Sketch Based Image Retrieval

Santhoshini Gongidi and Chris Andrew

Abstract— **Sketch based image retrieval systems are used to find image that are similar in content to a given sketched image. The aim of this project is to build a system that is able to retrieve relevant and appropriate images when a sketch image is given as a query to the system. The retrieved images may be sketches or colored/digital images. The system will be trained to identify the best possible match for a given sketch image to a subset of relevant images.**

I. INTRODUCTION

Image retrieval systems are used for searching a given query in image databases using a retrieval system, this query can be in form of text, images, rough drawings or sample sketches. For building a text-based query image retrieval system, every image in the database has to be associated with tags or words that describe the image and these tags are used for comparison between images and text search queries. Building this tag-word dictionary for image database is a time-consuming process and sometimes these tag words may fail to explain content of the images. Therefore, building image retrieval systems that can explain and understand image content becomes necessary to produce reliable and good retrieval systems. Some of the input query formats capable of capturing image content are example images, rough color drawings of images or outline sketches. Using example images and color drawings as inputs requires us to first create these image for our query, which in itself is a resource consuming task. Users using retrieval systems might not be able to find results coresponding to the query images every time they have a query to search, thus causing a wastage of resources. Similarly, generating detailed color drawings for querying is not a feasible option because not all users are capable of drawing detailed images and producing a detailed drawing is also a time consuming process. Outline sketches are rough drawings of images that convey the content of the image and are simple without any detailed information about the image. Producing an outline sketch of an object would take less than a few seconds and yet these sketches carry a rich amount of information. Therefore, input in the form of outline sketches is a user-friendly way to record a query and at the same time these sketches capture the image content as well. Since outline sketches satisfy the criteria to describe image content, building image based retrieval systems that take a outline sketch as query can generate desired results. Such image retrieval systems with sketch input are referred as Sketch based Image Retrieval (SBIR) systems.

In an SBIR system, a given query sketch needs to be compared with images available in a database to retrieve images that are relevant to the query. This is a special type of Content Based Image Retrieval(CBIR) system, similar to the one used in Google's Image search(Search by Image), where an image is used to find similar images in Google's database. The problem is not as simple as it seems, because of a large number of factors that introduce errors in the retrieval process. Digital/color images vary from sketches in that sketches only contain information about the edges. Comparing this edge information is not always easy and as such we must try and extract features from images that are invariant to color and extract information from the shape of the image. Apart from color, the images in the database will have a large variance in scale as well as the orientation of objects in the image. The features used to compare two images must also be scale and rotation invariant to allow a much more exhaustive search of the database.

The content of this report is organized as follows: Section II discusses the different approaches considered for building a SBIR system and also the previous work done in this area. Section III provides an overview of the pipeline used to build a retrieval system and the dataset used for training and testing purposes. In Section IV different aspects of pipeline will be discussed in detail. In Section V the results obtained for different scenarios are reported and in Section VI the results obtained are discussed along with the drawbacks of the current pipeline. We also throw light on how to improve the current pipeline and features in Section VI. The final section Section VII is to mention of contributions of the individual project members. The link for the code written is mentioned in the last section.

II. IDEAS & SOLUTIONS

To build a naive SBIR system, we can process the images so that they contain only the dominant edges and resize all the images to one standard size and store in a database. For a given sketch, resize it to chosen the standard resolution and do pixel by pixel comparison. Retrieve those images that have lowest error as appropriate results. The major drawback with this approach is that the intensity values and foreground pixel positions don't convey the image content always and therefore can't be used to retrieval. For example, consider an image where an object is present in the top left corner and the rest of the background. For querying let the sketch have the same object drawn at the bottom right. In this case, the comparison between these two will lead to large error and might not be retrieved. This can be avoided by looking at smaller patches of the images and sketches to define and compare their content.

Another way to build a SBIR is to break the images into smaller blocks and store these blocks for comparison with sketches. Although, this approach is better than the above mentioned approach, this also fails because a user input sketch is bound to have large amount of variance in the way the sketches are drawn. Therefore, it becomes necessary to extract higher level features like shapes, lines etc rather than pixel level comparison.

