CMPE 537 Computer Vision
Group Assignment #3:
Local Descriptor based Image Classification

Abdurrahman Dilmaç, Yusufcan Manav, Barış Yamansavaşçılar, Mansur Yeşilbursa

9 January 2021

1 Introduction

In this homework, we implemented image classification pipelines based on local descriptors which are extracted from images using several different approaches including HOG, SIFT, and ORB. These local descriptors are computed for each image considering their scale and location invariant features. Afterwards, local descriptors are grouped using Hierarchical K-Means and GMM, separately, and then histogram of the images are calculated based on the clustering model. Finally, a corresponding classifier is trained using the histogram of training images.

The general pipeline described above is performed by each project member in this homework. We explain our experiences and results in following sections. In Section 2, we give the details about local descriptors that are utilized in this project. Next, we describe the clustering techniques in Section 3. We investigate the feature quantization and classifiers approaches in Section 4 and 5, respectively. Finally, we analyze our pipelines and evaluations in Section 6 and 7 and then conclude this report.

2 Local Descriptors

2.1 HOG

Histogram of gradients (HOG) is window based image descriptor. Originally, it was proposed for human detection purposes. But later on, it found itself various other applications and its widely used as a image descriptor for machine learning based vision tasks. In this assignment, HOG is implemented manually. First, images were resized to 64x128 since usually a cell size of multiple of 8 is used. This is for the ease of the implementation. Images were converted to gray scale to facilitate gradient computations. In order to compute gradients on RGB image, for each cell, dominant color has to be decided and the gradient filters are applied on that color channel.

Gradients are calculated by using $[-1, 0, 1]$ and $[1, 0, -1]$ filters along the x and y axis respectively. Gradients in x and y directions are converted to magnitude and 'orientation' representation. Then, for each cell, gradient histograms are generated by dividing 0-180 degrees into 9 bins and summing the magnitudes of gradients that are within the range of the corresponding orientation bin.

After computing the histogram of gradient cells, normalization applied on the 2x2 blocks of cells to remove lighting effects. For example, if cell size is chosen to be 8x8, then normalization is applied on the sliding blocks of 16x16.

As images were previously resized, number of features for each image is constant. Hence, extracted features can be directly used in classification without applying clustering and quantization as the original paper does. We've experimented with both of the cases, further comparison can be found in the later sections.

2.2 SIFT

SIFT, namely Scale Invariant Feature Transform, is a feature detection algorithm that is used to extract important features from an image. The most significant property of SIFT is that it is not affected by the orientation and size of the image.

In this homework, we used SIFT method of OpenCV in Python in order to extract keypoints and its corresponding descriptors. When this method is run on the images of the dateset, we observed that SIFT extracts different number of keypoints from each image. In general, it extracts the number of keypoints between 300 and 400. On the other hand, each keypoint has a descriptor which consists of 128 attributes.

2.3 ORB

For the Orb descriptor we used the OpenCV's orb descriptor with nfeatures=500 to hard limit the maximum number of features extracted from images. Generally about 400 features per image extracted, each descriptor having 32 features.

3 Creating Dictionary

3.1 Hierarchical K-Means

We implemented an Hierachical K-Means method for the dictionary creation part. This method works for three layers. In first layer it splits the space into n groups. Then in the second layer it works on each created group from the previous stage, and splits them to n groups. In the third step it splits the second layer groups in to n, in result it creates n^3 groups. At the end of the algorithm the means of the groups stored, each representing an visual word.

3.2 GMM

We had problems over the vlfeat library so we used the GMM from the sklearn library.

3.3 Spectral Clustering

Spectral clustering is a powerful clustering method, especially when clusters are geometrically entangled/inseparable (i.e.nested circles). It projects sample space to another space where samples are more 'separable' and clusters them using k-means. While projecting, it makes use of an affinity matrix of the size of (#samples x #samples). This affinity matrix hinders the use of spectral clustering for large data as it is requires very large amount of memory.
We've also encountered this problem during implementation. We've tried to cluster HOG features using spectral clustering, however, it required very high memory usage (as high as 512 GB) for HOG features with cell size of 16. Affinity matrix only fitted into the memory when HOG features with cell size of 32 was used. However, as the cell size increases features per image decrease drastically (3 descriptor per image with 36 features) thus, extracted features become useless.
Spectral clustering implemented using sklearn library. However, this model only provides cluster labels for given data. We can't predict the cluster of the new data or have access to cluster centers. This creates problem for testing phase of the system as the new data can't be labeled with cluster centers. To overcome this problem, we've computed artificial cluster centers by simply taking the mean of the samples that are assigned to the same cluster by the model. It is unknown how valid or correct this approach is but it was the only way we could come up with which allows label assignment for the new data.

