International Journal of New Technology and Research (IJNTR)
ISSN:2454-4116, Volume-4, Issue-8, August 2018 Pages 76-83

Video Classification Using Low-Level Components and Computable Features Assessment

Enwerem Udochukwu, Charles Robert

***Abstract*— Video classifications are usually tailored towards categorizing videos into one or more predefined categories (e.g. genres) using the contexts associated with such categories. This limits their application to only "production videos" (i.e. video produced and edited for a viewing audience). We seek to make the classification criteria more flexible by classifying videos using low-level computable features that can be determined for any type of video independent of the context associated with its predetermined genre. The methodology adopted was based on choosing unrestricted computable features for developing a classification scheme. It extracted and analyzed the low-level components (key frames) and computable features (such as dominant color, lighting condition, and color dynamics) from sample videos. It then generated a model SVM classifier that was able to discriminate between tested videos to be classified. It finally, developed an interactive application to automate the extraction and analysis process.**

***Index Terms*— Computable features, Features extraction, Support Vector Machines, Video analysis, Video segmentation.**

I. INTRODUCTION

Video, a form of multimedia is one of the most elaborate forms of presenting data because it provides a multi-modal sensory experience which is closest to the way we encounter the world we live in [1].The amount of videos available today (through the broadcast media and internet) is growing. There is the need to manage them due to their relatively larger storage requirements when compared to other media types. Online video repositories play a significant role in social networking, e-learning, news media, documentation and entertainment. Billions of dollars are generated from the video industry with particular emphasis on online video repositories. Examples include: YouTube, Netflix, Yahoo Screen, Hulu e.t.c. Extensive works were done in areas such as video search and retrieval. Instead of searching for a video from a large heterogeneous video collection why not classify each video entry in the collection to optimize search? The answer to this question was behind this project. The objective was to classify videos from a given collection of videos (e.g. an online video database) into selected categories using computable features such as dominant color and lighting condition. Fig. 1 depicts the traditional way of classifying videos where classification is achieved by using the context set by the video producer or an observer to discriminate between video classes. On the other hand, we seek to replace the abstract context with more concrete data derived from the low level features of the videos. We then use this to facilitate the video classification process as shown in Fig. 2.

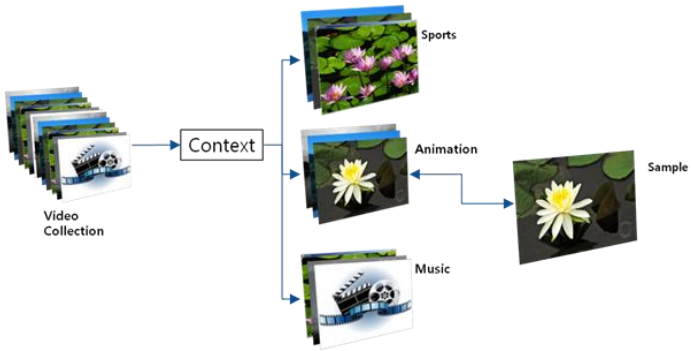


Figure 1: Classification using context

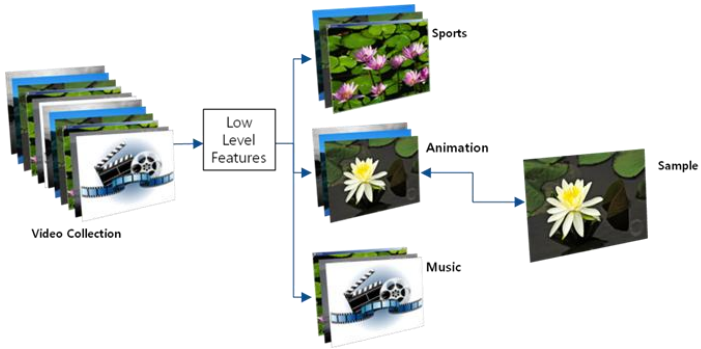


Figure 2: Classification using low level features

II. THEORETICAL BACKGROUND

One of the simplest methods for detecting shots is to take the difference of the color histograms of consecutive frames, with the assumption that the difference in color histograms of frames within the same shot will be smaller than the difference between frames of different shots [2][3][4]. Darin et al. [6] however, noted that while this approach is easy to implement, it has a number of potential problems. One is deciding what threshold the differences must exceed in order to declare a change in shots. Shots that contain a lot of motion require a higher threshold value than those with little motion. Also, the threshold value is likely to be different for different videos and even within the same video no particular value may correctly identify all shot changes [7]. A threshold value that is too low will identify shot changes that don't exist while a threshold value that is too high will miss some shot changes.

Shot changes can also be detected using the Kullback-Leibler distance between histograms of consecutive frames that have been transformed to the RGB color space [8]. The RGB values are calculated using (1).