Most state of the art sketch based retrieval systems rely on Deep Learning to extract useful features about the content of the image which is then matched to a database of images. Before Deep Learning based approaches became popular, the state of the art in SBIR was the bag of visual words model. In this model, we try and identify regions in an image that contain important content and extract descriptors from that region. These descriptors are then classified into words and the a distribution of these words are used to match images. In this project, we have tried to extract different distributions of such high level features like different shapes, their sizes, edges, etc to build a bag of visual words based SBIR system.

III. OVERVIEW & DATASET

The SBIR system built in the project uses the benchmark dataset provided by [4]. For this project, the training set consists of RGB colored images of different objects and test set comprises of sketch images. This benchmark dataset consists of 31 sketch images of different objects that form the test set. For every sketch, 40 color images corresponding to the sketch are provided. These images form the training set for the system and it contains 1240(31x40) images in total. Images are matched only if the provided sketch coresponds to a class of objects in the training set. This is because the system is unaware of the composition of other objects that the user might provide and has only learned to recognize the objects of the training set. A few examples of the sketches and their corresponding images are shown in the Fig. 1.

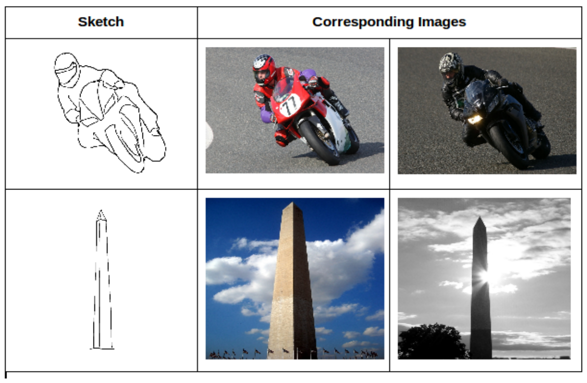
Sketch
Corresponding Images

Fig. 1: First column displays the sketch images. Second and Third columns shows the images corresponding to the sketches of its respective row available in training set

The pipeline of the SBIR system is divided into two phases: Training phase and Testing phase. In the training phase, for each image necessary feature descriptors are generated and stored to form a set of keypoints. These keypoints are then clustered to form a set of visual words. The model to classify a feature into a visual word is saved and for every image, we find the distribution of the visual words on every image and store it in the database for later comparison.

In the testing phase, feature descriptors for the user provided sketch are generated and the appropriate visual words are generating using the pre-trained models created in the training phase. A distribution of the words for the test image is compared with the database of the distributions generated in training phase to retrieve top similar images. Figures 2 & 3 correspond to training phase and testing phase respectively.

