Indonesian Journal of Electrical Engineering and Computer Science
Vol. 31, No. 2, August 2023, pp. 866∼875
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v31.i2.pp866-875

Learning color names using least-squares probabilistic classifiers

Janya Sainui, Chouvanee Srivisal
Division of Computational Science, Faculty of Science, Prince of Songkla University, Songkhla, Thailand

Article Info

Article history:

Received Nov 12, 2022
Revised Feb 18, 2023
Accepted Mar 12, 2023

Keywords:

Color classification
Color naming
Least-squares probabilistic classifiers
Local scaling parameter
Probabilistic classification

ABSTRACT

Color name is one of the important features for computer vision. Many existing methods proposed to classify colors into a small number of color names. In this paper, we propose an alternative method with the goal to improve the accuracy for assigning a color name to an object in the given image. We here use the least-squares probabilistic classifiers (LSPC) with the local scaling parameters for solving this task. The benefit of the LSPC is that its solution can be computed analytically so that the obtained solution is global optimum, while the local scaling parameters play an important role to deal with the data including clusters with different local statistics as appeared in the real-world data. To deal with this task, the LSPC is learned to assign a color name to each pixel with the highest of the class-posterior density distribution. Then, the estimations of the class-posterior density distributions are utilized to compute the scores for predicting a color name to the given object. Lastly, the color name with the highest score is chosen as a predicted color name for that object. The experimental results on the eBay data set show the improvements over previously proposed methods.

This is an open access article under the CC BY-SA license.

Corresponding Author:

Janya Sainui
Division of Computational Science, Faculty of Science, Prince of Songkla University
15 Karnjanavanich Rd., Hat Yai, Songkhla 90110, Thailand
Email: janya.s@psu.ac.th

1. INTRODUCTION

Color name is one of the most important attributes used to describe images in computer vision application like image retrieval [1]-[3] object recognition [4], [5], and object tracking [6]-[8]. More specifically, for example, Schauerte and Fink [9] required color term model in human-robot interaction system for communicating a color of the intended object such as “red book”, “yellow cup”. Schauerte *et al.* [10] used the color attribute to help blind finding things by using a small camera to detect the desired object; the color names are utilized in this context. The user is then guided toward the location of the target object. Heer and Stone [11] introduced using the color name model to simplify the user interface of image editors (e.g., Adobe Photoshop and Illustrator) resulting in, for example, the user can select image pixels using color names. Weijer and Schmid [12] used the color name model in the automatic image annotation system, which will be useful for image retrieval systems. Therefore, given an image, our goal here is to provide an algorithm that automatically assigns a color name to an object in the image in the same way as a human describes it. This is a challenging task in computer vision because there is a limitation of color names available.

Many existing methods in the literature review reduced the millions of colors into 11 English basic color names: pink, purple, red, orange, yellow, green, blue, blown, white, gray, and black [9], [13]-[19]. To deal with this problem, they collected images from the internet for training data, and most of them learned the color names based on statistical models. For instance, Weijer *et al.* [14], [15] proposed learning color names from the Google images. They used probabilistic latent semantic analysis (PLSA) model [13] to discover colors in a bag of pixel representation, where every pixel was represented by its lab value which was divided into $10 \times 20 \times 20$ bins. An image is then represented by a histogram indicating how many pixels are assigned to each bin. Similarly, Schauerte and Fink [9] proposed to learn color names on images acquired from the internet. Instead of using the PLSA model with the histogram of Lab, they applied a probabilistic hue, saturation, lightness (HSL) model, where the HSL color is divided into $32 \times 8 \times 8$ histogram bins. The process to remove outliers in the training data set was included. They reported the performance compared with the method proposed by [14], [15], showing that the results were comparable. However, the number of histogram bins used in their method is lower than that of proposed in [14], [15]. Schauerte and Stiefelhagen [20] proposed to use the supervised latent Dirichlet allocation (SLDA) [21] to train the color name model. They used Lab color space with $32 \times 32 \times 32$ histogram bins. They preprocessed the training data by applying Kullback-Leibler divergence ratios to remove outliers from the training data set. However, the classification results on the eBay data set indicate that the performance of their proposed method is not drastically different compared to the color name models proposed by [9], [14], [15].

The key problem of the above mentioned methods is that the solution of their statistical model is local optimum. Moreover, it is necessary to estimate other densities to obtain the final class-posterior probability. Therefore, the estimation of the class-posterior probability may not be reliable. In this paper, we propose an alternative method to color naming by exploring the probabilistic classifier with the global solution. We also focus on reducing the color space to the 11 basic color names, where our purpose is to learn the color classifiers for image pixels to end up with a color name for an object in such a way that the performance of the proposed method is closer to human performance. We also use images collected from the internet to train our model; however, the representation of colors is not histogram, we study based on raw pixel intensities, and the statistical classifier called the least-squares probabilistic classifiers (LSPC) is applied to learn the model from the densities of pixels of each color term. The contribution of this paper is a novel approach for assigning a color name to an object in the image. We use the LSPC which allows us to obtain a global solution analytically, in order to estimate the class-posterior probability for predicting color names to the pixels of the object. As the training data contains clusters with different distributions, we adopt the local scaling parameters instead of using a single parameter for computing the Gaussian width. We then utilize the class labels with their confidences (i.e., the class-posterior probabilities) for assigning a color name to the given object. We compare the proposed method with the counterpart methods resulting in that the proposed method outperforms the competitive methods. Lastly, we discuss the quality and the limitations of the LSPC to this work.

