COMPUTER VISION

CW-02

Presented By:
Monish Manjunatha - 100286321
Pranav Gujjar - 100443924

Introduction

- In the previous coursework, we classified scenes into one of various categories (one of Kitchen, Store, industrial, tall building etc.)
- Using Color Indexing' method method described by Swain and Bal- lard (S&B) based on image's color histogram (similar color histogram = similar scenes)
  - We tweaked the parameters – Color spaces, quantization, etc
- Also used Tiny Images , where we be down sampled to a 16 × 16 thumbnail , where these 256 colors were stretched to a long feature vector
  - Again we tweaked color spaces, normalized it etc
- Used KNN as the classifier tweaking values of K and the distance metrics

Recap of Previous Findings:

- **Color Histograms with HSV**: Achieved a peak accuracy of 35.6% with a k = 15 and **Tiny Image Technique with RGB**: Reached a maximum accuracy of 34.6% using a [4x4] output size.
- Conducted Lit review into SURF, SIFT, ORB, Bag of Features and Spatial Pyramids and Deep learning methods to enhance accuracy

WHY ?

- **Generalization Issues**: Both methods capture global color distribution or coarse spatial resolution without considering local patterns and textures, which are crucial for distinguishing between similar categories.
- **Sensitivity to Variations**: Color histograms and tiny images are sensitive to variations in lighting, viewpoint, and background, which can significantly affect the color distribution and the resized image representation.
- **Lack of Discriminative Power**: These techniques do not capture distinctive local features that can differentiate between complex image categories effectively.

Objectives:

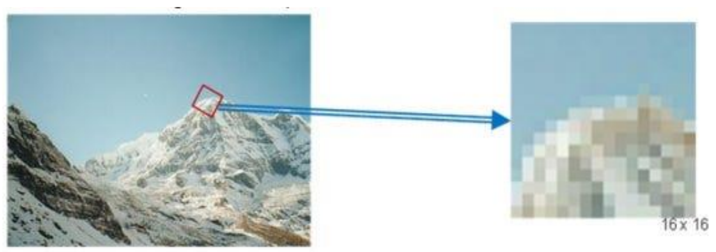
- Improve the accuracy and efficiency of image classification on SUN database
- Need for advanced Feature Extraction Methods
  - **SIFT** : Scale-Invariant Feature Transform (SIFT) captures key points that are invariant to image scale and rotation, and it provides robust matching across a substantial range of affine distortions, changes in 3D viewpoint, addition of noise, and changes in illumination.
  - **Bag of Words Model**: Transforms these features into a fixed-size vector using a visual dictionary, which can capture more meaningful and discriminative information from the images, potentially improving classification accuracy and robustness.
  - **+ Enhanced Feature Representation** ;
  - **+ Better Handling of Variability (lighting and Viewpoints)**
  - **+ Scalable: Use spatial pyramids to capture image layout**

Overview of New Implemented Techniques:

1. Discussion of Bag of Sift Feature , Implementation, Result Presentation
   - KNN classifier with Grayscale sift
   - SVM classifier with Grayscale sift
   - KNN classifier with sift + color
   - SVM classifier with sift + color
2. Discussion of Spatial Pyramids with sift Feature, Implementation, Result Presentation
   - KNN classifier with Grayscale sift
   - SVM classifier with Grayscale sift
   - KNN classifier with sift + color
   - SVM classifier with sift + color
3. Discussion of Histogram of Gradient Feature, Implementation, Result Presentation
   - Random forest classifier with Grayscale
   - Random forest classifier with Color

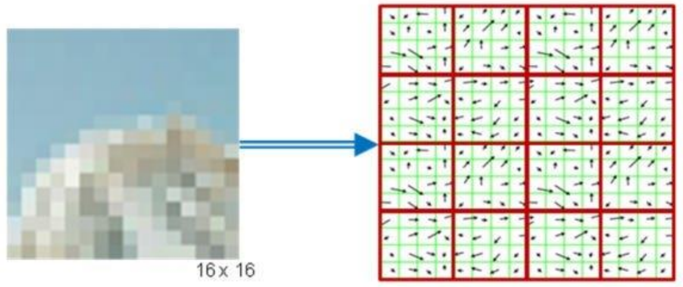
1. Bags of Sift Feature:

- Working
- Function Implementation
- Result Presentation

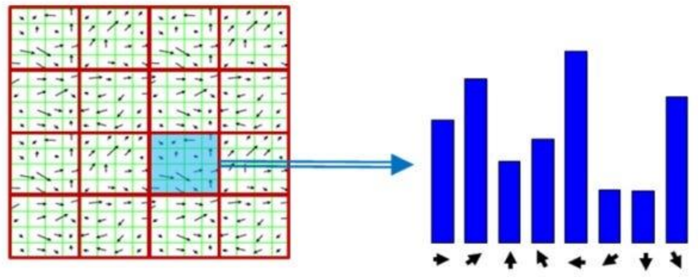


Feature extraction with Sift

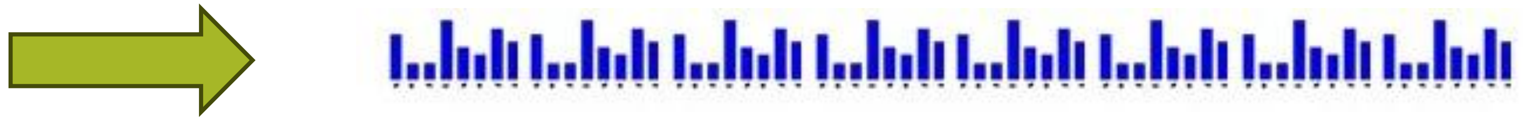
- Detect and identify Key points (extremas of DoG), scale and resize the region to a 16 x 16 pixels.
- SIFT vector = (PSR)F Location ; Scale; Orientation ; 128D descriptor



- 16 x 16 pixels will be **divided into 16 4x4 pixel squares, for each square** SIFT **will produce a gradient vector (in 8 directions)**



- The 16 histograms (one for each square) will be concatenated to obtain a 128-dimensional feature vector(16*8) where 16 is the total number of histograms and 8 represents the number of vector directions.



Bagging of Sift Features :

- The Bag of SIFT features technique, as explained in detail by Lazebnik and colleagues in 2006, converts images into a consistent representation of a fixed size, making tasks such as image classification and object recognition easier.

- **Creating a Visual Vocabulary**
  - First detect and describe unique Key points in the image using the Scale-Invariant Feature Transform (SIFT).
  - These descriptors are clustered to form a visual vocabulary. The k-means clustering algorithm creates k clusters, the centroid of each representing a “Visual word” in the vocabulary. The vocab thereby captures the most prominent and recurring local patterns.

- **Quantizing Features into Histograms**
  - For each new image to be classified, SIFT descriptors are again extracted by associating each of them with the closest cluster center in predetermined vocabulary of visual words.
  - Following this process, as explored by Sivic and Zisserman in 2003, the descriptors are quantized to each visual word resulting in a histogram, where each bin is a word, and the value is the count of descriptors of each image assigned to that word.

- **Next, Use a Classifier algorithms to compare and classify images based on their visual content of these histograms**

- Build vocabulary function Implementation:
- Bag of sift feature function Implementation:

Build Vocabulary function implementation:

- The "build_vocabulary1" function creates a collection of visual words by extracting SIFT descriptors from a set of training images.
- It runs through each image, dealing with both grayscale and color images.
- If an image has colors, it separates them and extracts descriptions for each color channel.
- Then it merges all of the descriptors into a single list.
- After putting these descriptors into an appropriate format, it uses a clustering technique called k-means to group them into clusters, resulting in the visual vocabulary.
- This vocabulary represents common visual patterns identified in training images and is utilized to better understand new images.

Bag of Sift Feature Implementation:

- The get_bags_of_sifts1 function first initializes feature storage matrices according to the quantity and vocabulary size of images.
- After loading each image sequentially, it preprocesses them in accordance with predefined modes, processing each color channel independently in the case of a "color" mode and converting to grayscale in the other case.
- The function uses the functions in the VLFeat package to extract SIFT descriptors either over regular intervals or the full image, depending on whether standard or Dense SIFT is given.
- The closest vocabulary word for each descriptor is then found by using vl_alldist2 to calculate the distances between them.
- The results of these matches are used to create a histogram that shows the number of times each visual word appears in an image.
- This histogram, which is kept in the image-corresponding matrix row, basically functions as a "bag of words" to represent the features of the image.

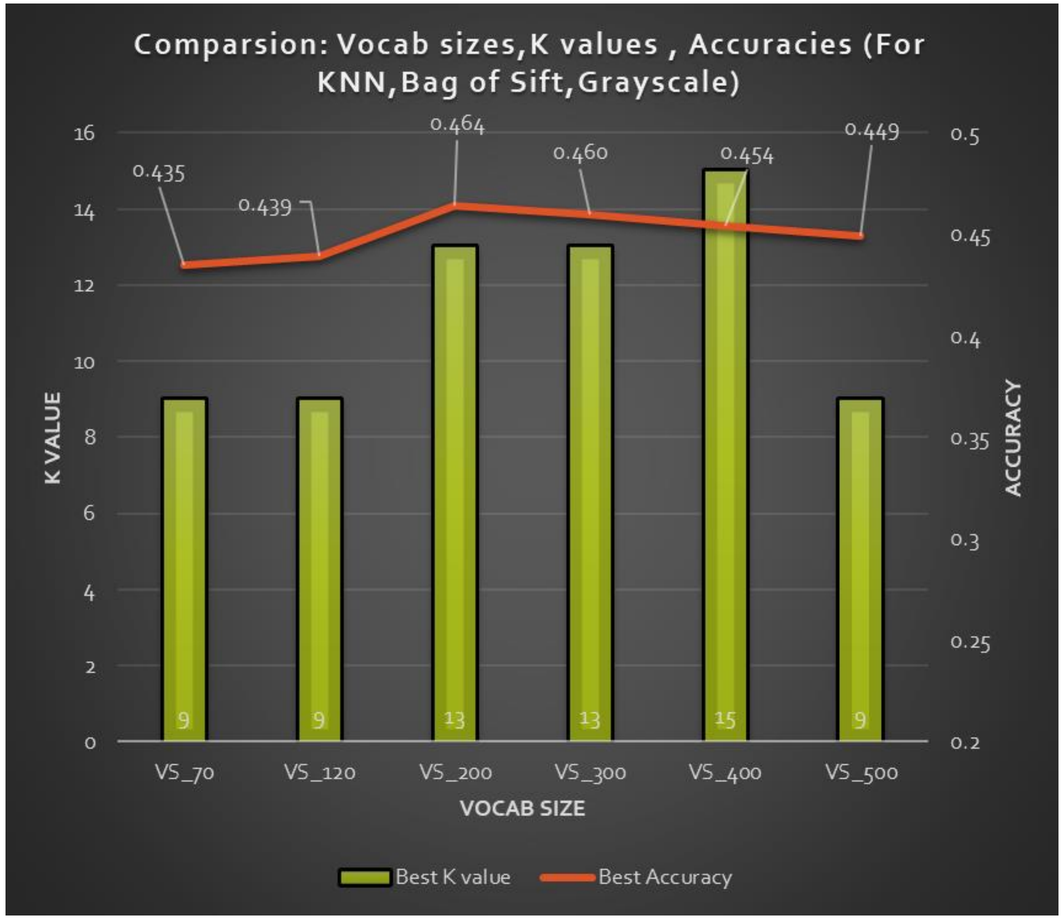
DENSE SIFT VS SIFT

Feature	Regular SIFT	Dense SIFT
Keypoint Detection	Detects keypoints at locations of high contrast and interest.	Computes descriptors at fixed intervals over a grid.
Scale and Orientation	Operates at multiple scales and orientations.	Typically computed at a predefined scale and orientation.
Descriptor Calculation	Descriptors based on gradient orientations around keypoints.	Similar descriptors, but computed densely across the grid points.
Robustness	Highly robust to changes in scale, rotation, and lighting.	Provides comprehensive texture information, less focus on robustness.
Applications	Ideal for image matching, object recognition, 3D reconstruction.	**Suited for texture recognition, image segmentation, Scene classification tasks.**
Feature Distribution	Sparse and based on image content (adaptive).	Uniform and comprehensive coverage of the image area.
Computational Load	Depends on the number of detected keypoints.	Generally higher due to dense feature extraction.