Enwerem Chinonyerem Udochukwu, Department of Computer Science, Federal University of Technology, Owerri/ School of Physical Sciences/ FUTO, Imo State, Nigeria
Charles Robert, Department of Computer Science, University of Ibadan/ Faculty of Science/ UI, Oyo State, Nigeria

$$r = \frac{R}{R+G+B}, \quad g = \frac{G}{R+G+B}, \quad b = \frac{B}{R+G+B} \qquad (1)$$

The Kullback-Leibler distance is calculated using (2).

$$K(p||q) = -\sum_{i=1}^{N} p(x_i) log \frac{q(x_i)}{p(x_i)} \qquad (2)$$

Where N is the number of bins in the histograms, $p(x_i)$ is the probability of color x_i for one frame and $q(x_i)$ is the probability of color x_i for the other frame [6].

Object-based features seem to be uncommon, perhaps because of the difficulty in detecting and identifying objects as well as the computational requirements to do so. When they are used, they tend to focus on identifying specific types of objects, such as faces [9][10]. Once objects are detected, features derived from them include dominant color, texture, size, and trajectory [6].

There are two major types of motion that can be perceived in a video. They are (a) Motion on the part of the objects being focused and (b) Motion due to camera actions. In other cases, there might also be other types of movement, such as text scrolling at the bottom of a news program. Motion-based features are usually detected using either MPEG motion vectors or by calculating the optical flow generated by objects within the video. Fischer et al. [11] detect total motion in a shot by comparing the histograms of blocks of consecutive frames. In order to detect object motion, they first calculate optical flow as described by Horn et al. [12]. Motion due to camera movement (e.g., panning) would result in all blocks having motion. Using this, camera motion can be subtracted, leaving only the motion of objects. These objects are identified by segmenting pixels with parallel motion. Roach et al. [13] detected the motion of foreground objects using a frame-differencing approach. Pixel-wise frame differencing of consecutive frames is performed using the Euclidean distance between pixels in the RGB color space. These values are kept within a threshold to better represent the motion and doing so for the sequence of pixels produces a 1D signal in the time dimension. To reduce this signal's sensitivity to camera motions, it is differentiated to produce a final motion signal [6].

III. Methodology

There were three major phases in the methodology developed for this work. These were (a) Key-Frame Extraction (b) Feature Extraction and (c) Video Classification. These stages are shown diagrammatically in Fig. 3.

A. Key-frame Extraction

The aim of the Key-frame extraction process is to select frames that are to a large extent, representative of the entire video. According to Ferman et al. [14], key frame-based methods to represent the color features of a group of frames are highly dependent on the selection criterion of the representative frame and may lead to unreliable results, if not done properly.

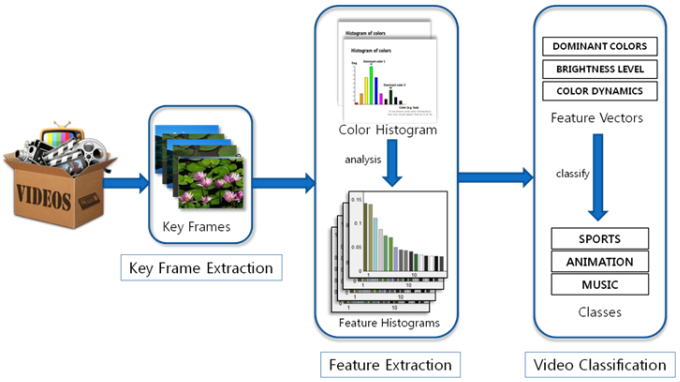


Figure 3: Overview of our Methodology

The key-frame extraction process begins with the breaking down of the sample video into its constituent frames. The total number of frames is usually determined by the video frame rate and the time-length of the video. The frames are then grouped into shots or windows. We use the appropriate term "windows" because by technical definition the grouping is not actually a shot. Usually a shot is defined a piece of video frame sequences taken from a single camera containing no camera changes or scene transitions [15]. In our case, the grouping only serves as a "window" through which key-frame selection can be made.

After the grouping, key-frame selection can be made directly from all the frames within a window. However, due to the computational expense of doing this, we further introduce gaps at regular intervals into each window in order to reduce the number of frames from which we have to make key-frame selections from. The gaps are regular so that the remaining frames within each window are evenly distributed.

Finally, we determine the color histogram of the remaining frames within each window and compute the absolute histogram difference of each frame to every other frame within the same window. We choose as the key-frame the frame with histogram H_k that minimizes the error function:

$$E(H_k) = \frac{1}{M}\sum_{i} \left|\left|H_i - H_k\right|\right|, \quad i = 1, 2, 3, \ldots, M \qquad (3)$$

where M denotes the number of frames in the window and H_i are the histograms of every frame in the window apart from H_k.

