# Binary scene classification

Sri Harshitha K (sriharshitha.k16@iiits.in)
B. Tech, ECE, Indian Institute of Information Technology Chittoor, Sri City, A.P, India

***Abstract* — In this project we present a procedure to classify real world scenes into two groups of natural and mad-made. The process requires the inclusion of various methods in the computer vision, image processing and feature recognition for identifying the category of the input image by comparing it against the given training dataset by means of the feature descriptors and classification algorithms.**

**The procedure takes into account the diagnostic information stored in the power spectrum of each category of images using the scale invariant feature transform (SIFT) and through supervised learning i.e., support vector machine (SVM) separates characteristic feature vectors of each class in separate groups that helps us to classify the new test image into its respective group.**

## I. INTRODUCTION

Classifying the world into natural and man-made environment or classification of an object in a single glance is one of the most accomplished feats of the human brain. It takes only a few tens of milliseconds to recognize the category of an object or environment, emphasizing an important role of feedforward processing in visual recognition. The mechanisms subtending efficient human visual recognition is our capacity to learn and remember a diverse set of places and exemplars by sampling the world several times per second, our neural architecture constantly registers new inputs even for a very short time, reaching an exposure to millions of natural images within just a short period of time. Inspired by the working of human brain, the conception of learning algorithms took birth. These algorithms in essence provide us with a methodology to find parameters which would be able to identify and classify different objects in a given signal input. How much would an artificial system have to learn before reaching the scene recognition abilities of a human being?

**Scene classification** is aimed at labeling an image into semantic categories (natural and man-made). It is an important task to classify, organize and understand thousands of images efficiently. From application point of view, scene classification is useful in-content based image retrieval. As accurate classification of an image, as better as it helps in better organization and browsing of the image data. Scene classification is highly valuable in remote navigation also.
As scene classification is dependent on feature extraction and classification

**TYPES OF FEATURES:**

**GIST:** The GIST descriptor focuses on the shape of scene itself, on the relationship between the outlines of the surfaces and their properties, and ignores the local objects in the scene and their relationships. The representation of the structure of the scene, termed spatial envelope is defined, as well as its five perceptual properties: naturalness, openness, roughness, expansion and ruggedness, which are meaningful to human observers. The degrees of those properties can be examined using various techniques, such as Fourier transform and PCA. The contribution of spectral components at different spatial locations to spatial envelope properties is described with a function called windowed discriminant spectral template (WDST), and its parameters are obtained during learning phase.
The implementation we used first preprocesses the input image by converting it to grayscale, normalizing the intensities and locally scaling the contrast. The resulting image is then split into a grid on several scales, and the response of each cell is computed using a series of Gabor filters. All of the cell responses are concatenated to form the feature vector

**SIFT:** A SIFT feature is a selected image region (also called Keypoints) with an associated descriptor. Keypoints are extracted by the SIFT detector and their descriptors are computed by the SIFT descriptor. It is also common to use independently the SIFT detector (i.e. computing the Keypoints without descriptors) or the SIFT descriptor (i.e. computing descriptors of custom Keypoints).

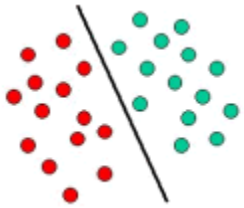
**HOG:** The technique counts occurrences of gradient orientation in localized portions of an image. This method is similar to that of edge orientation histograms, scale-invariant feature transform descriptors, and shape contexts, but differs in that it is computed on a dense grid of uniformly spaced cells and uses overlapping local contrast normalization for improved accuracy.

**TYPES OF CLASSIFIERS:**

**NEAREST NEIGHBOR**: Among the various methods of supervised statistical pattern recognition, the Nearest Neighbour rule achieves consistently high performance, without a priori assumptions about the distributions from which the training examples are drawn. It involves a training set of both positive and negative cases. A new sample is classified by calculating the distance to the nearest training case; the sign of that point then determines the classification

of the sample. The k-NN classifier extends this idea by taking the k nearest points and assigning the sign of the majority. It is common to select k small and odd to break ties (typically 1, 3 or 5). Larger k values help reduce the effects of noisy points within the training data set, and the choice of k is often performed through cross-validation.

