FIT 3081

IMAGE PROCESSING

ASSIGNMENT 2 REPORT

GROUP 13

LIM ZHEN KANG 31886876

PRIYESH NILASH PATEL 32182058

LIM CHENG EN 30720028

FARAYHA ZAHEER ALAM 31164943

Table of Contents

Introduction

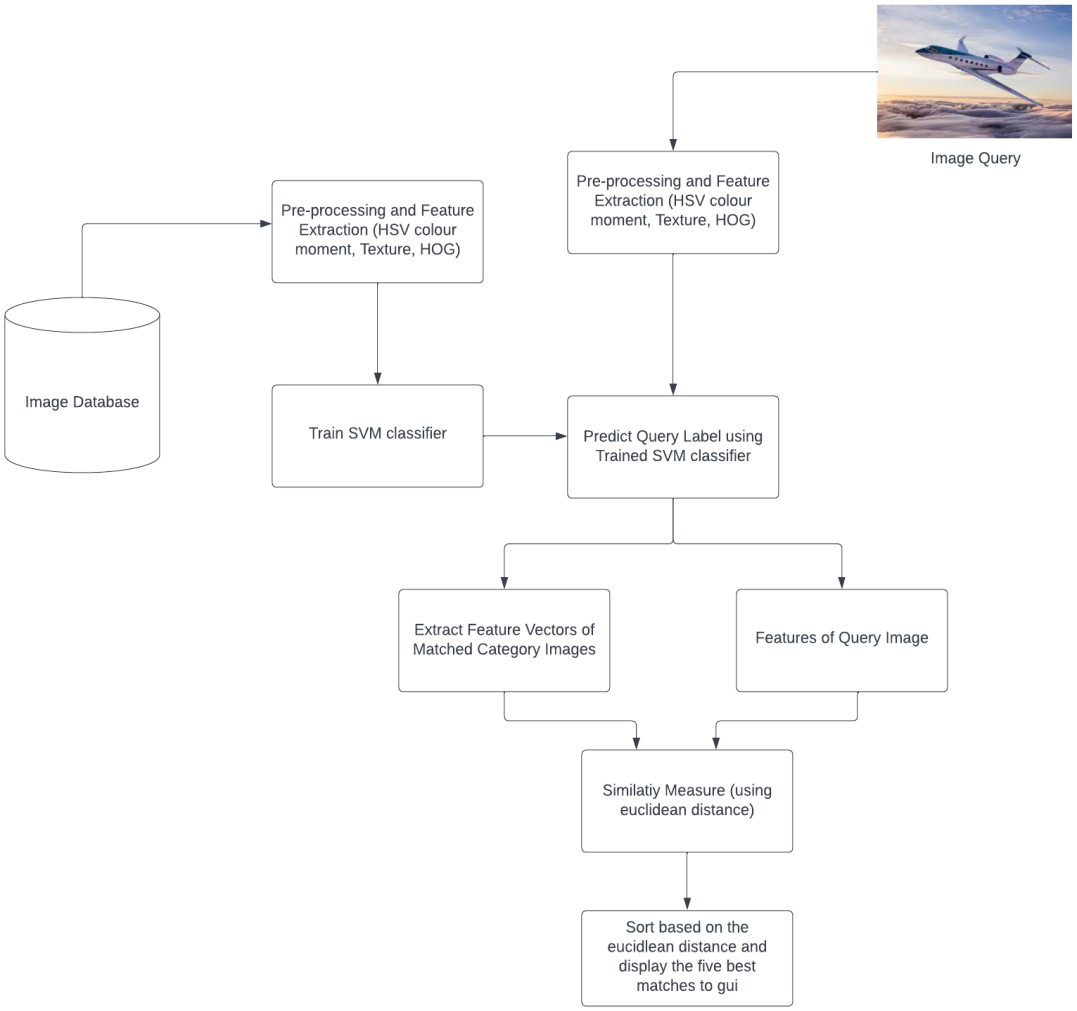
Content-based image retrieval (CBIR), also known as "Query-by Image Content(QBIC)" and "Content-Based Visual Information(CBVIR)," is a computer vision-based approach to image retrieval. It's a more sophisticated version of classic concept-based image retrieval methods. Traditional approaches relied on manual annotations such as keywords, tags, and so on, which can take a long time considering the size of the database, and there's a good chance that the annotation won't adequately represent the image, raising concerns about the approach's accuracy.

