Sketch Based Image Retrieval using Deep Learning Based Machine Learning

Deepika Sivasankaran, Sai Seena P, Rajesh R, Madheswari Kanmani

***Abstract**: Sketch based image retrieval (SBIR) is a sub-domain of Content Based Image Retrieval(CBIR) where the user provides a drawing as an input to obtain i.e retrieve images relevant to the drawing given. The main challenge in SBIR is the subjectivity of the drawings drawn by the user as it entirely relies on the user's ability to express information in hand-drawn form. Since many of the SBIR models created aim at using singular input sketch and retrieving photos based on the given single sketch input, our project aims to enable detection and extraction of multiple sketches given together as a single input sketch image. The features are extracted from individual sketches obtained using deep learning architectures such as VGG16 , and classified to its type based on supervised machine learning using Support Vector Machines. Based on the class obtained, photos are retrieved from the database using an opencv library, CVLib , which finds the objects present in a photo image. From the number of components obtained in each photo, a ranking function is performed to rank the retrieved photos, which are then displayed to the user starting from the highest order of ranking up to the least. The system consisting of VGG16 and SVM provides 89% accuracy*

Keywords: Sketch Based Image Retrieval, Machine Learning, Deep Learning, Content Based Image Retrieval

I. INTRODUCTION

To obtain information about something, it is required to be able to express it unambiguously and with relevance. Verbal descriptions, pictographs and sketches are one of the few ways to describe one's need for relevant information. With the advent of technology, many aim to gain knowledge through information remotely, by using search engines. This explicit description about the object is also used in search engines to get relevant results in the form of images, websites, videos etc. Information retrieval is a domain of interest to many problem solvers where the main difficulty lies in the ability to bridge the gap between information available and the query given by the user. Content based image retrieval is a sub-domain under information retrieval

Manuscript received on May 15, 2021.
Revised Manuscript received on May 22, 2021.
Manuscript published on June 30, 2021.
* Corresponding Author

Deepika Sivasankaran*, Department of Computer Science and Engineering, Sri Sivasubramaniya Nadar College of Engineering, Chennai, India. Email: deepika17033@cse.ssn.edu.in

Sai Seena P, Department of Computer Science and Engineering, Sri Sivasubramaniya Nadar College of Engineering, Chennai, India. Email: saiseena174322@cse.ssn.edu.in

Rajesh R, Department of Computer Science and Engineering, Sri Sivasubramaniya Nadar College of Engineering, Chennai, India. Email: rajesh17121@cse.ssn.edu.in

Madheswari Kanmani, Department of Computer Science and Engineering, Sri Sivasubramaniya Nadar College of Engineering, Chennai, India. Email: MadheswariK@ssn.edu.in

Relevance to the image. In Sketch Based Image Retrieval where, to obtain images, the user describes the images to retrieve in text form or as a drawing such. In query by text, the images are mapped to respective tags that are of (SBIR) [1], a sketch or a drawing is the input given to get images. For example, a drawing of an apple retrieves images of apples. These drawings can be professional, free-hand sketches, symbols and pictures defined by edges. A hand drawn sketch is abstract with roughly drawn shapes and may not contain edges and features as sharply defined in the photos. Thus relevance in the images retrieved depends on how efficiently a model can relate the abstract features of sketches to the photos. The ability to retrieve images heavily depends on the user's subjectivity in expressing them, thus increasing the need to obtain a dataset containing images with both heavy and light levels of abstractions in terms of features.

II. LITERATURE SURVEY

Ayan Kumar Bhunia, Yongxin Yang, Timothy M. Hospedales et al address the main concern involving the user's ability to draw a complete sketch. A sketch is already a product of a user's subjectivity and ability in expressing a pictographic content. Thus completion of a diagram also depends on such a user's skills in drawing a sketch. Hence the technique used in [2] is a reinforcement learning-based cross-modal retrieval framework that learns from the strokes provided and identifies the class to retrieve images. These strokes are compared with photos to obtain images containing similar stroke patterns, or edges.This enabled for faster retrieval of images compared to the time taken by conventional SBIR which involves complete sketching of a query

Most SBIR systems use singular sketches as input and retrieve images. Thus, to retrieve images containing many different classes of sketch, one cannot send a single sketch image containing various sketches belonging to different classes. Hence in [3], Sounak Dey, Anjan Dutta et al addressed this where text and sketches are used for singular sketch input image and a combination of 2 texts and 2 sketches are used to provide input for multiple classes. Thus a cross-modal deep network architecture is used to learn common embedding between text and images and between sketches and images. An attention model in included to detect the individual multiple input objects in the query on the photo dataset

SBIR models are created by training them with extensivedatasets for different classes. Training a model with new classes everytime is not a practical and feasible process. Sasi KiranYelamarthi et al [4] proposed a novel generative model, where the model can classify classes that are not seen during training. Instead of trying to fit within the existing classes seen during training, they are considered novel and generalized for further associations This is described as the Zero-Shot (ZS) framework where a model is able to classify

an image from a novel class that was not used during the training process, giving the name zero for the number of examples used in that class.