**SUPPORT VECTOR MACHINE:** Support vector machines are based on the concept of decision planes that define decision boundaries. A decision plane is one that separates between a set of objects having different class memberships. A schematic example is shown in the illustration below. In this example, the objects belong either to class green or red. The separating line defines a boundary on the right side of which all objects are green and to the left of which all objects are red. Any new object (white circle) falling to the right is labeled, i.e., classified, as green (or classified as red should it fall to the left of the separating line).



**NEURAL NETWORKS:** A neural network consists of units (neurons), arranged in layers, which convert an input vector into some output. Each unit takes an input, applies a (often nonlinear) function to it and then passes the output on to the next layer. Generally, the networks are defined to be feed-forward: a unit feeds its output to all the units on the next layer, but there is no feedback to the previous layer. Weightings are applied to the signals passing from one unit to another, and it is these weightings which are tuned in the training phase to adapt a neural network to the particular problem at hand.

## II. DATASET

The 15-class scene category data set is used for testing and training the project. From this data set we made a new data set which consists of two categories i.e., natural and man-made each category consisting of 200 images for training. Images are about 300 X 250 in average size.

Man-made images consist of scenes like:

- Suburban/Street
- Tall buildings
- Kitchen
- Living/Bed room/Office

Natural images consist of scenes like:

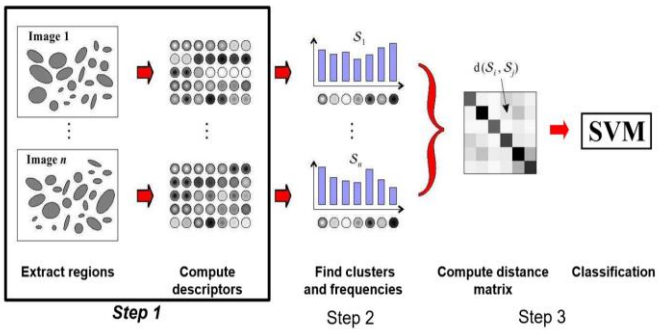
- Forests
- Mountains
- Countryside
- Coast

(i)Dataset for Artificial Scenes

(ii)Dataset for Natural Scenes

## III. IMPLEMENTATION

We essentially need to figure out features such that Image = array of features. The most important characteristic of feature that we require is a very strong co-relation between the feature value and the class of the image. This makes learning an easier, faster and a less error prone job. Thus, feature identification is among the most important task. Once, the features are identified a classifier can be constructed using state of art algorithms like SVM's, k-means etc.




The approach toward the scene classification in this project is:

- **Image resize**

  The image is scaled and the image is converted into gray scale by taking the average of Red, Green and Blue Components.

- **Finding the Pyramid of Sift Descriptors**

  The first step is to find the sift descriptors of the image. This done by using the dense sift implementation in **VLFeat: vl_dsift.**

  **vl_sift** bundles a feature detector and a feature descriptor. The detector extracts from an image a number of frames in a way which is consistent with variations of the illumination, viewpoint and other viewing conditions. The descriptor associates to the regions a signature which identifies their appearance compactly and robustly. vl_dsift extracts a dense set of SIFT features from the image.

  Creating the pyramid of sift descriptors involve splitting the image into multiple levels of sub-images. Once this is done, the vl_dsift is made to run on the smaller portion of the images in smaller steps and appended to the images' feature vector.

- **Creating the Vocabulary**

  To classify the new test image into its respective category, we need to generate a vocabulary using the descriptor we have. Using this vocabulary later histograms are created that defines the image's pyramid of sift features.

  To do this, we choose a random set of images in our data set and constructed a vocabulary using the pyramid of sifts descriptor described above. Next these descriptors were grouped together into 200 clusters using K-Means clustering with **VLFeat: vl_kmeans**.

  **vl_kmeans** is a clustering algorithm. Its purpose is to partition a set of vectors into KK groups that cluster around common mean vector. This can also be thought as approximating the input each of the input vector with one of the means, so the clustering process finds, in principle, the best dictionary or codebook to vector quantize the data.

