Computer Vision
Multi-object Detection and Segmentation
Final Report

Rohil Bansal **Ishita Ganjoo** **Archana Molasi**

1. Introduction:

The main aim of this project is to compare the combination of various classifiers with the classification algorithms for multi object detection in a scene. And for the scope of this project, we coupled the Histogram of Gradients (HOG) with a Support Vector Machine (SVM) and the Scale Invariant Feature Transform (SIFT) with Convolutional Neural Networks (CNN). We also applied two segmentation techniques, Canny Edge detector and Otsu's method on the entire image to fine tune the exact boundary of the objects in the image.
The code is present at: https://github.iu.edu/bansalro/FinalProject

2. Background and Related Work:

A lot of research has been done on object detection and segmentation techniques. The state of the art techniques like R-CNN and Girshick's Fast R-CNN[5] efficiently classify objects using deep convolutional networks and have improved detection accuracy. When tested on VOC12, Fast R-CNN gives a MAP of 65.7%. For image segmentation, graph based methods like Markov Random Fields perform better segmentation as compared to other edge detection techniques.

The purpose of implementing feature extraction algorithms like SIFT and HOG and testing on different classifiers in this project was to understand the basic building blocks of the state of the art techniques. Also, our focus was on evaluating and comparing the classification results of SVM when coupled with HOG and CNN with SIFT.

3. Methods

3.1 Histogram of Oriented Gradients (HOG)

Histogram of Gradients[3] is a famous image feature descriptor in Computer Vision. The idea behind Histogram of Oriented Gradients is that the local object appearance and shape within an image can be described by the distribution of intensity gradients and edge directions.

3.2 Support Vector Machines (SVM)

SVMs are trained with Radial Basis Function kernel for multiclass classification of 20 classes. SVMs are trained with the HOG training data generated for all the PASCAL VOC training images.

3.3 Implementation and Results (HOG):

To prepare training data following steps are performed:

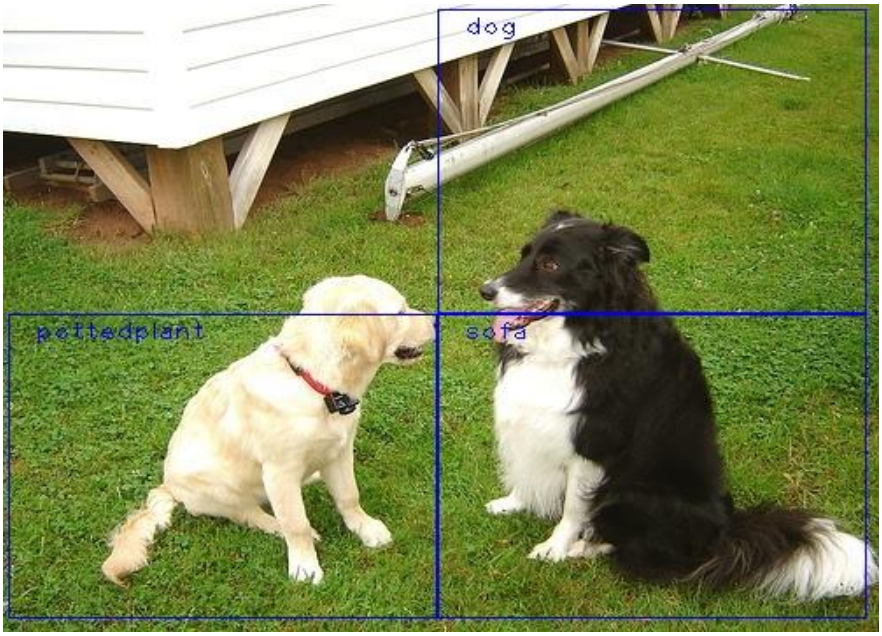
- Extract patches of objects from the training images by parsing the xml "Annotations" file of each training image.
- Calculate HOG features for each object patch with a stride of (64, 128) and padding of (8, 8).

The following steps are performed to test an image for object detection:

- Slide a window of size (width/3, height/3) across the image.
- Calculate HOG for each sliding window and for 48 adjacent windows varying from (c-3, r-3) to (c+3, r+3) where (c, r) is the co-ordinates of the top left corner of the image window.
- The adjacent 48 windows and the original window patch predicts a class for the sliding window. If more than 35 predictions are same for a class, we assign the class to the sliding window.