4 Feature Quantization

4.1 Bag Of Visual Words

In this method it gets the means of the cluster centers from the dictionary methods, and the list of features extracted for each image. First this method creates a zeros for each cluster center. Then for each feature of image it first finds the closest mean then increase the count of this feature by one. It creates a feature vector which has the size of the count of the cluster centers. Then it divides the count vector to its norm and returns the feature vector of the image.

4.2 Fisher Vectors

We had problems over the vlfeat library so we used the code from the github repository
`https://gist.github.com/danoneata/9927923`.

5 Classifiers

5.1 SVM

We used SVM Classifier implemented in sklearn with rbf kernel. This pipeline also had StandartScaler to normalize the data. C and gamma hyperparameters are optimized using GridSearchCV from sklearn. Used possible values for those hyper parameters are svc__C: [0.1, 1, 10, 100] and svc__gamma: [1, 0.1, 0.01, 0.001, 0.0001]. We evaluated method with 5-fold cross-validation.

5.2 Random Forest

We used Random Forest implementation of Sklearn library to carry out the performance evaluation. As hyper-parameters, we utilize 100 and 1000 trees with 42 random states.

6 Pipelines

6.1 ORB - Hierarchical K-Means - Bag of Visual Words - SVM

This pipeline uses the ORB features extracted by OpenCV library. On top of those features we used Hierarchical K-Means with n=4 and 3 layers which in returns give 64 clusters. This number is chosen since we have 20 classes and we need at least different parts of the images to represent those 20 class. On top of that since those descriptors represent a very local point in image such as corner of the mouth of face or wheel of airplane, we wanted to extract multiple describing visual part in each class so we select 64 as the number of classes. After that we used the Bag of Visual Words method to represent each image using the dictionary created by Hierarchical K-Means algorithm and descriptors extracted using ORB.
After that we used SVM classifier described above with GridSearchCV for hyperparameter optimization. The selected values for the hyperparameters were svc__C: 1, svc__gamma: 0.01.

6.1.1 Model Results

Table 1: Scores of the ORB - Hierarchical K-Means - Bag of Visual Words - SVM Pipeline

Class	**Precision**	**Recall**	**F1-Score**	**Count**
Faces	0.28	0.95	0.43	20
airplanes	0.16	0.80	0.26	20
anchor	0.56	0.50	0.53	20
background_class	0.00	0.00	0.00	0
barrel	0.00	0.00	0.00	20
camera	0.00	0.00	0.00	20
car_side	0.79	0.55	0.65	20
dalmatian	0.75	0.15	0.25	20
ferry	0.00	0.00	0.00	20
headphone	0.53	0.45	0.49	20
lamp	0.00	0.00	0.00	20
pizza	0.00	0.00	0.00	20
pyramid	0.00	0.00	0.00	20
snoopy	0.71	0.50	0.59	20
soccer_ball	0.00	0.00	0.00	0
stop_sign	0.86	0.30	0.44	20
strawberry	0.33	0.10	0.15	20
sunflower	0.60	0.15	0.24	20
water_lilly	0.00	0.00	0.00	20
windsor_chair	1.00	0.25	0.40	20
yin_yang	0.90	0.45	0.60	20
Macro Avg	0.36	0.25	0.24	380

6.2 SIFT - K-Means - Bag of Visual Words - Random Forest

In this pipeline, SIFT is used for the extraction of keypoints and their corresponding descriptors. By using the SIFT implementation of OpenCV, 1612181 descriptors with 128 features are extracted including the all of the images in the training set. Next, the K-Means algorithm is trained based on those descriptors since Hierarchical K-Means algorithm causes out of memory error considering the extracted SIFT features and given cluster number, which is 20. Based on 20 clusters, the model is trained. Afterwards, for each image in training set, the corresponding histogram is created. This histogram map is used as the input for the training of the classifier, which is Random Forest. Same operation is performed on the test images to create their histogram