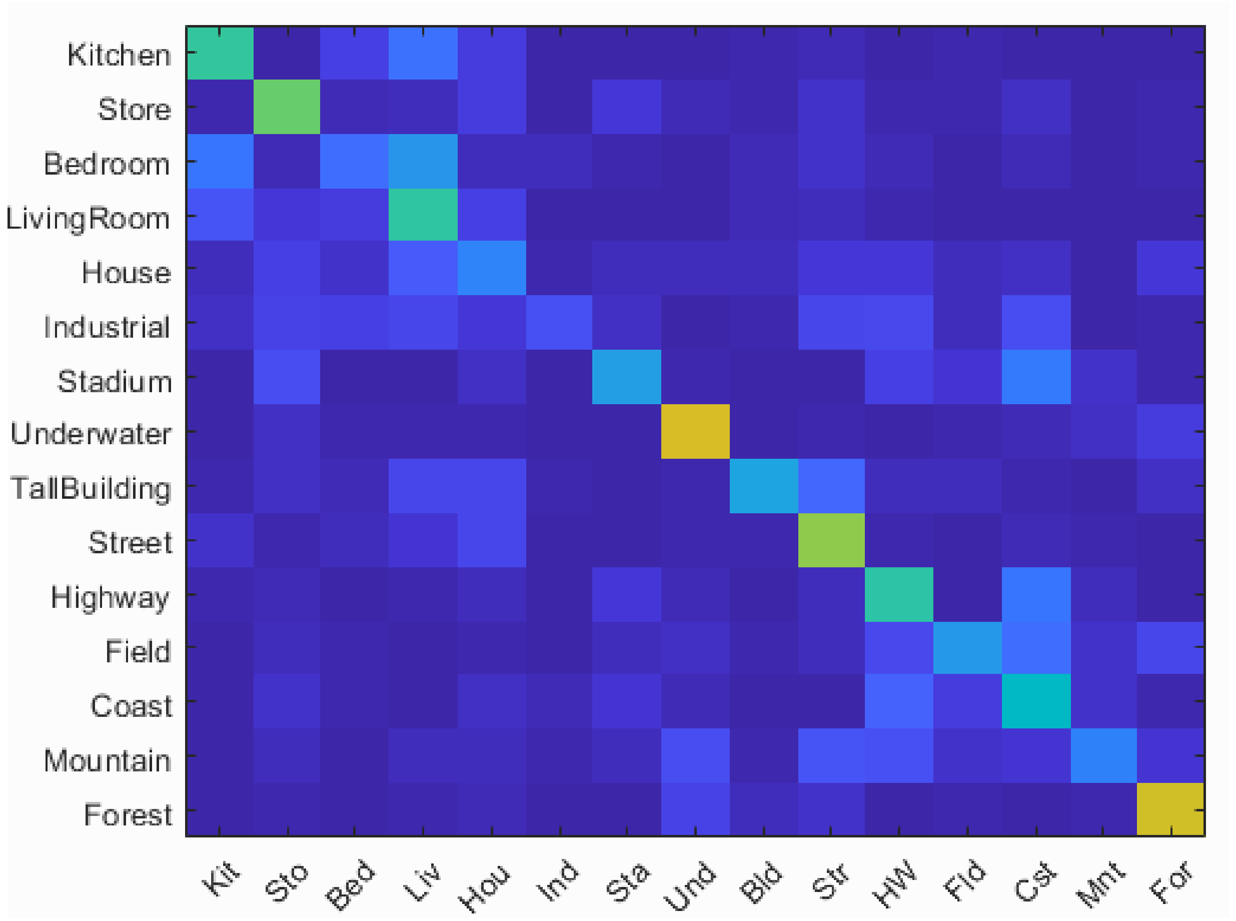
- Result Presentation of Bag of Sift Feature

1. Bag of Sift Feature: Result Presentation

- KNN classifier with Grayscale sift



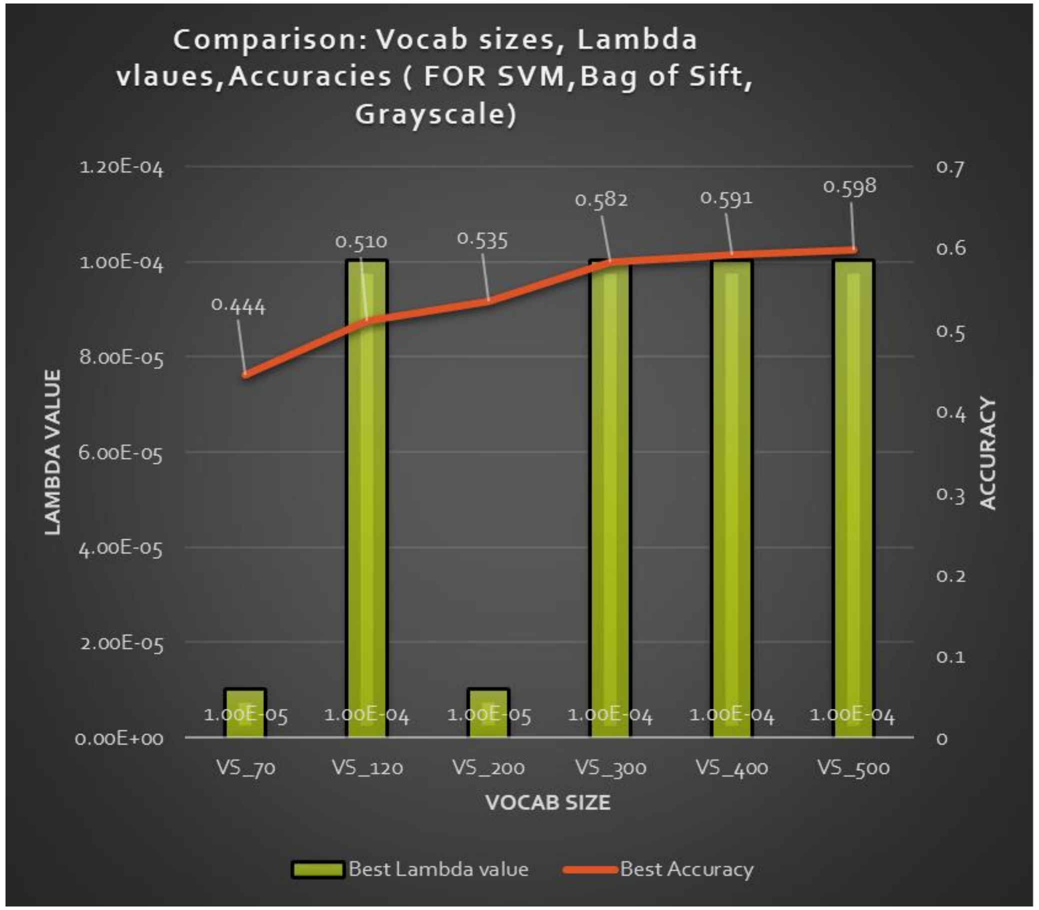
- Confusion Matrix:
- Vocab size : 200 , K = 13 , Accuracy = 0.464



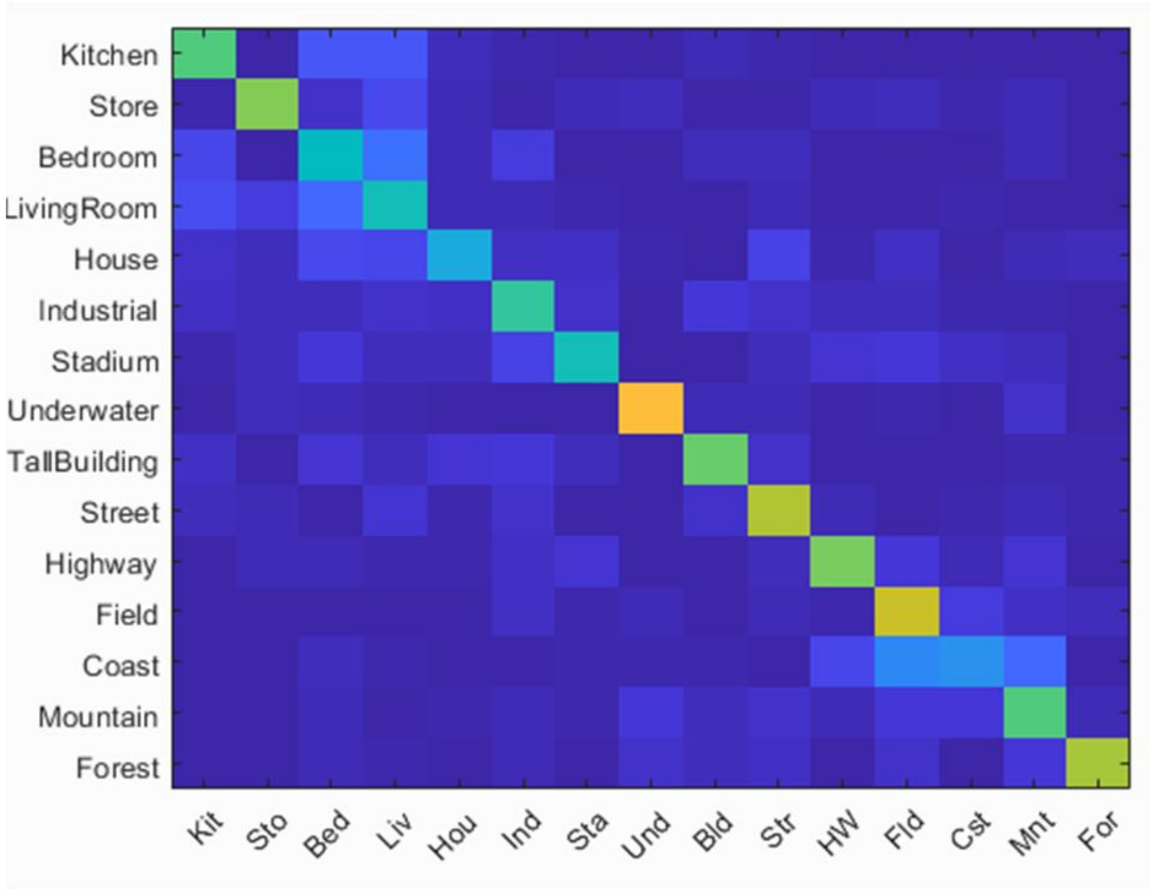
- DSIFT parameters: Fast, step size =4 , bin size = 4,without Normalize