We have trained the SVM model and placed it in the Computer Vision folder at
https://iu.box.com/s/83jplufus4cs9cj55xo58oreaomdczlc
To run the Hog+SVM code:

- Go to the folder HoG+SVM
- Make all
- Run from command line:
  ./HogSVM Test



3.4 Scale Invariant Feature Transform (SIFT)

David Lowe[2] introduced the idea of Scale Invariant Feature Transform and used this idea for object detection. After generating SIFT points for training images, we found that the density of the points along the objects in an image is very high as compared to the rest of the regions. Some examples with sift points are shown below:

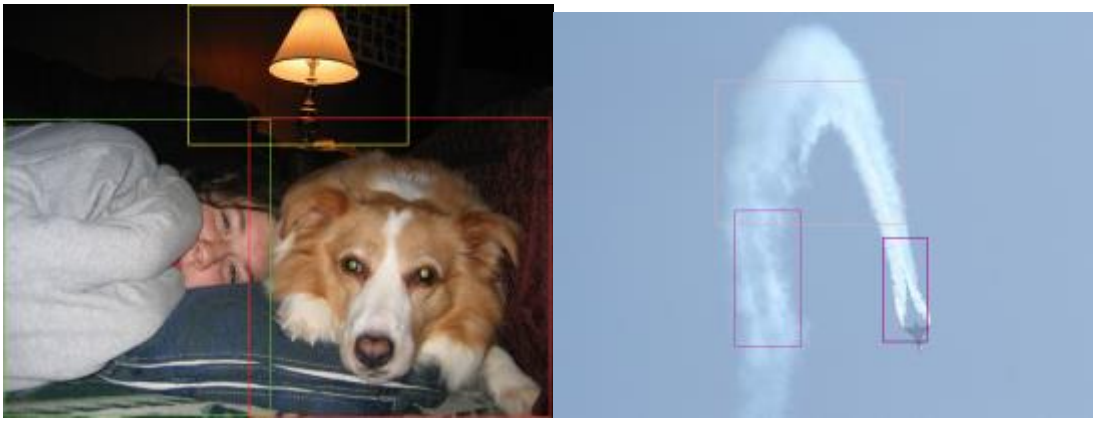

3.4.1 Clustering of SIFT points

Since a lot of SIFT points are generated on an image, we try to cluster those points and try to imagine each cluster as an object in an image. We used the hierarchical clustering model for clustering the SIFT points.

3.4.1.1 Hierarchical Clustering

We have used Agglomerative clustering technique which is a bottom-up clustering approach. Initially, we assumed each SIFT point to be a cluster and then we recursively we join the two nearest points or clusters based on the Euclidean distance. If a cluster contains more than one point, then its distance from the other clusters is the distance between their centroids i.e. the average of the sum of x and y coordinates.

We employed a very naive approach when deciding the number of clusters to be formed. We assumed that at most there'll be only three objects in a scene and hence, we stopped the clustering algorithm as soon as three clusters are formed and then we create the bounding boxes on those three clusters.



3.4.2 Extracting the regions and Labelling (CNNs)

Once we have the clusters marked on the image, we crop out those regions from the image and send it to a Convolutional Neural Network for labeling of the object. We used Caffe – a deep learning framework for labelling and classification of our objects.

3.5 Implementation and Results (SIFT):

3.5.1 How to run SIFT code

- Go to the folder Sift+CNN.
- Make all
- Run from command line:
  ./SiftCNN

This code by default runs in Test mode.

3.5.2 Caffe Training

- To train Caffe[4], we used a training set of 5,011 images.
- Extract object patches from each image by parsing through the "Annotations" xml file.
- Each extracted object is placed in its respective folder. This way we had 20 folders, one for each class.
- We perform Caffe training for 25000 iterations using the scripts provided by Bardia Doosti.
- The Caffe model and solver state are placed in the ComputerVision folder at https://iu.box.com/s/83jplufus4cs9cj55xo58oreaomdczlc

3.5.3 Results

- We have performed testing on 100 images.
- We count the number of correctly classified clusters out of the total number of clusters (in our case total clusters = 3)

- We obtained an accuracy of 49.3% by using the above approach.
- We also used another approach to calculate the accuracy of the classifier, named as union over intersection, where we calculate the overlap area between the ground truth box and the detected rectangular box.
- We initially considered a localized object to be correctly detected if the predicted class matched with the ground truth and the overlap area was greater than 50% but this approach gave a poor accuracy due to significant difference in the box coordinates, hence we carried out testing using the former approach.

4. Segmentation

Image segmentation is the process of partitioning an image into multiple segments. It is used to typically locate objects and boundaries in images. The result of image segmentation is a set of segments that collectively cover the entire image, or a set of contours extracted from the image.

As per our analysis, we found that image segmentation can be performed in several ways, below are some of the techniques that can be used:

- Clustering
- Thresholding
- Edge Detection
- Graph partitioning methods
- Histogram based methods

We have used Canny edge detection and Otsu's method to perform segmentation on the images.