B. Feature Extraction

In this phase, we extracted several color information from each key-frame. The aim of the feature extraction process was to form feature vectors for each sample video containing values from the different color information. Four major representative feature sets were used for our video classification. These were (a) The Dominant Color Vector, (b) The Reference Point Vector (c) The Brightness Level Vector, and (d) The Color Dynamics Vector.

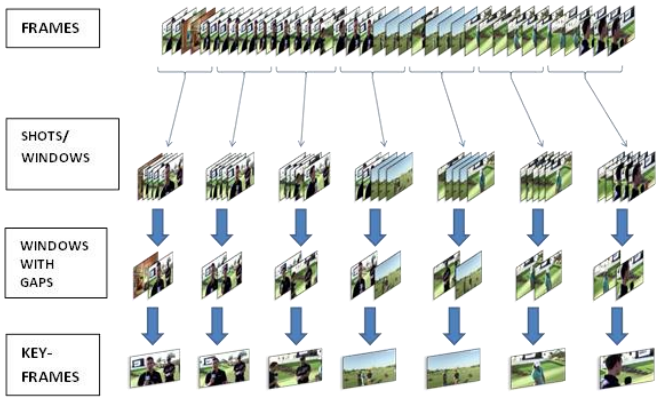


Figure 4: The Key-Frame Extraction Stages

C. The Dominant Color Vector

The purpose of the Dominant Color Vector DCV is to show how much of a color appears in each video frame and how frequently they do so throughout the entire video frame sequence.

First, a single color map m is created for converting all RGB key-frame images to index images. In the process, all pixel colors in the frame are remapped to the closest color in the new color map. In our case, this single color map is created using the MATLAB colorcube(n) function which select colors uniformly all throughout the RGB color space and always creates the same color map for a given number of colors, n. This offers a twofold advantage:

1) It provides a way of reducing the number of colors in each video frame image to a constant number of colors, hence eliminating the need for color palettes.
2) It reduces the computational complexity in processing each video frames, since every frame has the same color map.

Secondly, we compute the color histogram $H_k(i)$ for the key-frame at the window position k using (4).

$$H_k(i) = \sum_{x,y} \begin{cases} 1, if\ P(x,y) = i, \\ 0, if\ P(x,y) \neq i \end{cases} \qquad (4)$$

where P(x,y) is an index value in the color map m for the color of the pixel at position (x,y) on the key-frame; i is a bin number in the new color map m.

Next, we compute the family histogram for the entire video by summing up the bin counts at each bin position i for all histograms of the key-frames and computing the mean across the entire video. This is represented as:

$$H(i) = \frac{1}{M}\sum_{k=1}^{M} H_k(i), \qquad for\ i = 1 \sim n, \qquad (5)$$

where M is the number of windows/key-frames in the sample video and n is the number of bins in $H_k(i)$.

Both $H_k(i)$ and H(i) have the same number of bins n, since this is the number of bins in the color map produced by the MATLAB function colorcube(n). Therefore, from H(i) we have n potential dominant colors and consequently a feature vector composed of n values.

Increasing the value of n has the advantage of increasing the accuracy of our color mapping function, as there are more colors to choose from. However, if we have a video consisting of very few colors (say 10) and n is large (say 256), then there will be large number of redundant values (i.e. values that are insignificantly small or zero) in our feature vector, thereby leading to a high degree of arbitrariness in our classification scheme. The Dominant Color Vector is designed to solve this problem.

The Dominant Color Vector DCV is developed in the final stages of the process. It is composed of w components. Its components [D_1, D_2, …, D_w] represent the w most dominant colors in H(i) in order of their bin counts. w is chosen in such a way that all the dominant colors selected from H(i) do not have bin counts of 0 value.

Before D_1, D_2, …, D_w are deduced we first compute B_1, B_2, …, B_w which are the w respective bin numbers of the corresponding top w most dominant colors in the histogram H(i). They are determined by the following equations:

$$B_1 = \arg_i \left\{ \max_{i=1\sim n} [H(i)] \right\},$$

$$B_2 = \begin{cases} c, if\ H(c) > \left[\frac{1}{T} H(B_1) \times t_1\right] \\ 0, otherwise \end{cases} \qquad t_1 = 0 \sim 1,$$

$$B_w = \begin{cases} d, if\ H(d) > \left[\frac{1}{T} H(B_1) \times t_{w-1}\right] \\ 0, otherwise \end{cases} \quad t_w > t_{w-1} \qquad (6)$$

Then D_1, D_2, …, D_w can be deduced from the normalized bin counts of B_1, B_2, …, B_w by the following equations:

$$D_1 = \frac{1}{T} H(B_1),$$

$$D_2 = \frac{1}{T} \begin{cases} H(B_2), if\ H(B_2) > [D_1 \times t_1] \\ 0, otherwise \end{cases}$$

$$D_w = \frac{1}{T} \begin{cases} H(B_w), if\ H(B_w) > [D_1 \times t_{w-1}] \\ 0, otherwise \end{cases} \qquad (7)$$

where T is the total number of pixels that were analyzed in the video. It is computed by summing all the bin values in H(i) as in the equation:

$$T = \sum_{i=1}^{N} H(i), \qquad (8)$$

Also, t_1, t_2, …, t_{w-1} are threshold values between 0 and 1 that determine the allowable difference between the value of D_1 and those of D_2, D_3, …, D_w.