1. Bag of Sift Feature: Result Presentation

- SVM classifier with Grayscale sift



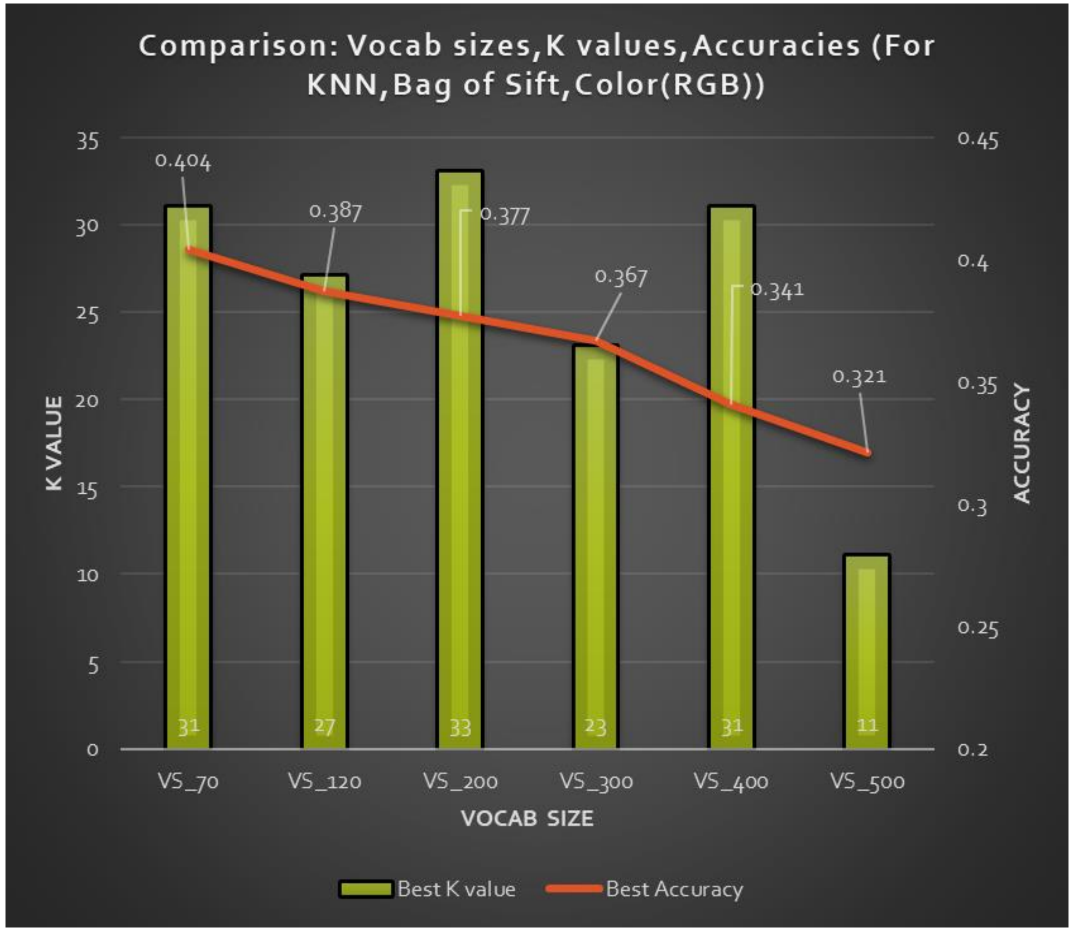
- Confusion Matrix
- Vocab size : 500 , L = 1.0000e-04 , Accuracy = 0.598



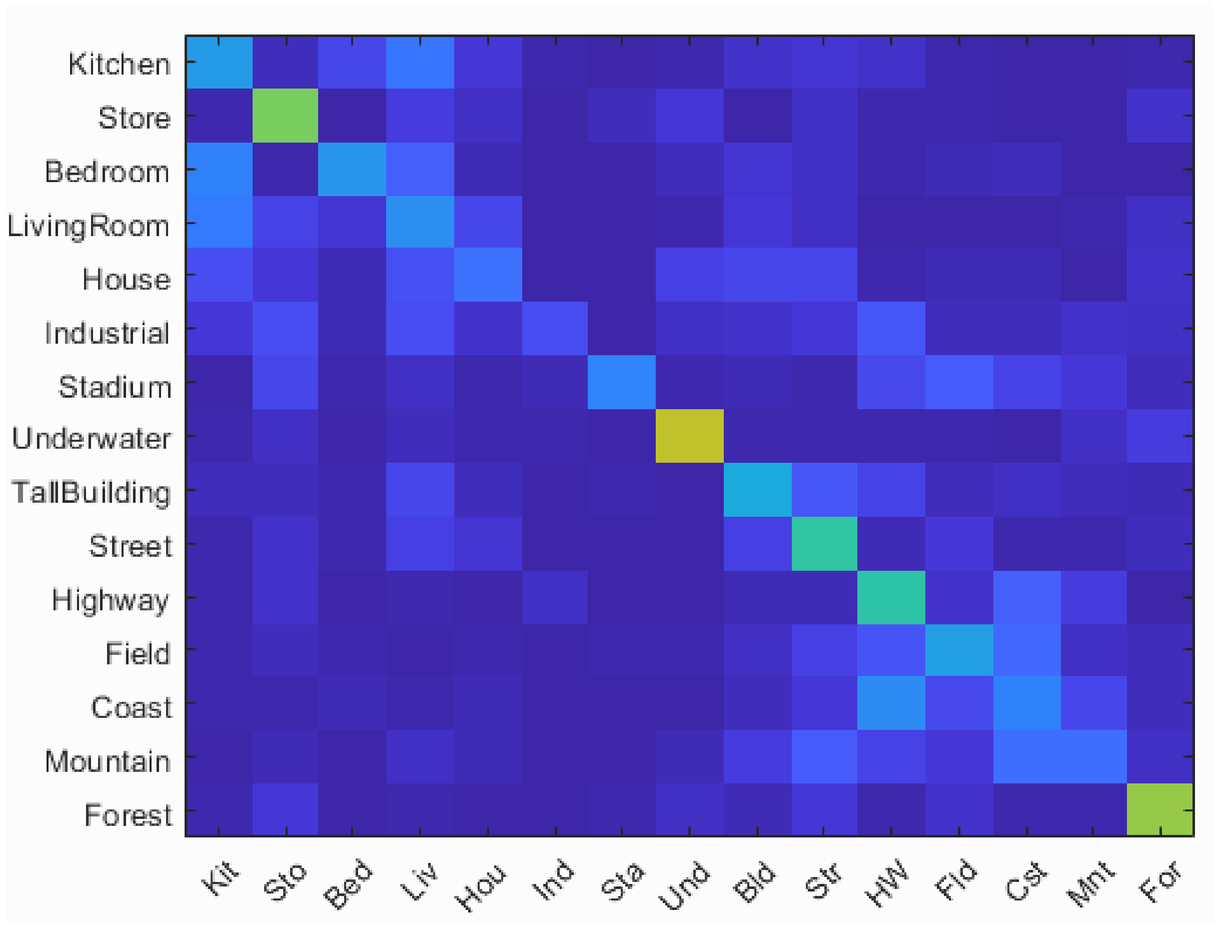
- DSIFT parameters: Fast, step size =4 , bin size = 4,without Normalize

1. Bag of Sift Feature: Result Presentation

- KNN classifier with sift + Color(RGB)



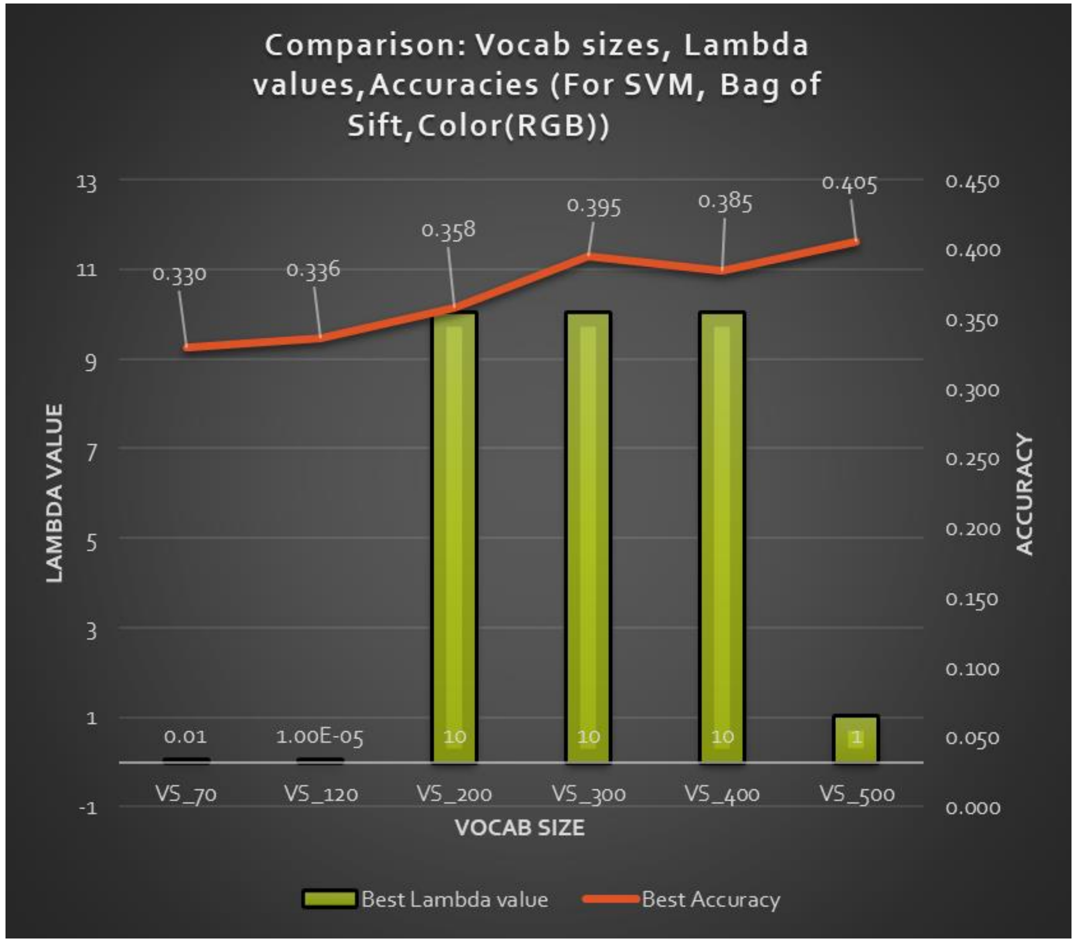
- Confusion Matrix
- Vocab size : 70 , K = 31 , Accuracy = 0.404



