Image Search Engine

Deep Talati(1401085), Harsh Dalal(1401022),
Krupa Gajjar(1401031), Mansi Thakkar(1401036) and Varad Bhogayata(1401042)

***Abstract*—Image Search Systems is unable to correlate with human perception, as words have limited ability to convey the exact meaning and context which leads to ambiguity in different contexts. An Image is the best way to convey the concept in the right context. An Image is the concept that exhibits the information in a more relevant way. Its very arduous task to map the query word to retrieve the exact image as a relevant answer. Image search system mainly works on the principle on keyword as queries and likewise, work on surrounding information like tags annotation to find the relevant image. The commercial search Engines like Google and Bing are using Image Search Systems.**

***Keywords*—Feature Index, Histogram, Haar Features, Convolutional Neural Network**

I. Introduction

Image Search Engine is information retrieval system, which will facilitate the process of finding the image. The focus of the image search is to retrieve the image with respect to user input query from the given large database.User can provide query as keywords and the system would return images similar to the input query.The criteria used for search could be meta tags, colour distribution in images.Indexed meta-data of the image is stored in large database ,when a query arrives and is performed , the search engine looks up the index and queries are matched with the information which is stored.The results of the search are sorted by relevancy.

II. Procedure

Our focus is text to image retreival search where input of the system is text and the result query obtained is/are image(s).

The current state-of-the-art algorithm for Text to Image Retreival Search implemented by Google is:

The algorithm parses web sites and relates the data on the website with the images. Based on the correlation of the data on the website and the image, it provides a correlation score to the image.

The system performs ranking based on the correlation scores and its internal algorithms, and shows the results to the user.

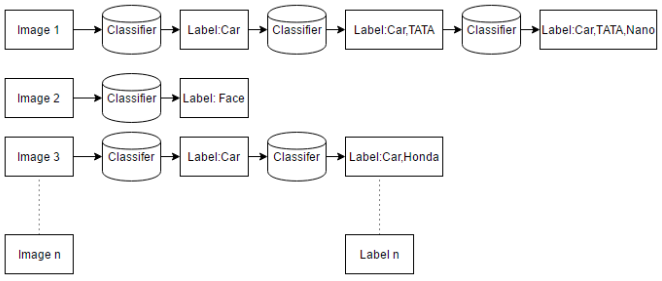

Figure 1: Our proposed algorithm

Our Proposed Algorithm:

The system uses a hierarchy of classifiers as shown in Fig 1, which are trained with some test data, and we finally obtain a label set for all the images present in our dataset.

When a user enters a text query, all the labels are searched, and if a match is obtained, the images are displayed.

Moreover, the system also uses the metadata from the image to further refine the search.

The system extracts latitude and longtitude from the image, to provide more options to search.

Example: If a user wants to search for a car in Navrangpura, they can add that as a parameter to get the desired output.

III. Different Types of Classifiers

Three different types of Classifiers we used are:

A. *Histogram*

We need to know how are we going to represent an image based on some features which can include color, texture, shape etc. Our feature index can be a single aspect or a combination of multiple aspects. Feature index is an object for comparison of the query and pre-processed images.

For this project, we consider using histogram as a feature index for our image search engine.In image processing context, the histogram of an image basically refers to the histogram of the values of pixel intensities. Colour histograms can be used as content descriptors and analysis can be performed by utilizing both RGB and HSV histograms.

The histograms of these colour channels are then concatenated to create a 1-D histogram, making it easier to perform similarity checks.Now, the similarity between the query image and the preprocessed image is made using a distance metric. Most common choices for distance metric are Euclidean, Correlation, and Chi-Squared. An assumption has been made that images having similar color distributions are similar to each other.

Procedure:

1. Creating feature index of all the pictures in the directory As mentioned above using histogram as a feature index, we generate histograms for all the images in our database and store the relevant details.

2. Taking the query image as an input from the user In this step, the user provides us the image for which the user wants to get semantically similar images.