D. Reference Point Vector

The Reference Point Vector RPV is a color representation that enables a multi-dimensional color value to be converted to an equivalent single value.

The color values of each pixel within a key-frame image

are usually stored in RGB format consisting of three values for each of the red, green and blue color channels for a single color. This poses a major challenge in our classification scheme because when we choose to store color values as components in our feature vector it increases the dimensionality of the vector and the complexity of our classification scheme. In order to maintain the linear dimensionality of our feature vector while including color values a novel color representation known as the Reference Point Vector RPV was introduced.

RPV is a feature vector with components that are equivalent to the color values of the dominant colors at B_1, B_2, …, B_w respectively. Its components [C_1, C_2, …, C_w] represent the Euclidean distance of each of the w most dominant colors in H(i) from a fixed point on the CIELUV color space. The CIELUV color space has the advantage of Perceptual Uniformity, i.e. the same distance between two different points makes equal perceived color difference [16]

In order to compute the values of C_1, C_2, …, C_w the color values at bin locations B_1, B_2, …, B_w are first converted to their equivalent CIELUV values b_1, b_2, …, b_w. A fixed point P on the CIELUV color space is chosen and the corresponding Euclidean distances C_1, C_2, …, C_w from P are computed thus:

$$C_1 = \sqrt{(b_1(l) - P(l))^2 + (b_1(u) - P(u))^2 + (b_1(v) - P(v))^2}$$

$$C_2 = \sqrt{(b_2(l) - P(l))^2 + (b_2(u) - P(u))^2 + (b_2(v) - P(v))^2}$$

$$C_w = \sqrt{(b_w(l) - P(l))^2 + (b_w(u) - P(u))^2 + (b_w(v) - P(v))^2} \quad (9)$$

where l is the luminance value and (u,v) are the two dimensional chromaticity values of each color in the CIELUV color space.

E. Brightness Level Vector

The next color representation for our feature vector is the Brightness Level Vector BLV. It serves as a means for representing the general lighting condition of the video. As with the previously discussed vectors, we restrict the components in BLV to w values [I_1, I_2, …, I_w]. Each component in BLV is the Brightness value of each of the w most dominant colors in H(i) with respect to the HSV color space. The HSV color space represents a color value in terms of its hue, saturation and value. The "value" component of a color in the HSV format is closely equivalent to the brightness of the color.

To form the BVL, each dominant color value in B_1, B_2, …, B_w is first converted to its equivalent value in the HSV color space. We represent each dominant color with the "value" component of its HSV format and ignore the hue and saturation components. If s_1, s_2, …, s_w are the respective HSV formats for B_1, B_2, …, B_w, then the BVL components are:

$$I_1 = s_1(v),$$
$$I_2 = s_2(v), \quad and$$
$$I_w = s_w(v) \quad (10)$$

where v is the "value" component of each color in the HSV color space.

F. Color Dynamics Vector

The three previously discussed vector representations have been derived from the dominant color histogram and were all meant to extract spatial color information about the video. The final vector representation known as the Color Dynamic Vector CDV on the other hand has the sole purpose of retrieving temporal color information about the video. The method used here for feature extraction is a slight variant of that proposed by Chan et al. [17].

First, we compute the color histogram $H_k(i)$ for each window k in the video as shown in (2). There are several ways of tracking the color changes between video frames.

Chan et al. [17] proposed a method whereby the color difference between two colors is computed in order to track changes in color between frames. However, Darin et al. [6] noted that it is impossible to determine from a color histogram the positions of pixels with specific colors, consequently making it difficult to track pixel-wise color changes and extract any spatial information from the color histogram. They suggested a method where the frame is first sub-divided into regions and the color histogram of each region is extracted in order to capture some spatial information. But, this method will prove to be computationally expensive for our application because there are a large number of frames to be processed.

For this work, we overcome the challenge of tracking color changes between frames by using a single color map m for all key-frames. As mentioned earlier, this is achieved by remapping all pixel colors in the frame to the closest color in m. Hence, we are able to know how the amount of each bin color in H_k (i) varies in between frames with respect to time.

To create CDV we estimate the absolute difference in the count of each color bin between two consecutive windows $H_r(i)$ and$H_{r+1}(i)$ and then compute the mean difference V(i) across all windows using the equation:

$$V(i) = \frac{1}{T(M-1)} \sum_{r=1}^{M-1} |H_r(i) - H_{r+1}(i)| \quad for\ i = 1 \sim n \quad (11)$$

where M is the number of windows/key-frames.