4.1 Methods of Segmentation

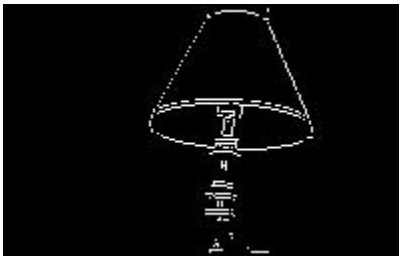
4.1.1 Canny Edge Detection

Canny edge detection[6] is a technique to extract useful structural information from different vision objects. We performed the following steps as part of the edge detection algorithm:

1. Convert the input image into grayscale image.
2. Apply Sobel filter to the image.
3. Find the image gradients by computing the magnitude and orientation at every pixel.
4. Now apply non-maximum suppression to further enhance the edges in the image. In non-maximum suppression, we compare the value of the current pixel with its neighbors and add the value to the final matrix if it's greater than its neighboring pixels.
5. Pass the output matrix obtained from the above step to hysteresis thresholding. In hysteresis thresholding, we apply dual thresholding by setting an upper and

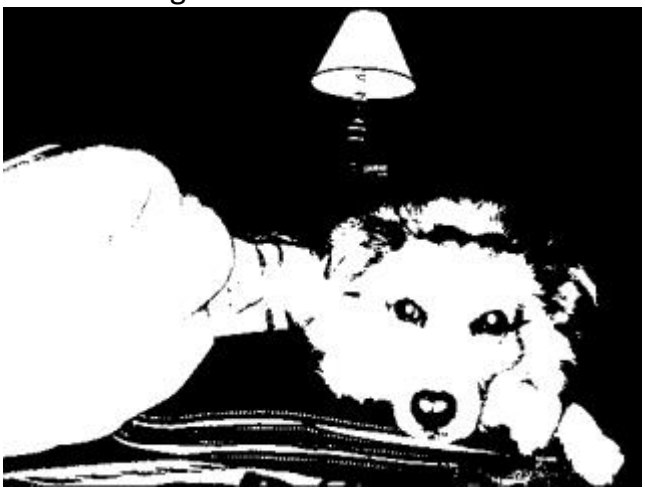
lower threshold.

We also tried the Canny edge detector on the localized objects and the results are as below:



4.1.2 Otsu's Method

Otsu's[1] image segmentation comes under the histogram based thresholding methods of image segmentation. The algorithm divides an image into two parts by finding a threshold which minimises in-class variance. This algorithm works well for images which have two classes of image pixels.



4.2 How to run Segmentation

- Go to the Segmentation folder and keep the input images under this folder.
- Make all
- To run Otsu, enter from command line:
  ./Segment otsu "InputFile"
- To run Canny, enter from command line:
  ./Segment canny "InputFile"

5. Dataset

PASCAL VOC 2007 Dataset

The PASCAL VOC 2007 is a supervised learning dataset for object detection and segmentation. This PASCAL challenge was held in 2007 and the goal of this challenge was to recognize several visual object classes in realistic scenes (i.e. not pre-segmented objects). The twenty object classes in this challenge were:

- Person: Person
- Animal: Bird, Cat, Cow, Sheep, Dog, Horse
- Vehicle: Aeroplane, Bicycle, Boat, Bus, Car, Motorbike, Train
- Indoor: Bottle, Chair, Dining Table, Potted Plant, Sofa, TV/Monitor

6. Conclusion

After thorough analysis of the results of both the classifiers, we can conclude that CNN gives a better classification rate than SVM and achieves an accuracy of 49.3% when tested on 100 images.

8. Improvements & Future Scope

- SIFT Clustering: We have employed a very naive approach where we assume that each image will contains three clusters. A more robust technique is needed in which the number of clusters in an image can be decided based on the density of points over the entire image.
- Hog Window: The sliding window approach used while calculating Hog features can be modified and made more robust. Currently, we have used a window of a fixed size. By using windows of multiple sizes over the image can lead to better feature extraction, further enhancing the accuracy of SVM.
- SVM: The SVM can be further improved by experimenting with the parameters used for SVM model creation.
- The segmentation results can be significantly improved by using techniques like Semantic texton forests where each pixel is assigned a class label and the classification and segmentation is done using Random Decision Forests.

9. References

[1] A C++ Implementation of Otsu's Image Segmentation Method Juan Pablo Balarini, Sergio Nesmachnow

[2] D. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. International Journal of Computer Vision 60(2), 91–110, 2004

[3] Navneet Dalal and Bill Triggs. Histograms of Oriented Gradients for Human Detection

[4] Caffe from caffe.berkeleyvision.org

[5] Fast R-CNN: http://www.cv-foundation.org/openaccess/content_iccv_2015/html/Girshick_Fast_R-CNN_ICCV_2015_paper.html

[6] John Canny," A Computational Approach to Edge Detection,"IEEE Transactions on Pattern Analysis and Machine Intelligence vol.pami- 8, no. 6, November 1986