CBIR systems, on the other hand, rely on the analysis of image content, where "content" refers to the color, shape, or other information that can be extracted directly from the image itself. Handcrafted features, local features, global features, and machine learning techniques are all examples of feature levels and approaches that can be used in the CBIR system. To apply their methods and approaches, each feature makes use of different image attributes. Despite the fact that each feature has some limitations that may impair the system's performance and efficiency, the fact that this technique isn't just based on metadata, which is dependent on the quality of annotation, makes it a more desirable and preferred approach in today's world.

Our initial design included a mix of local, global, and handcrafted features, but after some trial and error, we discovered that they weren't significantly contributing to improving the accuracy of our model, so each of the team members looked for other feature extraction methods that could benefit us, and we switched to them. For example, instead of extracting the RGB color space with K-mean clustering, we moved to HSV color moments, which is easier to implement and improves the accuracy.

Our final tentative design, algorithms, methodology used, results, and analysis, as well as challenges, weaknesses, and strengths related to our training model, are all discussed in the accompanying report.

Finalized System Design



Algorithms and Methodology

1. Algorithms

Training Model:

1. Read training set from **cifar10Train**
2. Extract **HSV color moments** from training images
    a. Convert each image from **RGB** to **HSV**.
    b. Extract and vectorize each channel and compute color moments for each one of them (H, S,V).
3. Extract **Texture features** from training images
    a. Convert each image from **RGB** to **gray**, apply **unsharp masking**, and extract texture features,
4. Extract **HOG features** from training images
    a. Convert each image from **RGB** to **gray**, apply **unsharp masking**, and extract hog features.
5. **Concatenate** all features into one feature vector.
6. Train **SVM classifier** using training features/labels

Query Image:

1. **Retrieval** of query image from GUI
2. **Resizing** query image to match the size of database images' dimension (32 x 32)
3. Convert query image from **RGB** to **gray**, apply **unsharp masking**(For HOG and Texture **only**).
4. **Extract** HOG features, Texture features, and HSV color moments of the query image.
5. **Concatenate** features into one vector
6. **Predict** Query label by making use of **trained SVM classifier** and concatenated features.
7. **Extract** all of the images in the training subfolder with the **same label** (category).
8. Perform **Similarity Check** by calculating the **Euclidean distance** between the query image and all the images under the particular label.

```
9. Sort matched images using the Euclidean distance and
   pass the 5 best-matched images to GUI for
   displaying.
```

2. Methodology

This part is divided into five sections which are mentioned below.

1. Description of the Dataset

Our dataset consists of 60,000 photos with a 32 by 32 image dimension divided into ten classes, each with 6000 images. There are 50,000 images in the training dataset and 10,000 images in the testing dataset. However, for this assignment, we only used 8 classes due to some reasons which will be discussed in *later sections*, resulting in a total of 40,000 training and 8,000 testing images.

2. Pre-processing

Before extracting color features from the image, convert RGB to HSV for later computation of each of the channels (H, S, V). For texture and HSV features, conversion from RGB to gray is done followed by image sharpening using unsharp masking.

3. Feature Extraction

3.1. HSV Color Moments

Colour is one of the most important features that can be found in an image. Images are normally stored in red, green, and blue (RGB) format though RGB is less robust towards external lighting changes compared to hue, color, and saturation(HSV) color space. The HSV color space allows us to separate the luminance from the color information. Therefore, a function *colorMoment* is created where the image read is first converted into HSV and each channel is extracted and vectorized. The

mean, standard deviation, skewness, and max values of each channel are then individually extracted. The extracted values are then put into a feature array that holds the information for each picture in one row.