The rest of this paper is organized as follows. We briefly overview the related works in section 2. In section 3, we explain our proposed color naming method. The experimental results and discussion are presented in section 4. Finally, the conclusion is provided in section 5.

2. RELATED WORKS

In this section, we first review the problem of probabilistic classification. Next, its solution via least-squares method called least-squares probabilistic classifier [22] is explained. Finally, we describe the local scaling parameters [23] for Gaussian widths.

2.1. Probabilistic classification

Let us suppose that we are given a training set of n paired samples of input $\boldsymbol{x} \in \mathbb{R}^d$ and its corresponding label $y \in \{1, \ldots, c\}$:

$$\{(\boldsymbol{x}_i, y_i)\}_{i=1}^{n},$$

drawn independently from a joint probability distribution with density $p(\boldsymbol{x}, y)$, where d denotes the dimensionality of input $\boldsymbol{x}$, and c denotes the number of classes. The goal of the probabilistic classification is to approximate the class-posterior probability $p(y|\boldsymbol{x})$ from the samples $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^{n}$ in order to classify a test sample $\boldsymbol{x}$ to the class $\widehat{y}$ by:

$$\widehat{y} := \underset{y \in \{1,\ldots,c\}}{\operatorname{argmax}} \ p(y|\boldsymbol{x}),$$

with confidence $p(\widehat{y}|\boldsymbol{x})$.

2.2. Least-squares probabilistic classifiers

The class-posterior probability $p(y|\boldsymbol{x})$ can be expressed as:

$$p(y|\boldsymbol{x}) = \frac{p(\boldsymbol{x}, y)}{p(\boldsymbol{x})},$$

where $p(\boldsymbol{x}, y)$ is the joint probability density of $\boldsymbol{x}$ and y, and $p(\boldsymbol{x})$ is the marginal density of $\boldsymbol{x}$, where $p(\boldsymbol{x})$ is assumed to be nonnegative for all $\boldsymbol{x}$. LSPC models the class-posterior probability $p(y|\boldsymbol{x})$ for class y by:

$$q(y|\boldsymbol{x}; \boldsymbol{\theta}_y) := \sum_{\ell=1}^{b} \theta_{y,\ell} \boldsymbol{\psi}_{y,\ell}(\boldsymbol{x}) = \boldsymbol{\theta}_y^\top \boldsymbol{\psi}_y(\boldsymbol{x}),$$

where $\top$ denotes the transpose of a matrix or a vector, b is the number of parameters, $\boldsymbol{\theta}_y = (\theta_{y,1}, \ldots, \theta_{y,b})$ are parameters to be learned from training samples and,

$$\boldsymbol{\psi}(\boldsymbol{x}) = (k(\boldsymbol{x}, \boldsymbol{x}_{y,1}), \ldots, k(\boldsymbol{x}, \boldsymbol{x}_{y,b}))$$

is the basic function, where:

$$k(\boldsymbol{x}, \boldsymbol{x}') = \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{x}'\|^2}{2\sigma^2}\right) \tag{1}$$

is the Gaussian kernel with width σ. Then, the model $q(y|\boldsymbol{x}; \boldsymbol{\theta}_y)$ is fit with the true class-posterior probability $p(y|\boldsymbol{x})$ by least-squares as:

$$J_y(\boldsymbol{\theta}_y) := \frac{1}{2} \int \left(q(y|\boldsymbol{x}; \boldsymbol{\theta}_y) - p(y|\boldsymbol{x})\right)^2 p(\boldsymbol{x}) d\boldsymbol{x}.$$

An empirical and regularized solution of LSPC is given by:

$$\widehat{\boldsymbol{\theta}}_y := \underset{\boldsymbol{\theta}_y \in R^b}{\text{argmin}} \left[\frac{1}{2} \boldsymbol{\theta}_y^\top \widehat{\boldsymbol{H}}_y \boldsymbol{\theta}_y - \widehat{\boldsymbol{h}}_y^\top \boldsymbol{\theta}_y + \lambda \boldsymbol{\theta}_y^\top \boldsymbol{\theta}_y\right]$$

where $\lambda \geq 0$ is the regularization parameter, $\widehat{\boldsymbol{H}}_y$ and $\widehat{\boldsymbol{h}}_y$ are defined as:

$$\widehat{\boldsymbol{H}}_y := \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{\psi}(\boldsymbol{x}_i) \boldsymbol{\psi}(\boldsymbol{x}_i)^\top, \widehat{\boldsymbol{h}}_y := \frac{1}{n} \sum_{j: y_j = y} \boldsymbol{\psi}(\boldsymbol{x}_j).$$

therefore, the parameters $\widehat{\boldsymbol{\theta}}_y$ can be computed analytically, where the solution can be obtained as:

$$\widehat{\boldsymbol{\theta}}_y = (\widehat{\boldsymbol{H}}_y + \lambda \boldsymbol{I}_b)^{-1} \widehat{\boldsymbol{h}}_y.$$

finally, given a test input point $\boldsymbol{x}$, the normalized estimator of the class-posterior probability for class y is given by:

$$\widehat{p}(y|\boldsymbol{x}) = \frac{\max(0, q(y|\boldsymbol{x}; \widehat{\boldsymbol{\theta}}_y))}{\sum_{y'=1}^{c} \max(0, q(y'|\boldsymbol{x}; \widehat{\boldsymbol{\theta}}_{y'}))}$$

the main advantage of LSPC is that it can obtain the globally optimal solution just by solving a system of linear equations, so that the computational time is efficient. For more details of LSPC please refer to [22], [24]. A MATLAB implementation of LSPC is available from [25].

2.3. Local scaling parameters

The results of the learned parameters, $\widehat{\boldsymbol{\theta}}_y$, depend on the choices of Gaussian width σ included in the basis function (i.e., eq.(1)), and we found that, as we randomly selected the pixels for training data, the distributions of training pixel values of each color class are different as illustrated in Figure 1, where Figure 1(a), Figure 1(b), and Figure 1(c) show the distribution of our training data points in RGB, HSV, and Lab color spaces, respectively. To overcome this problem, instead of choosing a single parameter σ, we apply the local scaling parameters introduced by [23] to compute the parameter $\sigma_{\boldsymbol{x}}$ for a pixel value $\boldsymbol{x}$. Thus, the basis function can be expressed as:

$$k(\boldsymbol{x},\boldsymbol{x}') = \exp\left(-\frac{\|\boldsymbol{x}-\boldsymbol{x}'\|^2}{2\sigma_{\boldsymbol{x}}\sigma_{\boldsymbol{x}'}}\right)$$

where $\sigma_{\boldsymbol{x}}$ and $\sigma_{\boldsymbol{x}'}$ are local scaling parameters for $\boldsymbol{x}$ and $\boldsymbol{x}'$ respectively, and they are defined as:

$$\sigma_{\boldsymbol{x}} = \|\boldsymbol{x}-\boldsymbol{x}^{(k)}\| \text{ and } \sigma_{\boldsymbol{x}'} = \|\boldsymbol{x}'-\boldsymbol{x}'^{(k)}\| \tag{2}$$

$\boldsymbol{x}^{(k)}$ is the k^{th} nearest neighbor sample of $\boldsymbol{x}$, and $\boldsymbol{x}'^{(k)}$ is the k^{th} nearest neighbor sample of $\boldsymbol{x}'$, where the number of the nearest neighbor sample k can be chosen by cross validation. We found through the experiments that the estimation of the class-posterior probability $p(y|\boldsymbol{x})$ by LSPC becomes more reliable.

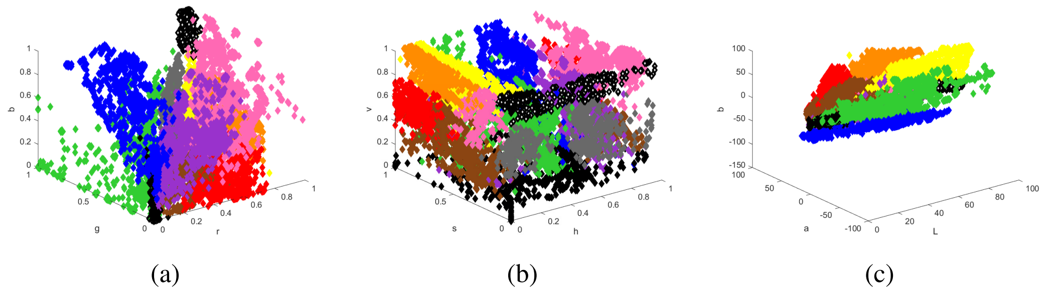


Figure 1. Training data points in different color spaces (a) RGB, (b) HSV, and (c) Lab

3. THE PROPOSED COLOR NAMING METHOD

In this section, we explain our proposed method to assign a color name to an object in the given image. There are two main steps. We firstly use the LSPC with the local scaling parameters to build the classification model for assigning a color name to each pixel of the object. Then, the class-posterior probabilities are used to compute the score to assign a color name to the object. The details are as follows.