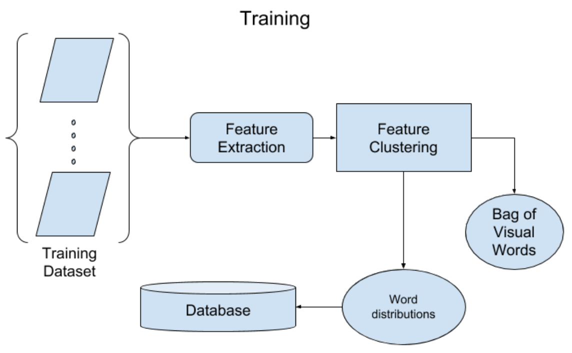

Fig. 2: Training phase pipeline

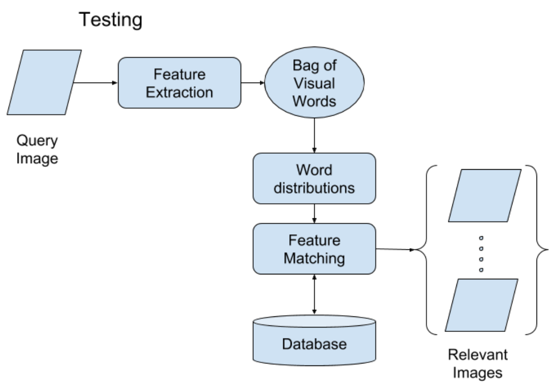

Fig. 3: Test phase pipeline

IV. METHOD

In this section, our approach to build an SBIR system is discussed in detail. Our SBIR system uses Bag-of-Words model to build feature descriptors for the images in the dataset. Each image in the training dataset is preprocessed to generate its edge image. This edge image is used to identify different keypoints in the images and descriptors for these keypoints are generated. The keypoints obtained from all the images of the training set are clustered to generate ***visual words*** that can describe the dataset. These visual words are used to generate histogram of words for each image in the training set and this histogram is used as a feature for searching and retrieval of relevant images. The features extracted for all the training images form the database as shown in 2

During the test phase or the retrieval phase, except for the preprocessing phase, the same procedure explained above is used to extract features for sketches. For a given query sketch, fits feature descriptor is extracted and is compared to features in the database. This comparison is explained in detail in the later subsections.

A. *Image Preprocessing*

In the training phase, firstly images are converted to grayscale. Then, these grayscale images are converted to edge images that have display only the dominant and strong edges present in the images. These edges images have resemblance to sketches and therefore feature extraction is performed on these images to generate image descriptors. To generate these edge images, Canny Edge detection is applied. As we want only strong edges to be preserved, the parameters for the edge detector are set as σ =5, low threshold as 0.05 and high threshold as 0.2

B. *Descriptors*

In this step, we want to identify potential points/areas in the processed image that can best describe the image content. Features are extracted from the neighbourhood of these potential points to form keypoint descriptors.

The different keypoint descriptors that are generated for each image are ***HOG descriptors, Shape Context descriptors and Spark descriptors***. Experiments were done using different combinations of these descriptors to find the best combination. These experiments are discussed in the results section.

1) HOG Descriptors: Histogram of Oriented Gradients(HOG) descriptors [3] encode distributions of the gradient direction in small local regions. To generate these descriptors, 4x4 spatial bins are used and 8-bin gradient orientation histogram is generated. These descriptors are generated at 500 randomly sampled points from the image foreground(edge lines or sketch lines). Distributions are generated for different sizes of local windows with their radii as 5, 10, 15, 20, 25, 30.

2) Shape Context Descriptors: The shape context descriptors [2] encode the distribution of the sample point locations on the shape relative to each of the other sample points. These distributions are encoded as log polar histograms with 5 bins for logarithmic distance between the sample points and 12 bins for angles. 500 sample points are chosen for each image from the foreground image only,i.e., points are samples from the edge lines only. Shape context features are generally computed at global level. In our case, since we want to build visual words and these words don't need to contain global information, therefore these distributions are calculated only within local windows of different radii 5, 10, 15, 20, 25, 30.

3) Spark Descriptors: The spark descriptors encode the distribution of sample point locations in the background relative to the edge points(points on the foreground). To generate these descriptors, 500 random points are sampled from the image background. For each chosen sample point, a local neighbourhood of 50x50 is considered and from the sample point we search for the first edge point in different directions. In some cases, we may not find anything in the chosen neighbourhood. We use angle and distance between the sample points and their identified edge points to form distributions that encode local information. Similar to the shape context features generated above, 5 bins for distance and 12 bins for angles were considered.

C. *Image Descriptors*

The keypoints identified above are used to generate visual words. As we use words in a sentence to understand the meaning of a sentence/ its content, similarly we use these visual words to identify the content of the images. To generate these visual words, all the keypoint descriptors obtained for all the images are clustered. The centroids of these clusters are the best representation of these clusters and therefore, are chosen as the visual words. These visual words represent all the the possible words that can be used to represent the images in the dataset. Therefore, to represent an image's content, the distribution of these visual words across the image is chosen. This histogram generated for each image is used as feature descriptor for comparison in the matching phase of the retrieval system.

