ABSTRACT
Logos sometimes also known as trademark have high importance in today’s marketing world. Logo or trademark is of high importance because it carries the goodwill of the company and the product. Logo matching and recognition is important to discover either improper or unauthorized use of logos. Query images may come with different types of scale, rotation, affine distortion, illumination noise, highly occluded noise. Sift descriptor, surf descriptor and hog descriptor are very good features to use among the existing techniques to recognize the logo images from such difficulties more accurately.

General Terms
Logo Recognition, invariant to scale, rotation, invariant to illumination noise, occluded objects.

Keywords
KeyPoint localization, KeyPoint descriptor, Interest Point Descriptor

1. INTRODUCTION
Logos of different formats and styles exist in the database. To recognize the different types of logo images of different logo features can be used for extracting the feature values. From the training and testing logo images similar set of features are extracted and matching algorithm is used to match the similarity between training and testing logo images.

In this study, experimentations of logo images are done with three existing technique, which is Scale invariant feature transform (SIFT), Speeded up robust feature (SURF) and Histogram oriented Gradient (HOG). For extracting the shape based feature vectors of testing and training logo images scale invariant feature transform (SIFT), Speeded up robust feature (SURF) and Histogram Oriented Gradient (HOG) is used. Finally, Manhattan Distance classifier is used to measure the similarity between the sample images of training logo images and testing logo images of the classes.

2. METHODOLOGY
For experimentations of logo recognition Logo dataset from University of Maryland [19], google images and flickrlogos are used which includes 110 logo images in the TIFF format and resize to 150×150 pixel. Convert images into Binary image and into double.

2.1. Scale Invariant Feature Transform
SIFT Keypoints are invariant to scale, rotation, invariant to illumination differences and partially occluded objects.

As David G. Lowe described SIFT [12] consists of four major stages:
(1) Scale-space peak selection
(2) KeyPoint localization
(3) Orientation assignment
(4) KeyPoint descriptor

First of all scale space have to be generated. To generate scale spaces apply Gaussian over sample logo image with varying σ value. Then down sample it and find Laplacian of Gaussian. To efficiently locate the key point location in scale space difference of Gaussian D(x,y,σ)(DOG), is calculated from differences of two nearby scale separated by a multiplicative factor. A point is decided as a local minimum or maximum with respect to neighborhood points. The scale of the KeyPoint is used to select the Gaussian smoothed image, with the closest scale, so that all computations are performed in a scale-invariant manner. For each image samples, the gradient magnitude and orientation are calculated. An orientation histogram is formed from the gradient orientation of the sample points within a region around the KeyPoint. Then a distinctive descriptor is achieved for each KeyPoint. The image gradient and magnitudes and orientations, relative to major orientation of the KeyPoint are sampled within a 16×16 region. Each sample is weighted by its gradient magnitude and by a Gaussian circular window centered at the local maximum. These samples are then accumulated into orientation histogram summarizing the contents into 4×4 sub region. Each histogram has 8 bins covering 360 degree of orientation. Therefore having a 4×4×8=128 feature vector is generated for each KeyPoint.


2.2. Histogram Oriented Gradient

Histogram of Oriented Gradients (HOG) [17] is inspired on Scale-Invariant Feature Transform (SIFT) descriptors proposed by [15]. To compose HOG, the cell histograms of each pixel within the cell cast a weighted vote. In this work the histogram channels are calculated over rectangular cells by the computation of unsigned gradient. The cells overlap half of their area, meaning that each cell contributes more than once to the final feature vector. In order to account for changes in illumination and contrast, the gradient strengths were locally normalized, i.e. normalized over each cell. The nine histograms with nine bins were then concatenated to make a 1x81 dimensional feature vector.

2.3. Speeded up Robust Feature

The major steps of SURF are,

- Fast-Hessian Detector
- Constructing the Scale-Space
- Accurate Interest Point Localization
- Interest Point Descriptor
- Orientation assignment
- Descriptor Components

Much of the performance increase in SURF is due to Integral Image [8]. The integral image \( I \Sigma (x, y) \) of an image \( I(x, y) \) represents the sum of all pixels in \( I(x, y) \) of a rectangular region formed by \((0,0)\) and \((x,y)\). Now calculate the Hessain matrix, as function of both space and scale. Then calculate Hessain determinant using the approximated Gaussians. The task of localizing the scale and rotation invariant interest points in the image can be divided into three steps. First determine threshold value for the responses such that all values below the predetermined threshold are removed. Then, find a set of candidate points. To do this each pixel in the scale-space is compared to its 26 neighbors, comprised of the 8 points in the native scale and the 9 in each of the scales above and below. Once interest points have been localized both in space and scale, the next steps are:

1. KeyPoint descriptor
2. Orientation assignment

The SURF descriptor describes how the pixel intensities are distributed within a scale dependent neighborhood of each interest point detected by the Fast-Hessian. This approach is similar to that of SIFT but integral images used in conjunction with filters known as Haar wavelets are used in order to increase robustness and decrease computation time. In order to achieve invariance to image rotation each detected interest point is assigned a reproductible orientation.

- The image is convoluted with two first-order Haar wavelets.
- The filter responses at certain sampling points around the KeyPoint are represented as a vector in a two-dimensional space.
- A rotating window of \( 60^\circ \) is used to sum up all vectors within its range, and the longest resulting vector determines the orientation.

Now the present worker has Interest Point descriptor vector of length 1x64 for each Interest Point. For a image the present worker have got 20-120 Interest Points. Suppose for image no. of Interest Points are 80. So a 2D matrix \( D1 \) of size 64x80 is generated.

\[
AB=\text{norm}(D1(x,\cdot));
\]
\[
D3(1,x)=AB; \text{ Where } x=64;
\]

And a scalar value from Orientation assignment. Combining these two features the present worker has got a vector of length \( 1 \times 65 \) for each Image sample.

3. Training Phase

In the experiment total 10 classes has taken and to form training set total training set 5 images is used to form training set per class. First of all the images are resized to \( 150 \times 150 \) pixels converted binary image and SIFT descriptor, SURF descriptor and HOG descriptors are calculated for first 5 image belongs to a particular class where feature vector’s size is \( 128+65+81=274 \).

\[ \text{fv}_t(SA, ; \text{CL})=F1; \]

\( F1 \) is feature vector of length \( 1 \times 274 \). \( SA \) represents training sample no and \( \text{CL} \) represents class no. and \( : \) is for feature vector.

\[ \text{fv}_{t1}=\text{fv}_{t(i,: ,1)}; \]

\( \text{fv}_{t1} \) contains all 5 feature vectors of 5 training images of a class1. \( \text{fv}_{t1} \) (i=1 to 10)contains all 5 feature vectors of 5 training images of a class. Now size of \( \text{fv}_{t1}=5 \times 274 \). So mean value, \( mn_{t1} \) is calculated for \( \text{fv}_{t1} \). Similarly mean value is calculated for all classes, \( mn_{t}\)=mean (fv_{ti}) And those mean are the featue vector of size \( 1 \times 274 \).

\[ mn_{t}=\text{mean} (fv_{t}); \]

size of (mn_{t}) is \( 1 \times 274 \).

4. Testing Phase

For testing 6 images are chosen per class out of eleven images and feature vector is extracted for each 6 image belongs to ten different classes. The feature are then stored into a different variable. This feature vector can be referred as \( \text{fv}_{\text{ij}} \) - Where \( i= \) Test image number and \( j= \) Class number.

5. Classification Phase

After feature vector (vector size \( 274 \)) extraction of each testing images, this vector are then subtracted from mean of different class (Feature vector of size \( 274 \)) using Manhattan distance. Using Manhattan distance, difference of two feature vector of size \( 274 \) is finally converted into scalar. Each difference is stored in different variable.

\( mn_{t}i= \) Mean value of ith class (Feature vector of size \( 274 \))

The differences and variables are shown below:

\[ d_{tk}= \] Difference of ith testing image of jth Testing class with mean of kth Training class

In this phase 10 different classes are taken and 60 images present per class out of which 5 images are used for training and 6 images is used for testing. Using the proposed approach described in this paper \( 274 \) element feature vector is calculated for each training sample of each class. After that mean of all 5 training samples are calculated for all 10 class. After that \( 274 \) element feature vector is calculated for each 6 samples of each 10 class. Then difference is calculated using the equation,

\[ d_{tk}=\text{sum} (\text{abs} (\text{fv}_{\text{ij}} - mn_{tk})); \]

and some example of calculations is given below for first sample of class 1 with all ten classes:

First sample of first class with mean of the first class:
For \( j = 1 \) to 6 test classes
For \( i = 1 \) to 6 test sample
For \( k = 1 \) to 10 mean of train class

\[
d_{ijk} = (\text{abs value}(1\text{st element of } 274\text{ element vector of } ith \text{ test sample of } jth \text{ class})\text{ - }1\text{st element of } 274\text{ element vector of mean of class } k) + \text{abs value}(2\text{nd element of } 274\text{ element vector of } ith \text{ test sample of } jth \text{ class} - 2\text{nd element of } 274\text{ element vector of mean of train } kth \text{ class}) + \text{abs value}(273th element of } 274\text{ element vector of } ith \text{ test sample of } jth \text{ class} - 274th element of } 274\text{ vector of mean of train class } kth)
\]

After calculation of difference between testing image and mean of training image (belongs to different class), it is to be identified that all testing image of a particular class belongs to that particular class or not and it is the responsibility of the identifier to correctly identify an image whether an image belongs to the proper class or not.

Now to calculate the accuracy individually for every class we have to check the following code:

\[
count_1 = 0;
\]

\[
\text{if min}\{d_{111}, d_{112}, d_{113}, d_{114}, d_{115}, d_{116}, d_{117}, d_{118}, d_{119}, d_{1110}\} = d_{111}, count_1 = count_1 + 1;
\]

It identifies that first testing sample is a member of first class and in that case value of counter \( count_1 \) is incremented by 1. Same process calculated for all testing sample of first class and if the minimum value then the counter is incremented by 1. This \( count_1 \) is divided with number of testing sample for class 1 which is 6. After that this division is multiplied with 100 to calculate the percentage accuracy for class 1 and stored in variable \( \text{acc}(1) \). Same process is applied for all testing samples for all classes and the accuracy is calculated.

After calculation of percentage accuracy of different classes, mean of all accuracy are calculated and stored in a variable \( \text{acc}_\text{comb} \).

<table>
<thead>
<tr>
<th>Class</th>
<th>( d_{111} )</th>
<th>( d_{112} )</th>
<th>( d_{113} )</th>
<th>( d_{114} )</th>
<th>( d_{115} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sample</td>
<td>34.64</td>
<td>44.05</td>
<td>64.58</td>
<td>58.22</td>
<td>44.63</td>
</tr>
<tr>
<td>Class</td>
<td>( d_{116} )</td>
<td>( d_{117} )</td>
<td>( d_{118} )</td>
<td>( d_{119} )</td>
<td>( d_{1110} )</td>
</tr>
<tr>
<td>Sample</td>
<td>41.59</td>
<td>35.33</td>
<td>55.55</td>
<td>41.77</td>
<td>50.88</td>
</tr>
</tbody>
</table>

Table 1a. Testing Sample1 of Class1 identified

This tabulation shows that value of \( d_{111}, d_{211}, d_{311}, d_{511}, d_{611} \) has the least value in the respective table. This identify that scalar value of the difference between first testing image of first class and mean of first class (from training set) is minimum. So the first testing image can be identified as a part of first class, because in this classification process the minimum value of the difference is taken, and if a particular testing image has minimum difference with a particular class then it signifies that the image belongs to that particular class.

6. EXPERIMENTAL RESULT

So overall accuracy for all classes using the Manhattan distance is 96.66%. It has been checked that different feature extraction process and there combination gives different accuracy for all classes. But in all cases classifier is Manhattan distance.

<table>
<thead>
<tr>
<th>Process</th>
<th>Percentage Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Only SIFT</td>
<td>56.66%</td>
</tr>
<tr>
<td>Only HOG</td>
<td>76.66%</td>
</tr>
<tr>
<td>Only SURF</td>
<td>81.66%</td>
</tr>
<tr>
<td>SIFT with HOG</td>
<td>73.33%</td>
</tr>
<tr>
<td>SIFT with SURF</td>
<td>83.33%</td>
</tr>
<tr>
<td>SURF with HOG</td>
<td>85.00%</td>
</tr>
<tr>
<td>SIFT, HOG and SURF</td>
<td>96.66%</td>
</tr>
</tbody>
</table>

Table 1b. Percentage Accuracy with different method
7. CONCLUSIONS AND FUTURE SCOPE

From experiment with different logos, it has been seen that this code is not very much invariant to illumination. It can tolerate little bit illumination noise. In future the code could be improved by making it more illumination invariant.

8. REFERENCES


[12] Stefan Romberg, “From Local Features To Local Regions”, PP. 841-845