  This is one the lengthy process because large amount of clusters is required to cluster. The when training for the first time, the vocabulary that has been generated is saved for future use in **.mat** file.

- **Creating the Histogram**

  Once the vocabulary was defined, we can define an image's pyramid of sifts with a histogram using **VLFeat: vl_kdtreequeryfunction.**

  **vl_kdtreequeryfunction** uses a best-bin first search heuristic. This is a branch-and-bound technique that maintains an estimate of the smallest distance from the query point to any of the data points down all of the open paths. **vl_kdtreequeryfunction** supports two important operations: approximate nearest-neighbor search *and* k-nearest neighbor search. The latter can be used to return the *k* nearest neighbors to a given query point Q.

  When we define an image's histogram we first run the pyramid of sift descriptors function defined initially on the image. This gives us a large vector of features defining an images sifts at multiple levels steps. We then run vl_kdtreequeryfunction to compare these features to our vocabularies features to give us the distance between the features we found and its closest vocabulary feature. Finally, we make a histogram of these closest vocabulary features. These histograms are used to compare the training and the test images. So as to reduce the running time of the code, the histograms that are created are also save as a **.mat** file.

- **Support Vector Machine**

  Once the features are defined for the images, we need to classify the test images by comparing with the training images set. The classifier we used here is support vector machines.

  For each category that we are trying to define, we perform a one vs many strategies against our test images. We call vl_svmtrain against training images' features to generate weight and offset vectors. We then multiply our weights against our test image features and add the offsets to get our one vs many result. We place this result into a prediction matrix **(i.e., the confusion matrix)** as we go through each category and take the max to figure out which category the image is closest to our test image.

In our project first we trained the system with the training which consist of 200-300 images under each category.

**Testing is done as following:**

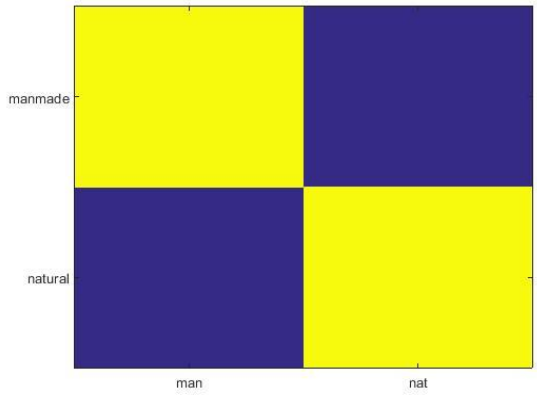
i. First the same set of images that are used for training (about 200-300 images per category) are used for testing and the output is observed.

ii. Later a new set of images (about 200-300 images per category) that are not included in the training set are given as an input for testing.

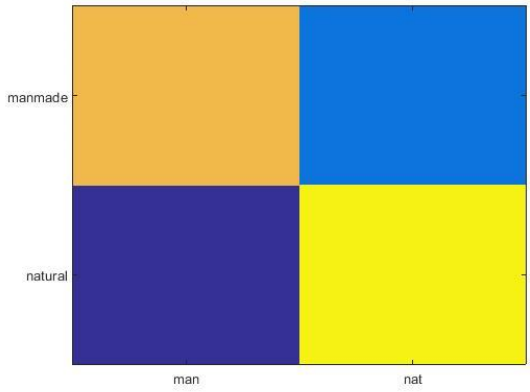
iii. Later when testing is done with single image, it is done at 3 different levels of the tree. As the levels are increased the numbers of strong features increase. In this project the code is made to run for levels 1, 2 and 3 and the output is observed.

