CONTENTS **Page No.**

	Abstract	i
1	**INTRODUCTION**	1
	1.1 Objective	1
	1.2 Motivation	2
	1.3 Background	1
2	**Related works**	3
3	**Proposed System**	4
	3.1 Drawback/Limitations of Existing System .	4
	3.2 Your Solution/Finding	4
	3.3 Sample Code .	4
4	**Result Analysis and Screenshots**	12
5	**Conclusion** .	22
6	**References**	23

1. INTRODUCTION

1.1. OBJECTIVE

Sketch-based image retrieval (SBIR), which allows the user to search images with a free-hand sketch has been an active research topic under the field of content-based image retrieval (CBIR) for a long time. The main objective of this project is to be able to perform accurate image retrieval from Flickr15K dataset via sketch based query images. The input sketch drawn with a free hand that roughly describes the holistic shape and salient local shapes of the searched object/scene in the dataset . On the other hand, the dataset images are generally realistic photographs which are very different from the input sketch. The core task of SBIR is to find images which have some similar object/scene to the holistic shape and salient local details of the input sketch.

1.2. MOTIVATION

The following project was chosen because sometimes when we are searching for an object/ scene but don't exactly remember the name we can draw that object/scene and get to know what exactly it is . Also SBIR is used in forensics , e-commerce, etc. Even children can use SBIR to learn new places and objects. Thus this topic seemed interesting and easy to explore.

1.3. BACKGROUND

Sketch-based image retrieval (SBIR) has been studied since the early 1990s and has drawn more and more interest recently. Yet, a comprehensive review of the SBIR field is still absent. Fine-grained SBIR has become the main topic for the recent research. . In computer vision and computer graphics, sketches also have many applications, such as sketch recognition, sketch synthesis and sketch-based image retrieval.

Pre-processing Image For Retrieval:

- Prior to using the harvested photos as input to CNN training, the images are pre processed. First, images are resized maintaining aspect ratio such that longest side (height or width) of the image is 256 pixels.
- Second, a 'soft' edge map is computed from the resized image using the contour detection method . The soft edge map is converted into a binary edge map by thresholding to retain the 25% strongest edge pixels, and removing the weakest 25%
- Edge following (hysteresis thresholding) is applied to the remaining pixels as per the Canny edge detection algorithm to detect the edge points connected to the "strong edges" and remove the isolated edge pixels. The resulting edgemap is padded with non-edge pixels to achieve a fixed dimension of 256 × 256 pixels.

Triplet loss: It is a loss function for machine learning algorithms where a baseline (anchor) input is compared to a positive (truthy) input and a negative (falsy) input.

The goal of the triplet loss is to make sure that:

1) Two examples with the same label have their embeddings close together in the embedding space
2) Two examples with different labels have their embeddings far away.

To formalise this requirement, the loss will be defined over **triplets** of embeddings:

1. an **anchor**
2. a **positive** of the same class as the anchor
3. a **negative** of a different class

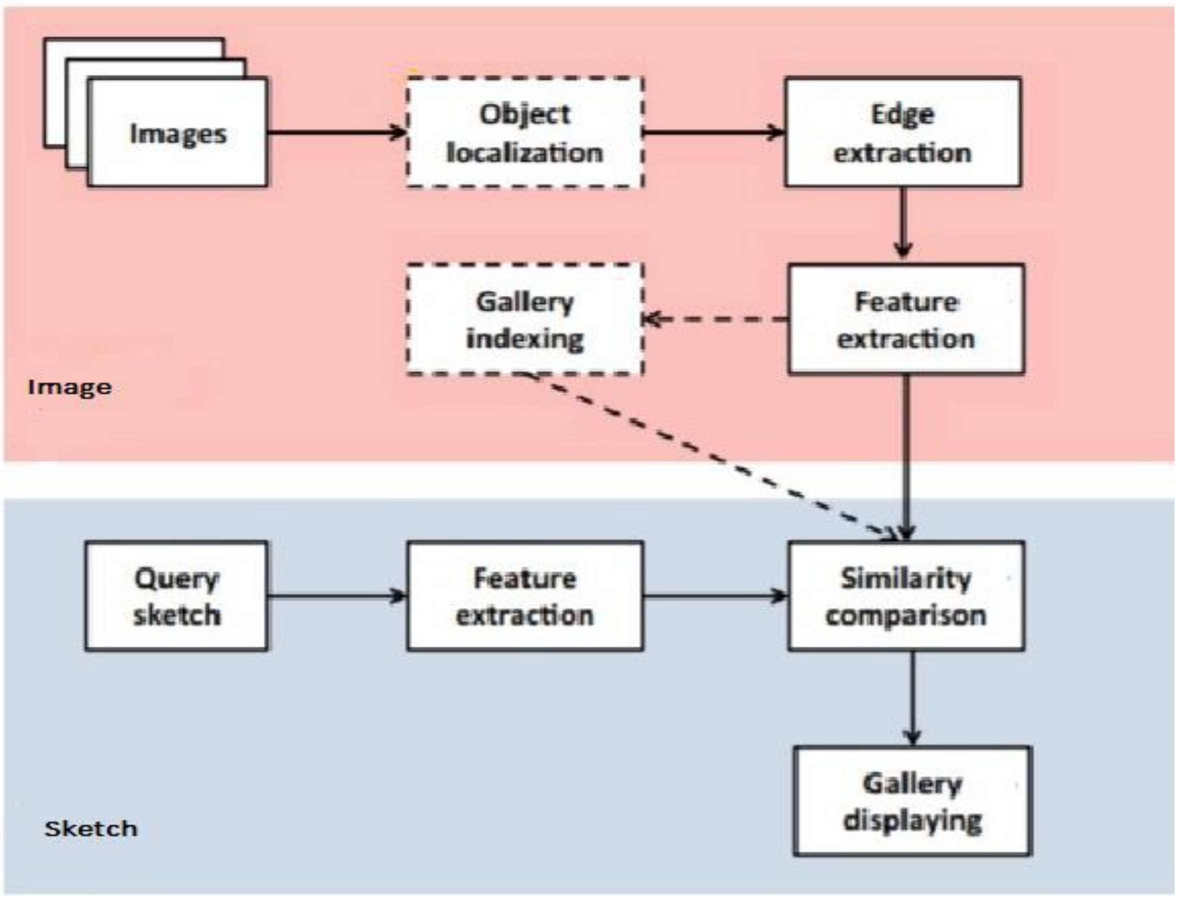
For some distance on the embedding space dd, the loss of a triplet (a,p,n)(a,p,n) is:

L=max(d(a,p)−d(a,n)+margin,0)L=max(d(a,p)−d(a,n)+margin,0)

We minimize this loss, which pushes d(a,p)d(a,p) to **00** and d(a,n)d(a,n) to be greater than **d(a,p)+margind(a,p)+margin**. As soon as nn becomes an “easy negative”, the loss becomes zero.

Convolutional Neural Networks: They have a different architecture than regular Neural Networks. Regular Neural Networks transform an input by putting it through a series of hidden layers. Every layer is made up of a set of neurons, where each layer is fully connected to all neurons in the layer before. Finally, there is a last fully-connected layer — the output layer — that represent the predictions. Convolutional Neural Networks are a bit different. First of all, the layers are organized in 3 dimensions: width, height and depth. Further, the neurons in one layer do not connect to all the neurons in the next layer but only to a small region of it. Lastly, the final output will be reduced to a single vector of probability scores, organized along the depth dimension. The convolution is performed on the input data with the use of a filter or kernel (these terms are used interchangeably) to then produce a feature map. We execute a convolution by sliding the filter over the input. At every location, a matrix multiplication is performed and sums the result onto the feature map.

The figure shows the SBIR system stems used in the project.



2. RELATED WORKS

There has been a lot of related work done in the field of SBIR . SBIR began to gain momentum in the early nineties with the color-blob based query systems of Flickner et al. and Kato et al. that matched coarse attributes of color, shape and texture using region adjacency graphs defined over local image structures (blobs). More efficient matching schemes for blob based queries, using spectral descriptors were subsequently proposed. This early wave of SBIR systems was complemented in the late nineties by several algorithms that accept line-art sketches, more closely resembling the free-hand sketches casually generated by lay users in the act of sketching a throw away query that required robust edge matching algorithms are to deliver usable results. Early approaches adopted optimization strategies that deformed a model of the sketch query to fit dataset images, ranking these according of their similarities with the sketch. Models explored by these approaches include elastic contour and super-pixel aggregation . However, model fitting approaches scale at best linearly with dataset size with each comparison often taking seconds leading to poor scalability. In contrast, the deep-learning based methods learn some variants of end-to-end cross-modal retrieval models to address this sketch-to-image domain shift. Some of the recent methods are siamese networks , triplet-loss or contrastive-loss based models. describes a hybrid multi-stage deep network, which combines both contrastive and triplet networks. Cao et al.proposed a new descriptor called Symmetric-aware Flip Invariant Sketch Histogram (SYM -FISH) which included three steps for extraction. The Flip Invariant Sketch Histogram (FISH) descriptor is first extracted on the input image followed by exploring the symmetric character of the image by calculating the kurtosis coefficient followed by generating the SYM-FISH by constructing a symmetry table. Li et al. show that sketches can be used for fine grained retrieval within object categories. They introduce a mid-level representation of sketch that along with capturing the object pose also has the ability to traverse both the sketch and image domains. Wang et al. proposed a crossdomain embedding methods that train Siamese networks to learn a common feature space for Sketches and 3D models.

3. PROPOSED SYSTEM

3.1 Drawback/Limitations of Existing System

The existing system used a sketch based dataset like Sketchy that was already trained for the Flickr15K dataset image retrieval, so no new sketch could be drawn or used as query without fully training the datasets again . The sketches of the pretrained dataset are also a little well shaped which might not be a case if a layman is trying to draw it free handed. Thus the existing system although very accurate in results does not actually use the sketches that are mostly made free handed.

3.2 Your Solution/Finding .

This project uses on spot sketch training only for that particular sketch made on the canvas right at the moment. This ensures that any type of sketch can be used as query and retrieval is done based on that particular sketch query .Each sketch made on spot is pre-processed and then features are extracted from it which are saved to numpy file until no new sketch is made and saved Although the pretrained sketch dataset gives 100% accuracy while on spot sketches give around 80-90% accuracy based on how precise the sketches are made.