3.2. HOG(Histogram of Oriented Gradients)

The histogram of oriented gradients is one of the feature descriptors used in computer vision primarily for object detection. The approach counts the number of times a gradient orientation appears in a certain area of an image. This method is comparable to edge orientation histograms, scale-invariant feature transform descriptors, and shape contexts, but it is distinguished by the fact that it is computed on a dense grid of uniformly spaced cells and employs overlapping local contrast normalization for increased accuracy. For the extraction of HOG features from our gray-scaled image, we use *extractHOGFeatures(I*) in our working code. This function returns a 1 by N vector, where N denotes the length of the HOG feature and the vector denotes the encoding of local shape information from the image.

3.3. Local Binary Pattern (LBP)

Local Binary Pattern (LBP) algorithm is also used to extract the texture feature from the images. This algorithm is a simple and efficient texture operator which labels the pixels of an image by thresholding the neighborhood of each pixel and considers the result as a binary number. The MATLAB in-built function *extractLBPFeatures(I)* returns extracted uniform local binary pattern (LBP) from a grayscale image. The algorithm firstly splits the image into an 8x8 pixel matrix which contains the intensity of each pixel. After that, it applies the threshold value to calculate the binary value of neighboring pixels. Lastly, the algorithm converts the binary value into decimal value and output into a vector. The output contains the information on the texture

feature of the images and is further to be used by the classification.

All of the above-mentioned feature vectors are concatenated into one which uniquely represents a single image.

4. Training/Testing Model

The concatenated features are used to train the SVM classifier, and then a series of adjustments (described in the *following sections*) are made until an accuracy of 70% or higher is achieved on the testing dataset (which comprises 1000 images per class).

5. Query Image

After configuring the GUI (Graphical User Interface), a query image is uploaded and a number of pre-processing steps are performed on it (resizing the image to 32 × 32 to match the dimension of the images used to train the classifier, transforming the image from RGB to grayscale, sharpening the image using unsharp masking). Following this, the individual features (texture, HSV color moments, and hog) are extracted and combined into a single feature vector. The trained SVM classifier is then used to determine the label of the "query" image, and all images in the training dataset that match the label are retrieved. After running a Similarity Test, which involves calculating the Euclidean distance between the "query" image and all the images under the matched label, the sorting operation is applied and the top five matches (images) are extracted and passed to the GUI for display for the user.

Results and Analysis

The model proposed has been tested with 40000 training images from the CIFAR-10 dataset where there are 10 categories, airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck, with each of them containing 5000 images. Though 2 categories specifically cat and dog were not included in our final model as will be discussed later. For testing purposes, 8000 images were used from the CIFAR-10 dataset to qualitatively evaluate the retrieval effectiveness of our model. As shown in Fig 2.1, we were able to achieve a 73% accuracy when predicting the label of the testing images. The confusion chart shows the number of images matched for each category and their percentage. The automobile category has the highest accuracy at 81.2% as vehicles have a more accurate distinction between neighboring pixels to outline their texture. On the other hand, the bird has the lowest accuracy as birds' feathers and motion images make it more difficult to distinguish between neighboring pixels.

```
ans = 0.7315
```