As described in the previous section, different keypoint descriptors were detected. Experiments are conducted to identify visual words using just single descriptor features, combination of 2 descriptor features and all the three descriptor features. When using more than 1 type descriptor features, for every image the keypoints are concatenated and then clustering is performed using these concatenated vectors. K-means clustering technique is used to cluster the keypoint features. Each feature is clustered into 750 visual words.

D. *Searching Methods*

To compare the features extracted from the sketch with those in the database, two different methods were experimented. One way of comparison is to find top-n nearest features in the database and retrieve the corresponding images as features. Another way to retrieve similar images is to build a tf-idf table using the features in the database and to retrieve similar images based on tf-idf ranking for a given sketch.

V. RESULTS

We have reported below the classification accuracy for our SBIR system, where each class of objects is treated as a separate class for classification. This effectively makes the problem into a 31 class problem. Color images were used for training the model and sketch images were used for testing. We have reported the results using word distributions from each of the extracted features as well as a combination of those features in the Table below:

Features	**Classifier**	**Accuracy**
SC	Logistic Regression	0.1612
SC	Linear SVM	0.193
SC	RBF kernel SVM	0.0322
Spark	Logistic Regression	**0.451**
Spark	Linear SVM	0.3548
Spark	RBF kernel SVM	0.3548
HOG	Logistic Regression	0.1935
HOG	Linear SVM	0.1612
HOG	RBF kernel SVM	0.1612
HOG+Spark+SC	Logistic Regression	0.3870
HOG+Spark+SC	Linear SVM	0.387
HOG+Spark+SC	RBF kernel SVM	**0.443**

Cross validation was done to find the best parameters for the models. SVM with RBF kernel responded well to a gamma value of 1e-3 and C value 1. SVM with linear kernel responded well for C=1. Logistic Regression gave best results for C=0.005.

VI. OBSERVATIONS & IMPROVEMENTS

We tried to understand where the model failed for various cases and were able to identify some failure cases for the model as shown in Fig. 4. It can be observed that even though the actual image retrieved by the model does not belong to the class of the query image, there are obvious similarities between the two as the content of the images are similar. This goes on to show that the image actually does a good job of matching content of the image with that of the sketch.

To have a more non-quantitative approach to evaluating the model, we decided to build a demo search engine, where a query sketch image is given as input and the relevant images in the dataset are retrieved. For this, we tried to improve the method in which the images are retrieved. TF-IDF is a method used for retrieving documents based on a given set of query words. For a long time, it was considered the state of the art in document retrieval. The basic idea for TF-IDF and other information retrieval methods is discussed in [1].

To use TF-IDF, we treated visual words as words of a document, where the image represent a document. A TF-IDF index is created for the images and stored in the database. For retrieval, the visual words from the test image are extracted and then TF-IDF is used to match it to the relevant set of images from the database. A demo for the above system will be shown during the presentation.

VII. CODE & CONTRIBUTIONS

In this section, contributions of both members are mentioned for different modules of the project. The table below mentions the tasks done by each member. Debugging and testing done by both the members. The code for this project is available at SBIR Github repository.

Task	**Done by**
HOG Feature Extraction	Chris
Shape Context Feature Extraction	Santhoshini
Spark Feature Extraction	Santhoshini
Clustering and Visual Word generation	Chris
TF-IDF generation and comparison	Chris
Demo application for SBIR	Santhoshini

VIII. CONCLUSION

We have successfully created a SBIR system that relies on features that are used in Image Processing. We built and tested the system on a benchmark SBIR dataset. The system was trained and used to build a search engine for sketch images.

REFERENCES

[1] Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge & Data Engineering*, (6):734–749, 2005.

[2] Serge Belongie and Jitendra Malik. Matching with shape contexts. In *Proceedings of the IEEE Workshop on Content-based Access of Image and Video Libraries (CBAIVL'00)*, CBAIVL '00, pages 20–, Washington, DC, USA, 2000. IEEE Computer Society.