3. Searching the directory for all the pictures Now, that the user has inputted the query image, we go ahead and match our query feature index with all the feature indexes of the directory.

4. Evaluating the same feature for the query picture As we need semantically similar images we calculate Euclidean distances between two feature Indexes (One from Query and One from Database) and we store them in a relevant file.

5. Ranking the pictures based on the extracted values After obtaining all the Euclidean distances, we now sort them into a relevant order and assign a rank based on the similarity of the images.

6. Outputting the matched pictures We output the matched images based on the rankings.

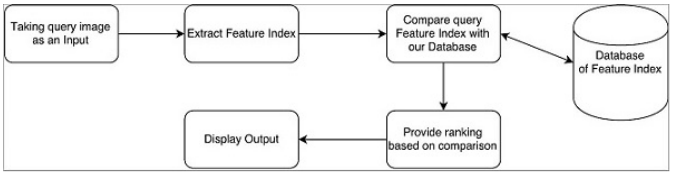

Figure 2.1: Histogram Procedure

B. *Haar Features*

Haar features are widely used for object recognition. Haar-like features were proposed by Viola and Jones as an alternative method for face detection. The general idea was to describe an object as a cascade of simple feature classifiers organized into several stages. It is a machine learning based approach where a cascade function is trained from a lot of positive and negative images. It is then used to detect objects in other images.

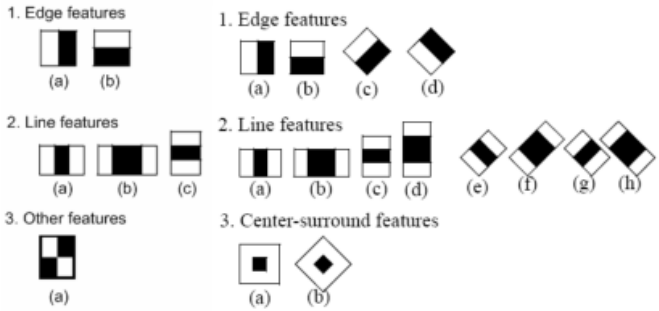

Figure 2.2: Haar Features

Each haar feature calculated by the system is a single value obtained by subtracting sum of pixels under white rectangle from sum of pixels under black rectangle. The computation complexity for this computation becomes too high. Hence, the system uses Adaboost to remove unnecessary features. But still, the complexity is pretty high as we have to check all the features for each window present.

The system applies the concept of cascade of classifiers. Instead of applying all the features on a window, we group the features into different stages of classifiers and apply one-by-one.If a window fails at a stage, discard it.

The system generates haar features from our dataset using the above algorithm.

Procedure:

1. Obtain the label set for each of the image already present in our dataset. After we obtain the haar features, we obtain the label set for the images present inside the dataset. For that we simply check if the output of classification is 1, then we add it to the label set.

2. Input search query. Now, after we have our label set built, we ask input from the user.

3. For search query, find the matching label set from our labels. Now, for search query, we iterate through all the labels inside our dataset, and find if the current label matches the search query. If it does, it is added to result.