Sketch Based Image Retrieval (SBIR) is a challenging domain mainly due the ambiguity and the abstraction in the sketches. A novel convolutional neural network based on the Siamese network for SBIR is proposed by Yonggang Qi, Yi-Zhe Song, Honggang Zhang, Jun Liu in [5]. By linking two Convolution Neural Networks (CNN), the triplets contain the class label, image and a boolean variable denoting True, if the image belongs to the same class or False, if different. The idea is to bridge the gap between such similar and dissimilar classes sketches such that it can work on novel data as well

Sounak Dey, Pau Riba et al addressed the zero shot learning process by first suggesting a novel ZS-SBIR dataset [6], QuickDraw-Extended, that consists of 330,000 sketches and 204,000 photos spanning across 110 categories was contributed. Highly abstract amateur human sketches are purposefully sourced to maximize the domain gap, instead of ones included in existing datasets that can often be semi-photo realistic. A ZS-SBIR framework was used to relate the sketch and the photo classes.

III. EXPERIMENTAL SETUP

Sketch Based Image Retrieval can be done in 2 ways:

- Class based image retrieval - Conventional method
- Sketch based image retrieval

Class based retrieval involves the training the model to classify the input sketch and retrieving the images from class-wise separated photo dataset.

In sketch based retrieval, the photo dataset is not classified, rather all images are put together. Here we use sketch based retrieval of images to obtain the photos from our dataset. The relevant images are retrieved using an object detection library CVlib [8]

A. Algorithm of Proposed System:

INPUT: Sketch containing various drawings
OUTPUT: Photos in ranked order

Step 1: An input sketch containing one or more drawings is given to the system
Step 2: The various components are identified individually using contours and bounding boxes are drawn around each of them.
Step 3: The individual sketch images are cropped and saved separately
Step 4: Each of the cropped image is sent to a classifier system and its class is identified
Step 5: Using CVLib, bounding boxes and labels the classes obtained from the classifier system are found
Step 6: Based on the number of classes contained in each photo, they are ranked.
Step 7: The photos are displayed in order of ranking.

B. Workflow of the Proposed System:

The architecture diagram of the system is given in Fig 1. The modules in the system are explained below:

1. Detection of individual sketches:

Here the input image containing sketches is given and using contours function, the various individual contours are obtained. These contours correspond to individual sketch components. From the contours obtained, left, right and top, bottom coordinates are extracted. Bounding boxes are drawn around each of these sketch components and saved separately.

2. Classification of the Sketch:

The separate sketches are loaded individually and their class is predicted.

3. Detection of Components in the Images:

Using CVLib, various objects detected in the photos are obtained. This contains a list of classes that may also be irrelevant to the query. So the classes of the query sketches are checked and bounding box coordinates of only such labels (query classes) are retained

4. Ranking:

A photo may contain one or more of the query sketch classes, Thus ranking is performed to order photos based on the number of query sketches found, If a photo contains all query classes, it is given rank 1. Images containing none of the query classes are discarded.

Dataset Used:
The dataset contains two parts:
1. Sketches - Drawings of various levels of abstractions
2. Photos - A dataset of photos to obtain from based on input query and order by relevance.

1. Sketch Dataset:

The sketches are obtained from three sources namely, QuickDraw! [10], TU-Berlin [11] and Sketchy [9]. Each of these sketch datasets contain 110, 251 and 125 classes respectively and the sketches given in each of the dataset are of varying complexities.
The number of sketch images with size of individual images in each of the dataset is given in Table- I.