## IV. RESULTS

- When the testing is done with same set of images that are used for training using **SIFT** and **GIST** the accuracy we got was **99% and 100%** respectively**.**

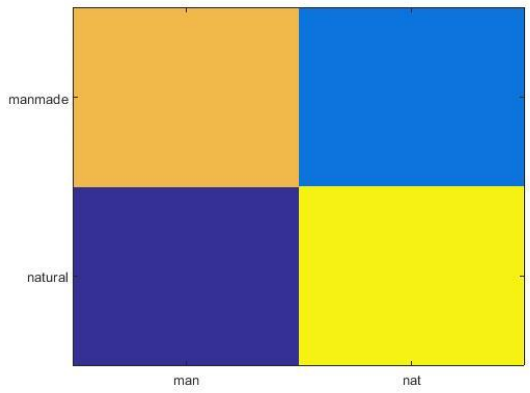



(i) Confusion matrix for GIST and SVM

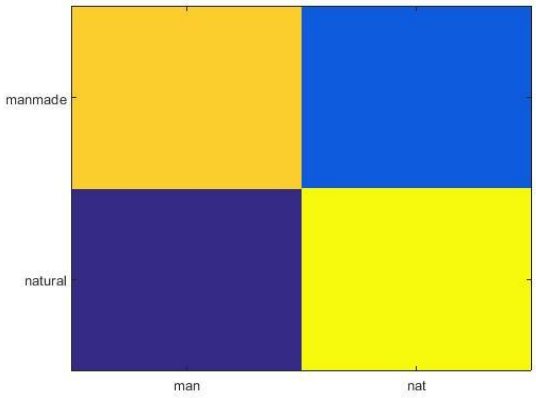



(ii) Confusion matrix for SIFT and SVM

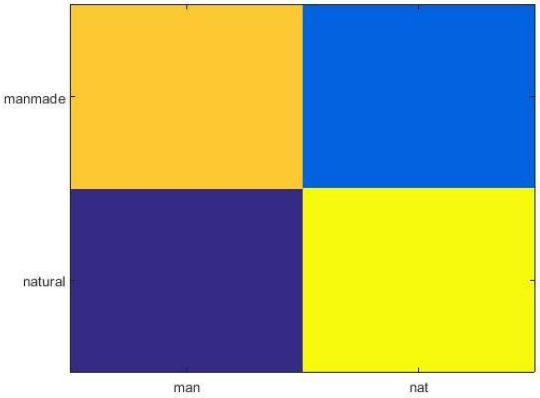
- When a new set of images are taken for testing, using **SIFT** we got an accuracy of **90%**, using **GIST** we got **93%** and using **SIFT + GIST**, we got approximately **95.6%**.




(i)Confusion matrix for SIFT and SVM

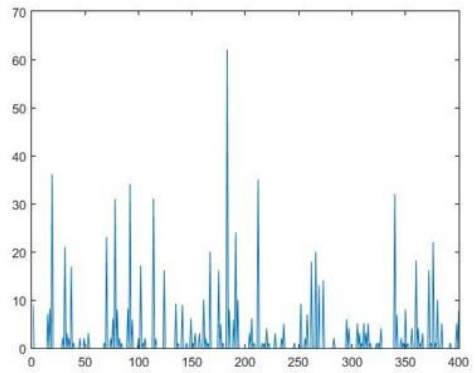



(ii)Confusion matrix for GIST and SVM

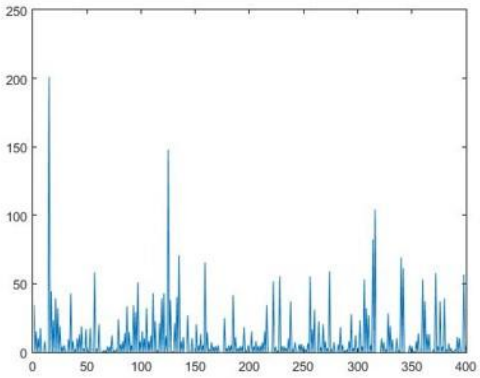



(iii)Confusion matrix for GIST + SIFT and SVM

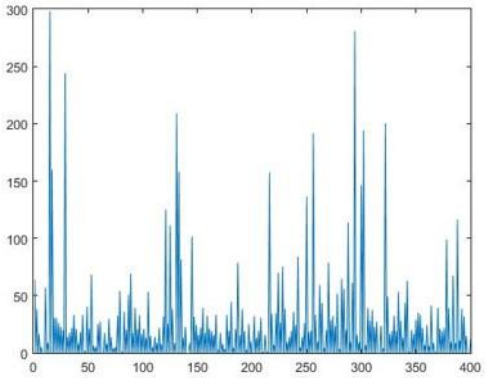
- When a single image is taken as an input the classification is done at three levels of the tree:
- At L = 1 the image is taken as a whole. The accuracy we got was **80%.**