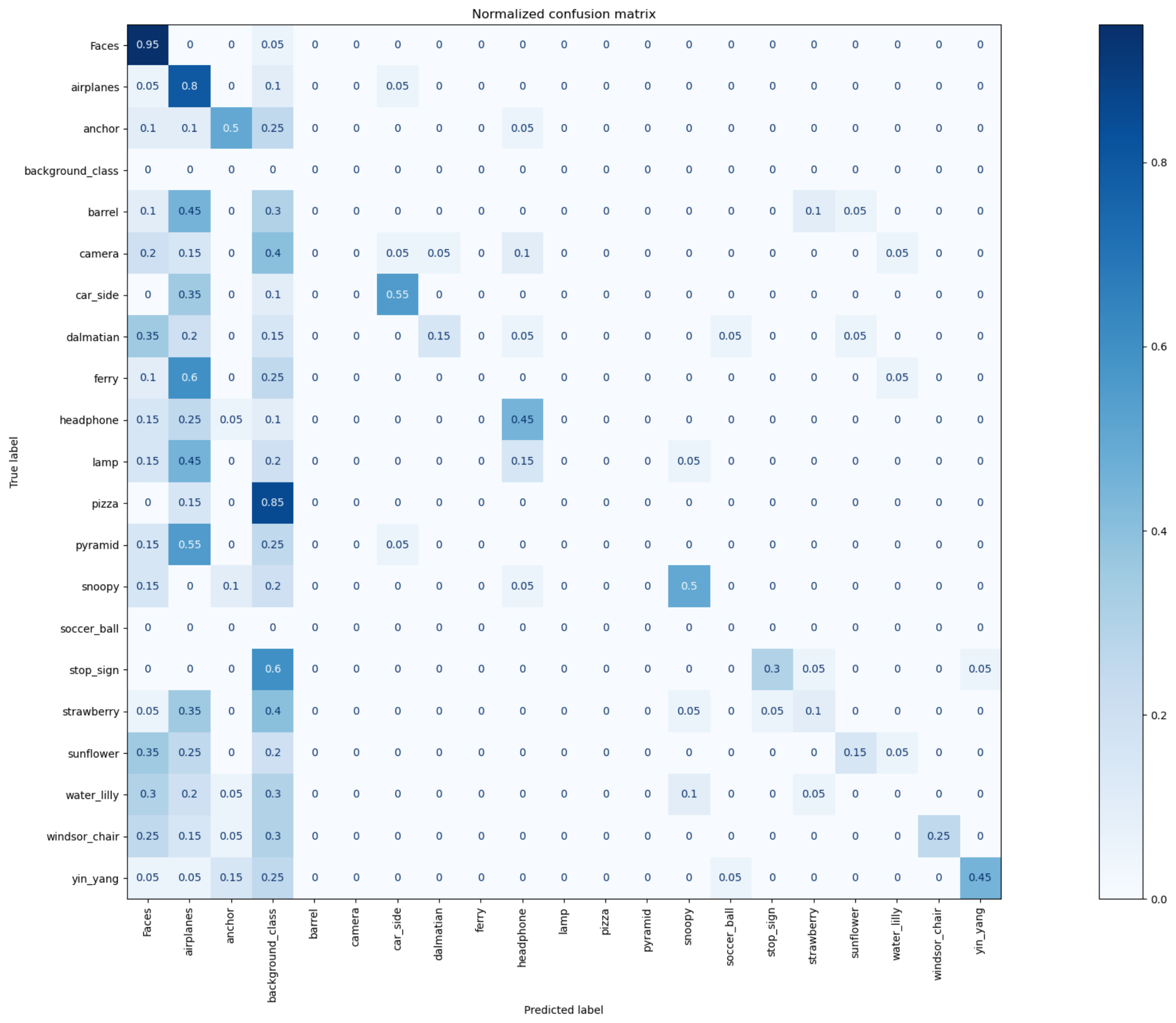

Figure 1: Confusion Matrix of the ORB - Hierarchical K-Means - Bag of Visual Words - SVM Pipeline

map. For the configuration of the Random Fores, 100 and 1000 tree estimators are used for the performance evaluation.

