



Vidyavardhini's College of Engineering & Technology

Department of Computer Engineering

56-Sahil Pawar

To create program to perform a retrieving Images and Searching
--

Date of Performance: 09/10/2023

Date of Submission: 16/10/2023



56-Sahil Pawar

Aim: To create program to perform a retrieving Images and Searching

Objective: The fundamental need of any image retrieval model is to search and arrange the images that are in a visual semantic relationship with the query given by the user.

Most of the search engines on the Internet retrieve the images based on text-based approaches that require captions as input.

Theory:

Image Retrieval is a fundamental and long-standing computer vision task that involves finding images similar to a provided query from a large database. It's often considered as a form of fine-grained, instance-level classification. Not just integral to image recognition alongside classification and detection, it also holds substantial business value by helping users discover images aligning with their interests or requirements, guided by visual similarity or other parameters.

Code:-

```
import os
import numpy as np
from keras.applications.vgg16 import VGG16, preprocess_input
from keras.preprocessing import image
from sklearn.metrics.pairwise import cosine_similarity
import matplotlib.pyplot as plt
```



```
from PIL import Image, ImageDraw, ImageFont
```

```
def extract_features(image_path):
```

```
    model = VGG16(weights='imagenet', include_top=False)
```

```
    img = image.load_img(image_path, target_size=(224, 224))
```

```
    img_array = image.img_to_array(img)
```

```
    img_array = np.expand_dims(img_array, axis=0)
```

```
    img_array = preprocess_input(img_array)
```

```
    features = model.predict(img_array)
```

```
    features = features.flatten()
```

```
    return features
```

```
def find_similar_images(query_features, dataset_features):
```

```
    similarities = {}
```

```
    for filename, features in dataset_features.items():
```

```
        similarity = cosine_similarity([query_features], [features])[0][0]
```

```
        similarities[filename] = similarity
```

```
    return similarities
```



```
def plot_images_with_similarity(images, similarity_ratios, query_image_path):  
    # Load the query image  
    query_img = Image.open(query_image_path)  
  
    # Plotting setup  
    fig, axs = plt.subplots(1, len(images) + 1, figsize=(15, 5))  
    axs[0].imshow(query_img)  
    axs[0].axis('off')  
    axs[0].set_title('Query Image')  
  
    # Load and annotate similar images  
    for i, (filename, ratio) in enumerate(zip(images, similarity_ratios), 1):  
        img_path = os.path.join(dataset_path, filename)  
        img = Image.open(img_path)  
        axs[i].imshow(img)  
        axs[i].axis('off')  
        axs[i].set_title(f'{filename}\nSimilarity: {ratio:.4f}')  
  
    plt.show()
```



Vidyavardhini's College of Engineering & Technology

Department of Computer Engineering

Path to your dataset

```
dataset_path = "/content/IMG"
```

Extract features for all images in the dataset

```
feature_vectors = {}
```

```
for filename in os.listdir(dataset_path):
```

```
    if filename.endswith(".jpg") or filename.endswith(".png"):
```

```
        image_path = os.path.join(dataset_path, filename)
```

```
        features = extract_features(image_path)
```

```
        feature_vectors[filename] = features
```

Path to your query image

```
query_image_path = "/content/Hyundai-Grand-i10-Nios-200120231541.jpg"
```

```
query_features = extract_features(query_image_path)
```

Find similar images

```
similarities = find_similar_images(query_features, feature_vectors)
```

Sort the results by similarity

```
sorted_similarities = sorted(similarities.items(), key=lambda x: x[1], reverse=True)
```



Vidyavardhini's College of Engineering & Technology

Department of Computer Engineering

```
# Extract filenames and similarity ratios for plotting
```

```
filenames, similarity_ratios = zip(*sorted_similarities)
```

```
# Plot images with similarity ratios
```

```
plot_images_with_similarity(filenames, similarity_ratios, query_image_path)
```

Output:-



Conclusion: In conclusion, the aim of studying the image retrieval using SIFT (Scale-Invariant Feature Transform) descriptors. It extracts distinctive features from a query image and a dataset of images, matches these features, and ranks images in the dataset based on the number of matching features. The program can be useful for applications like content-based image search and recommendation. While the code here uses SIFT, more advanced methods and deep learning can enhance retrieval accuracy.