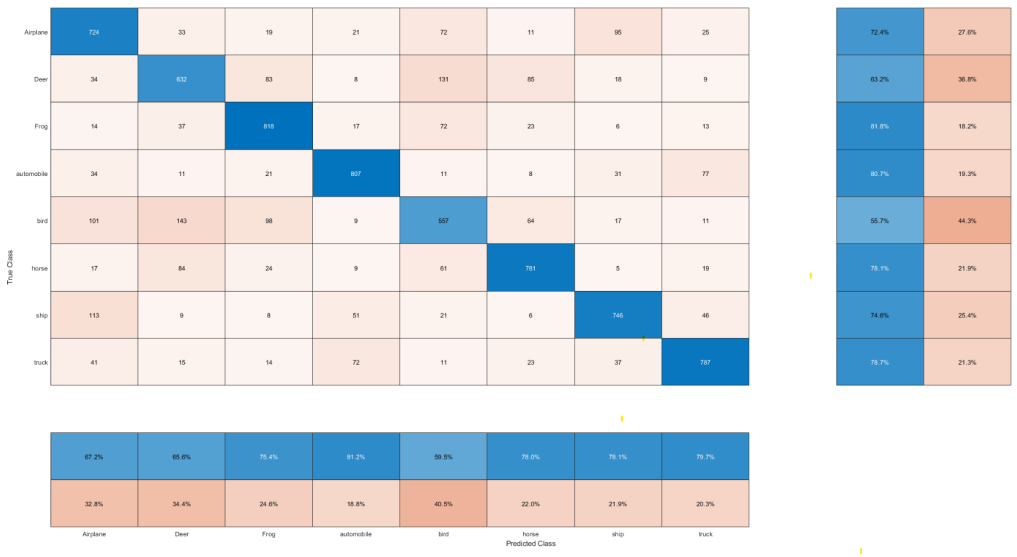


Fig 2.1 Result for 8 categories and total accuracy

As stated previously, the “Dog” and “Cat” subcategories were excluded from our final model. Though the model was run with all categories and the results are shown in Fig 2.2. The accuracy reduced significantly to 66% and runtime increased to 728 seconds from the previous 564 seconds. The confusion chart shows that cat and dog have high cross-matching which leads to the model having a lower accuracy. Dogs and cats are mostly found in common backgrounds and positions leading to more similar color features and HOG feature extraction.

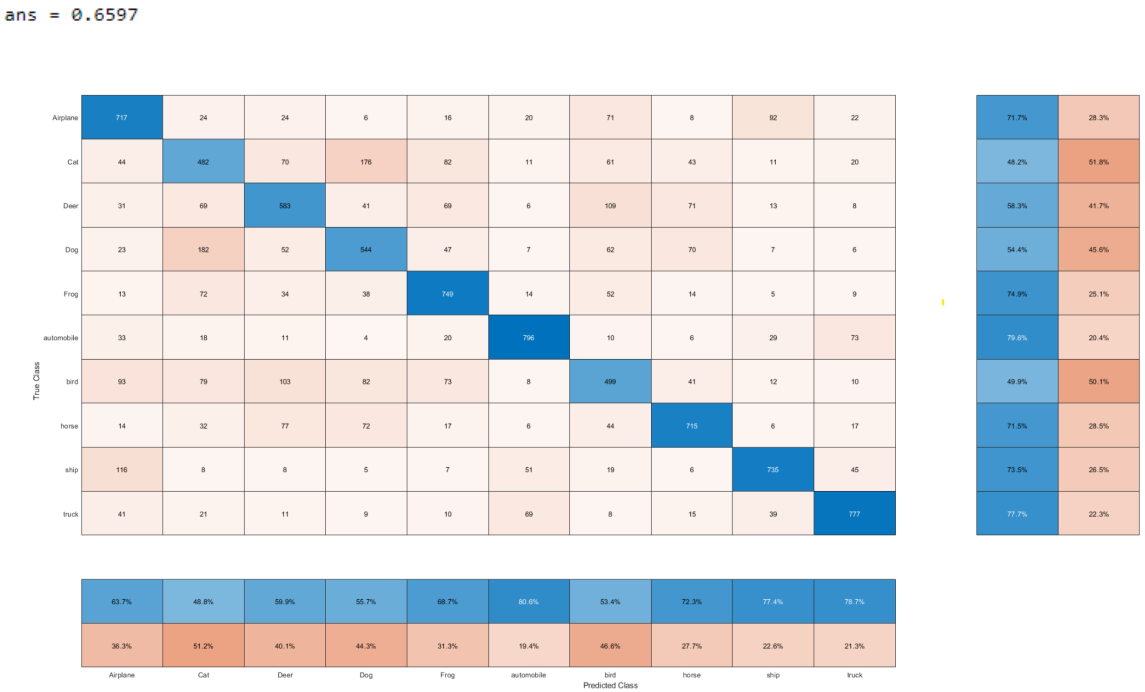


Fig 2.2 Accuracy results when using all categories

Once the model has been trained, query an image that is not in the testing or training dataset. Fig 2.3 shows a random plane image being queried and its results. As seen, the model is able to identify that the image has a plane but due to the more yellow color of the background, the model returns yellow airplanes.