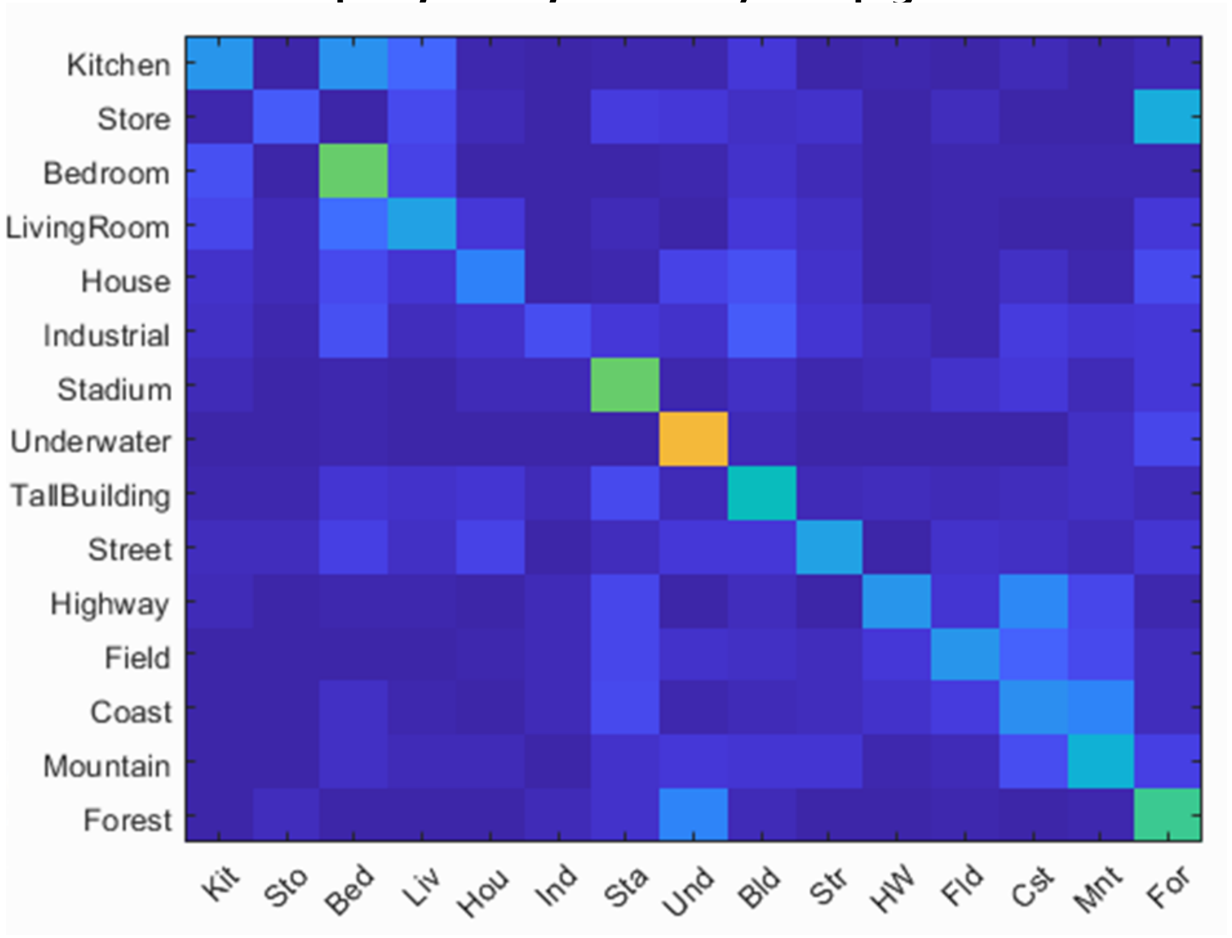
- **SIFT parameters:PeakThresh =0, EdgeThresh=10,NorThresh=-inf,Magnif=3,windowsize=2,without Normalize**

1. Bag of Sift Feature: Result Presentation

- SVM classifier with sift + Color(RGB)



- Confusion Matrix
- Vocab size : 400 , L = 1 , Accuracy = 0.405



- SIFT parameters:PeakThresh =0, EdgeThresh=10,NorThresh=-inf,Magnif=3,windowsize=2,without Normalize

Classifiers:

- K- nearest neighbor (KNN)
- Support Vector Machine (SVM)

K- Nearest Neighbor (KNN)Classifier:

- The nearest_neighbor_classify function applies the k-nearest neighbors technique to classify images by assessing a range of odd k values up to a user-specified maximum.
- This function computes the pairwise distances between training and test image feature vectors in order to find the nearest neighbors.
- For each odd k, the function determines the most common label among the k closest training image labels for each test image and evaluates the accuracy of these predictions by comparing them to the actual labels.
- It records the accuracies of all tested k values and selects the k with the highest classification accuracy.
- The function returns the predicted categories for the test images, the best k value, the maximum accuracy, and a table of accuracies for all analyzed k values, giving an in-depth understanding of how different neighborhood sizes effect classification performance.

Support Vector Machine (SVM) Classifier:

- The svm_classify1 function uses a linear SVM classifier to categorize images based on their characteristics. Using a 1-vs-all strategy across several image categories, the function compares multiple regularization levels (lambda values) to discover the best configuration for SVM training.
- SVMs are trained for each lambda, using binary labels to differentiate the target category from others.
- The function then runs these models on a test dataset, calculating the accuracy of each lambda by comparing predicted labels to true labels.
- Finally, it selects the lambda with the highest accuracy and applies this model to produce final predictions.
- The function returns the best lambda, highest obtained accuracy, and predicted categories for the test images, providing information about the usefulness of various regularization degrees in image classification tasks.

Why SVM performs better than KNN ?

1. **Model complexity and decision boundaries:**

- SVMs are effective in high-dimensional spaces, as is typical with image data processed with SIFT, where each image is represented as a high-dimensional vector *(Boser, Guyon, and Vapnik , 1992) .*
- In high-dimensional spaces, distance metrics of KNN becomes less effective due to the curse of dimensionality, where the volume of the space increases so much that the available data becomes sparse. This sparsity makes distance measurements less meaningful, explaining the lower performance of KNN compared to SVM *(Hastie and Tibshirani ,2001).*

2. **Generalization vs. Overfitting:**

- SVMs λ parameter allows for robust against overfitting and the kernel helps better differentiate the hyperplane. This prevents overfitting *(Boser, Guyon, and Vapnik , 1992) .*
- KNN classifiers have no underlying model other than storing the entire dataset, it susceptible to noise in the training data. Furthermore, the choice of k (the number of neighbors) and the distance metric can significantly impact performance but do not prevent overfitting *(Hastie and Tibshirani ,2001).*

Why Adding Color is Not Improving Accuracies ?

1. Dominance of Textural Features:

In scene recognition tasks, texture provided by grayscale SIFT features are more discriminative than color. Color variations can be subtle and susceptible to changes in lighting and shadows, whereas textural patterns provide more consistent cues for distinguishing between scene categories.

2. Colour Space and Representation:

The choice of colour space (RGB in our case) can significantly impact the effectiveness of colour features. RGB is sensitive to changes in lighting and may not be the best representation for capturing colour information that is useful for scene classification. Other colour spaces like HSV or YCbCR might provide features that are more robust and discriminative for scene recognition (*Khan et al. ,2012)* .

Due to time constraints this was not further investigated

Spatial Pyramid + Sift Feature:

- Introduction
- Function Implementation
- Result Presentation

2. Spatial Pyramid Feature :

- Spatial pyramids, first described by Lazebnik et al. (2006), improve on the standard bag of words structure by including spatial layout into the feature extraction process, resulting in more comprehensive spatial information about images.
- The approach separates each image into smaller sections at successive levels of a pyramid, from which local features are retrieved and quantified.
- This multilevel technique provides a more complete representation of the image by incorporating both global and local structural information. The multi-level feature extraction allows for capturing both coarse and fine spatial relationships between features, enhancing the discriminative power of the feature set.
- As Grauman and Darrell (2005) discovered, combining these features from different pyramid levels considerably enhances performance in tasks like scene classification and object recognition by taking into account not just the features but also their spatial arrangement within the image.

- Spatial Pyramid feature function Implementation:

2.Spatial Pyramid Feature Implementation:

- The get_spatial_pyramids function begins by converting a visual vocabulary to single precision and creating an array to store feature vectors for each image, based on the number of images and visual words.

- It processes each image at various pyramid levels, converting to grayscale as needed, and dividing it into increasingly smaller cells.

- Depending on whether Dense SIFT (DSIFT) or conventional SIFT (SIFT) is used, it extracts SIFT descriptors from each cell, matches them to the closest visual words, and generates histograms of these word instances.

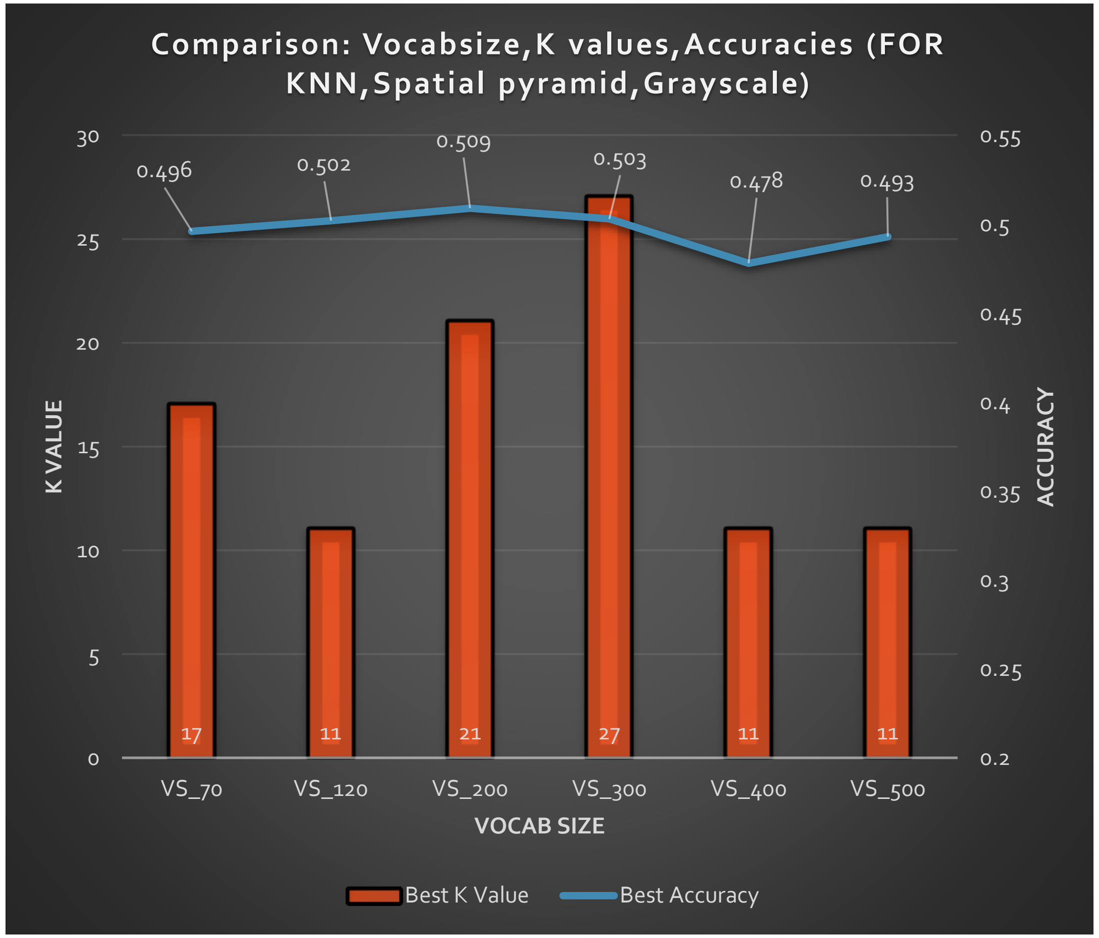
- These histograms are then weighted by pyramid level and merged to create a normalized feature vector for each image, which includes both detailed textures and larger structures.

- The function works well with both grayscale and color images, allowing various modes and SIFT types to provide a strong spatial feature representation beneficial for image classification and retrieval tasks.