We introduce T as in (6) in order to normalize V(i) between 0 and 1.

Consider the w dominant colors representations D_1, D_2, …, D_w in the family histogram H(i). If they represent colors at bin locations x_1, x_2, …, x_w respectively, then it can be deduced that any variation in these colors will have more significant impact on the perceptual characteristics of the video than the less dominant colors. Hence, we compose the Color Dynamic Vector CDV with values of V(i) as follows:

$$CDV = [V(x_1), V(x_2), \ldots, V(x_w)] \quad (12)$$

IV. VIDEO CLASSIFICATION

We chose the Support Vector Machine (SVM) classifier as our model classifier. The first step in the classification task is the separation of our data set into training and testing sets.

Each instance in the training set contains one "target value" (i.e. the class labels) and several "attributes" (i.e. the features or observed variables). The goal of SVM is to produce a model (based on the training data) which predicts the target values of the test data given only the data attributes [18]. We discuss the steps in our classification method in following section.

A. The Input Feature Vector

The input feature vector to our classifier is formed simply by concatenating the individual feature vectors created during the feature extraction phase. If w is the size of the individual feature vector, then the resulting vector comprises of a minimum of w components and maximum of 4×w components. The vector size varied because during testing, we tested individual vectors separately and collective in order to ascertain which feature vector combination gives better result. We represent the final input vector with feature data F as:

$$I = [F_1, F_2, \dots, F_n] \ \ where\, n = w, 2w, \dots, 4w \qquad (13)$$

B. The Training Set

Our training set consists of multiple input feature vectors I(i) with each corresponding to the "attributes" or features of a given video training sample i. Each training set entry i is assigned a class label indicating the type of "attributes" the video possesses. These labels may be set by an observer who examines the video content and determines that a given label best fits the sample video. Also, labels may be gotten from the "natural" class of the sample video (e.g. the genre of the video which was set by the producer). On the other hand, the attributes are gotten from the low level feature extraction process.

If the number of videos to be included in our training set is k, then the set is derived from (13) as follows:

$$I(i) = [F_1(i), F_2(i), \dots, F_n(i)] \ \ where\, i = 1, 2, \dots, k \qquad (14)$$

C. The Classifier

The classifier is formed from a training set of instance-label pairs (x_i, y_i), i = 1,...,l where $x_i \in R^n$ and $y \in \{1,-1\}^l$. It requires the solution of the following optimization problem:

$$\min_{w,b,\xi} \frac{1}{2} \boldsymbol{w}^T \boldsymbol{w} + C \sum_{i=1}^{l} \xi_i \qquad (15)$$

Such that:

$$y_i(\boldsymbol{w}^T \phi(\boldsymbol{x}_i) + b) \geq 1 - \xi_i,$$

and

$$\xi_i \geq 0.$$

Here, the training vectors x_i were mapped into a higher dimensional space by the function ϕ. The SVM classifier finds a linear separating hyper plane with the maximal margin in this higher dimensional space. C > 0 is the penalty parameter of the error term. Furthermore, $K(x_i, x_j) \equiv \phi(x_i)^T \phi(x_j)$ is called the kernel function [18].

The classification process involves the following steps:

1) Transform data to the format of an SVM environment: This involved creating the training and testing sets as described in (13) and (14).
2) Conduct simple scaling on the data: This is done in order to avoid attributes in greater numeric ranges dominating those in smaller numeric ranges. Scaling each attribute to the range [-1, +1] or [0, 1] is recommended. Since, most of our feature vectors were normalized the scaling problem is avoided.
3) Consider the Radial Basis Function (RBF) kernel function first before other kernel functions: This is because the RBF kernel ($K(x, y) = e^{-\gamma||x-y||^2}$) nonlinearly maps samples into a higher dimensional space. Therefore, it can handle the case where the relation between class labels and attributes is nonlinear.
4) Use cross-validation to find the best parameter C and γ: Here we use a technique known as v-fold cross-validation. We first divide the training set into v subsets of equal size. Sequentially one subset is tested using the classifier trained on the remaining v-1 subsets. Thus, each instance of the whole training set is predicted once. Hence, the cross-validation accuracy is the percentage of data which are correctly classified.
5) Use the best parameter C and γ to train the whole training set: These parameters may be affected by the size of the data set.
6) Carry out testing [18].

V. RESULTS

Results were collated from both the feature extraction and classification processes. They are briefly described and discussed in this section.

A. Feature Extraction Results

The graphical representation of the four feature vectors for four sample videos that were obtained from the feature extraction process are shown below. Fig. 5 visualizes the data from the feature vector of a video with dominant green (a) and that from a video with dominant blue (b). Fig. 6 visualizes the data from the feature vector of a video with dominant brown (a) and that from a grey level color video (b).