3.3 Sample Code

runner.py

```
from flask import Flask, render_template, request
import json
import os
import time
import base64
import torch
from datetime import timedelta

# SketchTriplet network--------------------
from SketchTripletHalf.SketchTriplet_half_sharing import BranchNet
from SketchTripletHalf.SketchTriplet_half_sharing import SketchTriplet as SketchTriplet_halfshare
from SketchTripletHalf.flickr15k_dataset import flickr15k_dataset_train
```

```python
from SketchTripletHalf.retrieval import retrieval

def load_model_retrieval():
    net_dict_path = './static/model/500.pth'
    branch_net = BranchNet()  # for photography edge
    net = SketchTriplet_halfshare(branch_net)
    if torch.cuda.is_available():
        net.load_state_dict(torch.load(net_dict_path))
        net = net.cuda()
    else:
        net.load_state_dict(torch.load(net_dict_path, map_location=torch.device('cpu')))
    net.eval()
    return net

# Retrieval model and dataset definition
flickr15k_dataset = flickr15k_dataset_train()
retrieval_net = load_model_retrieval()

app = Flask(__name__, template_folder='templates', static_folder='static')
app.send_file_max_age_default = timedelta(seconds=1)

@app.route('/canvas', methods=['POST', 'GET'])
def upload():
    if request.method == 'POST':
        sketch_src = request.form.get("sketchUpload")
        upload_flag = request.form.get("uploadFlag")
        sketch_src_2 = None
        if upload_flag:
            sketch_src_2 = request.files["uploadSketch"]
        if sketch_src:
            flag = 1
        elif sketch_src_2:
            flag = 2
        else:
            return render_template('canvas.html')

        basepath = os.path.dirname(__file__)
        upload_path = os.path.join(basepath, 'static/sketch', 'upload.png')
        if flag == 1:
            # base64 image decode
            sketch = base64.b64decode(sketch_src[22:])
            user_input = request.form.get("name")
            file = open(upload_path, "wb")
            file.write(sketch)
            file.close()

        elif flag == 2:
            # upload sketch
            sketch_src_2.save(upload_path)
            user_input = request.form.get("name")

        # for retrieval
        retrieval_list, real_path = retrieval(retrieval_net, upload_path)
        real_path = json.dumps(real_path)

        return render_template('panel.html', userinput=user_input, val1=time.time(), upload_src=sketch_src,
                               retrieval_list=retrieval_list,
                               json_info=real_path)

    return render_template('canvas.html')

@app.route('/canvas')
def homepage():
    return render_template('canvas.html')

if __name__ == '__main__':
    # open debug mode
    app.run(debug=True)
```

retrieval.py

```python
from PIL import Image
import numpy as np
import torchvision.transforms as transforms
from torch.autograd import Variable
import torch


def extract_feat_sketch(net, sketch_src):
    img_size = 256
    normalize = transforms.Normalize((0.4914, 0.4822, 0.4465), (0.2023, 0.1994, 0.2010))
    transform = transforms.Compose([
        transforms.Resize((img_size, img_size)),
        transforms.ToTensor(),
        normalize
    ])
    sketch_src = transform(sketch_src)
    sketch = Variable(sketch_src.unsqueeze(0))
    if torch.cuda.is_available():
        sketch = sketch.cuda()
    feat = net.get_branch_sketch(sketch)
    feat = feat.cpu().data.numpy()
    return feat


def get_real_path(retrieval_list):
    retrieval_list = list(retrieval_list)
    real_list_set = []
    for i in range(5):
        real_list = []
        for j in range(18):
            ori_path = retrieval_list[i * 18 + j]
            real_path = './images/dataset/' + ori_path.split('/')[-
2] + '/' + ori_path.split('/')[-1][:-4] + '.png'
            name = ori_path.split('/')[-2] + '/' + ori_path.split('/')[-
1][:-4]
            real_list.append((real_path, name))
        real_list_set.append(real_list)


    pathdic = []
    for i in range(90):
        tmp = {}
        ori_path = retrieval_list[i]
        real_path = './images/dataset/' + ori_path.split('/')[-
2] + '/' + ori_path.split('/')[-1][:-4] + '.png'
        name = ori_path.split('/')[-2] + '/' + ori_path.split('/')[-1][:-
4]
        tmp['path'] = real_path
        tmp['name'] = name
        pathdic.append(tmp)

    return real_list_set, pathdic


def retrieval(net, sketch_path):
    sketch_src = Image.open(sketch_path).convert('RGB')
    feat_s = extract_feat_sketch(net, sketch_src)
    feat_photo_path = './SketchTripletHalf/dataset/feat_photo.npz'
    feat_photo = np.load(feat_photo_path)

    feat_p = feat_photo['feat']
    cls_name_p = feat_photo['cls_name']
    cls_num_p = feat_photo['cls_num']
    path_p = feat_photo['path']
    name_p = feat_photo['name']

    dist_l2 = np.sqrt(np.sum(np.square(feat_s - feat_p), 1))
    order = np.argsort(dist_l2)
    order_path_p = path_p[order]

    return get_real_path(order_path_p)
```

flickr15k_dataset.py

```
import os
import os.path
from torch.utils.data import Dataset


class flickr15k_dataset_train(Dataset):
    def __init__(self, root='./static/images'):
        self.root = root
        self.gt_path = os.path.join(self.root, 'groundtruth')
        self.img_set_path = os.path.join(self.root, 'dataset')
        self.gt = {}

        for i in range(1, 34):
            self.gt[str(i)] = []
        file = open(self.gt_path)
        for line in file:
            sketch_cls = line.split()[0]
            img_path = line.split()[1][:-4] + '.png'
            img_cls = img_path.split('/')[0]
            img_name = img_path.split('/')[1][:-4]
            img_path = os.path.join(self.img_set_path, img_path)
            # check img exist
            if os.path.exists(img_path):
                self.gt[sketch_cls].append((img_path, img_cls, img_name))
        file.close()

        self.datapath = []
        for i in range(1, 34):
            item = str(i)
            for fn in self.gt[item]:

                self.datapath.append((fn[1], item, fn[0], fn[2]))

    def __getitem__(self, idx):
        photo = self.datapath[idx]
        return photo

    def __len__(self):
        return len(self.datapath)
```