3.1. Assigning a color name to each pixel

Given an image, the goal of this step is to label a color name to each pixel corresponding to the object region. To learn a color name model from probabilistic view, let the samples $\{\boldsymbol{x}_i \in \mathbb{R}^d\}_{i=1}^n$ are the pixel intensity values in any color space like red, green, blue (RGB), HSV, and Lab; however, through our experiments we found that Lab color space is the best choice for the LSPC, and the labels $\{y_i\}_{i=1}^n \in \{1,\ldots,c\}$ are the corresponding classes which refer to their color terms. In this work, the number of color terms, c, is 11, including pink, purple, red, orange, yellow, green, blue, blown, white, gray, and black. By using the LSPC, the class-posterior probability $p(y|\boldsymbol{x})$ is learned as described in section 2. To choose the Gaussian width parameter, σ, we found through the experiments that the use of a single parameter σ is not suitable. Instead, we adopt the local scaling parameters to compute $\sigma_{\boldsymbol{x}}$ for a data point $\boldsymbol{x}$ so that the estimation of the class-posterior probability $p(y|\boldsymbol{x})$ becomes more reliable, where the number of nearest neighbors k in (2) can be chosen by cross validation. Keep in mind that the candidate set of k is dependent on the size of training data. In order to set this candidate set of k, the common way is that the chosen parameter by cross validation should not be in the left most or the right most; otherwise, we have to change this candidate set. Finally, the class label $y^{(te)}$ for a test pixel $\boldsymbol{x}^{(te)}$ is the class y that gives the highest $\widehat{p}(y|\boldsymbol{x}^{(te)})$:

$$y^{(te)} = \underset{y\in\{1,\ldots,c\}}{\operatorname{argmax}}\ \widehat{p}(y|\boldsymbol{x}^{(te)}).$$

at the end of this step, the color names with their confidence $\widehat{p}(y|\boldsymbol{x}^{(te)})$ for each pixel of the object are obtained as our example results shown in Figure 2. Figure 2(a) shows the input images and Figure 2(b) shows the results of labeling color names to the pixels of the object.

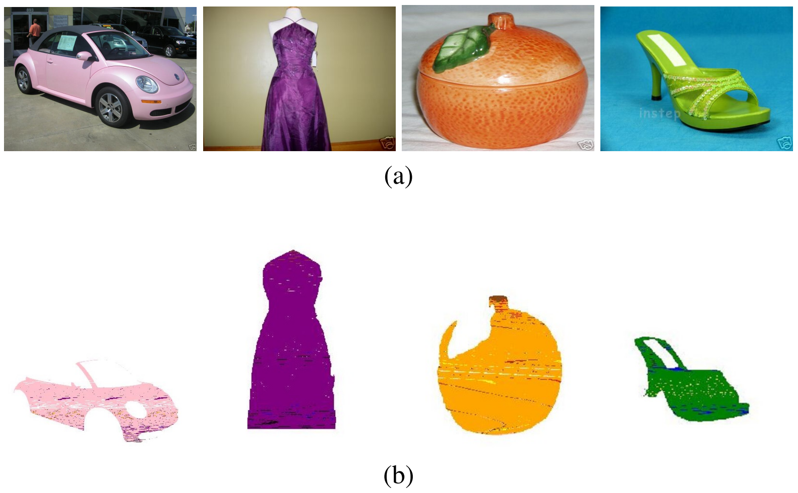


Figure 2. Example of (a) input images and (b) the results of labeling color names to the pixels of the object

3.2. Color naming to an object

Next, we assign a color name to an object in the image. To do so, we consider the class-posterior probabilities $\widehat{p}(y|\boldsymbol{x}^{(te)})$ corresponding to predicted class y by computing the score for each color class y as:

$$score_y = \frac{1}{n}\sum_{i=1}^{n_y}\widehat{p}(y|\boldsymbol{x}_i^{(te)}) \tag{3}$$

where n_y denotes the number of pixels that are predicted as class y, and n denotes the number of total pixels of the object so that $\sum_{y=1}^{c} n_y = n$. Then, the color name of the object is the class y that yields the highest $score_y$. For example, in Figure 2, the colors of a car, a dress, a glass, and a shoe are assigned to pink, purple, orange, and green, respectivley. Notice that the majority vote can be used to assign a color name to an object. However, we here used the summation of confident scores (3) to deal with the problem of ambiguous result as an example shown in Figure 3, where the ground truth color of a car is pink. The input image is shown in Figure 3(a), while the results of labeling color names to the pixels are shown in Figure 3(b). In this example, if we used the majority vote, the result of predicting color name is purple. However, if we used (3), the result is pink. Moreover, we found through our experiment that the confident scores (3) help improve overall performance.