Table -I :Images per class in each dataset

SNO	DATASET	IMAGES PER CLASS	IMAGE SIZE (pixels)
1	QuickDraw!	3001	256x256x3
2	TU-Berlin	80	1111x1111x3
3	Sketchy	500 to 800	256x256x3

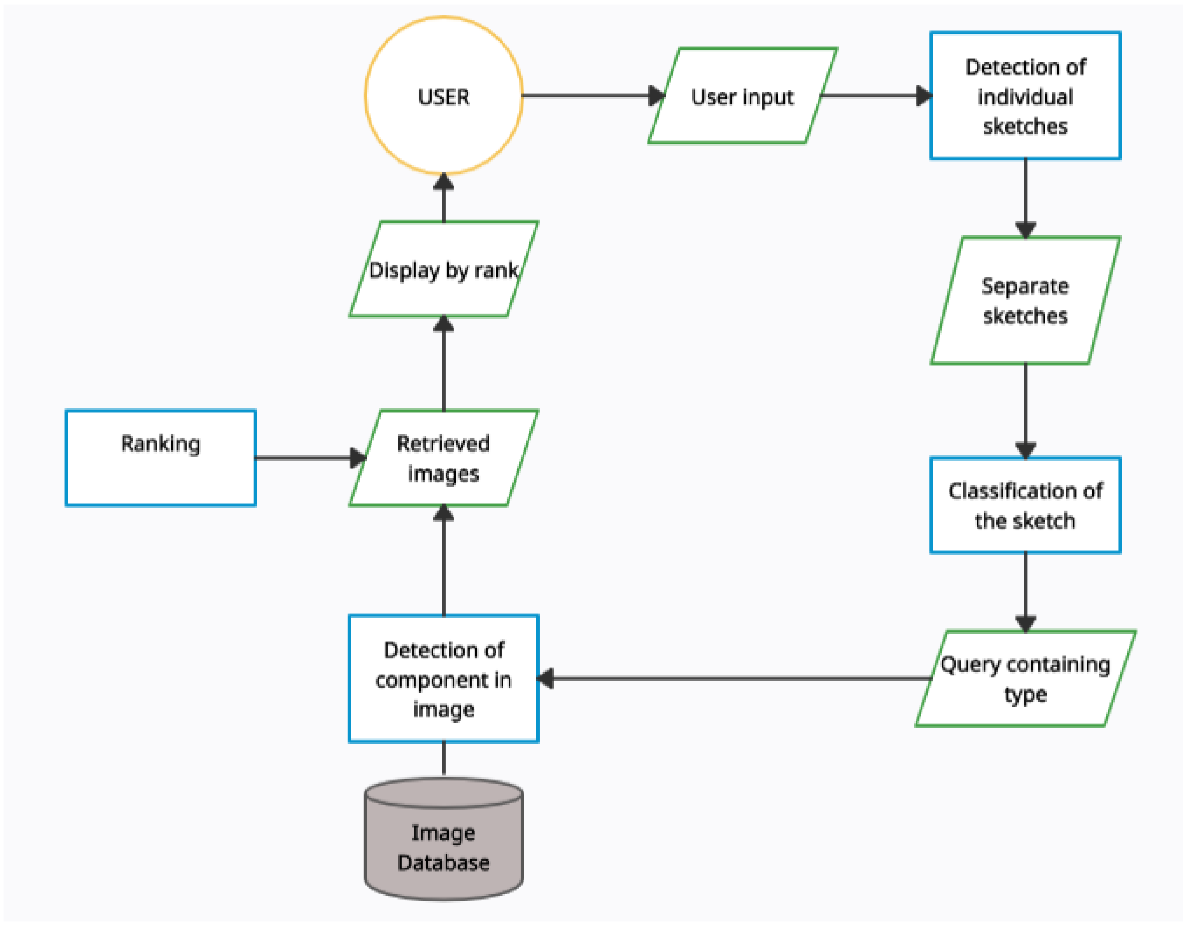


Fig. 1. Architecture diagram of the Proposed System

Fig 2 - 4 show some sketch images of the class apple from each of the three datasets
The level of abstraction between sketches of the same class from different datasets can be observed.

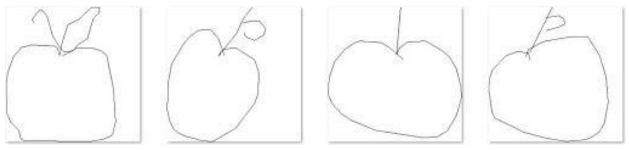


Fig. 2. QuickDraw! dataset

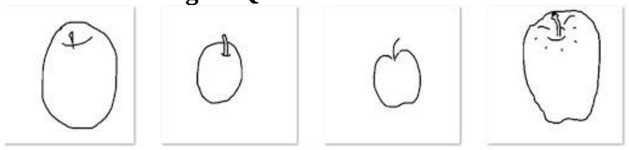


Fig. 3. Sketchy dataset

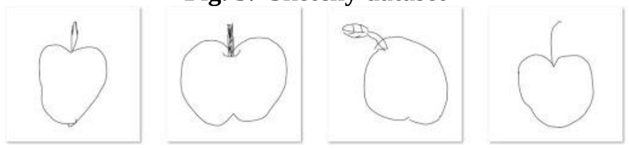


Fig. 4. TU-Berlin dataset

2. Photo dataset:

Sketch datasets QuickDraw! and Sketchy include sketches as well as photos. Since most photos contain utmost 2 components, a simple database was created such that photos contained more than 3 components. This was done by randomly selecting 2 images from each class and then concatenating them with other images of different classes. The combinations used are 4 photos, 3 photos and 2 photos, Fig. 5 to 7. These images are concatenated using python and an online software, Peko-Step[12]