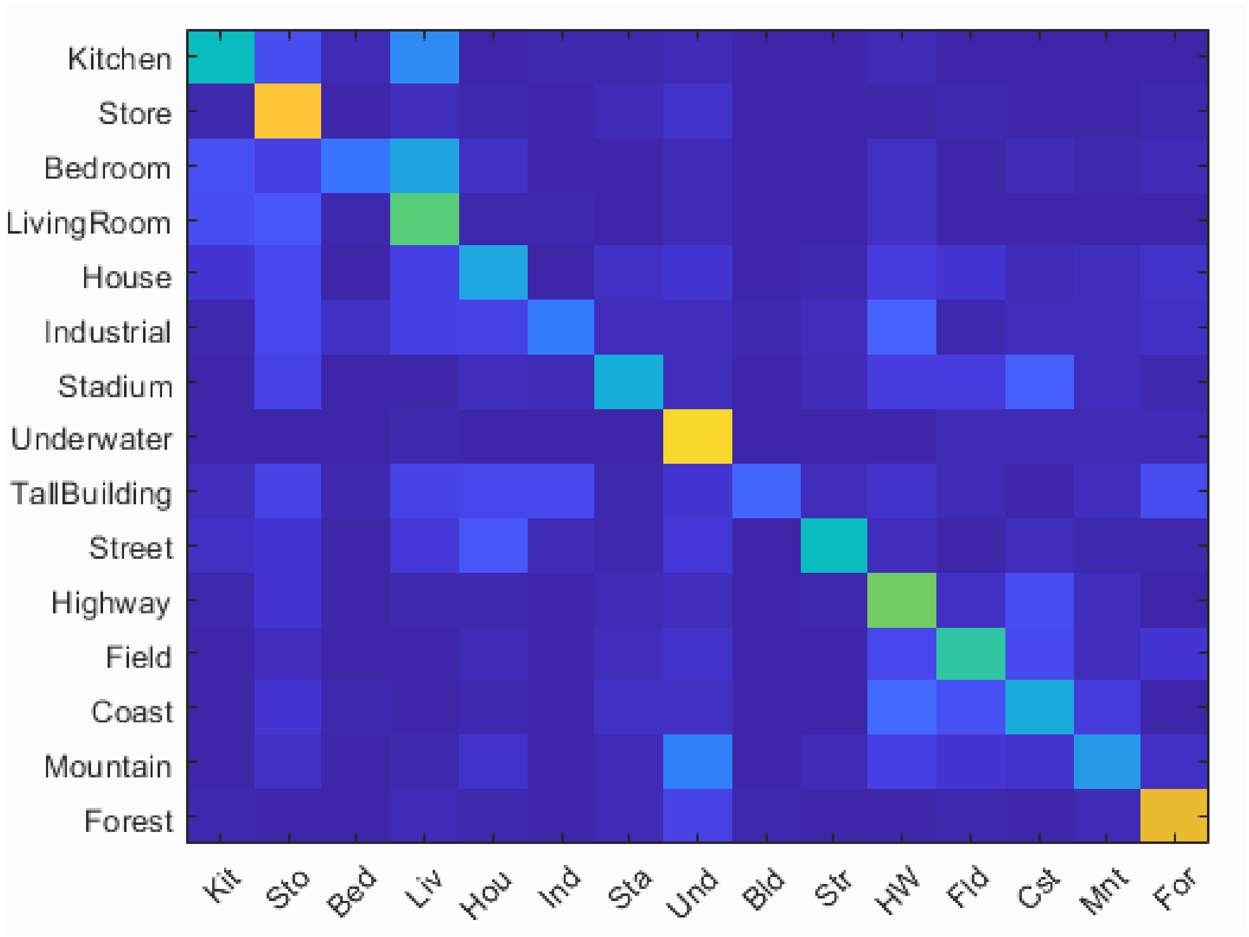
- Result Presentation of Spatial Pyramid + Sift Feature

2. Spatial Pyramids + Sift Feature: Result Presentation

- KNN classifier with Grayscale sift



- Confusion Matrix
- Vocab size : 200 , K = 21 , Accuracy = 0.509

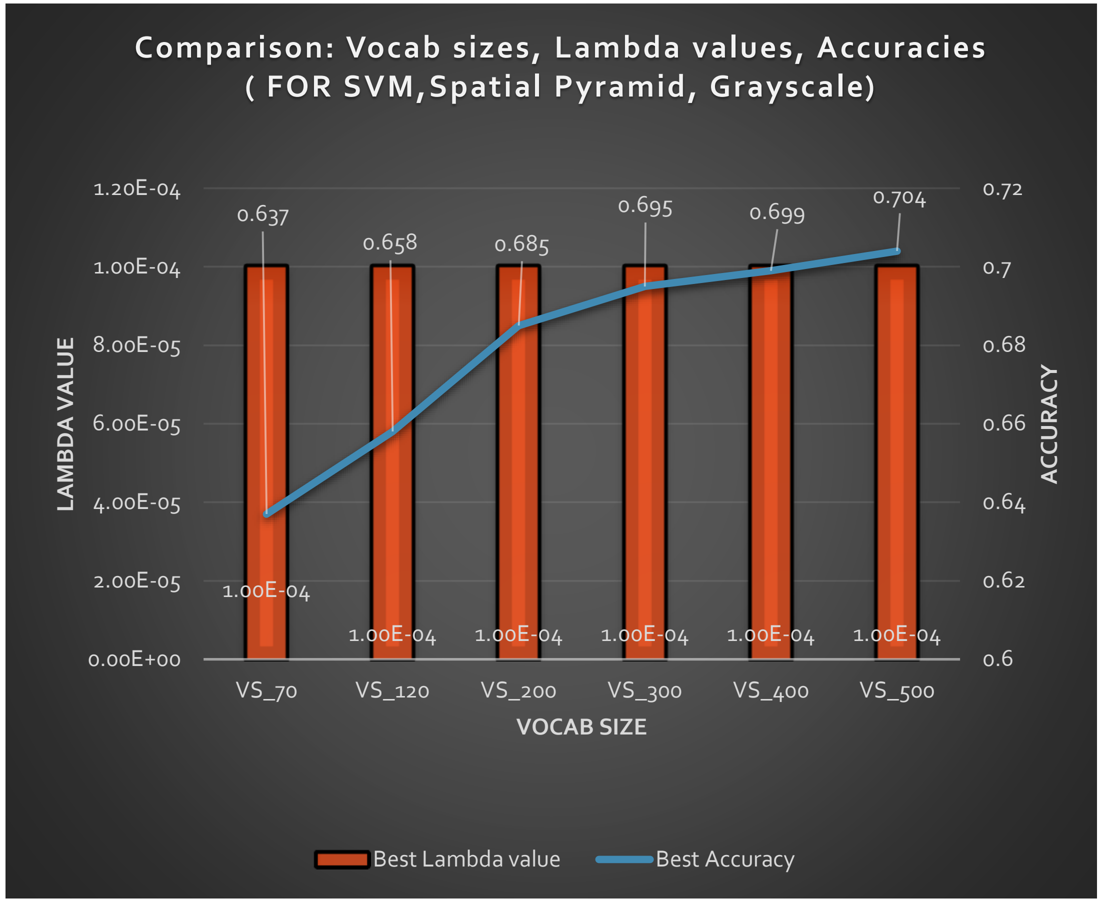


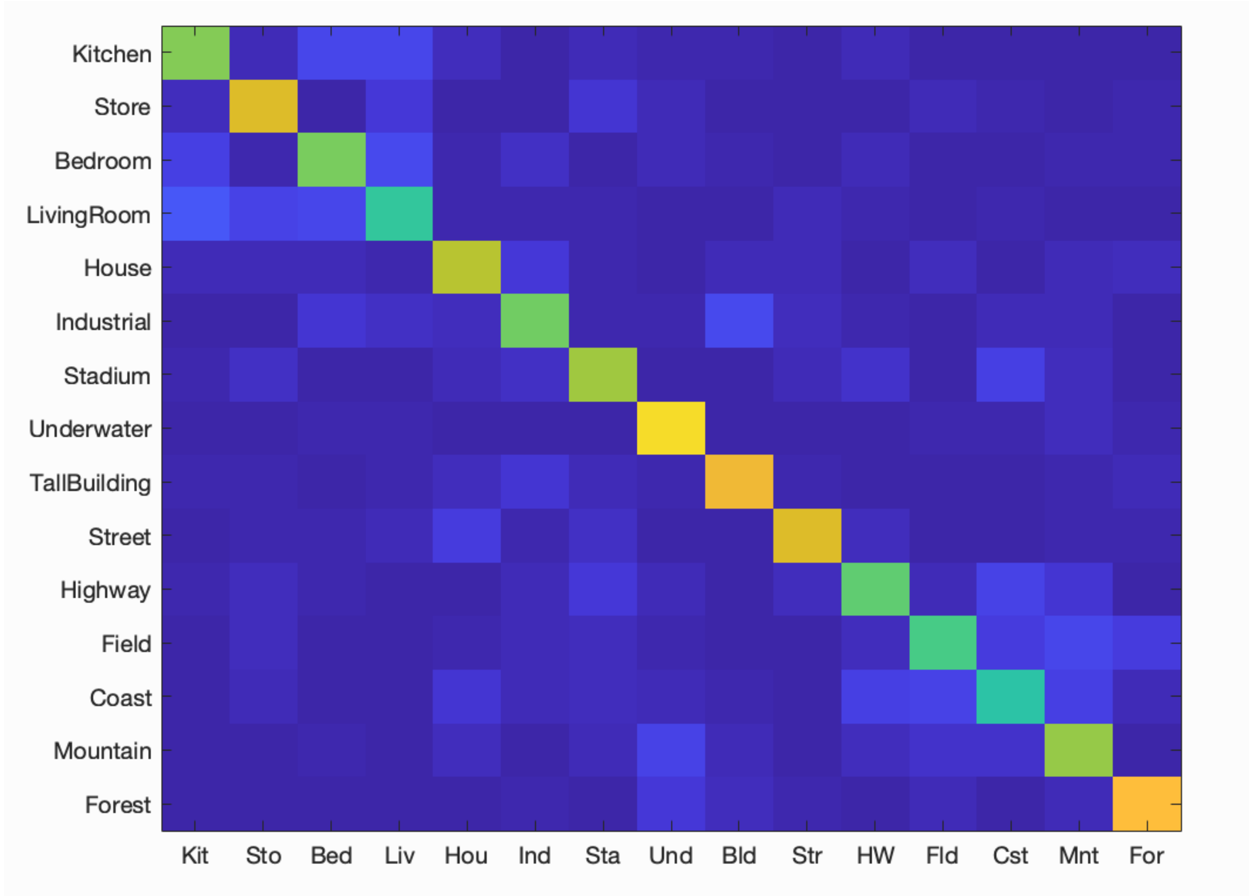
- DSIFT parameters: Fast, level =2, step size =4 , bin size = 4,Normalize