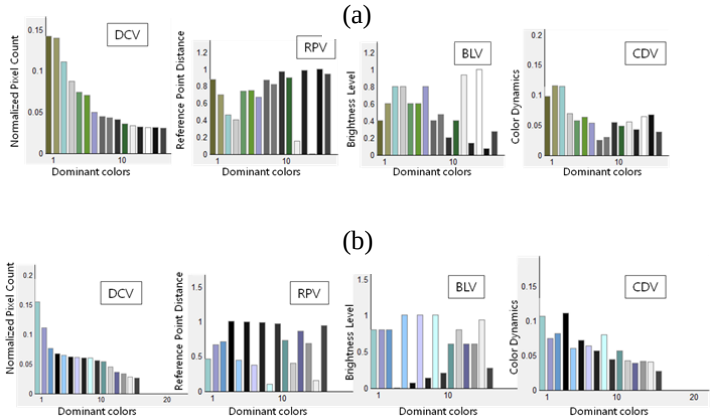
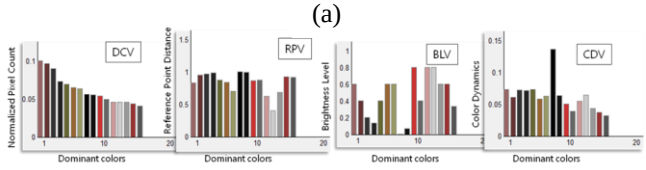


Figure 5: Feature vectors of videos with (a) Dominant green and medium lighting and (b) Dominant blue and high lighting

(a)



(b)

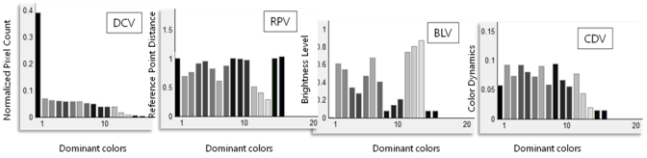


Figure 6: Feature vectors of videos with (a) Dominant brown and medium lighting and (b) Dominant grey and medium lighting

B. Classification Results

In deriving the results of our classification using SVM a total of 201 sample videos are analyzed. Our SVM classifier is meant to discriminate between three classes i.e. sports, animation and music videos. The training set consists of 171 videos with 57 videos for each class while the testing set comprises of 30 videos with 10 videos for each class.

The class labels for each training set video is known to the SVM classifier but those of the test videos are unknown. How well the SVM classifier is able to guess their class label determines the accuracy of the classifier.

In training the classifier, cross validation was performed and it was found that the Radial Basis Function (RBF) with a scaling factor (sigma) of 4 produced the best results for our classification. Also features where used individually and in different combinations to deduce their impact on the classification process.

Some of the most relevant results from the SVM classifier are shown below. Two sets of results were obtained from the classification process. The first set shown in table 1 is obtained from feature vectors where the grey level color data were included. The second set shown in table 2 is obtained from feature vectors where the grey level color data have been removed. A comparison of the two results is shown in table 3. Fig. 7 shows the bar chart of the classification results with the inclusion of grey level color data while Fig. 8 shows that of results without the grey level color data. Fig. 9 depicts the comparison between the two sets of results using a line graph.

Table 1: Classification results (with grey level color data included)

	sports (%)	animation (%)	music (%)
combined	90	80	80
dcv	20	80	70
rpv	20	80	80
blv	40	80	90
cdv	70	20	70
dcv+blv	90	80	80
rpv+blv	50	90	80
dcv+rpv+blv	80	80	80
dcv+blv+cdv	90	80	80

Table 2: Classification results (with grey level color data removed)

	sports (%)	animation (%)	music (%)
combined	50	80	50
dcv	10	70	20
rpv	30	80	80
blv	40	90	80
cdv	20	50	30
dcv+blv	40	80	80
rpv+blv	40	80	80
dcv+rpv+blv	40	70	70
dcv+rpv+cdv	50	80	70

Table 3: Result Comparison

	color + grey level data (%)	color data only(%)
combined	83.33	60
dcv	56.67	33.33
rpv	60	63.33
blv	70	70
cdv	53.33	33.33
dcv+blv	83.33	66.67
rpv+blv	73.33	66.67
dcv+rpv+blv	80	60
dcv+rpv+cdv	83.33	66.67

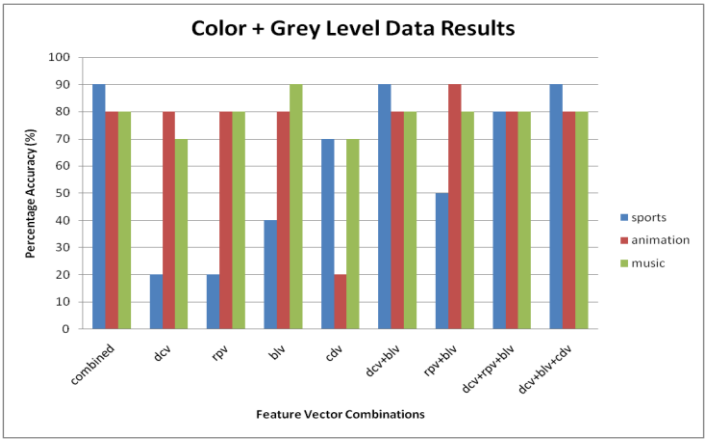


Figure 7: Bar chart for Classification Results (with grey level color data included)