SketchTriplet_half_sharing.py

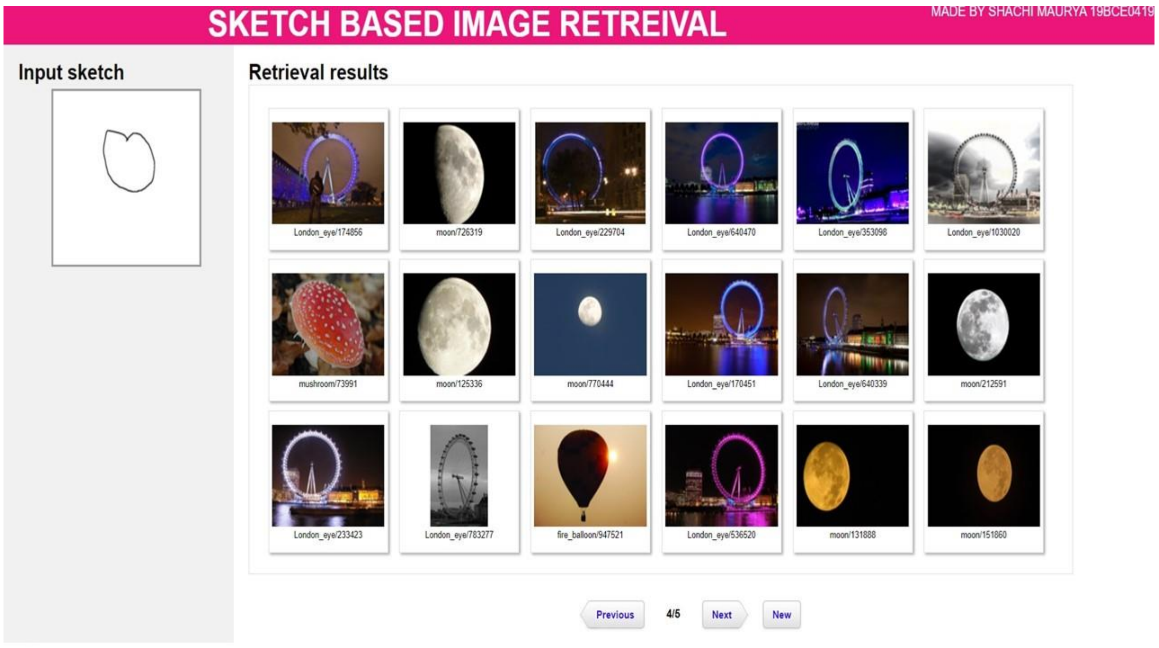
```
import torch.nn as nn
```

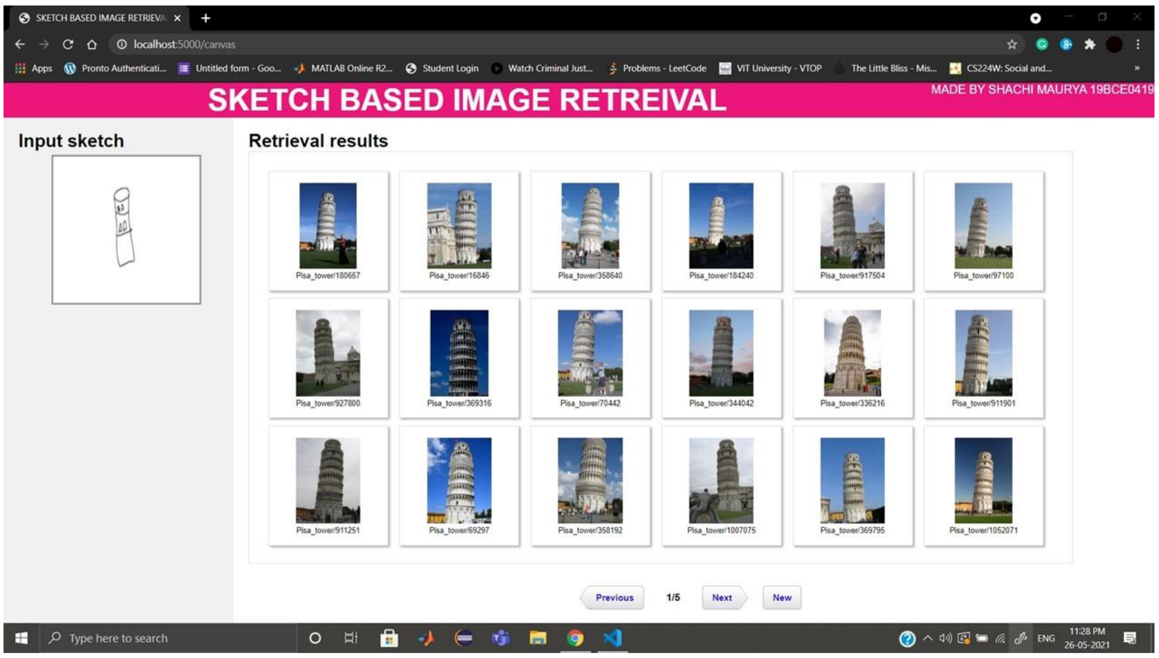
```python
class BranchNet(nn.Module):
    def __init__(self, num_feat=100):
        super(BranchNet, self).__init__()
        self.num_feat = num_feat
        self.conv1 = nn.Sequential(
            nn.Conv2d(
                in_channels=3,
                out_channels=64,
                kernel_size=15,
                stride=3,
                padding=0
            ),
            nn.ReLU(),
            nn.MaxPool2d(
                kernel_size=3,
                stride=2
            )
        )
        self.conv2 = nn.Sequential(
            nn.Conv2d(
                in_channels=64,
                out_channels=128,
                kernel_size=5,
                stride=1,
                padding=0
            ),
            nn.ReLU(),
            nn.MaxPool2d(
                kernel_size=3,
                stride=2
            )
        )
        self.conv3 = nn.Sequential(
            nn.Conv2d(
                in_channels=128,
                out_channels=256,
                kernel_size=3,
                stride=1,
                padding=0
            ),
            nn.ReLU()
        )
        self.conv4 = nn.Sequential(
            nn.Conv2d(
                in_channels=256,
                out_channels=256,
                kernel_size=3,
                stride=1,
                padding=0
            ),
            nn.ReLU()
        )
        self.conv5 = nn.Sequential(
            nn.Conv2d(
                in_channels=256,
                out_channels=256,
                kernel_size=3,
                stride=1,
                padding=0
            ),
            nn.ReLU(),
            nn.MaxPool2d(
                kernel_size=3,
                stride=2
            )
        )
        self.fc6 = nn.Sequential(
            nn.Linear(256 * 5 * 5, 2048),
            nn.ReLU(),
            nn.Dropout(p=0.55)
        )
        self.fc7 = nn.Sequential(
            nn.Linear(2048, 512),
            nn.ReLU(),
            nn.Dropout(p=0.55)
        )
        self.feat = nn.Linear(512, num_feat)

    def forward(self, x):
        x = self.conv1(x)
        x = self.conv2(x)
        x = self.conv3(x)
        x = self.conv4(x)
        x = self.conv5(x)
        x = x.view(x.size(0), -1)
        x = self.fc6(x)
        x = self.fc7(x)
        x = self.feat(x)
        return x

    def get_halfsharing(self, x):
        x = self.conv4(x)
        x = self.conv5(x)
        x = x.view(x.size(0), -1)
        x = self.fc6(x)
        x = self.fc7(x)
```

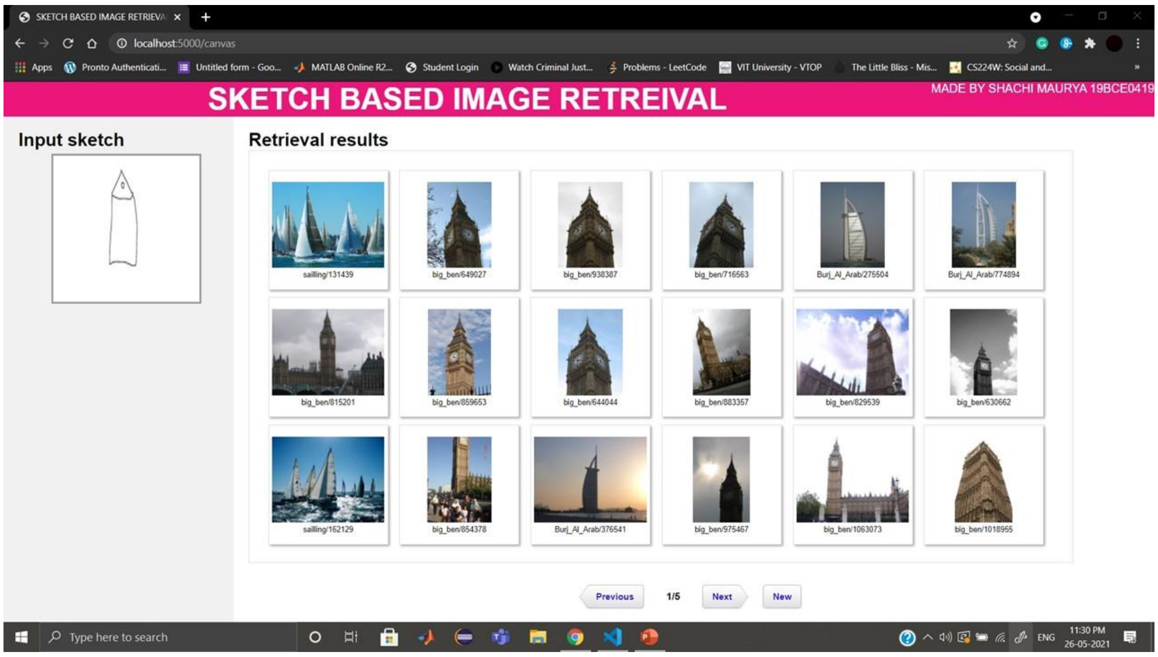
```python
        x = self.feat(x)
        return x

    def get_branch(self, x):
        return self.forward(x)


class SketchTriplet(nn.Module):
    def __init__(self, branch_net):
        super(SketchTriplet, self).__init__()
        self.branch_net = branch_net

        self.conv1_a = nn.Sequential(
            nn.Conv2d(
                in_channels=3,
                out_channels=64,
                kernel_size=15,
                stride=3,
                padding=0
            ),
            nn.ReLU(),
            nn.MaxPool2d(
                kernel_size=3,
                stride=2
            )
        )
        self.conv2_a = nn.Sequential(
            nn.Conv2d(
                in_channels=64,
                out_channels=128,
                kernel_size=5,
                stride=1,
                padding=0
            ),
            nn.ReLU(),
            nn.MaxPool2d(
                kernel_size=3,
                stride=2
            )
        )
        self.conv3_a = nn.Sequential(
            nn.Conv2d(
                in_channels=128,
                out_channels=256,
                kernel_size=3,
                stride=1,
                padding=0
            ),
            nn.ReLU()
        )

    def ancSeq(self, x):
        x = self.conv1_a(x)
        x = self.conv2_a(x)
        x = self.conv3_a(x)
        # half sharing
        x = self.branch_net.get_halfsharing(x)
        return x

    def forward(self, anc_src, pos_src, neg_src):
        self.anc_src = anc_src  # anchor source     (sketch input)
        self.pos_src = pos_src  # positive source   (photograph edge input
)
        self.neg_src = neg_src  # negative source   (photograph edge input
)
        feat_a = self.ancSeq(self.anc_src)
        feat_p = self.branch_net(self.pos_src)
        feat_n = self.branch_net(self.neg_src)
        return feat_a, feat_p, feat_n

    def get_branch_sketch(self, x):
        return self.ancSeq(x)

    def get_branch_photo(self, x):
        return self.branch_net(x)
```