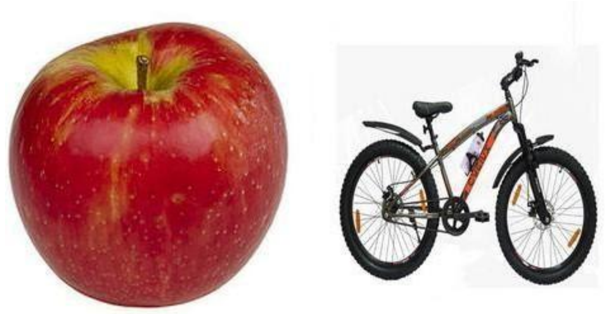


Fig. 5. Photo containing 2 classes

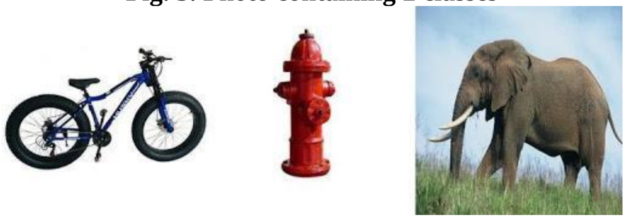


Fig. 6. Photo containing 3 classes

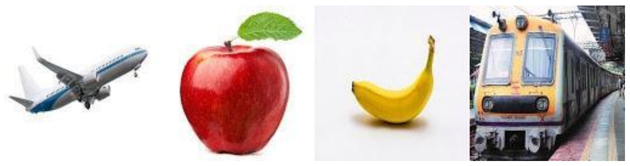


Fig. 7. Photo containing 4 classes

IV. METHODOLOGY

A. Detection of Individual Sketches:

The sketch dataset taken for this project is QuickDraw. It contains a total of 110 classes, of which 15 classes are chosen. The 15 classes are: Airplane, apple, banana, bicycle, bird, car, cat, chair, cup, elephant, fire hydrant, knife, pizza, teddy bear, train. The sketches in this dataset are in binary format. ie, black and white thus each pixel contains values from 0 to 255 where 0 indicates black, 255 indicates white and the intermediate values denote the grayscale range. An input image is sent to the system to detect the various individual sketches present in it. An input image (Fig. 8) is sent to the system to detect the various individual sketches present in it.

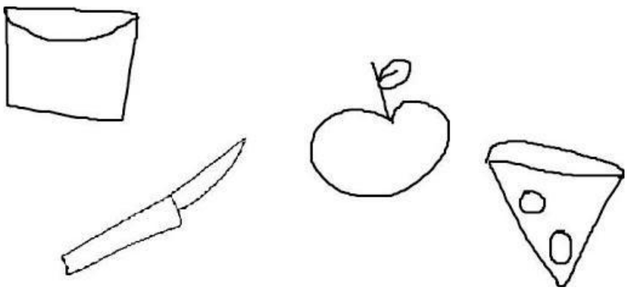


Fig. 8. Input image containing various sketches

B. Finding Individual Sketches:

From this input image, the individual sketches are obtained by finding the contours of every single sketch. A contour can be defined as a curve or a line of continuous points containing pixels of the same colour or intensity. These contours can be extracted using various edge detection algorithms such as Canny [17] and Sobel [18]. Here the OpenCV function findContours [19] is used to obtain the contours. To make use of this function, first the images are converted to grayscale and then a threshold function is applied. This threshold function sets the pixels values above a limit to the said maximum value. The thresholding done here converts pixel values greater than 127 to 255, where 127 is the lowest shade of grey that can be obtained before its transition to white. The thresholded image is inverted and then sent to the findContours function which returns us arrays containing contours of every individual sketch.

C. Drawing Bounding Boxes:

To obtain the individual sketch components from the input image, a bounding box is drawn. This can be drawn using the function, cv2.rectangle(), which requires the top left and the bottom right coordinates. These coordinates can be obtained from the contours array by finding the minimum and maximum values in the first column for the left and right coordinates and in the second column for the top and bottom coordinates. Thus from the coordinates a bounding box is drawn, Fig. 9, adding 10 extra pixel values to each coordinate and it is displayed to the user. To crop individual sketches, the coordinates obtained before the addition of the 10 pixel units are taken and individual images are snipped and saved on the local system.