6.2.1 Model Results

The results of the training model in terms of the precision, recall, and F1-score, are given in Table 2. It can be concluded that Random Forest is not a suitable classifier considering SIFT features since the overall accuracy, which is 0.08, is extremely low. Note that both configurations in which 100 trees case and 1000 trees case the overall accuracy did not change.

6.3 ORB - GMM - Fisher Vectors - SVM

For this method we used the described methods described above. This pipeline was implemented using only pre-implemented methods for test purposes only.
This method takes the ORB desciptors as input and uses GMM clustering with cluster size 32 to create words. After it is fitted, Fisher Vector quantization is utilized to generate 2080 dimentional feature vectors using GMM cluster centers and ORB descriptors of the image.
SVM with rbf kernel then utilized for classification task. The selected values for the hyperparameters were svc__C: 10, svc__gamma: 0.0001.

6.3.1 Model Results

Table 2: Scores of the SIFT - K-Means - Bag of Visual Words - Random Forest Pipeline

Class	**Precision**	**Recall**	**F1-Score**	**Count**
Faces	0.83	0.25	0.38	20
airplanes	0.17	0.10	0.12	20
anchor	0.03	0.05	0.04	20
background_class	0.00	0.00	0.00	0
barrel	0.07	0.10	0.08	20
camera	0.14	0.25	0.18	20
car_side	0.11	0.20	0.14	20
dalmatian	0.00	0.00	0.00	20
ferry	0.00	0.00	0.00	20
headphone	0.00	0.00	0.00	20
lamp	0.00	0.00	0.00	20
pizza	0.33	0.15	0.21	20
pyramid	0.00	0.00	0.00	20
snoopy	0.17	0.35	0.23	20
soccer_ball	0.00	0.00	0.00	0
stop_sign	0.86	0.30	0.44	20
strawberry	0.00	0.00	0.00	20
sunflower	0.00	0.00	0.00	20
water_lilly	0.00	0.00	0.00	20
windsor_chair	0.00	0.00	0.00	20
yin_yang	0.05	0.10	0.07	20
Macro Avg	0.10	0.08	0.07	380

Table 3: Scores of the ORB - GMM - Fisher Vectors - SVM Pipeline

Class	**Precision**	**Recall**	**F1-score**	**Count**
Faces	0.42	0.90	0.57	20
airplanes	0.26	0.80	0.39	20
anchor	0.76	0.80	0.78	20
background_class	0.00	0.00	0.00	0
barrel	0.00	0.00	0.00	20
camera	0.00	0.00	0.00	20
car_side	0.87	0.65	0.74	20
dalmatian	0.58	0.35	0.44	20
ferry	0.00	0.00	0.00	20
headphone	0.44	0.55	0.49	20
lamp	0.67	0.20	0.31	20
pizza	1.00	0.05	0.10	20
pyramid	0.67	0.30	0.41	20
snoopy	0.94	0.75	0.83	20
soccer_ball	0.00	0.00	0.00	0
stop_sign	1.00	0.65	0.79	20
strawberry	0.46	0.30	0.36	20
sunflower	1.00	0.55	0.71	20
water_lilly	0.33	0.15	0.21	20
windsor_chair	1.00	0.65	0.79	20
yin_yang	1.00	0.70	0.82	20
macro avg	0.54	0.40	0.42	380

6.4 HOG - SVM

As it was mentioned earlier, HOG features produce fixed size features for each image. Therefore, we can directly train a SVM model with the HOG features. Model wasn't trained on background class. I've applied standard scaling to the features in order to stabilize the training of SVM. For this pipeline, I trained the SVM model for 300 iterations and optimal learning rate. HOG features are extracted with cell size of 8 and the model wasn't trained on background class.
Scores can be seen in Table 4 and confusion matrix can be inspected in Figure 3. Accuracy of this model on the test set is 16%. When confusion matrix inspected it can be clearly seen that this model very frequently assigns