[3] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1 - Volume 01*, CVPR '05, pages 886–893, Washington, DC, USA, 2005. IEEE Computer Society.

[4] Mathias Eitz, Kristian Hildebrand, Tamy Boubekeur, and Marc Alexa. Sketch-based image retrieval: Benchmark and bag-of-features descriptors. *IEEE Transactions on Visualization and Computer Graphics*, 17(11):1624–1636, 2011.

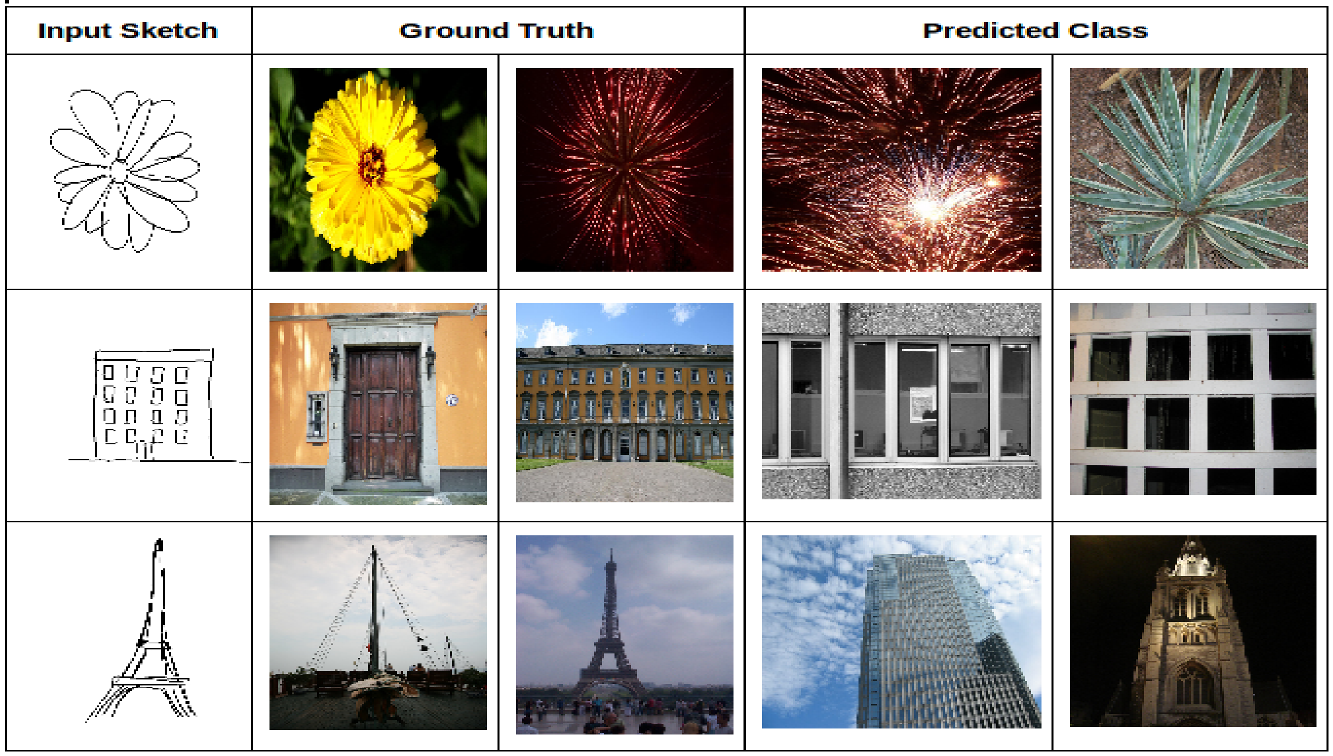

Fig. 4: This image shows the incorrect predictions by the SBIR. First column displays examples of the input sketches given to the system. 2nd and 3rd columns show the ground truth images corresponding these sketches. 4th and 5th columns show the images of predicted class corresponding to the given sketch.