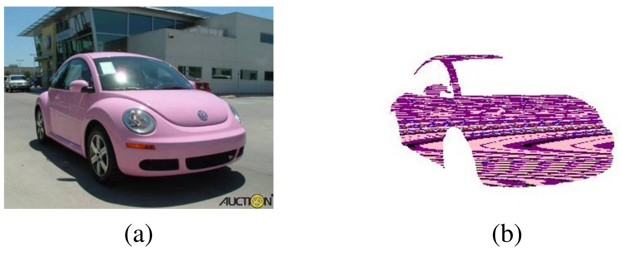


Figure 3. Example of (a) an input image (pink car) and (b) the ambiguous result (pink or purple?) of labeling color names to the pixels

3.3. Training data set

Training data is important and it can help to improve the accuracy of prediction. In this work, we are interested in the use of raw pixel intensities as training data, where our purpose is to have the pixel values representing the space of color terms to cover the entire color space as possible. However, it is not easy in practice because it lacks of ground truth available. In order to archive the training data set, many works [9], [14], [15] used the images obtained from the internet. Thus, inspired by these works, we extracted the pixel values of the images to represent samples of each color term, where the images are from the internet. In particular, we mainly extracted the pixel intensities and their corresponding labels from the 512-Google data set. This data set was provided by [9]; it consists of 512 images for each of the 11 basic color terms, and all images have their masks that indicate the sample pixels of each color term as examples shown in Figure 4. However, as shown in Figure 4, the masked pixels may not always be correct so that the obtained training data will have noise. Thus, we need to be careful to collect the image pixel values of the 11 basic color terms.

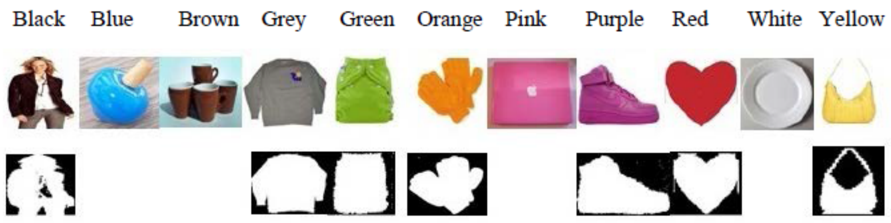


Figure 4. Example of images (middle row) with their masks (bottom row) and their labels (top row) from 512-Google data set

In this paper, we started collecting the training data points by uniformly sampling integer values from 0 to 255 for each channel of RGB for obtaining 200 pixel intensity values. We then asked 10 users to assign a color term of eleven basic color terms to these 200 data points without guiding any information to the users. Thus, they assigned a color name to each color pixel based on their experiences. After that we selected a color name for each pixel value by using the majority voting. These 200 pixel values and their obtained labels were used as training data for the color naming algorithm presented in [26], in order to assign color names to masked pixels of 512 images data set. Next, the obtained color labels by the color term model in [26] were compared with the corresponding ground truth color labels in order to keep only the pixels that the ground truth label and the label obtained from the algorithm in [26] are the same. We randomly selected at most 10 unique pixel intensities of each image. However, we rechecked the labels by human again and removed some pixels with ambiguous color. Moreover, in the same way, we also extracted the pixel intensities from the last two images of each category of each color term on the eBay data set. As reported in [14], adding the eBay images to the training data helps improve the overall performance. Finally, we remove outliers from each class. Our obtained training pixel values with their corresponding labels, where the number of overall pixels is 11,128, are illustrated in Figure 1, showing the distribution of data points in RGB, HSV, and Lab color spaces in Figure 1(a), Figure 1(b), and Figure 1(c), respectively. Notice that only the first 100 images from each color term on the 512-Google images data set were used to collect our training data points.

4. EXPERIMENTS

4.1. Set up

We used the eBay data set, which is available on the internet and was used in [9], [14], [15], [20], to evaluate the classification accuracy of our color term model. This data set consists of 4 categories: cars, glasses, shoes, and dresses with 12 images for each of the 11 basic color terms; we used the first 10 images for evaluation and the last 2 images added to the training data set. Therefore, there are 110 images for each category and overall 440 images for evaluation. Images in this data set are all collected from the eBay website and hand-segmented into foreground and background [14]. Thus, each image has its mask representing the region of the object as the same as the training data set, and we only consider the foreground object in our experiments. Moreover, the object in the image has only one color as examples shown in Figure 5.