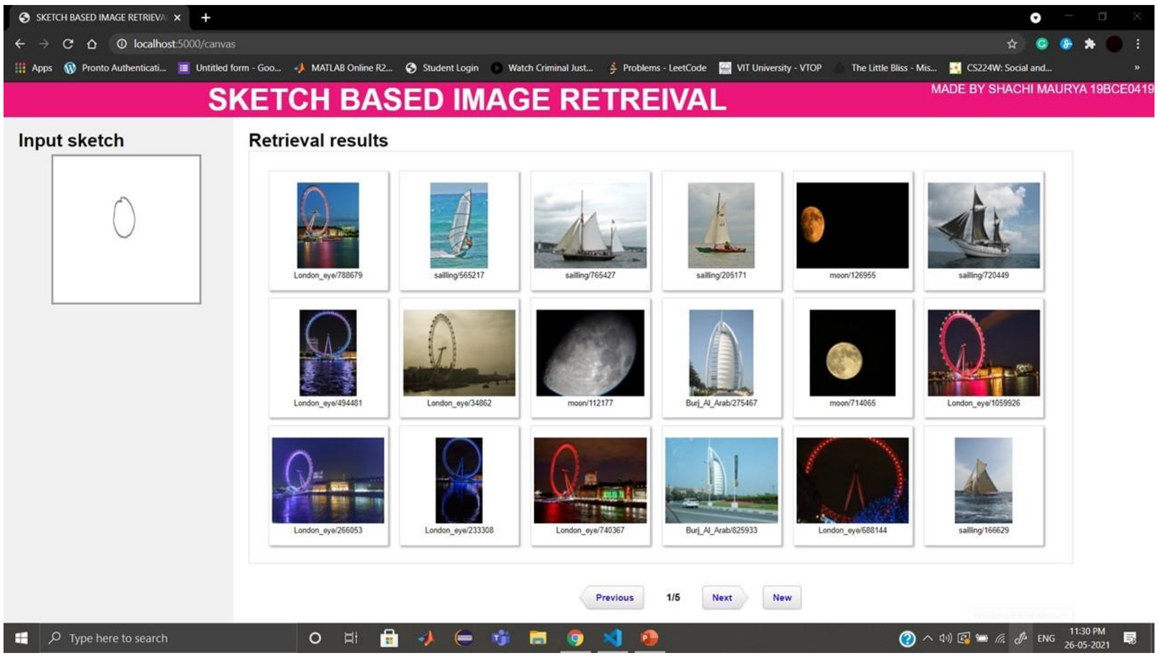
4 Result Analysis and Screenshots

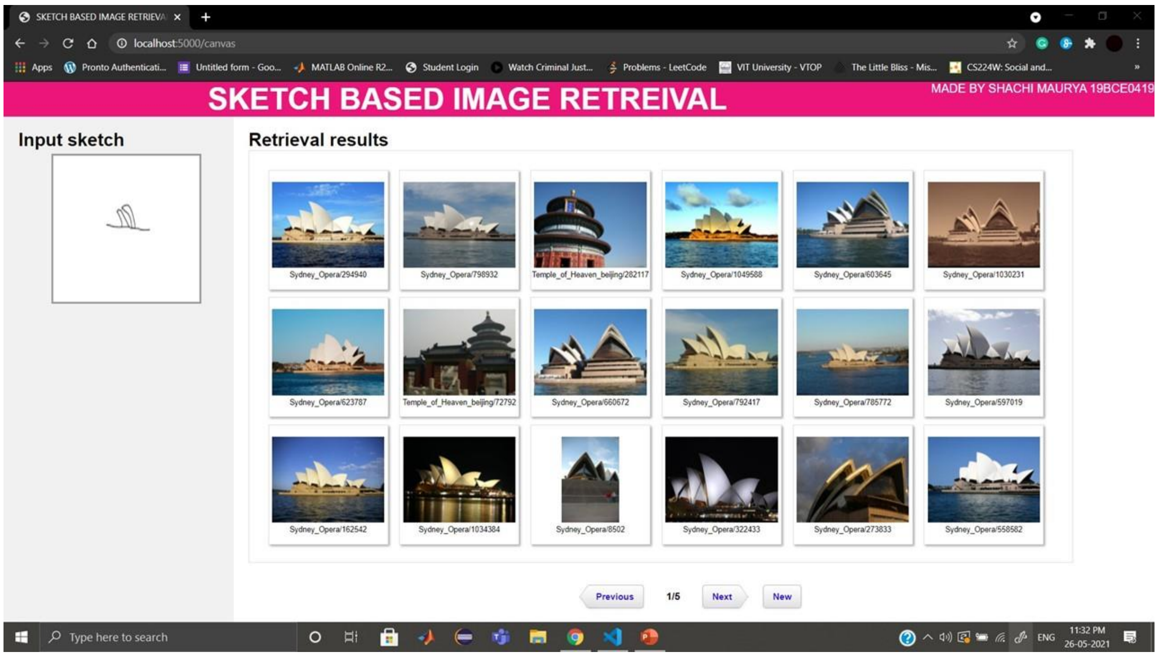
The images retrieved are 95-100% accurate for those categories which have either a illustrative sketch or has more than 500 images in the Flickr15K dataset .While those sketches which are simple curves or have less that 500 images in Flickr15K Dataset give varying results. Overall the whole retrieval process is a success as there are only a few categories out of 33 whose retrieval is not 100% accurate.
The categories like London Eye, Moon , Taj Mahal ,Big Ben ,etc have more than 500 Flickr15K images and are also giving highest accuracy for retrieval . While the categories like Heart, Mushroom, Butterfly, Swan, etc do not give very accurate results. The following screenshots give the justification for the result obtained.

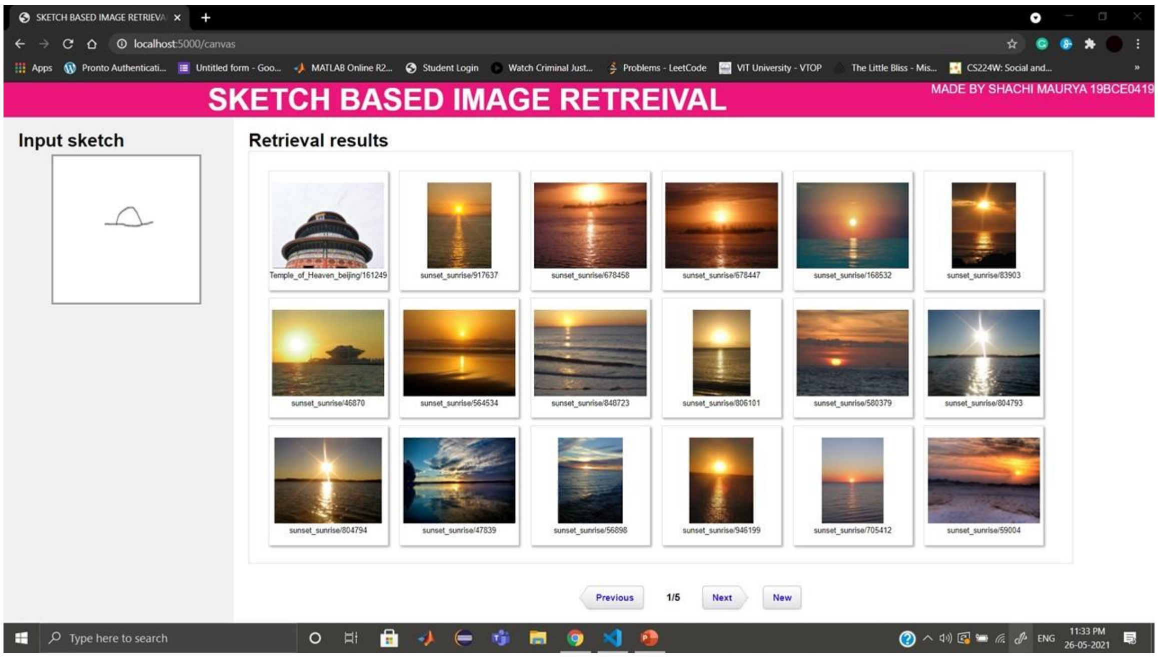

SKETCH BASED IMAGE RETREIVAL
MADE BY SHACHI MAURYA 19BCE0419
Input sketch
Retrieval results
London_eye/174856
moon/726319
London_eye/229704
London_eye/640470
London_eye/353098
London_eye/1030020
mushroom/73991
moon/125336
moon/770444
London_eye/170451
London_eye/640339
moon/212591
London_eye/233423
London_eye/783277
fire_balloon/947521
London_eye/536520
moon/131888
moon/151860
Previous
4/5
Next
New

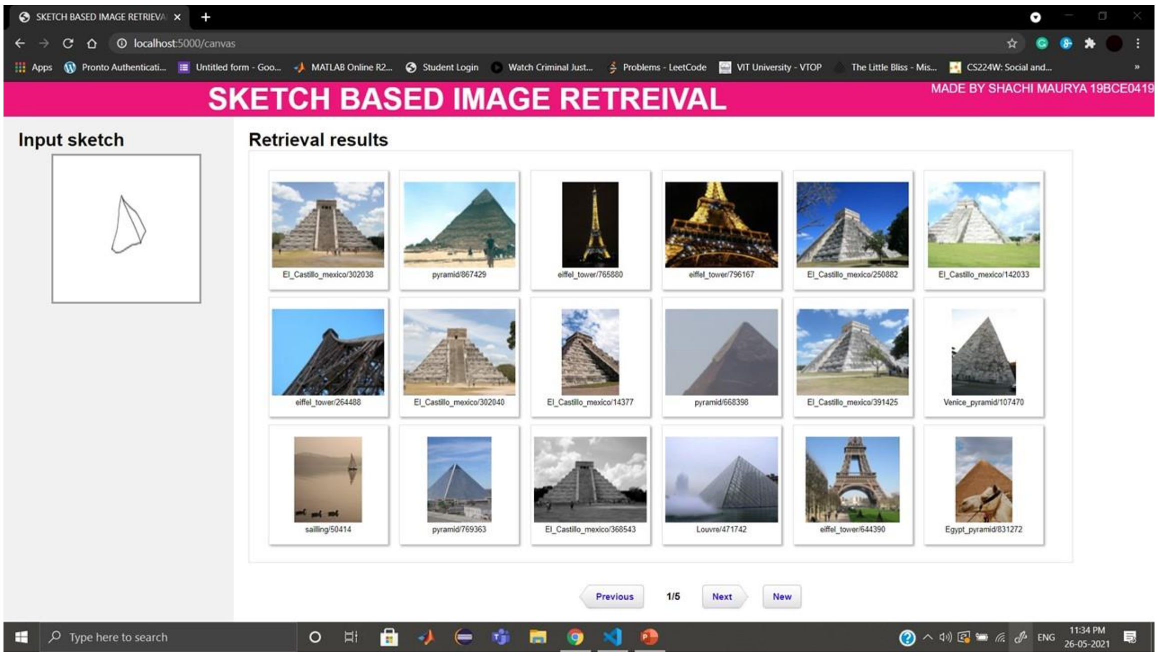

SKETCH BASED IMAGE RETREIVAL
localhost:5000/canvas
Apps
Pronto Authenticati...
Untitled form - Goo...
MATLAB Online R2...
Student Login
Watch Criminal Just...
Problems - LeetCode
VIT University - VTOP
The Little Bliss - Mis...
CS224W: Social and...
SKETCH BASED IMAGE RETREIVAL
MADE BY SHACHI MAURYA 19BCE0419
Input sketch
Retrieval results
Pisa_tower/180657
Pisa_tower/16846
Pisa_tower/358640
Pisa_tower/184240
Pisa_tower/917504
Pisa_tower/97100
Pisa_tower/927800
Pisa_tower/369316
Pisa_tower/70442
Pisa_tower/344042
Pisa_tower/336216
Pisa_tower/911901
Pisa_tower/911251
Pisa_tower/69297
Pisa_tower/358192
Pisa_tower/1007075
Pisa_tower/369795
Pisa_tower/1052071
Previous
1/5
Next
New
Type here to search
ENG
11:28 PM
26-05-2021