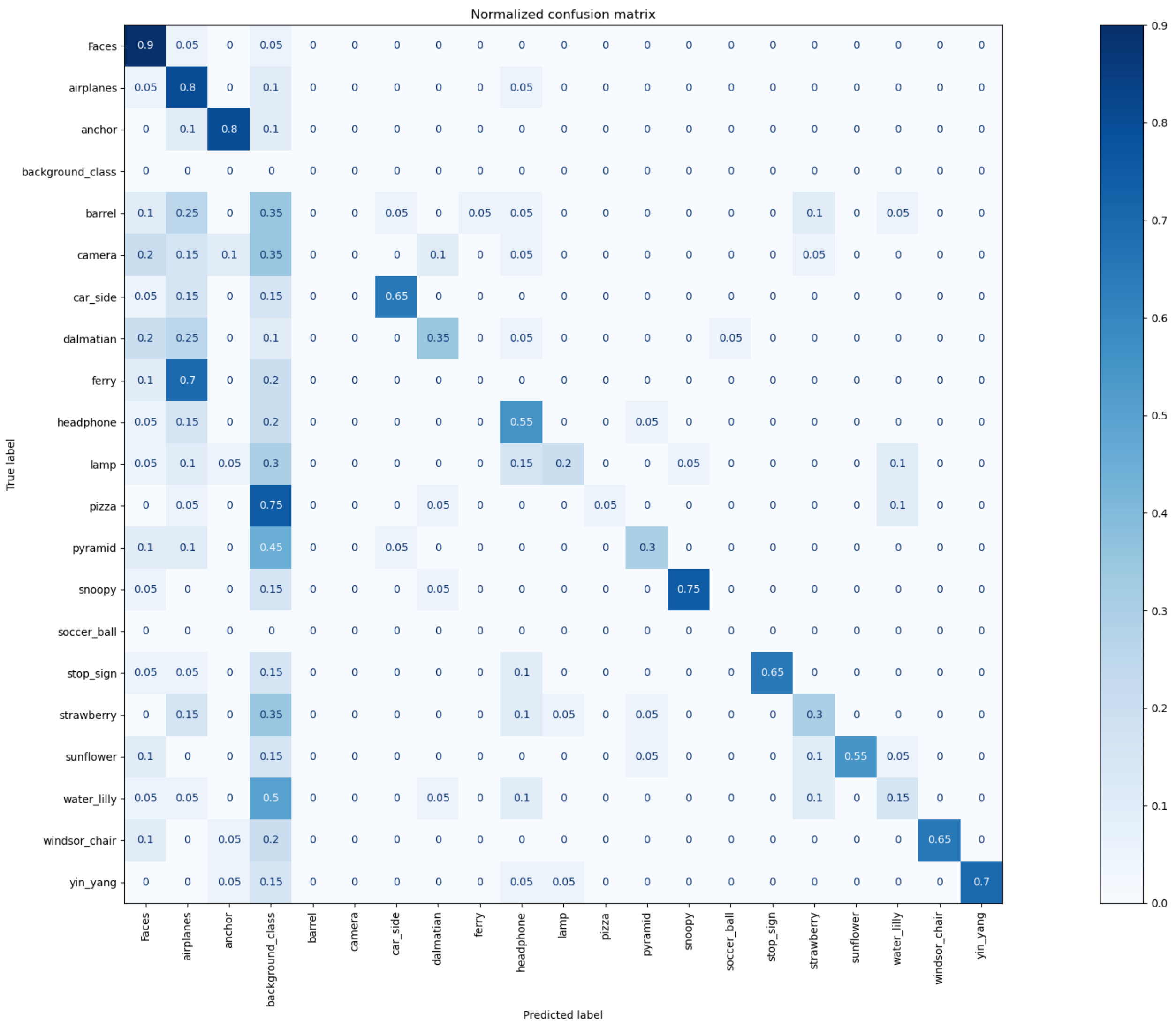

Figure 2: Confusion Matrix of the ORB - GMM - Fisher Vectors - SVM Pipeline

images to airplane class which approximately contains 1/3 of the dataset. It seems like model doesn't learn to differentiate between classes but rather guessing with high probability class.

6.5 HOG-Spectral Clustering-BoW-SVM

This method is completed just to show that why specteral clustering is not a good idea for this task because it can't handle large data. I've used cluster size 64 but as it was mentioned, only 3 features per image were generated because of the memory constraints spectral clustering brings on which makes bag of visual words pretty meaningless. Results also suggest that this method only a little better than random class assignment. I trained SVM model for 100 iterations and with optimal learning rate. HOG features are extracted with cell size of 32 and model wasn't trained on background class.
Scores can be seen in Table 5 and confusion matrix can be inspected in Figure 4. Accuracy of this model on the test set is 12% When confusion matrix is inspected, it can be seen that model usually assigns the input image to airplanes class except for few cases. This indicates that model didn't really learned the image classification task.