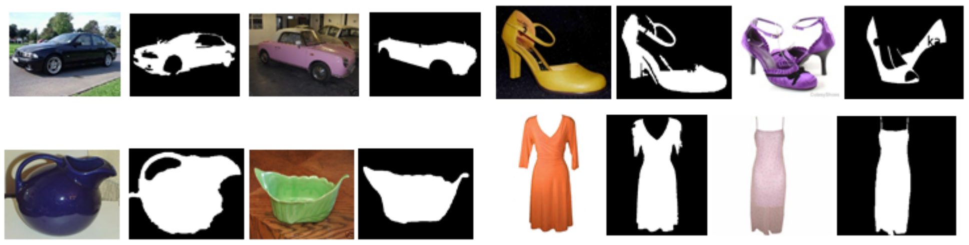


Figure 5. Example of images and their masks from eBay data set

We compared our proposed method with the methods proposed by [9], [15], [20], because they used the eBay data set for evaluation and their results are publicly available. They also used the first 10 images of each category and considered only the foureground object for the accuracy evaluation. In addition, we performed experiments on basic supervised machine learning algorithms with Lab color space, namely support vector machine (SVM) with rbf kernel, k-NN, and Naive bay. For our method, the parameter $k = 29$ and the regularization parameter $\lambda = 0.01$ were selected by cross-validation, where the candidate set of k is $\{21, 22, ..., 30\}$, the candidate set of the regularization parameter λ is logspace (-2, 0, 9), and the number of parameters b be equal to the number of training pixels n. For k-NN, we set k = 3, 5, and 7, where $k = 7$ yielded the best result. We then reported the results of k-NN with $k = 7$. To compare the quality of different methods, we used the accuracy as criteria, and it computes as:

$$accuracy = \frac{n_{correct}}{n_{test}},$$

where $n_{correct}$ denotes the number of correct labels to the objects, and n_{test} denotes the number of testing images for each category.

4.2. Results

First of all, we show you the accuracies of the proposed method with using different color spaces; include Lab, RGB, and HSV. The experimental results on the eBay data set are shown in Table 1, showing that Lab color space outperforms RBG and HSV color spaces. This indicates that Lab color space is the suitable choice for using the LSPC to assign a color name to an object in the given image. The reason is that the LSPC is based on density estimation, and it works well in which the size of training samples is large; in other words, the sample data should be dense such that the densities will be estimated more correctly. As demonstrated in Figure 1, the conversion of RGB space to Lab space will make the data points in the same class closer to each other. Consequently, the LSPC performs better. In contrast, the LSPC performs worst on the HSV color space because the data points in the same color term have more separation than that of Lab and RGB, especially, the colors “gray”, “black”, “white”, and “red”. This sums up that the Lab color space is the best choice for the LSPC in order to assign a color name to an image pixel. Next, we demonstrate the performance of using the local scaling parameters for the LSPC as the results show in Table 2. The input of this result is Lab color space, and a single parameter σ was obtained from cross validation. It can be found that the local scaling parameters help improve the performance of the LSPC.

Table 1. The averages of accuracies in percentages obtained from the proposed method comparing among the different color spaces

Color spaces	Cars	Dresses	Glasses	Shoes	Averages
Lab	**73.64**	**93.64**	**85.45**	**90.00**	**85.68**
RGB	67.27	90.90	80.91	90.00	82.27
HSV	70.91	71.82	64.55	76.36	70.91

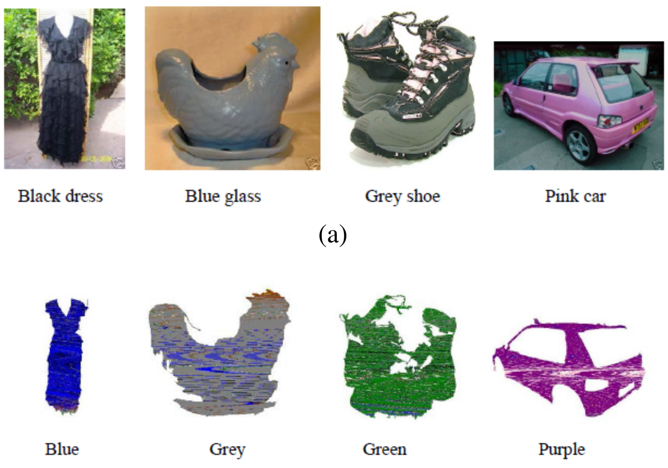
Table 2. The averages of accuracies in percentages obtained from the proposed method comparing between using a single parameter σ and the local scaling parameters σ in Lab color space

Parameter	Cars	Dresses	Glasses	Shoes	Averages
The local scaling parameters σ	**73.64**	**93.64**	**85.45**	**90.00**	**85.68**
A single parameter σ	67.27	84.55	82.73	86.36	80.23

Lastly, the averages of accuracies of each category and overall averages obtained from each method are shown in Table 3. The last two rows indicate the human accuracy from [9], [17]. The results show that our method outperforms others in categories of dresses and glasses, and be comparable to [9], [20] in category cars. Overall, we obtained the highest average of accuracies over four categories. The example of images that were incorrectly predicted are shown in Figure 6, where Figure 6(a) shows the input images with ground truth color name and Figure 6(b) shows the incorrectly results obtained from our method.