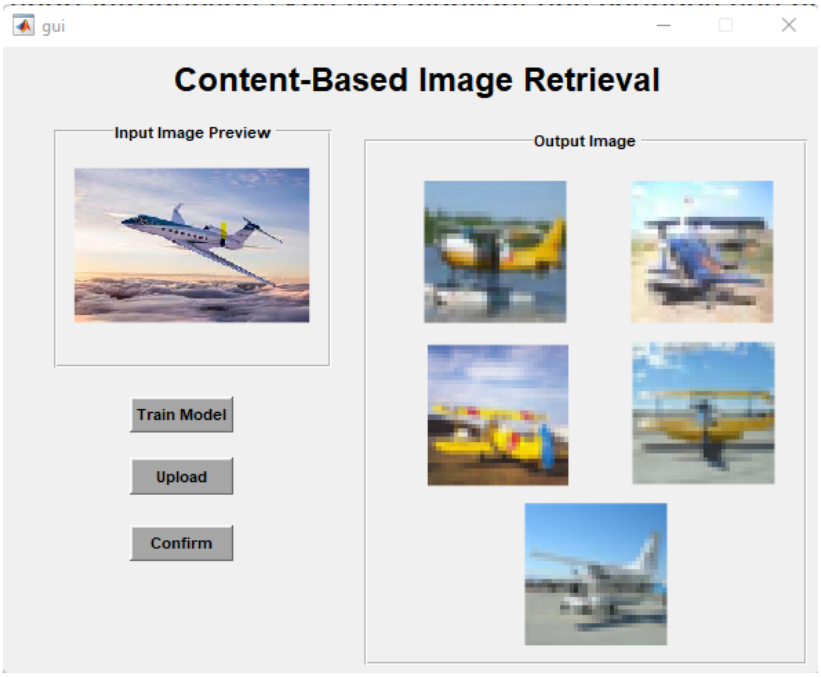


Fig 2.3 Airplane query

Additionally, a car image is provided to the model which identifies it as an automobile and returns images of automobiles in the database which have the lowest Euclidean distance as shown in Fig 2.4.