6.6 HOG-K-Means-BoW-SVM

This method is the best performing method which makes use of manually implemented image descriptor. HOG features are extracted with cell size of 8 and model wasn't trained on background class. 64 cluster samples were used for k-means and SVM trained for 800 iterations with optimal learning rate.

Scores can be seen in Table 6 and confusion matrix can be inspected in Figure 5. Accuracy of this model on

Table 4: Scores of the HOG - SVM Pipeline

Class	**Precision**	**Recall**	**F1-score**	**Count**
Faces	0.18	0.15	0.16	20
airplanes	0.09	0.85	0.16	20
anchor	0.00	0.00	0.00	20
background_class	0.00	0.00	0.00	0
barrel	0.67	0.10	0.17	20
camera	0.50	0.10	0.17	20
car_side	0.00	0.00	0.00	20
dalmatian	0.28	0.40	0.33	20
ferry	0.00	0.00	0.00	20
headphone	0.19	0.25	0.21	20
lamp	0.00	0.00	0.00	20
pizza	0.70	0.35	0.47	20
pyramid	0.00	0.00	0.00	20
snoopy	0.23	0.15	0.18	20
soccer_ball	0.00	0.00	0.00	0
stop_sign	0.44	0.20	0.28	20
strawberry	0.05	0.05	0.05	20
sunflower	0.38	0.30	0.33	20
water_lilly	0.00	0.00	0.00	20
windsor_chair	0.00	0.00	0.00	20
yin_yang	0.40	0.20	0.27	20
macro avg	0.20	0.20	0.14	380

Table 5: Scores of the HOG -Spectral Clustering - BoW - SVM Pipeline

Class	**Precision**	**Recall**	**F1-score**	**Count**
Faces	0.11	0.65	0.19	20
airplanes	0.09	0.30	0.13	20
anchor	0.00	0.00	0.00	20
background_class	0.00	0.00	0.00	0
barrel	0.00	0.00	0.00	20
camera	0.08	0.05	0.06	20
car_side	0.24	0.80	0.37	20
dalmatian	0.00	0.00	0.00	20
ferry	0.14	0.15	0.15	20
headphone	0.00	0.00	0.00	20
lamp	0.00	0.00	0.00	20
pizza	0.00	0.00	0.00	20
pyramid	0.15	0.20	0.17	20
snoopy	0.00	0.00	0.00	20
soccer_ball	0.00	0.00	0.00	0
stop_sign	0.00	0.00	0.00	20
strawberry	0.00	0.00	0.00	20
sunflower	0.00	0.00	0.00	20
water_lilly	0.00	0.00	0.00	20
windsor_chair	0.00	0.00	0.00	20
yin_yang	0.67	0.10	0.17	20
macro avg	-	0.12	0.07	380

the test set is 24% Confusion matrix of this method seems much better compared to other HOG based models, we can see a nice diagonal even though there is still very high number of wrong assignments.

Table 6: Scores of the HOG-K-Means-BoW-SVM Pipeline

Class	**Precision**	**Recall**	**F1-score**	**Count**
Faces	0.20	0.70	0.31	20
airplanes	0.18	0.25	0.21	20
anchor	1.00	0.00	0.00	20
background_class	0.00	0.00	0.00	0
barrel	0.22	0.70	0.33	20
camera	0.50	0.05	0.09	20
car_side	0.26	0.65	0.37	20
dalmatian	0.18	0.10	0.13	20
ferry	0.20	0.40	0.27	20
headphone	1.00	0.00	0.00	20
lamp	0.53	0.50	0.51	20
pizza	0.00	0.00	0.00	20
pyramid	0.21	0.45	0.29	20
snoopy	0.00	0.00	0.00	20
soccer_ball	0.00	0.00	0.00	0
stop_sign	0.40	0.30	0.34	20
strawberry	1.00	0.00	0.00	20
sunflower	0.29	0.10	0.15	20
water_lilly	0.00	0.00	0.00	20
windsor_chair	0.00	0.00	0.00	20
yin_yang	0.50	0.40	0.44	20
macro avg	0.33	0.28	0.17	380