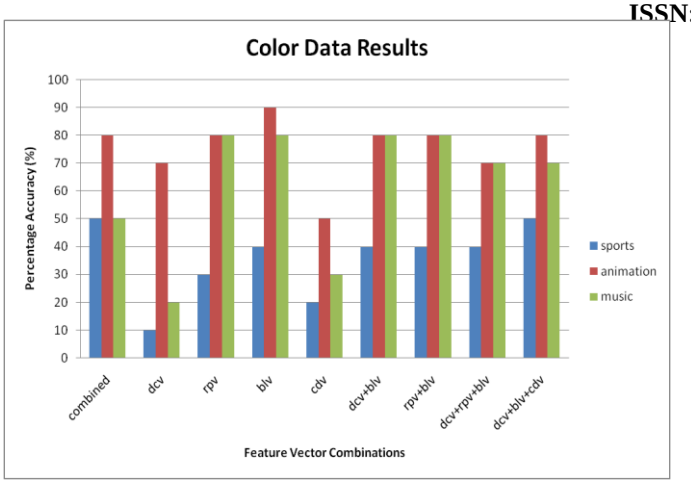


Figure 8: Bar chart for Classification Results (with grey level color data removed)

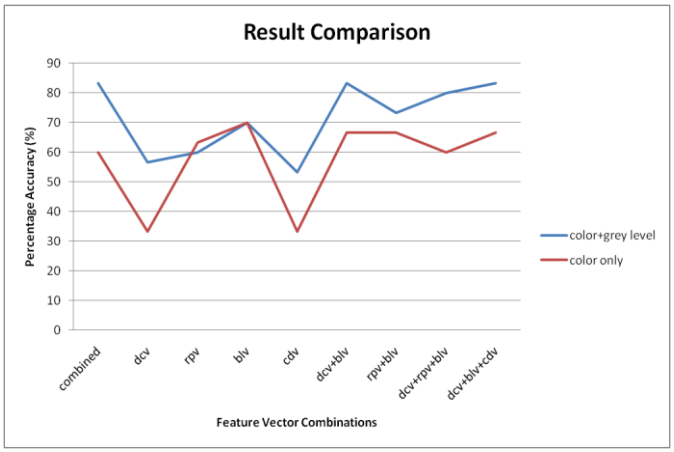


Figure 9: Line graph comparison of Classification Results

VI. RESULT DISCUSSION

We discuss the classification results based on the class labels that were chosen. The various combinations of feature vectors used for classification were observed to have varied effect on the SVM classifier's ability to identify test videos from the different classes accurately.

A. Sports

Sports videos had the highest classification accuracy when all the feature vectors were combined and with the grey level color data included. However, the classifier performed poorly in identifying sports videos when individual vectors were used. Only the CDV showed reasonable accuracy when used as a standalone vector. Sport videos produced the worst classification accuracy on the average when grey level color data was removed.

B. Animation

The classifier performed fairly well with all feature vector combinations both individually and collectively for animation videos with the exception of the CDV. Particularly, it was observed that a combination of the RPV and BLV produced the most accurate results for animation videos when used, both individually and collectively. When grey level color data was removed, the classifier performed best on the average in classifying animation videos. In this respect, the BLV produced the most accurate results.

C. Music

For music videos, the BLV was observed to produce the most accurate results with the SVM classifier with the inclusion of the grey level color data. However, on the average the SVM was able to classify music videos to a high level of accuracy with all combinations of feature vectors. On the other hand, when the grey level color data was removed, the SVM classifier had less accuracy. In particular, the DCV and BLV individually produced the least accuracy in classifying music videos.

In general, the SVM classifier performed better when grey level color data was included in the feature vectors than when it was removed. This can be deduced from the comparison in Fig. 9. Also, the combined vectors produced more accurate classification results on the average than the individual vectors.

VII. CONCLUSION

There are a wide range of visual features that enable the human visual system to able to perceive, recognize and distinguish between objects in the world. However, among these features, color information stands out as one of the characteristics that are easiest to perceive. Furthermore, most digital electronic systems such as television sets, computers, and camcorders provide abundant color data for encoding visual information. From our findings, it can be concluded that color information is an important determinant for understanding both temporal and spatial characteristics of a video. Hence, it is a useful tool for classification. Particularly, the Dominant color features have been shown in this work to be useful for video classification using machine learning. Among the major contributions of this work is the ability to perform classification on both restricted and unrestricted sample domain. This makes the project useful for developing new independent classification systems for videos based on color.