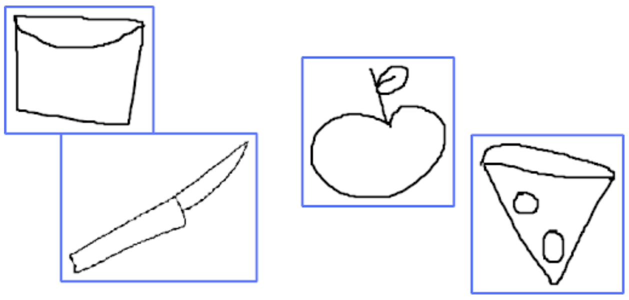


Fig. 9. Drawing bounding boxes around the individual sketches

The individual sketches are extracted and saved separately as given in Fig. 10 to Fig. 13.

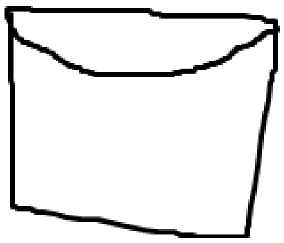


Fig. 10. Individual sketch – I

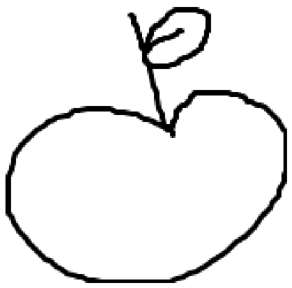


Fig. 11. Individual sketch – II

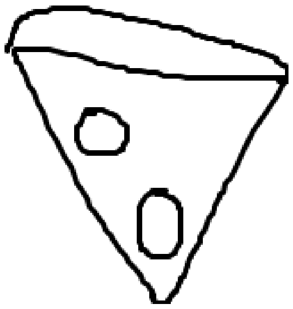


Fig. 12. Individual sketch – III

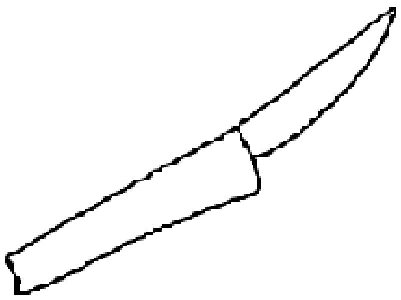


Fig. 13. Individual sketch - IV

D. Classifier Model for Sketch Classification:

Two methods were implemented to create the classifier. The first method uses purely deep learning for the classifier and the second method combines both deep learning and machine.

1. **Deep learning classifier**:

The total number of sketch images used in this dataset is 45015 images, which are divided to 38250 training images, 4500 testing images and 2265 validation images.

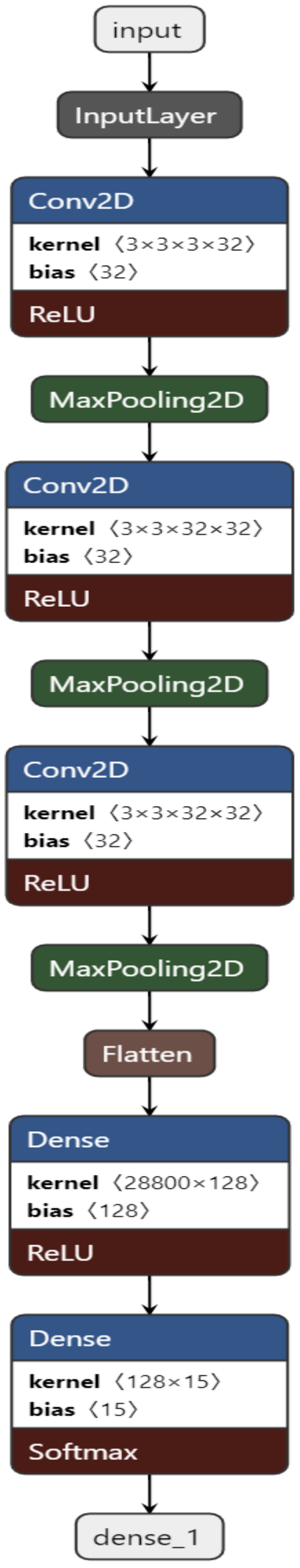


Fig. 14. Architecture of simple CNN

The images are resized to 100*100 for convenience. The model initially created for the classifier incorporated a deep learning approach using Convolution Neural Networks. The architecture of the CNN is given in Fig. 14The total number of layers in the CNN created is 10. This method proves to be inefficient due to the extreme loss in data. Hence a second method was proposed which combines both machine learning and deep learning.