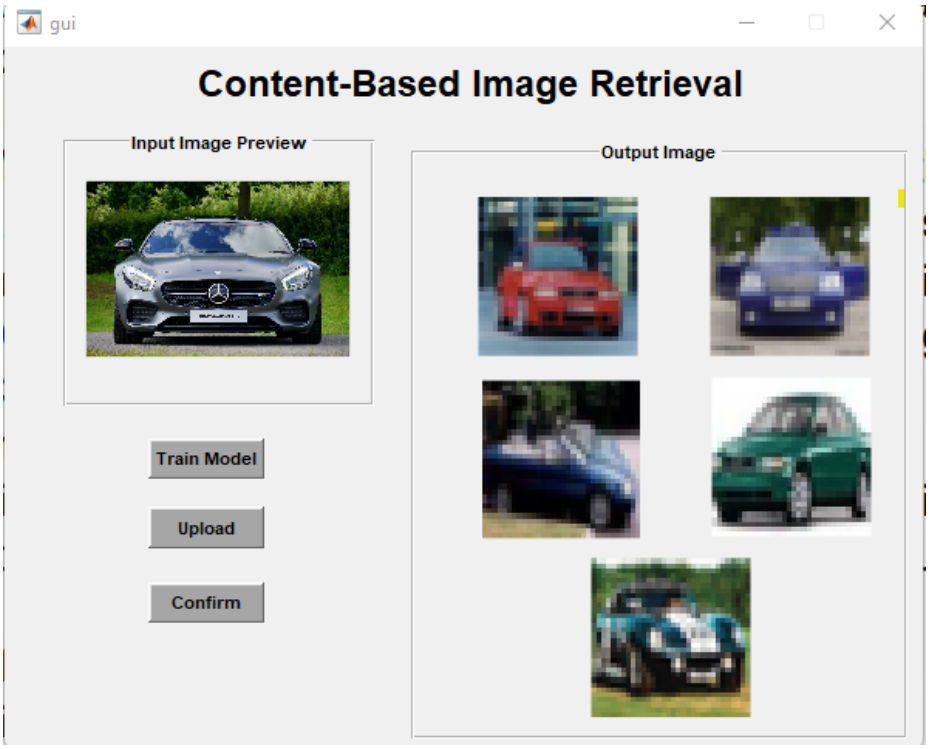


Fig 2.4 Query using car image

A few queries are provided as examples in Fig 2.5, 2.6, and 2.7.

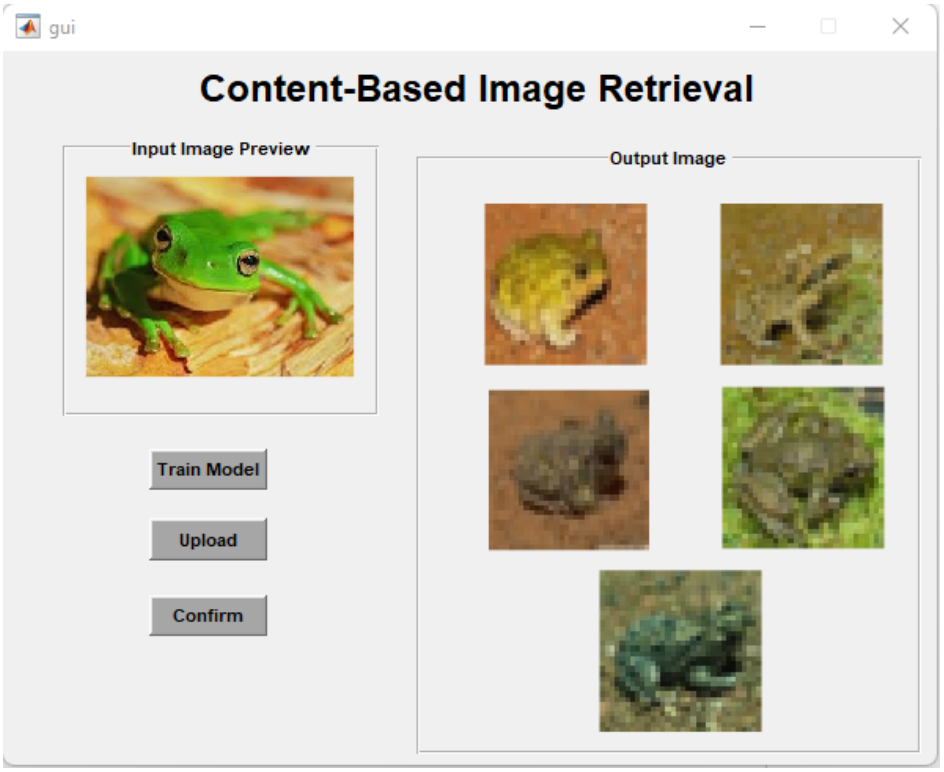


Fig 2.5 Frog query.

