Husada, M., Hermana, A., & Rumapea, S. (2019). *Fruit Ripeness Based on RGB, HSV, HSL, L*a*b* Color Feature Using SVM*. https://doi.org/10.1109/ICoSNIKOM48755.2019.9111486

Roseman, Mark. (2020). *Modern Tkinter for busy Python developers : quickly learn to create great looking user interfaces for Windows, Mac and Linux using Python's standard GUI toolkit*. 257.

SerpApi: Google Search API. (n.d.). Retrieved March 28, 2023, from https://serpapi.com/

Sharma, N., Jain, V., & Mishra, A. (2018). An Analysis of Convolutional Neural Networks for Image Classification. *Procedia Computer Science*, *132*, 377–384. https://doi.org/10.1016/J.PROCS.2018.05.198

Su, F., Xu, H., Chen, G., Wang, Z., Sun, L., & Wang, Z. (2019). *Improved Simple Linear Iterative Clustering Algorithm Using HSL Color Space BT - Intelligent Robotics and Applications* (H. Yu, J. Liu, L. Liu, Z. Ju, Y. Liu, & D. Zhou, Eds.; pp. 413–425). Springer International Publishing.

Yu, J., & Li, C. (2013). Face Recognition Based on Euclidean Distance and Texture Features. *2013 International Conference on Computational and Information Sciences*, 211–213. https://doi.org/10.1109/ICCIS.2013.63

Zhang, G., Qin, D., Yang, J., Yan, M., Tang, H., Bie, H., & Ma, L. (2022). UAV Low-Altitude Aerial Image Stitching Based on Semantic Segmentation and ORB Algorithm for Urban Traffic. *Remote Sensing*, *14*(23). https://doi.org/10.3390/rs14236013