2. Deep Learning And Machine Learning Combined:

Due to the extreme loss in data features obtained in the convolution neural network, an alternate method was proposed which included the feature extraction from sketches using deep learning architectures such as VGG16 and InceptionV3 and a basic CNN as well. The basic Convolution Neural Network contains 6 layers where the flattening and dense layers are eliminated because of the need for only features and not a classification from the CNN. The feature vectors obtained were then used to train a Support Vector Machine (SVM) to classify the various classes. The pretrained models were loaded with imagenet weights. From the classification models created using features from the pretrained architectures and the simple CNN, it was observed that the sketch classifier is more efficient with neural networks and architectures containing less number of layers. Hence VGG16 proved to work better than InceptionV3. Sample input and output for the novel classifier system is given in Fig. 15.

```
Predicted Label:
fire hydrant
Actual Label:
Fire_Hydrant
```

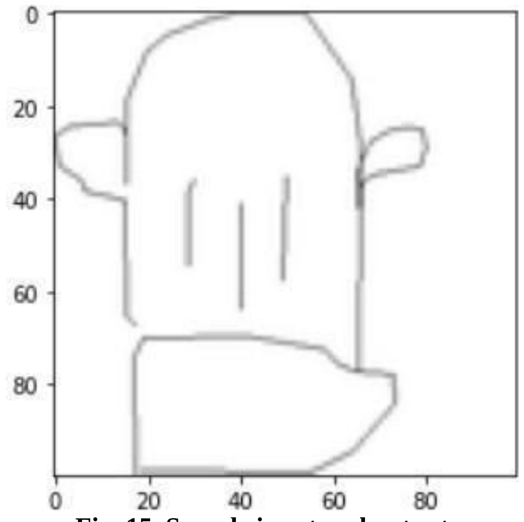


Fig. 15. Sample input and output

E. Object Detection in Photos:

The OpenCV library CVlib [8] is used to detect objects in an image. The library is trained on COCO dataset [24] and uses the pre-trained YOLO v3 weights. The dataset contains 80 classes of different objects. CVlib can performface detection, gender detection and object detection.

Herethe object detection function is used to detect multiple objects in a photo. Given a photo, cvlib's Detect Common Object function detects all objects seen in the photo, which

includes redundant objects as well. The function returns bounding boxes coordinates, labels and confidence scores of the objects. Hence the labels, bounding box coordinates and confidence scores of the objects that are not a part of the input query are removed. Consider an input photo read from the database

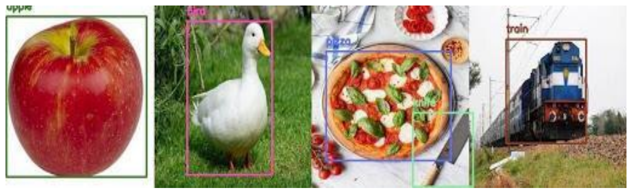


Fig. 16. CVLib object detection performed on a photo

The detect_common_objects function detected all objects in the photo, Fig. 16. Consider a query: (knife,pizza)

Since the photo contains detection for objects other than the queries, the bounding boxes and labels for those classes other than query are eliminated, thus giving Fig. 17

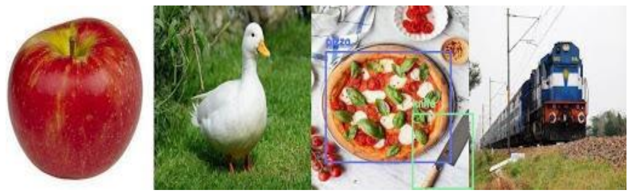


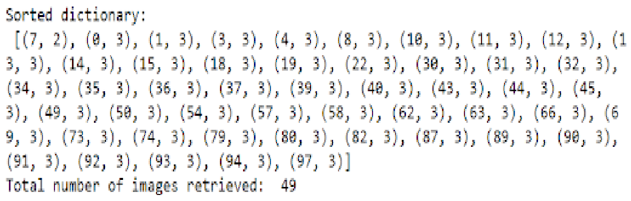
Fig. 17. Bounding boxes only for query objects

F. Ranking and Retrieval:

After filtering photos containing only components in the query, they must be ranked according to relevance. Relevance, here is a measure that counts the number of query objects found in the photos counting different instances (objects) of the same class as 1. Based on the number of query objects, given in the input sketch, found in each of the photos, they are given a rank. Rank=1 is given to photos containing all query objects, rank=2 is given to photos missing at-most 1 query object and rank=3 is given for all other images.