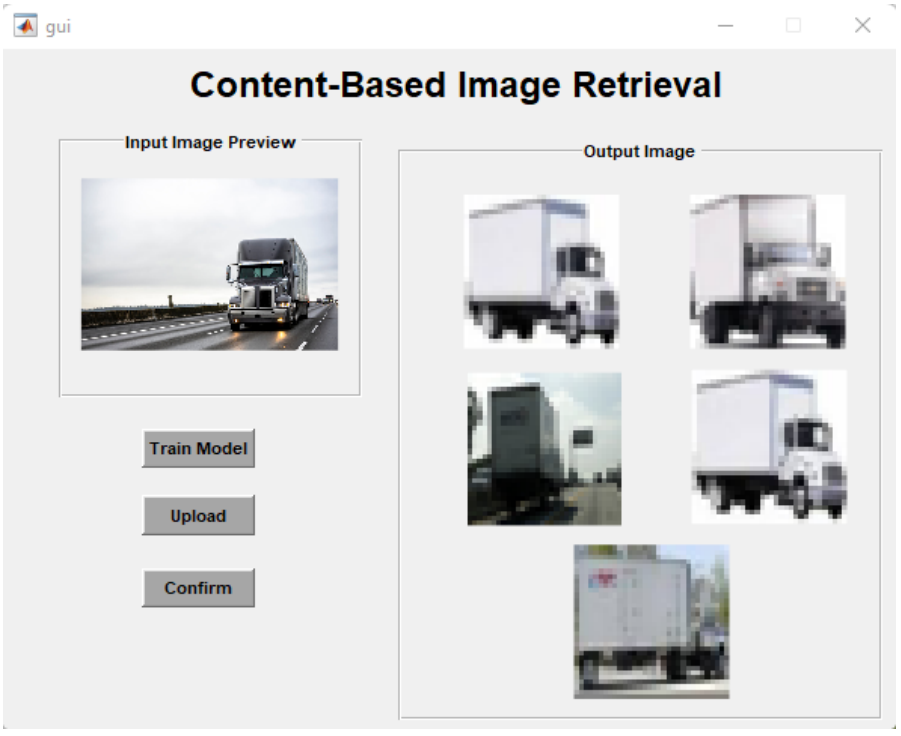


Fig 2.6 Truck query.

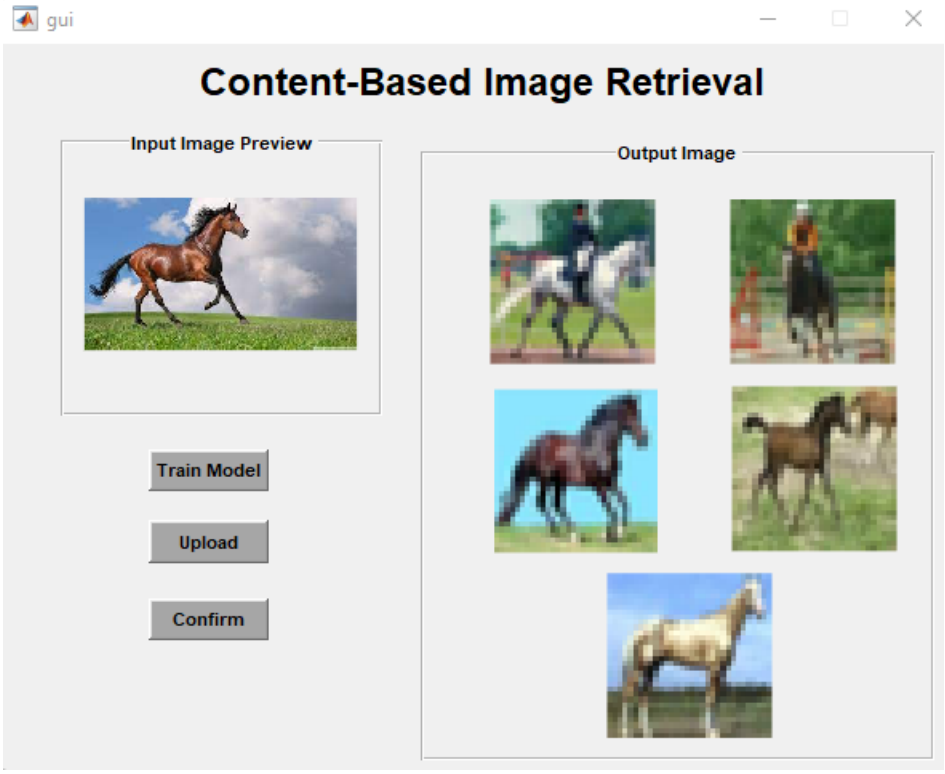


Fig 2.7 Horse query.