Table 3. The averages of accuracies in percentages on foreground objects of eBay data set

Methods	Cars	Dresses	Glasses	Shoes	Averages
Proposed	**73.64**	**93.64**	**85.45**	90.00	**85.68**
SLDA [20]	**73.63**	90.00	80.90	91.82	84.09
PLSA-bg [15]	71.82	86.36	83.64	**92.73**	83.64
X^2-rank [9]	**73.63**	88.18	79.01	**92.73**	83.41
SVM	73.64	90.00	80.91	72.73	79.32
7-NN	72.27	86.36	83.64	84.55	81.81
Naive bay	72.27	82.73	73.36	84.55	79.09
Human [9]	92.73	91.99	87.82	90.18	90.68
Human [17]	-	-	-	-	88.89



(b)

Figure 6. Example of incorrectly labeled color names to the masked pixels obtained from our method (a) input images with a ground truth color name and (b) predicted color names to the masked pixels

4.3. Discussion

Here, we discuss more about the factors that affect the performance of the LSPC to this work. First, the number of training samples (i.e., training pixels) should be cover the entire space of the selected color space (e.g., the size of training data is large) due to the accuracy may be higher; however, the computational time for learning color name model is also more complicated. Second, in this work, the number of parameters, b, was set to be equal to the size of training pixels, n; however, if the number of n is large, the computational time will be more complex. To address this problem, the number of b should be reduced. Notice that the smaller number of b yields the faster computational time and may affect the performance of the LSPC. Third, in order to speed up the computational time for testing step, we can reduce the number of testing pixels of an image by resizing the image (e.g., using imresize() in MATLAB). In this work, we downsized the image by a factor of 5, and we found that the performance was not lost; in contrast, the computational time is much less than the use of the original size of the image. The main limitation of the proposed method is the same as other probabilistic view methods; it depends on the estimation of the class posterior probability. For the LSPC, the estimation of the class posterior probability depends on the training data set, which is now not covering the entire color space. So, in the future, we plan to find out the method for collecting the training data set.

5. CONCLUSION

In this paper, we propose an alternative method to automatically assign a color name chosen from 11 basis color names to an object in the given image. Unlike the counterpart existing approaches that assigned the color names to image pixels by maximizing the likelihood function, we here directly estimate the posterior probability distribution $p(y|\boldsymbol{x})$ of color labels given a test pixel value. This leads to be more reliable as we solve the classification problem directly and never solve a general problem such as $p(\boldsymbol{x}|y)$ as an intermediate step. Our proposed method first applies the least-squares probabilistic classifiers (LSPC) to assign color names to image pixels. Because the distributions of training data points are different, we utilize the local scaling parameters to compute the Gaussian width included in the basis function for each data point. Next, the score used to decide a color name to the target object is presented. We did several experiments to analyze our proposed method. We found that the Lab color space is the best input vector for the LSPC to solve this task, as well as the local scaling parameters are more useful than using the single Gaussian width. Lastly, we show that the proposed method improves the overall performance over the counterpart methods.

REFERENCES

[1] J. Weijer and F. S. Khan, "An overview of color name applications in computer vision," in *Computational Color Imaging. CCIW 2015. Lecture Notes in Computer Science()*, Cham: Springer, 2015, pp. 16–22, doi: 10.1007/978-3-319-15979-9_2.

[2] Y. Liu, D. Zhang, and G. Lu, "Region-based image retrieval with high-level semantics using decision tree learning," *Pattern Recognition*, vol. 41, no. 8, pp. 2554–2570, 2008, doi: 10.1016/j.patcog.2007.12.003.

[3] G. Csurka, S. Skaff, L. Marchesotti, and C. Saunders, "Building look & feel concept models from color combinations," *The Visual Computer*, vol. 27, no. 12, pp. 1039–1053, 2011, doi: 10.1007/s00371-011-0657-9.

[4] O. Russakovsky and L. Fei-Fei, "Attribute learning in large-scale datasets," in *Trends and Topics in Computer Vision. ECCV 2010. Lecture Notes in Computer Science*, Berlin, Heidelberg: Springer, 2012, pp. 1–14, doi: 10.1007/978-3-642-35749-7_1.

[5] F. S. Khan, R. M. Anwer, J. Weijer, A. D. Bagdanov, A. M. Lopez, and M. Felsberg, "Coloring recognition in still images," *International Journal of Computer Vision*, vol. 105, no. 3, pp. 205–221, 2013, doi: 10.1007/s11263-013-0633-0.

[6] S. Hu, Y. Ge, J. Han, and X. Zhang, "Object tracking algorithm based on dual color feature fusion with dimension reduction," *Sensors*, vol. 19, no. 1, p. 73, 2018, doi: 10.3390/s19010073.

[7] C. Yue, W. Yu, Z. Hou, and S. Ma, "Visual tracking algorithm based on color name histogram," *IOP Conference Series: Materials Science and Engineering*, vol. 790, no. 1, p. 012063, 2020, doi: 10.1088/1757-899X/790/1/012063.

[8] W. Pei and X. Lu, "Moving object tracking in satellite videos by kernelized correlation filter based on color-name features and kalman prediction," *Wireless Communications and Mobile Computing*, vol. 2022, pp. 1–16, 2022, doi: 10.1155/2022/9735887.

[9] B. Schauerte and G. A. Fink, "Web-based learning of naturalized color models for human-machine interaction," in *2010 International Conference on Digital Image Computing: Techniques and Applications*, 2010, pp. 498–503, doi: 10.1109/DICTA.2010.90.