2. Spatial Pyramids + Sift Feature: Result Presentation

- SVM classifier with Grayscale sift

- Confusion Matrix
- Vocab size : 500 , L = 1.0000e-04 , Accuracy = 0.704

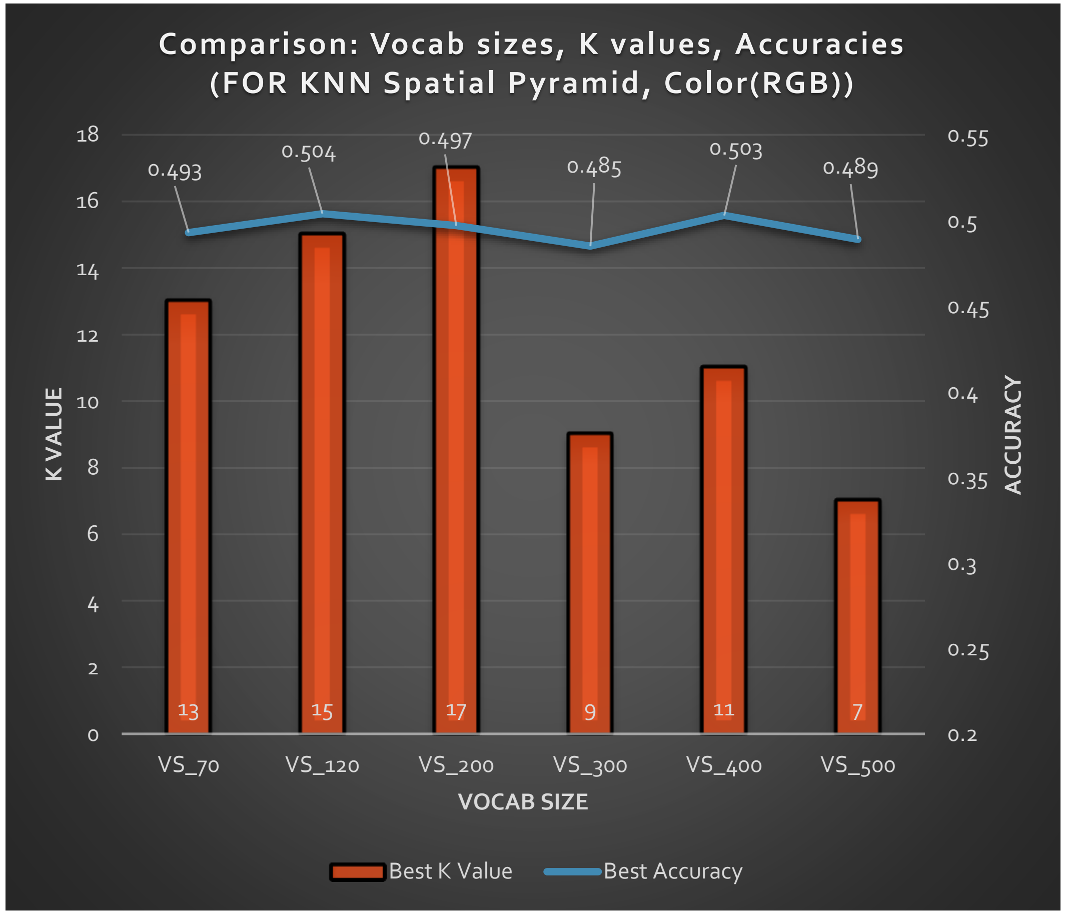




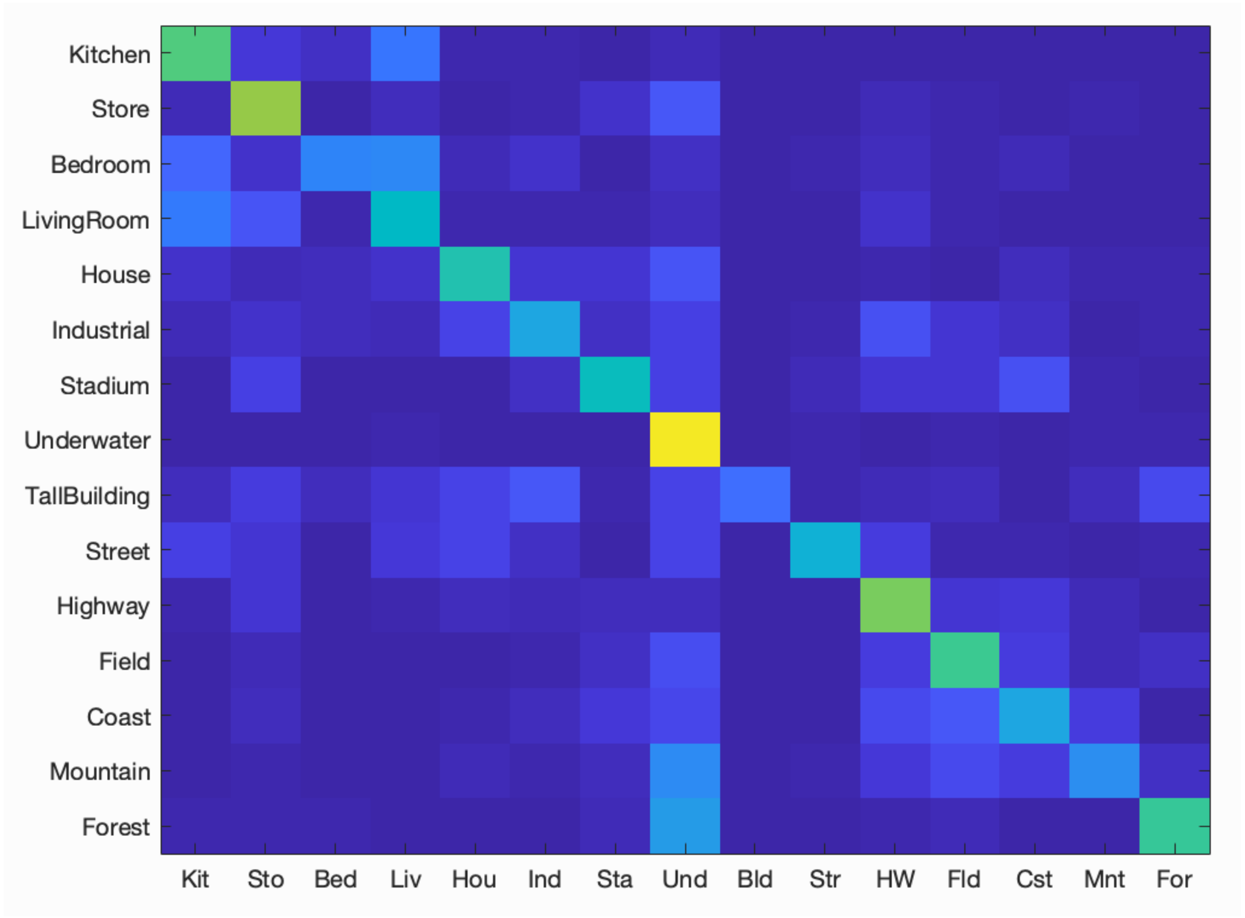
- DSIFT parameters: Fast, level =2, step size =4 , bin size = 4,Normalize

2. Spatial Pyramids + Sift Feature: Result Presentation

- KNN classifier with sift + Color(RGB)



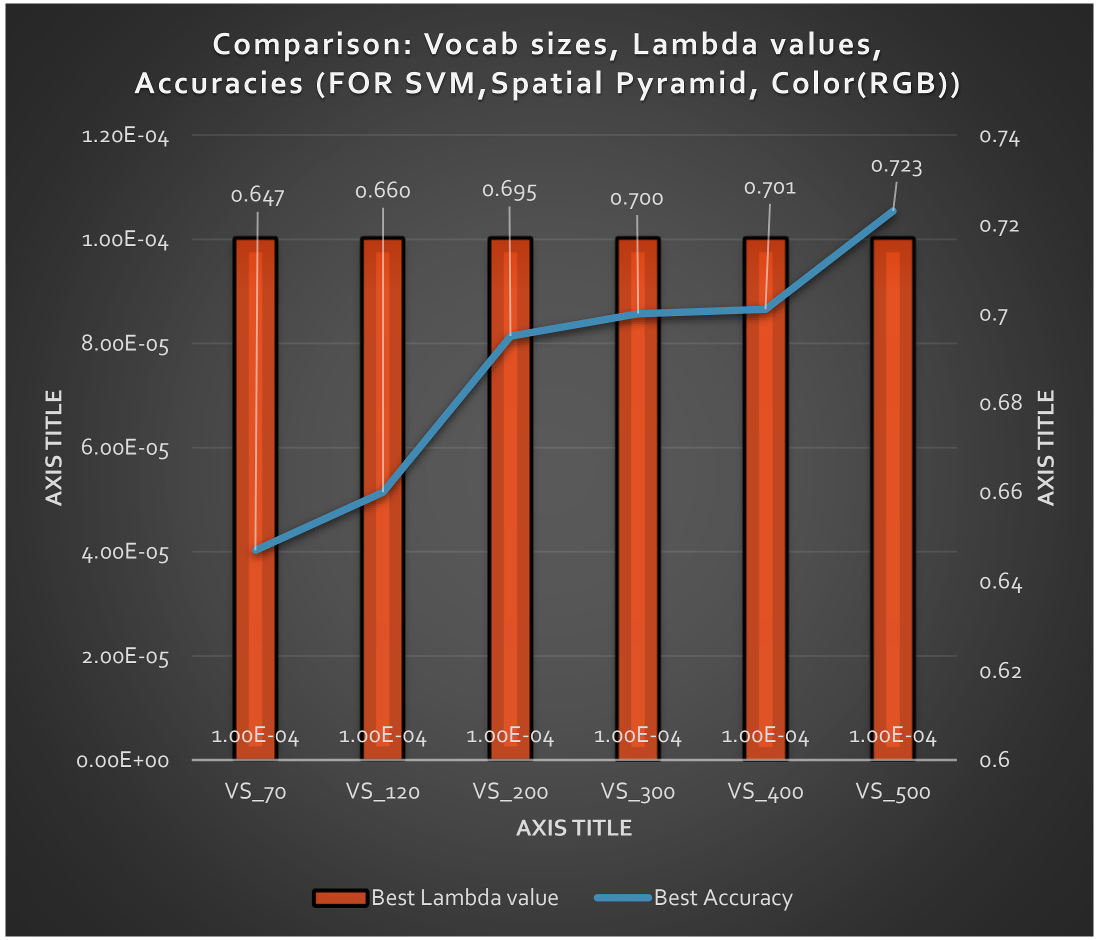
- Confusion Matrix
- Vocab size : 120 , K = 15 , Accuracy = 0.504



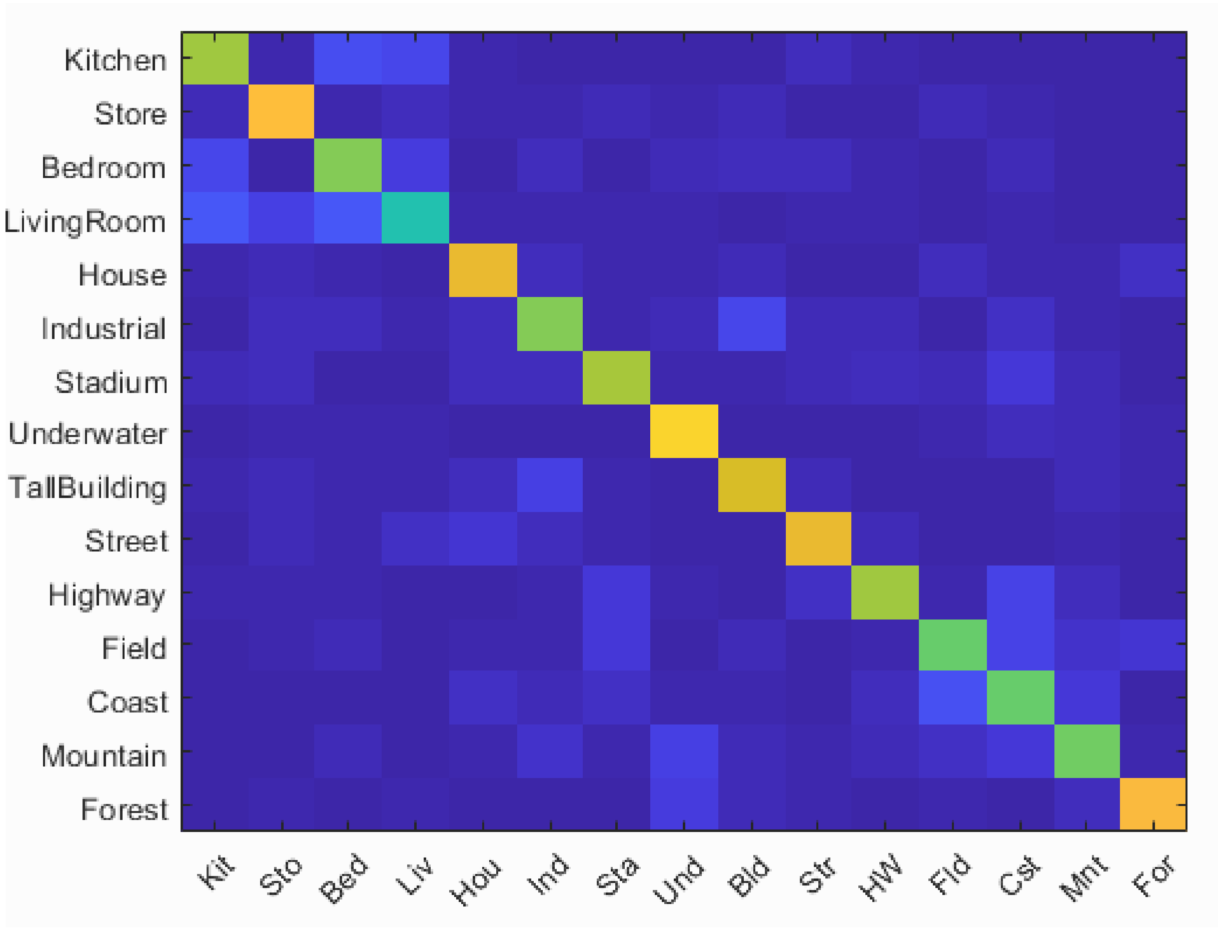
- DSIFT parameters: Fast, level =2, step size =4 , bin size = 4,Normalize