Discussion

I. Strength

From what has been discovered during the analysis, the proposed system above can accurately classify vehicles such as Trucks, Ships, and Automobiles. It shows that the algorithm can accurately measure the color feature being extracted from the images, with the combination of the HOG feature and LBP feature, it further improves the detection performance. Using Euclidean distance, we are also able to get the closest matching images when provided with a query image.

II. Weaknesses

The result of the experiments shows that during the process of classifying the images, the proposed system was not able to analyze the differences in images between the categories “Cat” and “Dog” efficiently. It appears to be confused between the features being extracted such as the shape and color might be a little bit the same between cats and dogs. Hence, the accuracy has been lowered.

III. Challenges

One of the challenges faced by our system is the classification of the dataset which includes “Cat” and “Dog. It is easy for humans to tell them apart but for the system to identify them automatically, it would require a better solution other than the proposed algorithm to overcome it. Our team had tried a lot of different algorithms such as Scale-Invariant Feature Transform (SIFT) with different image preprocessing methods to sharpen the image to solve this challenge. However, the results of those implementations are undesirable as those algorithms decrease the accuracy of the system.

According to Sun (2020), CNN was proposed as one of the methods that could efficiently classify “Cat” and “Dog”. It involves a few steps which are, Convolution, Max pooling, and Flattening. Convolution by applying feature detectors to the query image which is an array of numbers representing the features of the images and results in a feature map. Applying max-pooling which is a process to reduce the number of nodes in the fully connected layers without losing key features and spatial structure information in the images. Flattening will take all the pooled feature maps and process them into a single vector as the input for the fully connected layers. This also can be further improved by adding more convolution layers.

Conclusion

CBIR has been and will be a long topic of discussion which is still an ongoing challenge faced by the CBIR community. There are also many image processing features still being discussed and debated throughout the years of research up until now by the experts, all with a single aim, which is to design an algorithm that results in high accuracy of image retrieval and minimizes the computational cost of it.

In this report, a brief overview of the CBIR methods and the performance of the process has been presented. This includes methods based on color, textual, and HOG feature descriptors which are combined to improve the efficiency of the system in retrieving users' expected results. In fact, preprocessing methods could be used along with preprocessing techniques such as image sharpening and image conversion could enhance the precision of binarized statistical image features.

The experiment and outcome of the proposed system were performed by using the CIFAR10 dataset consisting of 60000 32x32 color images in 10 classes, with 6000 images per class. There are 50000 training images and 10000 test images. The results showed that the system was able to perform well with an average accuracy ranging from 60% to 70%+- by considering 8 classes which are 40,000 training and 8000 testing images due to the reasons being discussed earlier.

List of Contribution

Group Member	**Contribution**
Lim Zhen Kang	● Implement Local Binary Pattern Algorithm for CBIR system ● Implement interface for CBIR GUI ● Research on different image sharpening methods ● Records the video demonstration
Priyesh Nilash Patel	● Implement Colour Feature extraction Algorithm for CBIR System ● Implement Image Sharpening Function ● Implement query image retrieval code with euclidean distance matching ● Write report results and analysis section
Lim Cheng En	● Research on SIFT and SURF detection feature ● Write a report on Discussion ● Write a report on Conclusion
Farayha Zaheer Alam	● Implemented HOG Features for CBIR System. ● Written Introduction, Finalized System Design, Algorithm, and Methodology section.

Table 1. List of Contributions by the team member

References

Content-based image retrieval - Wikipedia. (2022). Retrieved 29 May 2022, from https://en.wikipedia.org/wiki/Content-based_image_retrieval

Histogram of oriented gradients - Wikipedia. (2022). Retrieved 29 May 2022, from https://en.wikipedia.org/wiki/Histogram_of_oriented_gradients

Sun, L. (2020, October 29). *CNN image classification: Cat or dog*. Medium. Retrieved May 29, 2022, from https://towardsdatascience.com/cnn-classification-a-cat-or-a-dog-568e6a135602