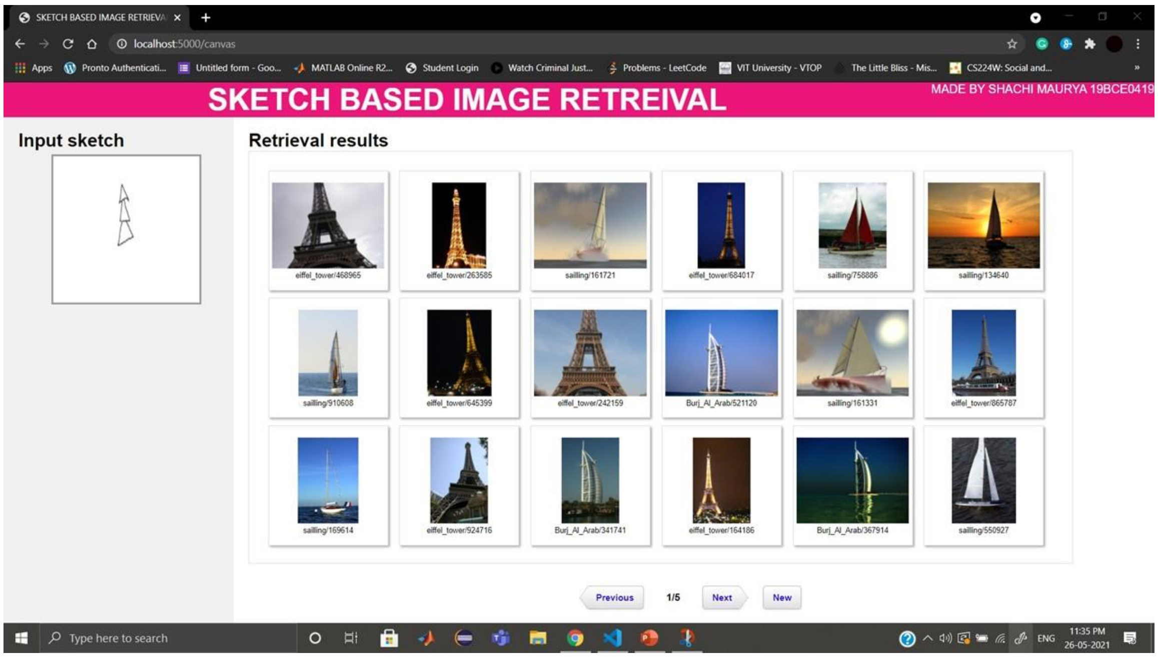

SKETCH BASED IMAGE RETRIEVAL
localhost:5000/canvas
SKETCH BASED IMAGE RETREIVAL
MADE BY SHACHI MAURYA 19BCE0419
Input sketch
Retrieval results
sailling/131439
big_ben/649027
big_ben/938387
big_ben/716563
Burj_Al_Arab/275504
Burj_Al_Arab/774894
big_ben/815201
big_ben/859653
big_ben/644044
big_ben/883357
big_ben/829539
big_ben/630662
sailling/162129
big_ben/854378
Burj_Al_Arab/376541
big_ben/975467
big_ben/1063073
big_ben/1018955
Previous
1/5
Next
New

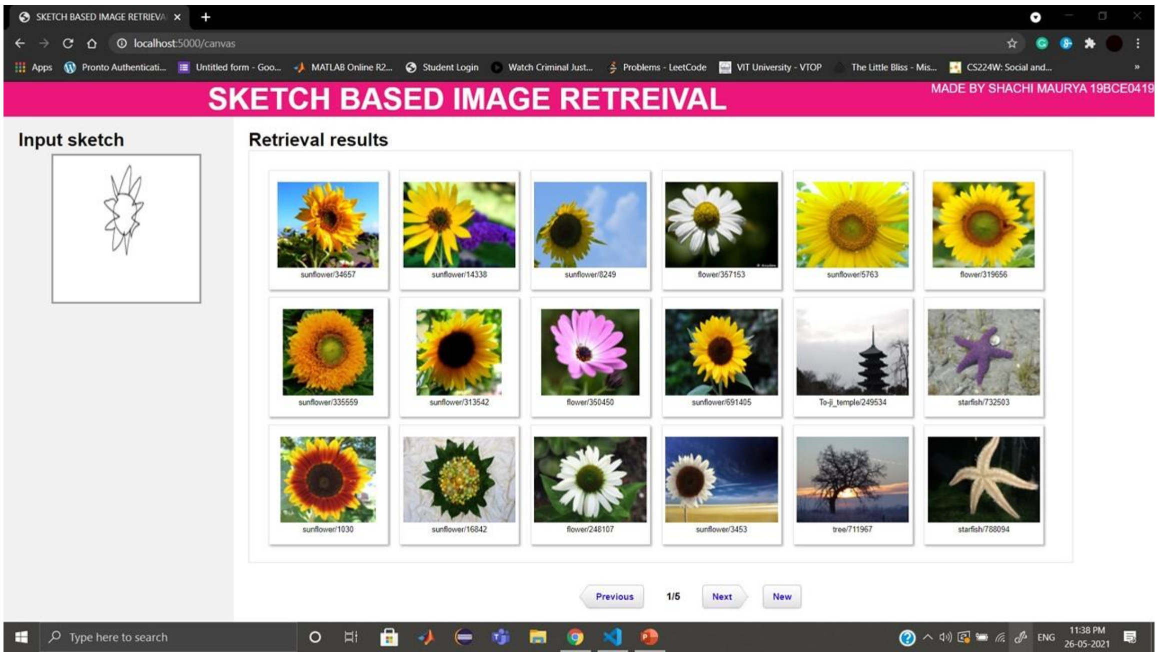

SKETCH BASED IMAGE RETRIEVAL
localhost:5000/canvas
SKETCH BASED IMAGE RETREIVAL
MADE BY SHACHI MAURYA 19BCE0419
Input sketch
Retrieval results
London_eye/788679
sailling/565217
sailling/765427
sailling/205171
moon/126955
sailling/720449
London_eye/494481
London_eye/34862
moon/112177
Burj_Al_Arab/275467
moon/714065
London_eye/1059926
London_eye/266053
London_eye/233308
London_eye/740367
Burj_Al_Arab/825933
London_eye/688144
sailling/166629
Previous
1/5
Next
New

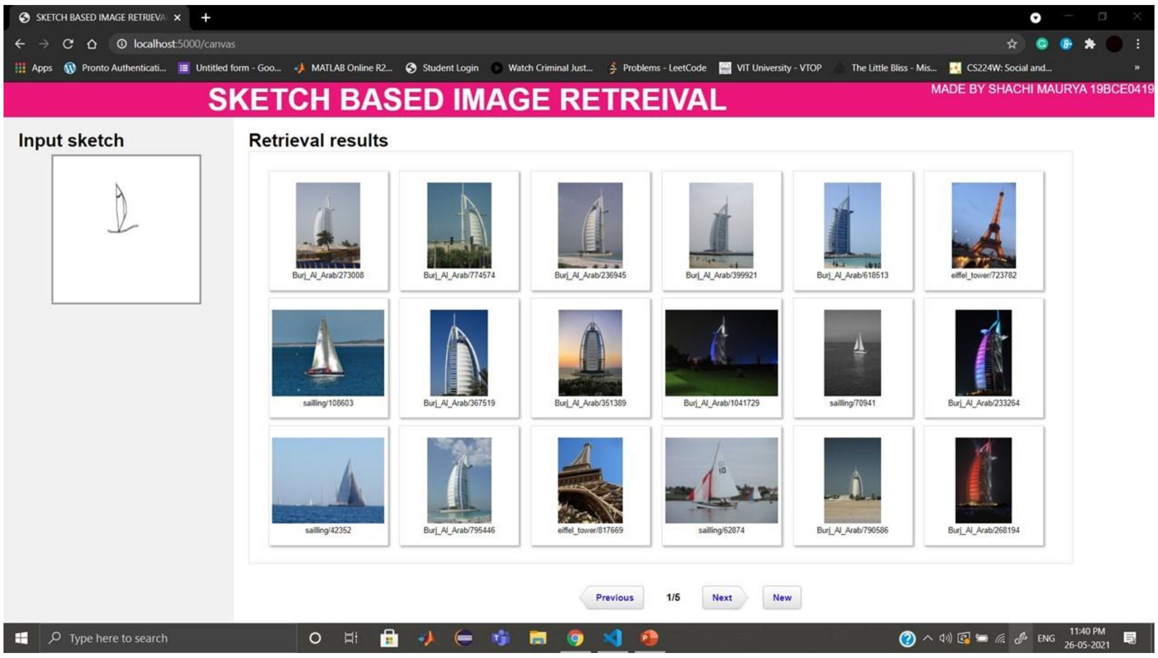

SKETCH BASED IMAGE RETREIVAL
MADE BY SHACHI MAURYA 19BCE0419
Input sketch
Retrieval results
Sydney_Opera/294940
Sydney_Opera/798932
Temple_of_Heaven_beijing/282117
Sydney_Opera/1049588
Sydney_Opera/603645
Sydney_Opera/1030231
Sydney_Opera/623787
Temple_of_Heaven_beijing/72792
Sydney_Opera/660672
Sydney_Opera/792417
Sydney_Opera/785772
Sydney_Opera/597019
Sydney_Opera/162542
Sydney_Opera/1034384
Sydney_Opera/8502
Sydney_Opera/322433
Sydney_Opera/273833
Sydney_Opera/558582
Previous
1/5
Next
New

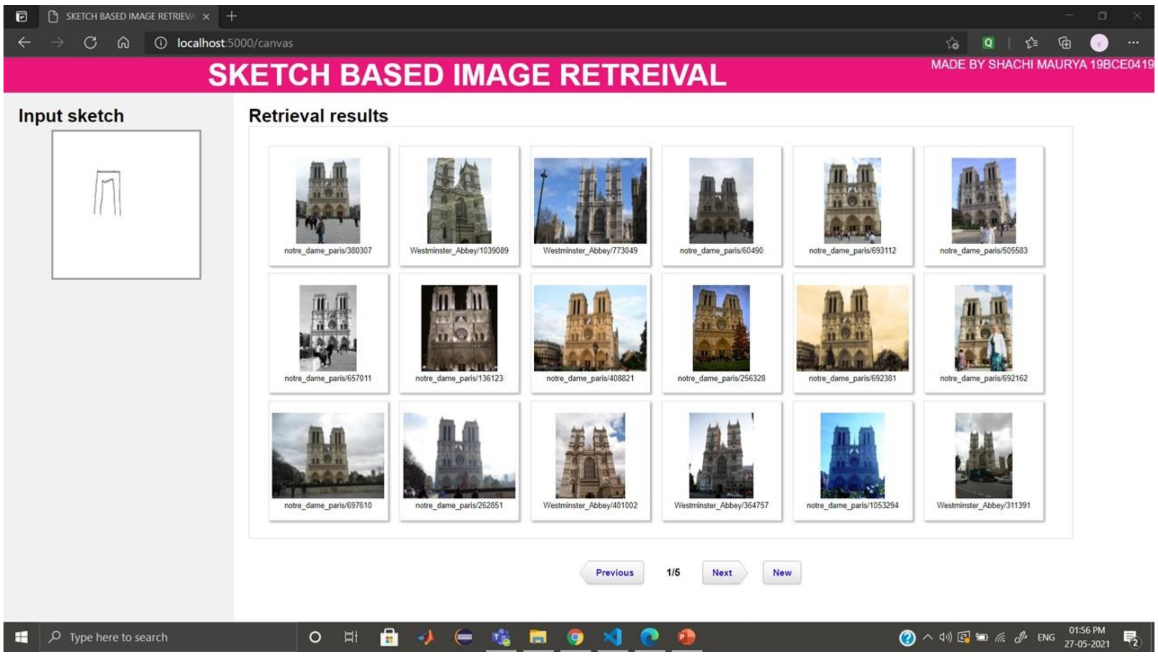

SKETCH BASED IMAGE RETREIVAL
MADE BY SHACHI MAURYA 19BCE0419
Input sketch
Retrieval results
Temple_of_Heaven_beijing/161249
sunset_sunrise/917637
sunset_sunrise/678458
sunset_sunrise/678447
sunset_sunrise/168532
sunset_sunrise/83903
sunset_sunrise/46870
sunset_sunrise/564534
sunset_sunrise/848723
sunset_sunrise/806101
sunset_sunrise/580379
sunset_sunrise/804793
sunset_sunrise/804794
sunset_sunrise/47839
sunset_sunrise/56898
sunset_sunrise/946199
sunset_sunrise/705412
sunset_sunrise/59004
Previous
1/5
Next
New