REFERENCES

[1] Hari S. (2002). Segmentation, Structure Detection and Summarization of Multimedia Sequences, Graduate School of Art and Science, Columbia University.
[2] Zhang H., Kankanhalli A., Smoliar S. W. (1993). Automatic partitioning of full-motion video, Multimedia Systems, Volume 1, pp. 10-28.
[3] Andrew B. W. (1994). Image Compression Using the Discrete Cosine Transform, Mathematica Journal, Volume 4, No. 1, pp. 81-88.
[4] Kobla V., Doermann D. S., Lin K, -I, Faloutsos C. (1997). Compressed-domain video indexing techniques using DCT and motion vector information in MPEG video, Storage and Retrieval for Image and Video Databases (SPIE), pp. 200-211.
[5] Oge M. (2011). Practical Image and Video Processing Using MATLAB, John Wiley & Sons, Inc.
[6] Darin B., Diane J. C. (2008). Automatic Video Classification: A Survey of the Literature, IEEE Transactions on Systems, Man, and Cybernetics, Volume 38, Issue 3, pp. 416 – 430.
[7] Jadon R., Chaudhury S., Biswas K. (2001). A fuzzy theoretic approach for video segmentation using syntactic features, Pattern Recognition Letters, Volume 22, Number 13, pp. 1359-1369.

[8] Iyengar G., Lippman A. (1997). Models for automatic classification of video sequences, Proceedings of SPIE Storage and Retrieval for image and Video Databases VI, Volume 3312, pp. 216-227.
[9] Wang H., Divakaran A., Vetro A., Chang S. -F., Sun H. (2003). Survey of compressed-domain features used in audio-visual indexing and analysis, Journal of Visual Communication and Image Representation, Volume 14, Number 2, pp. 150-183.
[10] Yuan X., Lai W., Mei T., Hua X. –S., Wu X. –Q., Li S. (2006). Automatic video genre categorization using hierarchical SVM, Proceedings of IEEE International Conference on Image Processing (ICIP), pp. 2905–2908.
[11] Fischer S., Lienhart R., Effelsberg W. (1995). Automatic recognition of film genres, MULTIMEDIA '95: Proceedings of the third ACM international conference on Multimedia, pp. 295–304.
[12] Horn B. K. P., Schunck B. G. (1981). "Determining optical flow," AI, Volume 17, Number 1-3, pp. 185–203.
[13] Roach M.J., Mason J.S.D., Pawlewski M. (2001). Video Genre Classification using Dynamics, IEEE International Conference on Acoustics, Speech, and Signal Processing, Volume 3.
[14] Ferman A. M., Tekalp A. M., and Mehrotra R. (2002). Robust Color Histogram Description for video segment retrieval and identification. IEEE Transactions on Image Processing, Vol. 11, No. 5, pp. 497 – 508.
[15] Zampoglou M., Papadimitrious T., and Diamantaras K. I. (2008). Integrating Motion and Color for Content Based Video Classification. In Proceedings of IEEE International Conference on Image Processing, pp. 194-199.
[16] Wei C. –Y., Dimitrova N., and Chang S. –F. (2004, June). Color-Mood Analysis of Films Based on Syntactic and Psychological Models, IEEE International Conference on Multimedia and Expo (ICME '04), Vol. 2, pp. 831-834.
[17] Chan E., Lee J., and Roy R. (2015, December). What Can We Learn From Movie Colors?, Machine Learning Project, University of Stanford, USA.
[18] Hsu C. W., Chang C. C., and Lin C. J. (2003). A Practical Guide to Support Vector Classification. https://www.cs.sfu.ca/people/Faculty/teaching/726/spring11/svmguide.pdf, retrieved in January, 2016.

Enwerem Chinonyerem Udochukwu is currently a Lecturer with the Department of Computer Science, Federal University of Technology, Owerri, Nigeria. He received his B. Tech. in Computer Science from the Federal University of Technology, Owerri and thereafter went on to obtain a MSc. in Computer Science from the University of Ibadan, Ibadan, Nigeria. His current area of research includes (but is not limited to) Software Engineering, Multimedia Data mining, Pattern Recognition and Artificial Intelligence.

Charles Robert is currently an Adjunct Professor at Campbellsville University, Campbellsville, Kentucky, U.S.A. and also a Senior Lecturer with Department of Computer Science at the University of Ibadan, Ibadan, Nigeria. He obtained his B. Eng. from the Federal University of Technology, Minna, Nigeria and thereafter went on to obtain his Masters and Ph.D. degrees from the Université Nancy 2, Nancy, France. His current area of research includes (but is not limited to) Web and Multimedia Data mining and Artificial Intelligence.