[10] B. Schauerte, M. Martinez, A. Constantinescu, and R. Stiefelhagen, "An assistive vision system for the blind that helps find lost things," in *Computers Helping People with Special Needs. ICCHP 2012. Lecture Notes in Computer Science*, Berlin: Springer, 2012, pp. 566–572, doi: 10.1007/978-3-642-31534-3_83.

[11] J. Heer and M. Stone, "Color naming models for color selection, image editing and palette design," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2012, pp. 1007–1016, doi: 10.1145/2207676.2208547.

[12] J. Weijer and C. Schmid, "Applying color names to image description," in *2007 IEEE International Conference on Image Processing*, 2007, pp. 493-496, doi: 10.1109/ICIP.2007.4379354.

[13] T. Hofmann, "Probabilistic latent semantic indexing," in *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, 1999, pp. 50–57, doi: 10.1145/312624.312649.

[14] J. Weijer, C. Schmid, and J. Verbeek, "Learning color names from real-world images," in *2007 IEEE Conference on Computer Vision and Pattern Recognition*, 2007, pp. 1–8, doi: 10.1109/CVPR.2007.383218.

[15] J. Weijer, C. Schmid, J. Verbeek, and D. Larlus, "Learning color names for real-world applications," *IEEE Transactions on Image Processing*, vol. 18, no. 7, pp. 1512–1523, 2009, doi: 10.1109/TIP.2009.2019809.

[16] Y. Wang, J. Liu, J. Wang, Y. Li, and H. Lu, "Color names learning using convolutional neural networks," in *2015 IEEE International Conference on Image Processing (ICIP)*, 2015, pp. 217–221, doi: 10.1109/ICIP.2015.7350791.

[17] L. Yu, Y. Cheng, and J. Weijer, "Weakly supervised domain-specific color naming based on attention," 2018, [Online]. Available: http://arxiv.org/abs/1805.04385.

[18] K. R. Jyothi and M. Okade, "Computational color naming for human-machine interaction," in *2019 IEEE Region 10 Symposium (TENSYMP)*, 2019, pp. 391–396, doi: 10.1109/TENSYMP46218.2019.8971326.

[19] J. Sainui and M. Tongsamrit, "Color naming for describing object in image using different classification algorithms and color spaces," in *Proceedings of the 2020 12th International Conference on Computer and Automation Engineering*, 2020, pp. 65–69, doi: 10.1145/3384613.3384622.

[20] B. Schauerte and R. Stiefelhagen, "Learning robust color name models from web images," in *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*, 2012, pp. 3598-3601.

[21] D. M. Blei and J. D. McAuliffe, "Supervised topic models," in *Advances in Neural Information Processing Systems 20 - Proceedings of the 2007 Conference*, 2008, pp. 121–128.

[22] M. Sugiyama, "Superfast-trainable multi-class probabilistic classifier by least-squares posterior fitting," *IEICE Transactions on Information and Systems*, vol. E93-D, no. 10, pp. 2690–2701, 2010, doi: 10.1587/transinf.E93.D.2690.

[23] L. Zelnik-manor and P. Perona, "Self-tuning spectral clustering," in *Advances in Neural Information Processing Systems*, 2004, pp. 1601–1608.

[24] M. Yamada, M. Sugiyama, G. Wichern, And J. Simm, “Improving the accuracy of least-squares probabilistic classifiers,” *IEICE Transactions on Information and Systems*, vol. E94-D, no. 6, pp. 1337–1340, 2011, doi: 10.1587/transinf.E94.D.1337.
[25] “Software.” http://www.ms.k.u-tokyo.ac.jp/sugi/software.html (accessed Mar. 31, 2023).
[26] J. Sainui and P. Pattanasatean, “Color classification based on pixel intensity values,” in *2018 19th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD)*, 2018, pp. 302–306, doi: 10.1109/SNPD.2018.8441141.

BIOGRAPHIES OF AUTHORS

Janya Sainui received her BS and MS in computer science from Prince of Songkla University, Hat Yai, Songkhla, Thailand, in 2005 and 2009, respectively. She is currently a Lecturer in computer science at Prince of Songkla University, Thailand. She has conducted research on topics such as bitmap indexing, video indexing and detection, clustering and unsupervised dimension reduction. She is interested in algorithm and application of machine learning, especially, applying machine learning techniques to real world applications such as image/video processing, medical imaging, bioinformatics, as well as information retrieval. She can be contacted at email: janya.s@psu.ac.th.

Chouvanee Srivisal is currently a lecturer in computer science at Faculty of Science Prince of Songkla University, Thailand. She received her BS in computer science from Prince of Songkla University and MS in information technology from King Mongkut’s Institute of Technology Ladkrabang, Thailand in 1997 and 2002, respectively. She has interested in algorithm and application of machine learning that for conducting research on topics such as image processing and information retrieval. She can be contacted at email: chouvanee.s@psu.ac.th.