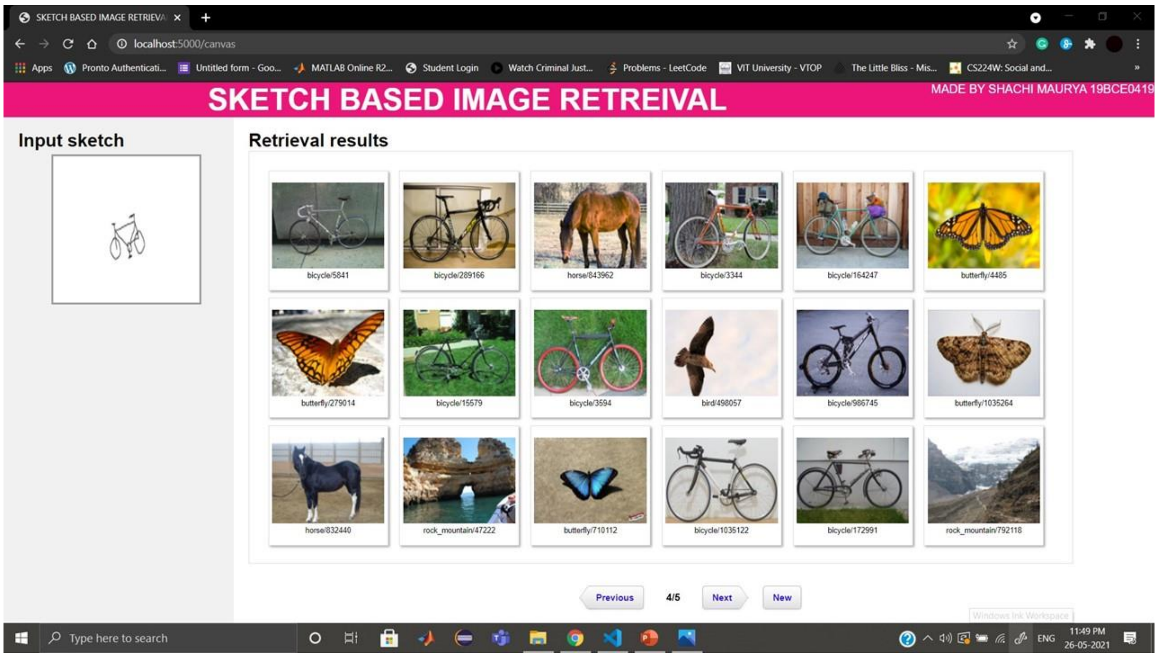

SKETCH BASED IMAGE RETREIVAL
MADE BY SHACHI MAURYA 19BCE0419
Input sketch
Retrieval results
El_Castillo_mexico/302038
pyramid/867429
eiffel_tower/765880
eiffel_tower/796167
El_Castillo_mexico/250882
El_Castillo_mexico/142033
eiffel_tower/264488
El_Castillo_mexico/302040
El_Castillo_mexico/14377
pyramid/668398
El_Castillo_mexico/391425
Venice_pyramid/107470
sailing/50414
pyramid/769363
El_Castillo_mexico/368543
Louvre/471742
eiffel_tower/644390
Egypt_pyramid/831272
Previous
1/5
Next
New

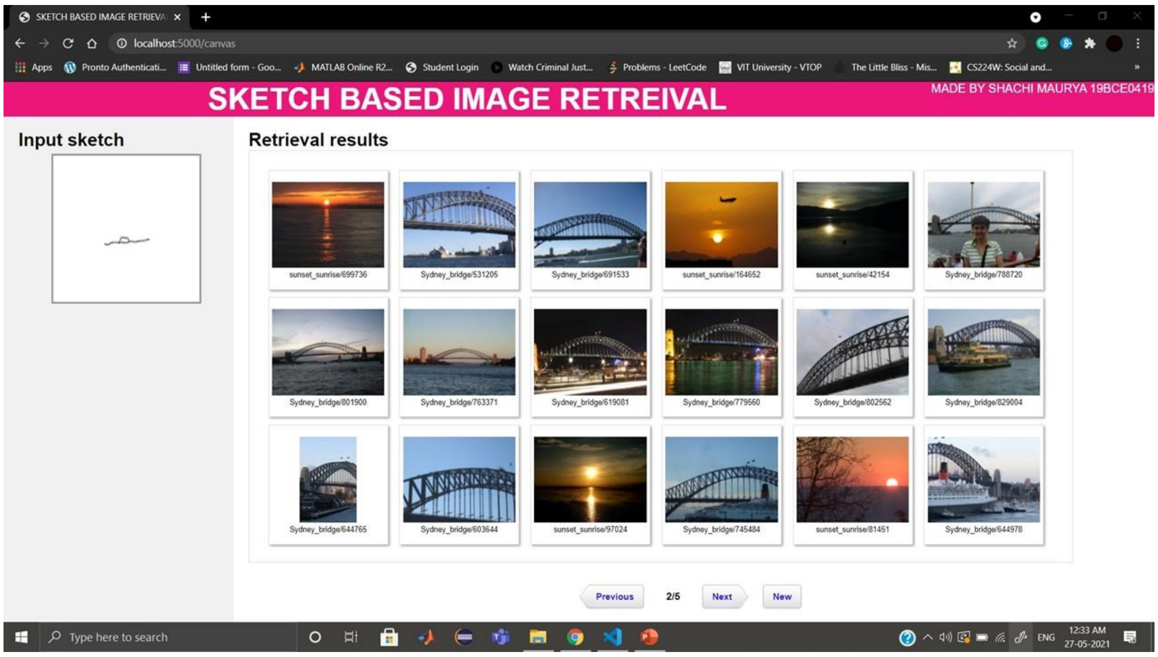

SKETCH BASED IMAGE RETREIVAL
MADE BY SHACHI MAURYA 19BCE0419
Input sketch
Retrieval results
eiffel_tower/468965
eiffel_tower/263585
sailing/161721
eiffel_tower/684017
sailing/758886
sailing/134640
sailing/910608
eiffel_tower/645399
eiffel_tower/242159
Burj_Al_Arab/521120
sailing/161331
eiffel_tower/865787
sailing/169614
eiffel_tower/924716
Burj_Al_Arab/341741
eiffel_tower/164186
Burj_Al_Arab/367914
sailing/550927
Previous
1/5
Next
New


SKETCH BASED IMAGE RETREIVAL
MADE BY SHACHI MAURYA 19BCE0419
Input sketch
Retrieval results
sunflower/34657
sunflower/14338
sunflower/8249
flower/357153
sunflower/5763
flower/319656
sunflower/335559
sunflower/313542
flower/350450
sunflower/691405
To-ji_temple/249534
starfish/732503
sunflower/1030
sunflower/16842
flower/248107
sunflower/3453
tree/711967
starfish/788094
Previous
1/5
Next
New


SKETCH BASED IMAGE RETREIVAL
MADE BY SHACHI MAURYA 19BCE0419
Input sketch
Retrieval results
Burj_Al_Arab/273008
Burj_Al_Arab/774574
Burj_Al_Arab/236945
Burj_Al_Arab/399921
Burj_Al_Arab/618513
eiffel_tower/723782
sailing/108603
Burj_Al_Arab/367519
Burj_Al_Arab/351389
Burj_Al_Arab/1041729
sailing/70941
Burj_Al_Arab/233264
sailing/42352
Burj_Al_Arab/795446
eiffel_tower/817669
sailing/62874
Burj_Al_Arab/790586
Burj_Al_Arab/268194
Previous
1/5
Next
New


SKETCH BASED IMAGE RETREIVAL
localhost:5000/canvas
SKETCH BASED IMAGE RETREIVAL
MADE BY SHACHI MAURYA 19BCE0419
Input sketch
Retrieval results
notre_dame_paris/380307
Westminster_Abbey/1039089
Westminster_Abbey/773049
notre_dame_paris/60490
notre_dame_paris/693112
notre_dame_paris/505583
notre_dame_paris/657011
notre_dame_paris/136123
notre_dame_paris/408821
notre_dame_paris/256328
notre_dame_paris/692381
notre_dame_paris/692162
notre_dame_paris/697610
notre_dame_paris/262851
Westminster_Abbey/401002
Westminster_Abbey/364757
notre_dame_paris/1053294
Westminster_Abbey/311391
Previous
1/5
Next
New
Type here to search
01:56 PM
27-05-2021


SKETCH BASED IMAGE RETREIVAL
localhost:5000/canvas
SKETCH BASED IMAGE RETREIVAL
MADE BY SHACHI MAURYA 19BCE0419
Input sketch
Retrieval results
bicycle/5841
bicycle/289166
horse/843962
bicycle/3344
bicycle/164247
butterfly/4485
butterfly/279014
bicycle/15579
bicycle/3594
bird/498057
bicycle/986745
butterfly/1035264
horse/832440
rock_mountain/47222
butterfly/710112
bicycle/1036122
bicycle/172991
rock_mountain/792118
Previous
4/5
Next
New
Type here to search
11:49 PM
26-05-2021