2. Spatial Pyramids + Sift Feature: Result Presentation

- SVM classifier with sift + Color(RGB)



- Confusion Matrix
- Vocab size : 500 , L= 1.0000e-04 , Accuracy = 0.723



- DSIFT parameters: Fast, level =2, step size =4 , bin size = 4,Normalize

Histogram of Gradient Feature:

- Introduction
- Function Implementation
- Result Presentation

3. Histogram of Gradient Feature:

- The Histogram of Gradients (HOG) descriptor, when combined with the Random Forest method, provides strong performance in image analysis applications like as scene recognition and object detection.
- HOG extracts important gradient and edge information from an image by generating histograms of gradient directions within localized areas.
- This generates a full feature set that represents the texture and shape information of the visual content.
- Random Forest, an ensemble learning method, improves forecast accuracy and reduces overfitting by averaging decisions across several trees.
- This combination is especially beneficial because Random Forest can efficiently handle the high dimensionality and variability of HOG features, resulting in better classification and detection performance in diverse and complicated visual scenarios.

- Histogram of Gradient Feature Implementation:

3. Histogram of Gradient Feature Implementation:

- It starts by initializing a random number generator with a specific seed to ensure reproducibility.
- The function sets important parameters like cell size, block size, and the number of orientation bins, which influence the resolution and detail of feature extraction.
- It computes the dimensions of the feature vector based on the image’s layout of cells and blocks.
- For each image, it selects either color or grayscale mode for processing; color mode involves extracting and combining HOG features from each color channel, while grayscale mode converts color images before feature extraction.
- Using MATLAB's **extractHOGFeatures**, it calculates gradients, sorts them into orientation bins, and compiles them into cell histograms.
- The features are then normalized using L2 normalization (Euclidean distance) to mitigate illumination effects and adjusted in length to maintain uniformity, facilitating consistent model input.
- Finally, these processed features are stored in an output matrix for subsequent use.

Random Forest Classifier:

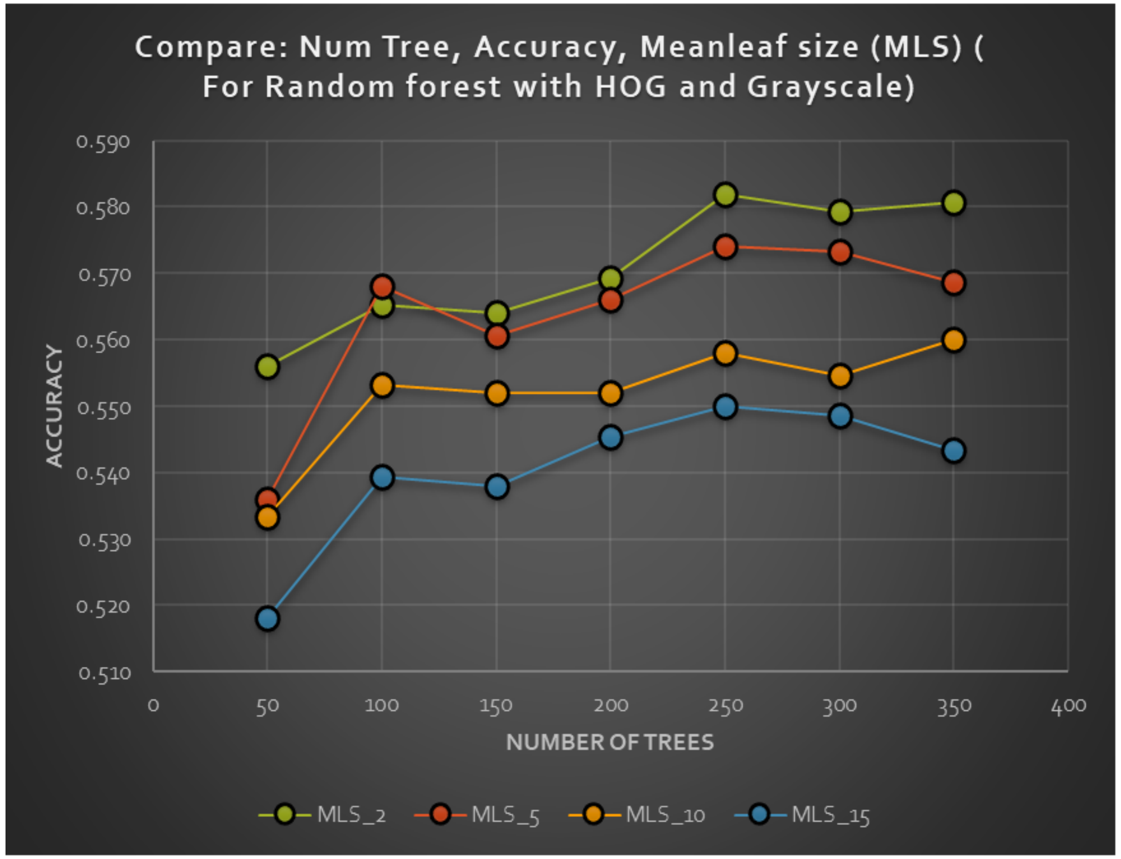
- The random_forest_classify function trains and tests Random Forest classifiers for image classification by iterating over a predefined list of tree counts using MATLAB's TreeBagger.
- This method uses bootstrap aggregation to create an ensemble of decision trees, defining parameters for best categorical outcomes and enables out-of-bag error calculation for robust model validation.
- It predicts test data categories, computes accuracy, and chooses the model with the highest accuracy.
- Additionally, it stores important information like as the number of trees, accuracy, and out-of-bag errors into a results table, allowing for a systematic comparison of performance across different setups.
- To ensure repeatability, the random number generator is started with a specific seed.

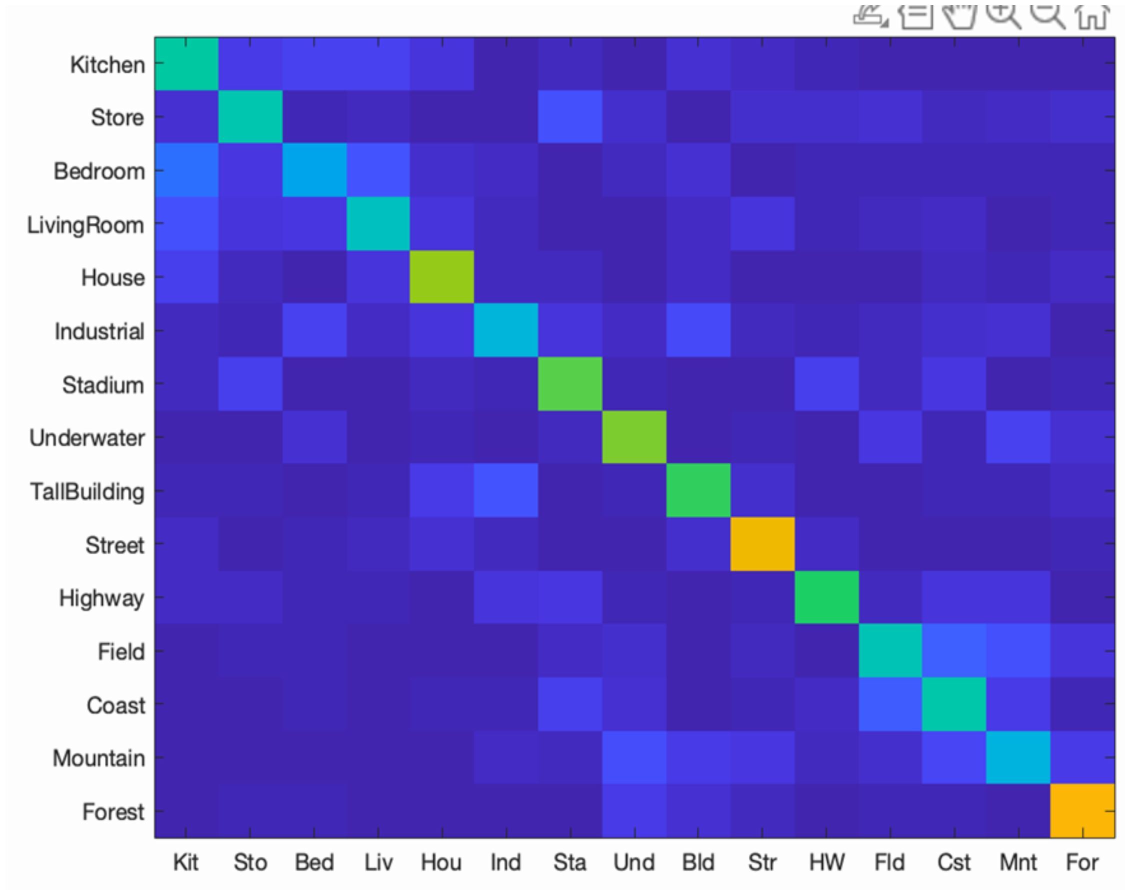
- Result Presentation of Histogram of Gradient Feature :

3. Histogram of Gradient Feature : Result Presentation

- Random Forest classifier with Grayscale:

- Confusion Matrix
- Random Forest parameters : Num Tree: 250 , Minleaf size: 2, Max num split : 500, Accuracy : 0.582