- At L = 2 the image was divided into smaller parts. The accuracy we got was **96%.**




- At L = 3 the image was further divided into smaller parts. As we get more amount of strong features the accuracy we got was 9**4%.**




As this is binary classification, using SIFT or GIST feature extraction and SVM as classifiers gives us almost equal accuracy. But as the number of categories increase the feature matching becomes complex. In such cases using GIST or GIST + SIFT is more useful for feature extraction.

The accuracy increases as the number of levels increase.

As we go from level 2 to level 3, the image is finely divided into small portions. As it is fine divided, individual frames yield too less matches.

So the accuracy is more at L = 2 is more when compared to L = 3.

## V. Conclusion

The proposed model has been designed by using the scale invariant feature transform (SIFT) with the support vector machine (SVM) for the binary-class classification in the adaptive manner. The proposed binary-class enabled support vector machine (SVM) has been developed with the adaptive classification in the iterative manner. One primary class is compared against all other classes signified the secondary class for the realization of the support vector machine (SVM). In this project, the scale invariant feature transform (SIFT) along with the binary-class support vector machine (SVM) has been proposed for the natural and artificial scene recognition. The probabilistic classification with the binary-class support vector machine has been utilized for the robustness in the classification. For binary classification the chosen feature extraction and the classifiers are sufficient.

**Future scope**

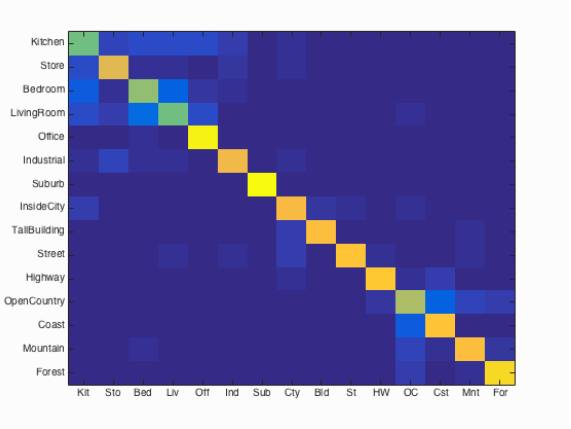
The scene classification can be extended for multi-class classification. This can be implemented at a higher level like using neural networks and generally in multi-class classification, the mistakes that the system tends to make is while classifying between

i. Coast and open country,

ii. Forest and mountain,

iii. Tall buildings and street,

iv. Office, living room and bedroom etc.




(i) confusion matrix acc. to the **Olivia Torralba** works.

## References And Links

[1] *Bosch, Anna, Andrew Zisserman, and Xavier Muñoz. "Scene classification via pLSA." Computer Vision–ECCV 2006 (2006): 517-530.*

[2] Gokalp, Demir, and Selim Aksoy. "Scene classification using bag-of-regions representations." *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*. IEEE, 2007.

[3] Hu, Junlin, and Ping Guo. "Combined Descriptors in Spatial Pyramid Domain for Image Classification." *arXiv preprint arXiv:1210.0386* (2012).

[4] Xiao, Jianxiong, et al. "Sun database: Large-scale scene recognition from abbey to zoo." *Computer vision and pattern recognition (CVPR), 2010 IEEE conference on*. IEEE, 2010.

[5] Tomašev, Nenad, and Dunja Mladenić. "MODIFIED K-MEANS ALGORITHM FOR FINDING SIFT CLUSTERS IN AN IMAGE."

[6] http://web.mit.edu/torralba/www/

[7] Li, Li-Jia, et al. "Objects as Attributes for Scene Classification." *ECCV Workshops (1)*. 2010.