SKETCH BASED IMAGE RETREIVAL

MADE BY SHACHI MAURYA 19BCE0419

Input sketch

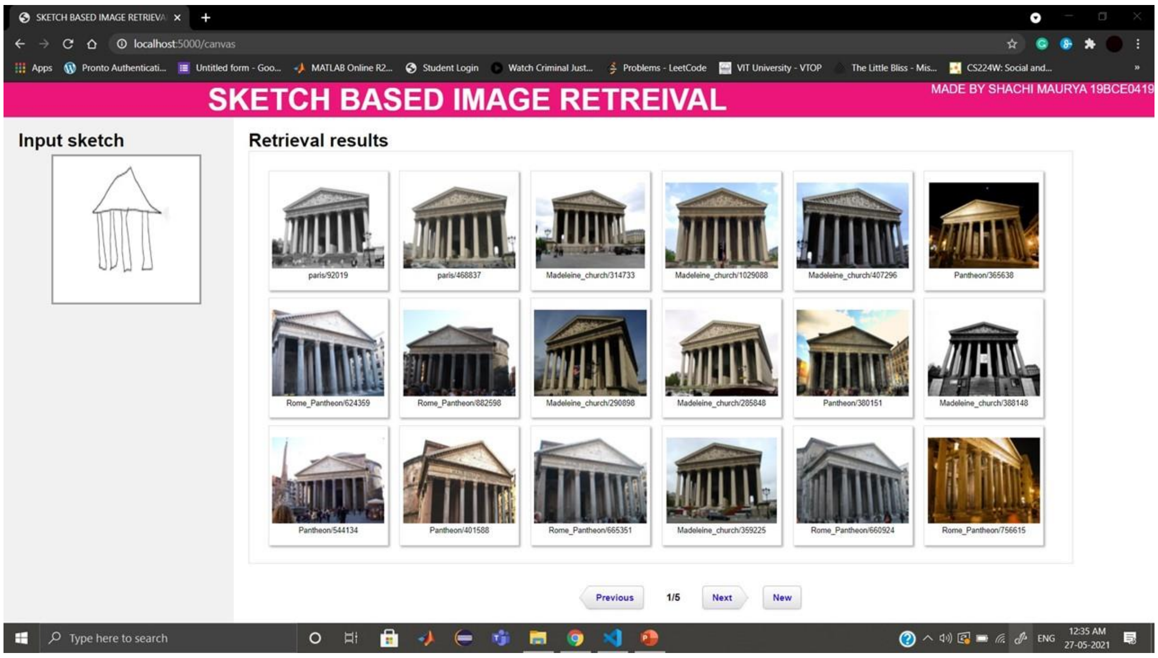
Retrieval results

sunset_sunrise/699736 Sydney_bridge/531205 Sydney_bridge/691533 sunset_sunrise/164652 sunset_sunrise/42154 Sydney_bridge/788720

Sydney_bridge/801900 Sydney_bridge/763371 Sydney_bridge/619081 Sydney_bridge/779560 Sydney_bridge/802562 Sydney_bridge/829004

Sydney_bridge/644765 Sydney_bridge/603644 sunset_sunrise/97024 Sydney_bridge/745484 sunset_sunrise/81451 Sydney_bridge/644978

Previous 2/5 Next New



SKETCH BASED IMAGE RETREIVAL

MADE BY SHACHI MAURYA 19BCE0419

Input sketch

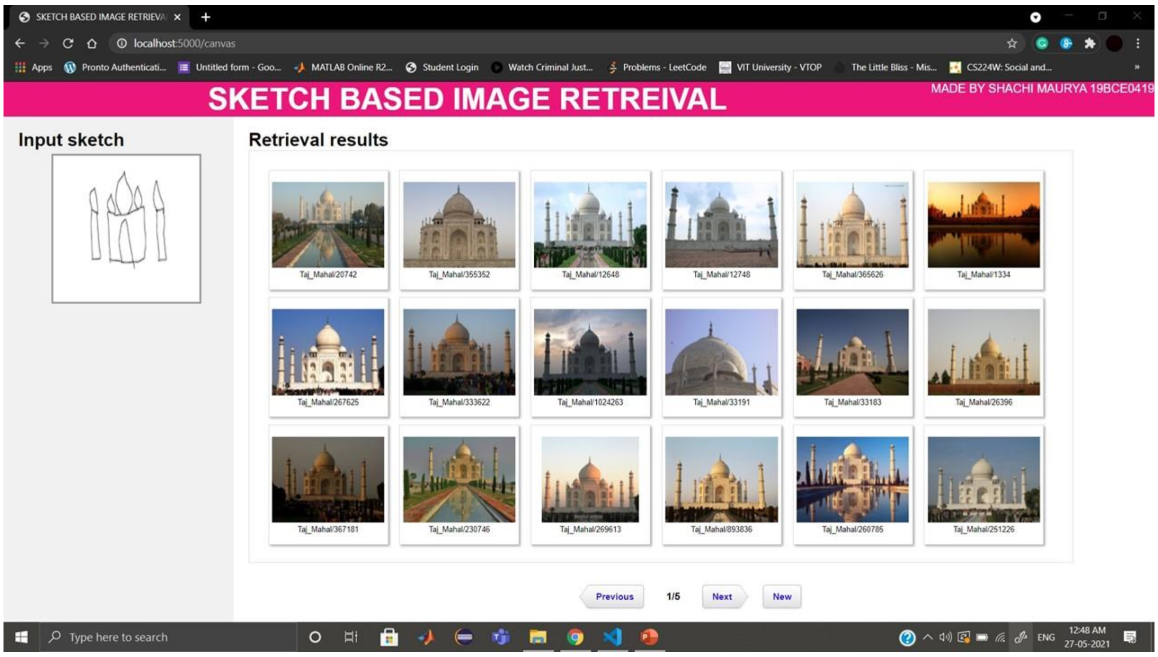
Retrieval results

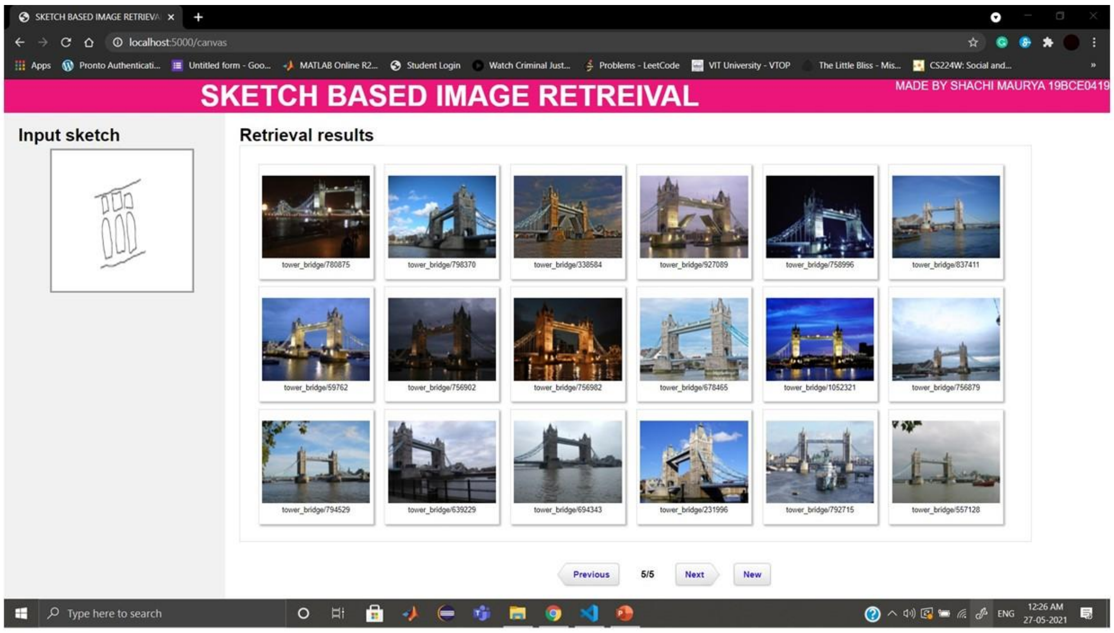
paris/92019 paris/468837 Madeleine_church/314733 Madeleine_church/1029088 Madeleine_church/407296 Pantheon/365638

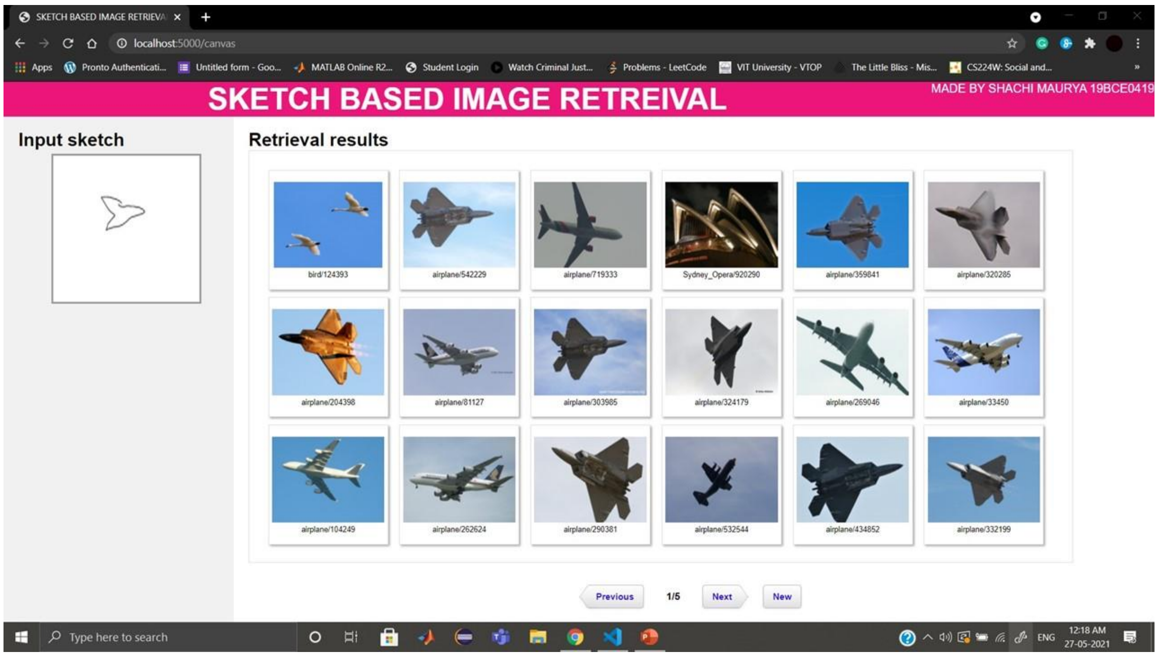
Rome_Pantheon/624359 Rome_Pantheon/882598 Madeleine_church/290898 Madeleine_church/285848 Pantheon/380151 Madeleine_church/388148