NOTE: Images that do not contain any of the query objects are discarded in the retrieval process.

The tuple in the sorted dictionary contains images in the ranked order. Consider a tuple from the dictionary (7,2) in Fig. 18. Here 7 is the photo number and 2 is the rank.



```
Sorted dictionary:
 [(7, 2), (0, 3), (1, 3), (3, 3), (4, 3), (8, 3), (10, 3), (11, 3), (12, 3), (1
3, 3), (14, 3), (15, 3), (18, 3), (19, 3), (22, 3), (30, 3), (31, 3), (32, 3),
(34, 3), (35, 3), (36, 3), (37, 3), (39, 3), (40, 3), (43, 3), (44, 3), (45,
3), (49, 3), (50, 3), (54, 3), (57, 3), (58, 3), (62, 3), (63, 3), (66, 3), (6
9, 3), (73, 3), (74, 3), (79, 3), (80, 3), (82, 3), (87, 3), (89, 3), (90, 3),
(91, 3), (92, 3), (93, 3), (94, 3), (97, 3)]
Total number of images retrieved:  49
```

Fig. 18. Images in ranked order

G. Performance Analysis:

1. Simple CNN - Deep learning approach:

Table - II: Performance analysis for deep learning

MODEL	F1 SCORE	ACCURACY
Basic cnn	0.44	0.4352
Vgg 16	0.21	0.2203

2. Performance Analysis of 3 models:

F1- Score alongside accuracy is considered as the major metric of efficiency. The classification report for the validation set of the dataset is given in Table- III.

Table - III: Performance analysis for validation set

MODEL	F1-SCORE	ACCURACY
Vgg16	0.89	0.8922
Inceptionv3	0.69	0.6866
Basic 6 layercnn	0.65	0.76

3.Performance Analysis VGG-16:

The validation classification report for the 15 class dataset in VGG-16 is included in Table -IV.

Table - IV: Performance analysis for validation set

CLASS	PRECISIO N	RECALL	F1- SCORE
airplane	0.85	0.87	0.86
apple	0.97	0.96	0.96
banana	0.92	0.93	0.92
bicycle	0.92	0.95	0.93
bird	0.82	0.81	0.82
car	0.88	0.87	0.88
cat	0.81	0.85	0.83
chair	0.95	0.91	0.93
cup	0.93	0.92	0.93
elephant	0.84	0.87	0.85
fire hydrant	0.81	0.83	0.82
knife	0.92	0.87	0.89
pizza	0.96	0.97	0.96
teddy bear	0.92	0.92	0.92
train1	0.90	0.86	0.88
Accuracy			0.89
Macro avg	0.89	0.89	0.89
Weighted avg	0.89	0.89	0.89

V. RESULTS AND DISCUSSIONS

From Table - II, it is inferred that a pure deep learning based approach proves to be inefficient due to the less features contained in the sketches. Since the sketches contain only black edges on a white background, it causes high data loss and relevantly low details to learn from. Pre-trained models proved to perform poorer than the basic CNN.

Table - III shows that the machine learning based approach using SVM works better with deep learning based feature extraction. Here the feature extraction is efficient when the pretrained model used for extraction contains less number of layers.

This method is also useful when the dataset is imbalanced as the SVM only learns from the extracted features and avoids overfitting of the class with majority which occurs in the case of deep learning. To improve the f1-score, classes with lower f1-score can be trained with more sketch images. The need for an extensive computing device occurs when using more data for training the SBIR classifier, which can be eliminated using this method as it requires comparatively less memory to function.

Since CVLib is based on the COCO dataset, it is trained to only detect objects belonging to those 80 classes. This limits us to use classes in our dataset common to both the COCO dataset and the sketch dataset. Hence, as an improvement, a separate object detection model can be created that allows for usage of all classes taken in the dataset. This object classifier can be created and deployed using Tensorflow's Object Detection API [25] using the guide given in [26]

VI. CONCLUSION

From the results obtained, it is understood that a model using pure deep learning does not classify images accurately due to the nature of the sketches being binary ie. black and white and the dataset being sparse, when compared to obtaining features using a deep learning model and then training on those features using a support vector machine.The sketch images contain only 0 to 255 as pixel values, which is greatly reduced as we move down the filtering layers from the input convolution layer to the final output dense layer. This is also a possible reason for the lower accuracy of the convolution neural network (CNN). Due to extracting features using pre-trained architectures like VGG16, the SVM model is able to predict highly detailed images although the training was done only on heavily abstract and light featured sketch images.