4. Display the result to the users. Finally display all the result computed in the fourth step above.

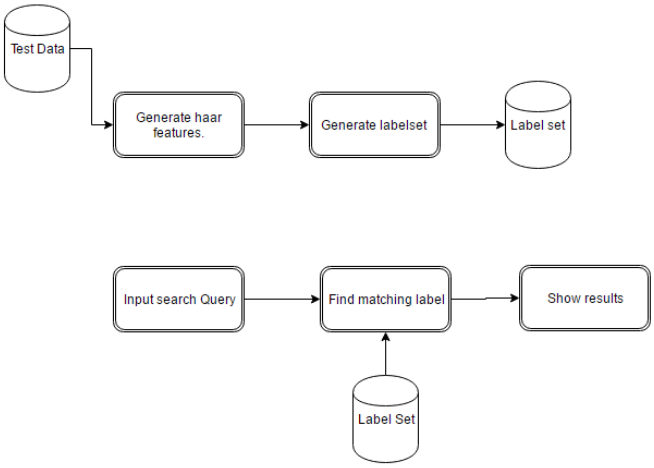

Figure 2.3: Haar Classification Procedure

C. *Convolutional Neural Network*

Regular Neural network dont scale well to full images. In our Database, images are only of size 150x150x3 (150 height,150 weight, 3 color channels), So a single fully-connected neuron in a first hidden layer of a regular neural network would have 150*150*3 = 67500 weights. But as image size increases, the number of weights at each level increases, the number of parameters increases. Hence, this full connectivity is wasteful and huge number of parameters quickly lead to overfitting. For this reason, we use Convolution neural network.

Convolutional Neural Network are very assemble to ordinary Neural Network. CNN make the explicit assumption that inputs are images, or any 2D or 3D data which can be made to look like an image. CNN are good at capturing local spatial patterns in data. We are dealing with images, hence we can model our data according to Convolutional neural network.

A CNN is made up of Layers. Every Layer has a functionality: It transforms an input image to an output image with some differentiable function that may or may not have parameters.

A simple Convolution Neural Network is a sequence of layers, and every layer of a Convolution Neural Network transforms one volume of activations to another through a differentiable function. We mainly use three types of layers to build Convolution Neural Network architectures: Convolutional Layer, Pooling Layer, and Fully-Connected Layer.

Convolution Layers:

1. Convolutional Layer: The convolution layer is the core building block of a convolutional network that does heavy computations. Firstly, each image pixel is multiplied by the feature pixel. Obtained feature matrix and input image is then convoluted. At convolution layer, one image becomes a stack of filtered images.

2. Normalization Layer: After the convolution operation, set value zero where negative pixel value is present. A stack of images becomes a stack of images with no negative values.

3. Pooling Layer: Pooling is the process of shrinking the image stack. For that, firstly we choose window size and iterate the window across the filtered image. From each window, take maximum value of the multiplied matrix with window matrix. This mainly used to lower down the dimension of the image characteristics which can be mapped to lower dimension matrix.

4. Fully-Connected Layer: Neurons in a fully connected layer have full connections to all activations in the previous layer, as seen in regular Neural Networks. After that, all the final feature matrix are converted into vector.

Procedure:

1. Train our CNN model from test data. The system trains the CNN model from the specification specified above.
2. Obtain the label set for each of the image already present in our dataset. For all the images in our dataset, the system obtain label set for every image.
3. Input search query. After the above sets are performed, the system inputs search query.
4. For search query, find the matching label set from our labels. Search query is compared with every label set of our image dataset, if search query matches with the label set, we append it to result.
5. Display the result to the users. The obtained result is displayed in this step.