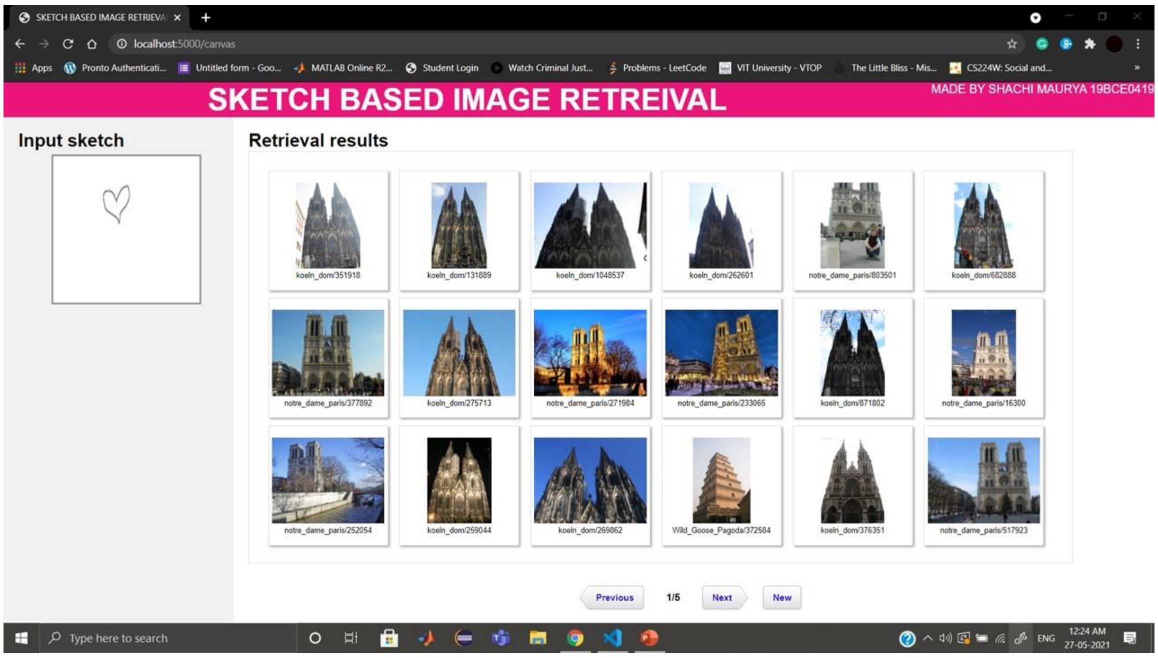
Pantheon/544134 Pantheon/401588 Rome_Pantheon/665351 Madeleine_church/359225 Rome_Pantheon/660924 Rome_Pantheon/756615

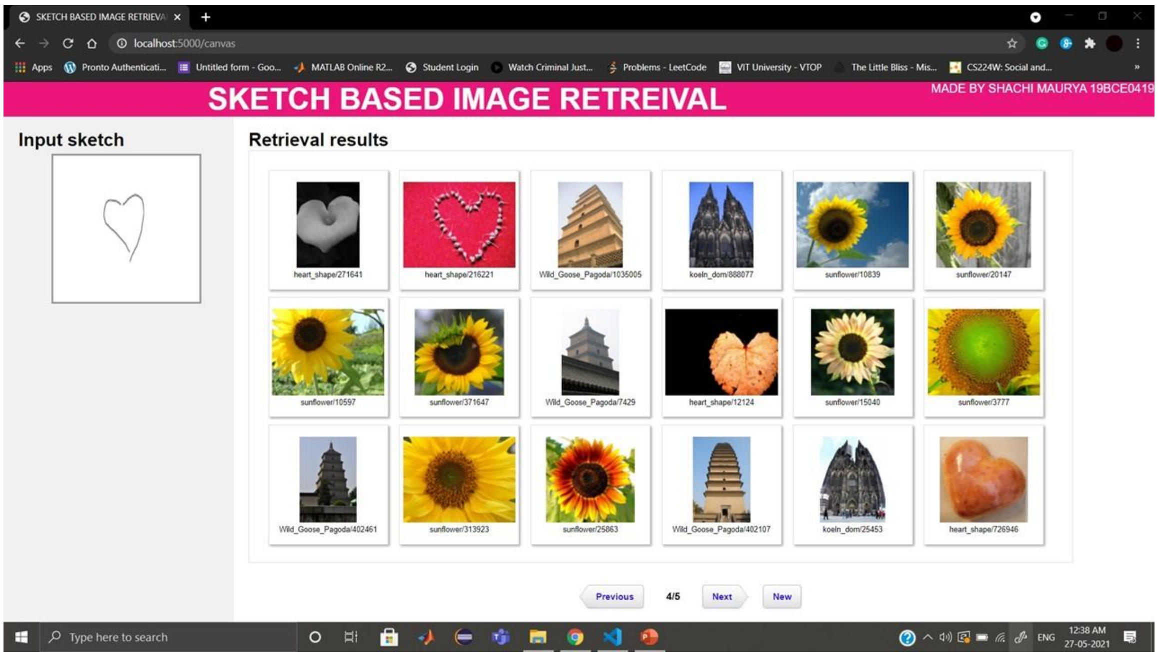
Previous 1/5 Next New


SKETCH BASED IMAGE RETRIEVAL
localhost:5000/canvas
SKETCH BASED IMAGE RETREIVAL
MADE BY SHACHI MAURYA 19BCE0419
Input sketch
Retrieval results
Taj_Mahal/20742
Taj_Mahal/355352
Taj_Mahal/12648
Taj_Mahal/12748
Taj_Mahal/365626
Taj_Mahal/1334
Taj_Mahal/267625
Taj_Mahal/333622
Taj_Mahal/1024263
Taj_Mahal/33191
Taj_Mahal/33183
Taj_Mahal/26396
Taj_Mahal/367181
Taj_Mahal/230746
Taj_Mahal/269613
Taj_Mahal/893836
Taj_Mahal/260785
Taj_Mahal/251226
Previous
1/5
Next
New


SKETCH BASED IMAGE RETRIEVAL
localhost:5000/canvas
SKETCH BASED IMAGE RETREIVAL
MADE BY SHACHI MAURYA 19BCE0419
Input sketch
Retrieval results
tower_bridge/780875
tower_bridge/798370
tower_bridge/338584
tower_bridge/927089
tower_bridge/758996
tower_bridge/837411
tower_bridge/59762
tower_bridge/756902
tower_bridge/756982
tower_bridge/678465
tower_bridge/1052321
tower_bridge/756879
tower_bridge/794529
tower_bridge/639229
tower_bridge/694343
tower_bridge/231996
tower_bridge/792715
tower_bridge/557128
Previous
5/5
Next
New


SKETCH BASED IMAGE RETREIVAL
MADE BY SHACHI MAURYA 19BCE0419
Input sketch
Retrieval results
bird/124393
airplane/542229
airplane/719333
Sydney_Opera/920290
airplane/359841
airplane/320285
airplane/204398
airplane/81127
airplane/303985
airplane/324179
airplane/269046
airplane/33450
airplane/104249
airplane/262624
airplane/290381
airplane/532544
airplane/434852
airplane/332199
Previous
1/5
Next
New


SKETCH BASED IMAGE RETREIVAL
MADE BY SHACHI MAURYA 19BCE0419
Input sketch
Retrieval results
koeln_dom/351918
koeln_dom/131889
koeln_dom/1048537
koeln_dom/262601
notre_dame_paris/803501
koeln_dom/682888
notre_dame_paris/377892
koeln_dom/275713
notre_dame_paris/271984
notre_dame_paris/233065
koeln_dom/871802
notre_dame_paris/16300
notre_dame_paris/252054
koeln_dom/259044
koeln_dom/269862
Wild_Goose_Pagoda/372584
koeln_dom/376351
notre_dame_paris/517923
Previous
1/5
Next
New



5 Conclusion

The SBIR is an interesting topic and one of the most popular researches still going on in CBIR . The project is just a representation of the findings that have been done so far in CBIR and illustrates the whole process . It is seen that more image in dataset helps to increase the rate of accurate retrievals using query image because we are using CNN and Triplet loss that require training which improves with large dataset images. The half shared weights are used instead of full because we are trying to get any image that is partially close to the query as our query image is a sketch drawn free- handed. The anchor image is the query sketch image while the positive is the one from the query image category and the negative is not from the query image category. This way triplet loss is very helpful in training the images on spot for every sketch.

.

6. References

Online Reference

1. https://github.com/zhoushuozh/drawingborad
2. https://github.com/TuBui/Triplet_Loss_SBIR

Journal Reference:

1. TuBui, Leonardo Sampaio Ferraz Ribeiro, M. Ponti , John Collomosse,” Generalisation and Sharing in Triplet Convnets for Sketch based Visual Search”, Computer Vision and Image Understanding; Center of Vision ,Speech and Signal Processing ;Institute of Mathematical and Computer Sciences, 2016

2. TuBui, L. Ribeiro , M. Ponti , John Collomosse,”Compact descriptors for sketch-based image retrieval using a triplet loss convolutional neural network “Computer Vision and Image Understanding; Center of Vision ,Speech and Signal Processing ;Institute of Mathematical and Computer Sciences, 2017

3. Yi Li ·,Wenzhao Li,” Doodle to Search: Practical Zero-Shot Sketch-based Image Retrieval”, Springer-Verlag GmbH Germany, part of Springer Nature 2018, 2018

4. M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, Query by image and video content: The qbic system, Computer 28 (9) (1995) 23–32.

5. T. Kato, Database architecture for content-based image retrieval, in: SPIEIS&T 1992 Symposium on Electronic Imaging: Science and Technology, International Society for Optics and Photonics, 1992, pp. 112–123

6. W. Fang, K. Le, and L. Yi. Sketch-based 3d shape retrieval using convolutional neural networks. arXiv preprint arXiv:1504.03504, 2015

7. L. Yi, H. Timothy, S. Yi-Zhe, and G. Shaogang. Fine-grained sketch-based image retrieval by matching deformable part models. Proceedings of the British Machine Vision Conference, 2014.

8. Xiaochun, Z. Hua, L. Si, G. Xiaojie, and L. Liang. Symfish: A symmetry-aware flip invariant sketch histogram shape descriptor. IEEE International Conference on Computer Vision, 2013.

9. R. Hu and J. Collomosse. A performance evaluation of gradient field hog descriptor for sketch based image retrieval. Computer Vision Image Understanding, 117(7):790–806, 2013.

10. J. M. Saavedra and J. M. Barrios. Sketch-based image retrieval using learned keyshapes (lks). In BMVC, 2015