REFERENCES

1. Hirata, K., Kato. T (1992) 'Query by visual example-content based image retrieval ', International Conference on Extending Database Technology: Advances in Database Technology, pp. 56– 71
2. Ayan Kumar Bhunia, Yongxin Yang, Timothy M. Hospedales, Tao Xiang, Yi- Zhe Song (2020) 'Sketch Less for More: On-the-Fly Fine-Grained Sketch Based Image Retrieval ', IEEE Conference on Computer Vision and Pattern Recognition (CVPR)
3. Sounak Dey, Anjan Dutta, Suman K. Ghosh, Ernest Valveny, JosepLlados,Umapada Pal (2018) 'Learning Cross-Modal Deep Embeddings for Multi-Object Image Retrieval using Text and Sketch', 24th International Conference on Pattern Recognition (ICPR)
4. Sasi Kiran Yelamarthi, Shiva Krishna Reddy, Ashish Mishra, and Anurag Mittal (2018)'A Zero-Shot Framework for Sketch Based Image Retrieval',The European Conference on Computer Vision (ECCV)
5. Yonggang Qi, Yi-Zhe Song, Honggang Zhang, Jun Liu (2016), 'Sketch-based image retrieval via Siamese convolutional neural network', IEEE International Conference on Image Processing(ICIP),
6. Sounak Dey, Pau Riba, Anjan Dutta, JosepLlados, Yi-Zhe Song (2019), 'Doodle to Search: Practical Zero-Shot Sketch-based Image Retrieval', The IEEE Conference on Computer Vision and Pattern Recognition (CVPR).
7. VGG-16, https://keras.io/api/applications/vgg/
8. CVLib - object detection library, https://www.cvlib.net/
9. Sketchy dataset, https://sketchy.eye.gatech.edu/
10. QuickDraw! dataset, https://github.com/sounakdey/doodle2search
11. TU-Berlin dataset, http://cybertron.cg.tu-berlin.de/eitz/projects/classifysketch/
12. Peko-Step, https://www.peko-step.com/en/tool/combine-images.html
13. Python 3.8, https://www.python.org/downloads/release/python-380/
14. Kaggle, https://www.kaggle.com
15. Google colab, https://colab.research.google.com/
16. Support vector machine, https://towardsdatascience.com/https-medium-com-pupalerushikesh-svm- f4b42800e989
17. Canny, https://medium.com/sicara/opencv-edge-detection-tutorial-7c3303f10788
18. Sobel, http://www.adeveloperdiary.com/data-science/computer-vision/how-to- implement-sobel-edge-detection-using-python-from-scratch/
19. findContours(), https://www.thepythoncode.com/article/contour-detection-opencv-python
20. Machine learning, https://www.geeksforgeeks.org/machine-learning/
21. Deep Learning, https://machinelearningmastery.com/what-is-deep-learning/
22. Convolution Neural Network, https://towardsdatascience.com/a-comprehensive-guide-to-convolutional- neural-networks-the-eli5-way-3bd2b1164a53
23. InceptionV3, https://keras.io/api/applications/inceptionv3/
24. COCO Dataset http://images.cocodataset.org/zips/val2017.zip
25. Tensorflow Object Detection API, https://tensorflow-object-detection-api-tutorial.readthedocs.io/
26. Tensorflow Object Detection API usage, https://towardsdatascience.com/how-to-train-your-own-object-detector-with- tensorflows-object-detector-api-bec72ecfe1d9

AUTHORS PROFILE

Deepika Sivasankaran, is pursuing her undergraduate study in Computer Science and Engineering at Sri Sivasubramaniya Nadar College of Engineering. Her interests include image processing, deep learning, machine learning and computer vision.

Sai Seena P, is pursuing his undergraduate study in Computer Science and Engineering at Sri Sivasubramaniya Nadar College of Engineering.

Rajesh R, is pursuing his undergraduate study in Computer Science and Engineering at Sri Sivasubramaniya Nadar College of Engineering. His interests include web development (MERN stack), Front end design,Backend development, Mobile application development(React native) and DBMS.

Madheswari Kanmani, Associate Professor in the Department of Computer Science and Engineering has 7 years of teaching experience. She received her PhD in the area of image fusion from Anna University, Chennai. She received her Bachelor's degree in Computer Science and Engineering from Hindusthan College of Engineering and Technology, Coimbatore. M.E. Software Engineering from Anna University. Her area of interest is on image processing, Optimization techniques, Software Engineering, Software Testing and Wireless Networks.