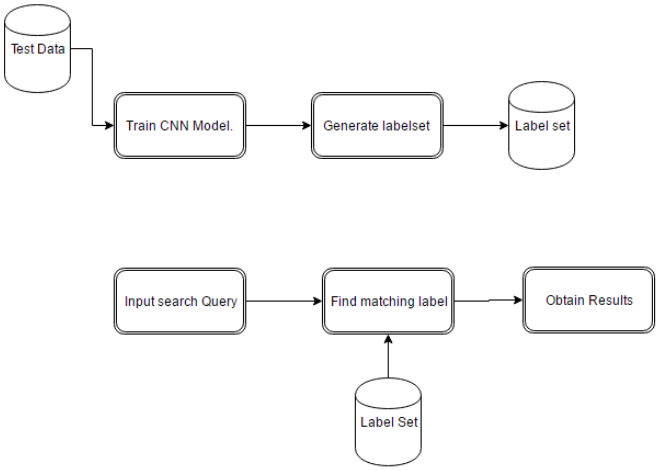

Figure 2.4: CNN Classification Procedure

IV. Simulation

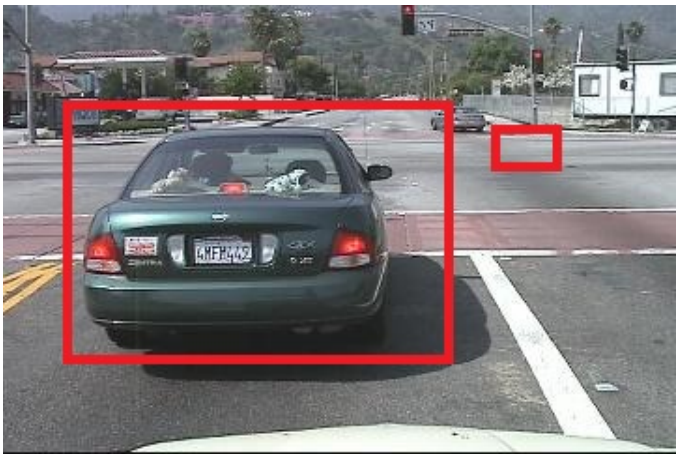

Figure 3.1: Car detection using haar features

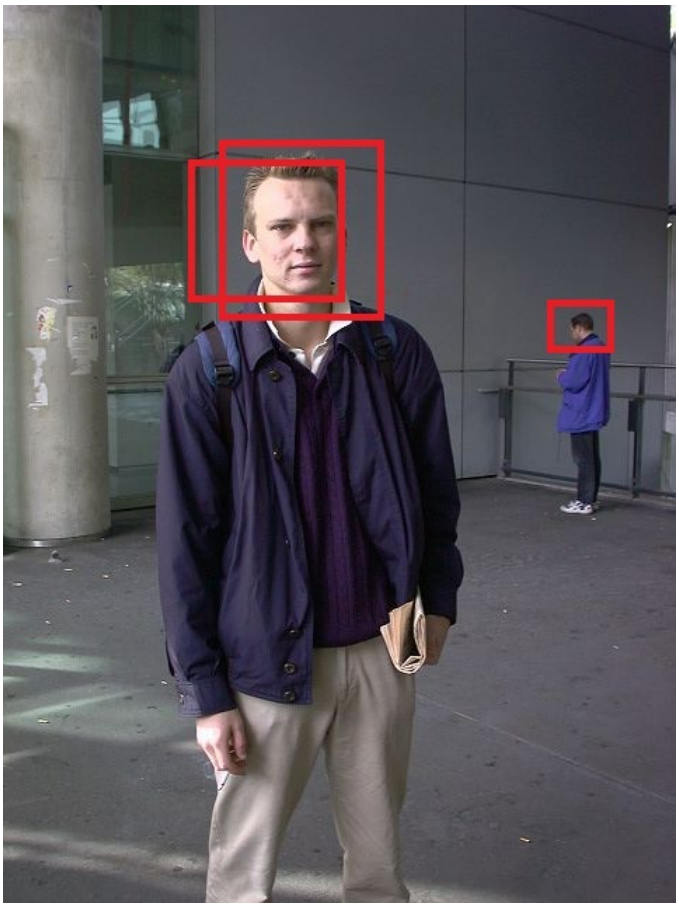

Figure 3.2: Face detection using haar features

V. Future work

1. Time taken for image search needs to be reduced.
2. Memory cost efficient model should be created
3. Scalable model needs to be discovered.

REFERENCES

[1] https://pythonformachinelearning.wordpress.com/2014/08/15/your-very-own-personalised-image-search-using-python/

[2] http://cs231n.github.io/convolutional-networks/

[3] http://www.pyimagesearch.com/2014/01/27/hobbits-and-histograms-a-how-to-guide-to-building-your-first-image-search-engine-in-python/

[4] http://www.pyimagesearch.com/2014/02/24/building-image-search-engine-searching-ranking-step-4-4/

[5] https://github.com/kzwang/elasticsearch-image

[6] http://www.sciencedirect.com/science/article/pii/S187705091503505X

[7] https://en.wikipedia.org/wiki/Haar-likefeatures

[8] http://docs.opencv.org/2.4/doc/tutorials/objdetect/cascadeclassifier/cascadeclassifier.html