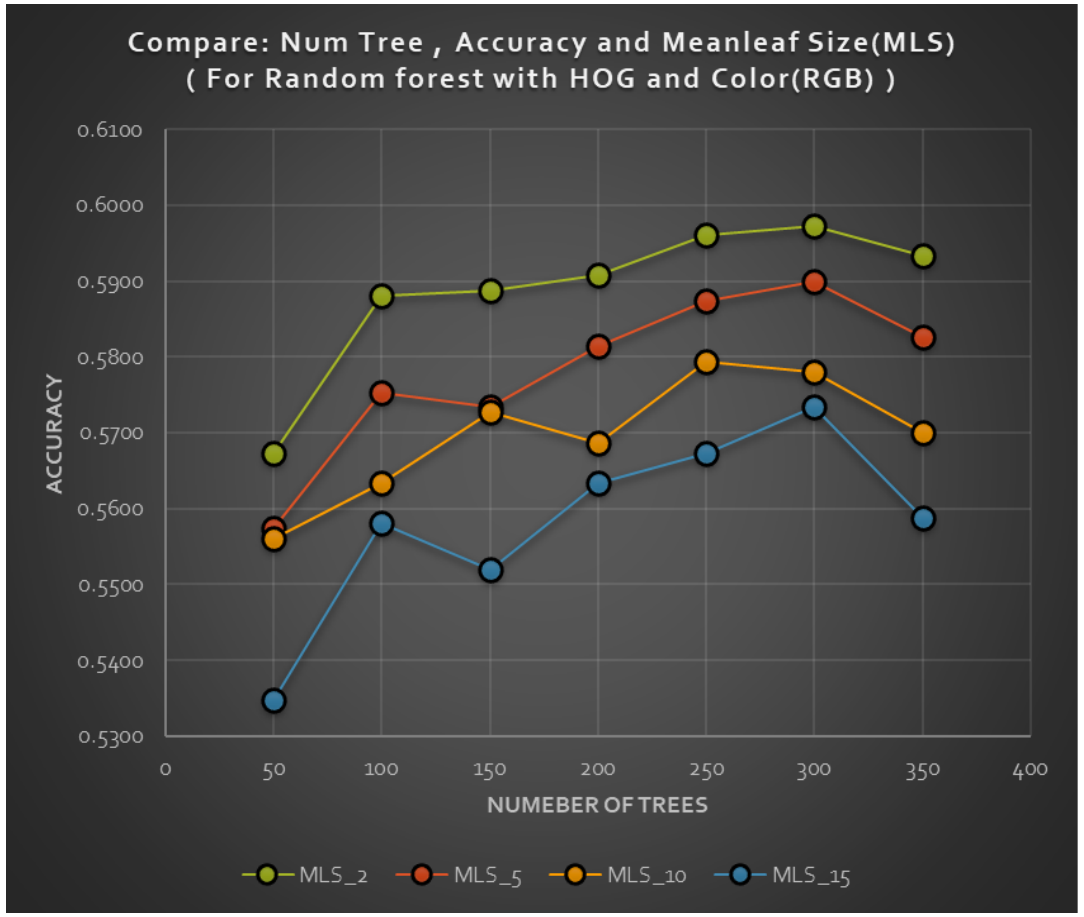




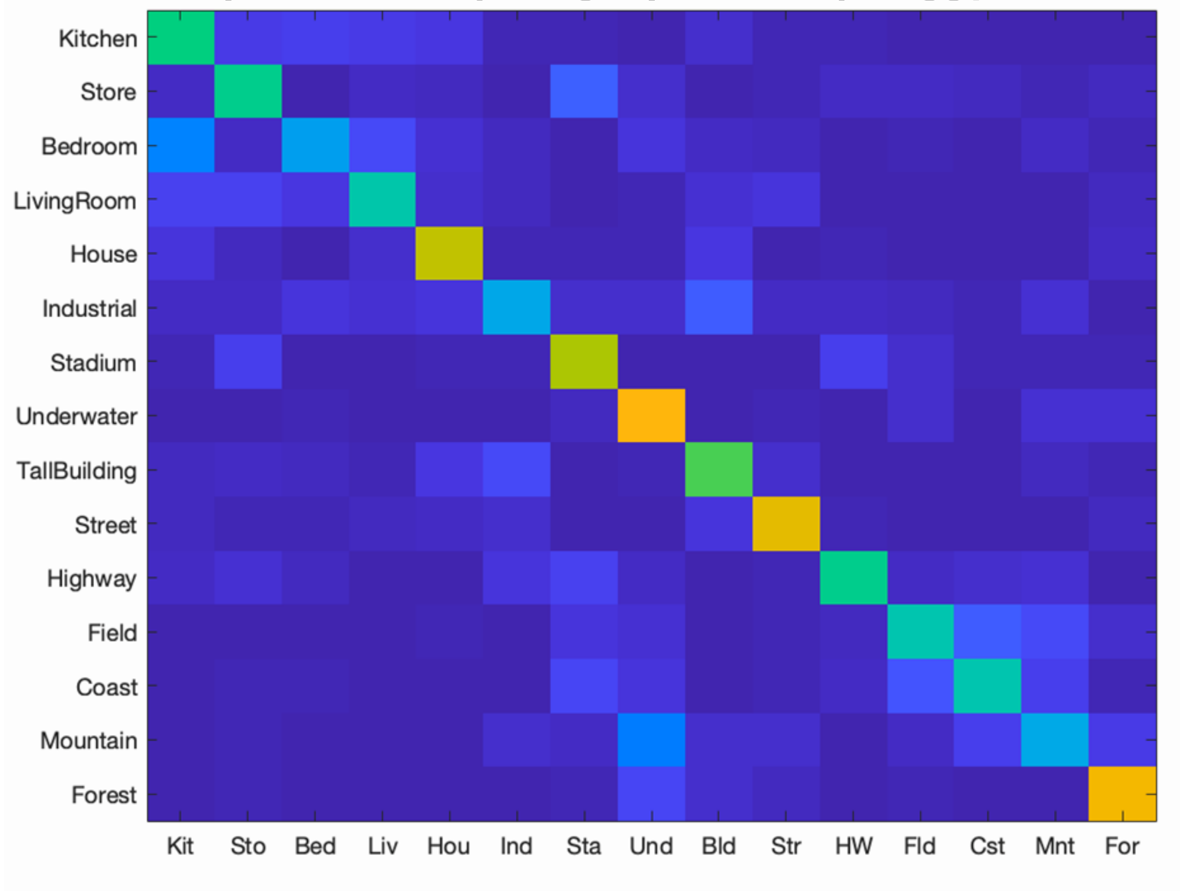
HOG parameters : cell size: 64, block size: 2, num bins: 83, image size: 256x256

3. Histogram of Gradient Feature : Result Presentation

- Random Forest classifier with Color(RGB):



- Confusion Matrix
- Random Forest Parameters : Num Tree: 300 , Minleaf size: 2, Max num split : 500, Accuracy: 0.597



HOG parameters : cell size: 64, block size: 2, num bins: 83, image size: 256x256

Conclusion

Experiment Summary Table:

1. Bag of SIFT feature with Various classifier and Color spaces as below:

Sr.No.	Classifier	Color Space	Build Vocab size	SIFT Type	Parameter Setup	Best K/Best Lambda value	Best Accuracy
1	KNN	Grayscale	200	DSIFT	Fast, step size =4 , bin size = 4, Normalize	K = 13	0.464
2	**SVM**	**Grayscale**	**500**	**DSIFT**	**Fast, step size =4 , bin size = 4, Normalize**	**L = 0.00001**	**0.598**
3	KNN	Color(RGB)	70	SIFT	PeakThresh =0, EdgeThresh=10,NorThresh=-inf,Magnif=3,windowsize=2	K = 31	0.404
4	SVM	Color(RGB)	500	SIFT	PeakThresh =0, EdgeThresh=10,NorThresh= inf,Magnif=3,windowsize=2	L = 1	0.405

2. Spatial Pyramid feature with Various classifiers and Color spaces as below:

Sr.No.	Classifier	Color Space	Build Vocab size	SIFT Type	Parameter Setup	Best K/Best Lambda value	Best Accuracy
1	KNN	Grayscale	200	DSIFT	Fast, level =2, step size =4 , bin size = 4, Normalize	K = 21	0.509
2	**SVM**	**Grayscale**	**500**	**DSIFT**	**Fast, level =2, step size =4 , bin size = 4, Normalize**	**L = 0.00001**	**0.704**
3	KNN	Color(RGB)	120	DSIFT	Fast, level =2, step size =4 , bin size = 4, Normalize	K = 15	0.504
4	**SVM**	**Color(RGB)**	**500**	**DSIFT**	**Fast, level =2, step size =4 , bin size = 4, Normalize**	**L = 0.00001**	**0.723**

3. Histogram of Gradient feature with Classifier and Color spaces as below:

Sr.No.	Classifier	Color Space	Num Tree	Max num split	Parameter Setup	MeanLeaf size	Best Accuracy
1	Random Forest	Grayscale	250	500	Cell size = 64 , Block = 2, Num bin = 83, img size = [256 x256]	2	0.582
2	**Random Forest**	**Color(RGB)**	**300**	**500**	**Cell size = 64 , Block = 2, Num bin = 83, img size = [256 x256]**	**2**	**0.597**

Conclusion Points:

1. The SVM with RGB and DSIFT (Spatial Pyramid) achieved the highest accuracy (0.723):
   - SVM is effective at managing the large dimensionality of RGB color information. DSIFT in the RGB space gathers a wide variety of features that increases image classification accuracy.
   - SVM's optimization technique maximizes the margin between classes, improving its capacity to categorize complex patterns effectively.
2. Second highest accuracy (0.704) for SVM with Grayscale and DSIFT (Spatial Pyramid):
   - The SVM's ability to handle many features, including texture and edge information, makes it suitable for DSIFT's dense feature vectors, even in grayscale.
   - SVM's linear decision boundaries in high-dimensional feature space offer accurate generalization, even with limited color information.
3. Third highest accuracy (0.598) for SVM using Grayscale and DSIFT (Bag of SIFT):
   - SVM with grayscale DSIFT is less effective than the Spatial Pyramid due to the simpler feature grouping method, which may not capture the spatial hierarchy in image features as well.
   - DSIFT offers dense local features, but lacks the spatial structure in the Bag of SIFT method, reducing performance slightly.

Conclusion Points:

4. Random Forest with RGB (Gradient Histogram) had the fourth highest accuracy (0.597).
    - Random Forest successfully manages variability in RGB color space and uses ensemble learning to reduce overfitting.
    - Random Forest's many decision trees use the Histogram of Gradient features to extract detailed texture and color information.
5. KNN with RGB and SIFT (Bag of SIFT) showed the lowest accuracy (0.404).
    - KNN's performance decreases considerably in high-dimensional spaces, such as RGB SIFT features, due to the constraint of dimensionality.
    - KNN's dependency on distance measurements becomes less important as feature dimensionality increases, resulting in lower performance.

SVM, particularly when paired with Spatial Pyramid and DSIFT features in RGB color space, excellent in handling complex, high-dimensional data and achieves the maximum classification accuracy. This demonstrates its capability to use spatial hierarchies and detailed feature representation for outstanding image classification.

References:

S. Lazebnik, C. Schmid and J. Ponce, "Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories," *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, New York, NY, USA, 2006, pp. 2169-2178, doi: 10.1109/CVPR.2006.68.

Sivic and Zisserman, "Video Google: a text retrieval approach to object matching in videos," *Proceedings Ninth IEEE International Conference on Computer Vision*, Nice, France, 2003, pp. 1470-1477 vol.2, doi: 10.1109/ICCV.2003.1238663.

S. Lazebnik, C. Schmid and J. Ponce, "Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories," *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, New York, NY, USA, 2006, pp. 2169-2178, doi: 10.1109/CVPR.2006.68.

Jian Li, S. K. Zhou and R. Chellappa, "Appearance modeling under geometric context," *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, Beijing, China, 2005, pp. 1252-1259 Vol. 2, doi: 10.1109/ICCV.2005.38.

Boser, B. E., Guyon, I. M., and Vapnik, V. N., 1992. A training algorithm for optimal margin classifiers. In: Proceedings of the fifth annual workshop on Computational learning theory - COLT '92. New York, USA: ACM Press, pp.144-152.

Friedman, J., Hastie, T., and Tibshirani, R., 2001. The elements of statistical learning. New York, USA: Springer series in statistics.

Khan, R., Hanbury, A., Stöttinger, J., and Bais, A., 2012. Color Attributes for Object Detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.3306-3313.

Varma, M. and Zisserman, A., 2009. A statistical approach to texture classification from single images. International Journal of Computer Vision, 62(1-2), pp.61-81.

Question and Answers

Thank you

Work Distribution for Course work 2:

Sr.No.	Task Description	Student name
1	Build Vocabulary Function	Monish
2	Bag of sift function	Monish
3	Spatial Pyramid Function	Pranav
4	Support vectore classifer	Pranav
5	Histogram of Gradient function	Monish
6	Random forest classifier	Pranav
7	Experiments	Pranav and Monish
8	Make Presentation	Monish and Pranav