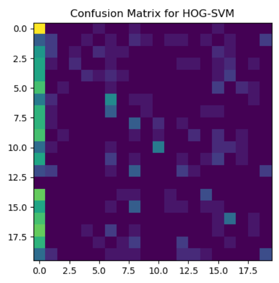

Figure 3: HOG-SVM Confusion Matrix

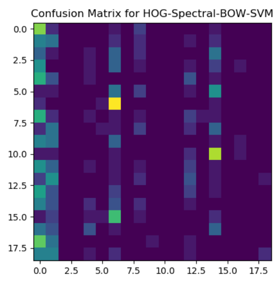

Figure 4: HOG-Spectral Confusion Matrix

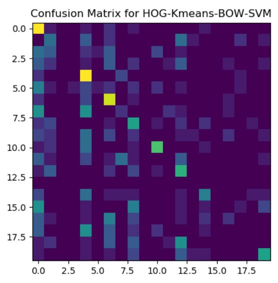

Figure 5: HOG-K-means Confusion Matrix

7 Evaluations

The ORB - Hierachical K-Means - Bag of Visual Words - SVM pipeline had macro averaged F1 score of 0.24 which is low for this task. The main problem was the background class since they had various images and also had many examples in the training set so it lowered the score of the model. Also this model detected most of the faces and airplanes also had about 50% accuracy on the car side and headphone. When we look at the training set we see that those are the classes with the most examples and it shows us that increasing the examples will probably increase our performance on this task. Also the unbalanced data is probably creating bias for these classes as we can see from the precision scores the classes with high recall generally had low precision. Only exception was the yin-yang class, which had only 40 training examples but since the images are generally very similar the classifier had 0.6 F1 score on this class.

For overcoming the unbalanced data problem we tried oversampling and undersampling but it did not increase the scores by meaningful amount so we did not include that results, but getting new samples for the classes will probably improve the model drastically. Also this unbalance may effect the dictionary we created since some classes will contribute so much descriptors and the can flood the clusters. But we were not able to try subsampling in the dictionary creation so we don't know the exact effects of them.

For HOG features, we tried to extract features for all rotations of the image to both increase sample size and also remove negative effects that stem from resizing. Resizing might be negatively impacting the feature quality as it messes up with the proportions. Especially for the images that is horizontally long and vertically short.

But it didn't affect the results at all. It was removed in the final version of the code.

HOG models performed moderately; better than SIFT based pipeline and worse than ORB based pipelines. Since HOG was manually implemented, its full capability may not be realized. Also spectral clustering proved to be the wrong choice for large datasets (even though this particular dataset was not very large by any means).

The ORB - GMM - Fisher Vectors - SVM Pipeline had the highest scores in the task generally with the macro average F1 score of 0.42. It also suffered same problem with the ORB - Hierachical K-Means - Bag of Visual Words - SVM pipeline. It still had high precision and low recall on the classes with more data, which leads us to think that it has bias because of data. This shows us that GMM and Fisher vectors are better than the Hierachical K Means and Bag of Visual Words for the image representation.

If we compare ORB - Hierachical K-Means - Bag of Visual Words - SVM pipeline and SIFT - K-Means - Bag of Visual Words - Random Forest Pipeline we can see that the scores of the previous one is better. Macro averaged F1 score of 24% and 7% respectively. Which can lead that SVM is better classifier for this task, since SIFT and ORB has comparable performance.

The best model is ORB - GMM - Fisher Vectors - SVM Pipeline but since we did not implement any methods in that, we only determine the dictionary size of GMM and SVM hyperparameter, we select our best pipeline as ORB - Hierachical K-Means - Bag of Visual Words - SVM. The images below are the wrongly classified images from that pipeline.

8 Member Contribution

Some methods were implemented more than once by different group members as the codebase for each member was separate until the submission.

8.1 Abdurrahman Dilmaç

8.2 Yusufcan Manav

- Implement and test ORB - Hierachical K-Means - Bag of Visual Words - SVM pipeline.
- Configured and test ORB - GMM - Fisher Vectors - SVM pipeline for this task.
- Implement Hierachical K-Means and Bag of Visual Words methods

8.3 Barış Yamansavaşçılar

The pipeline of SIFT - K-means - Bag of Visual Words - Random Forest is implemented.

8.4 Mansur Yeşilbursa

HOG features, Spectral clustering, K-means, BoW, SVM
HOG based pipelines

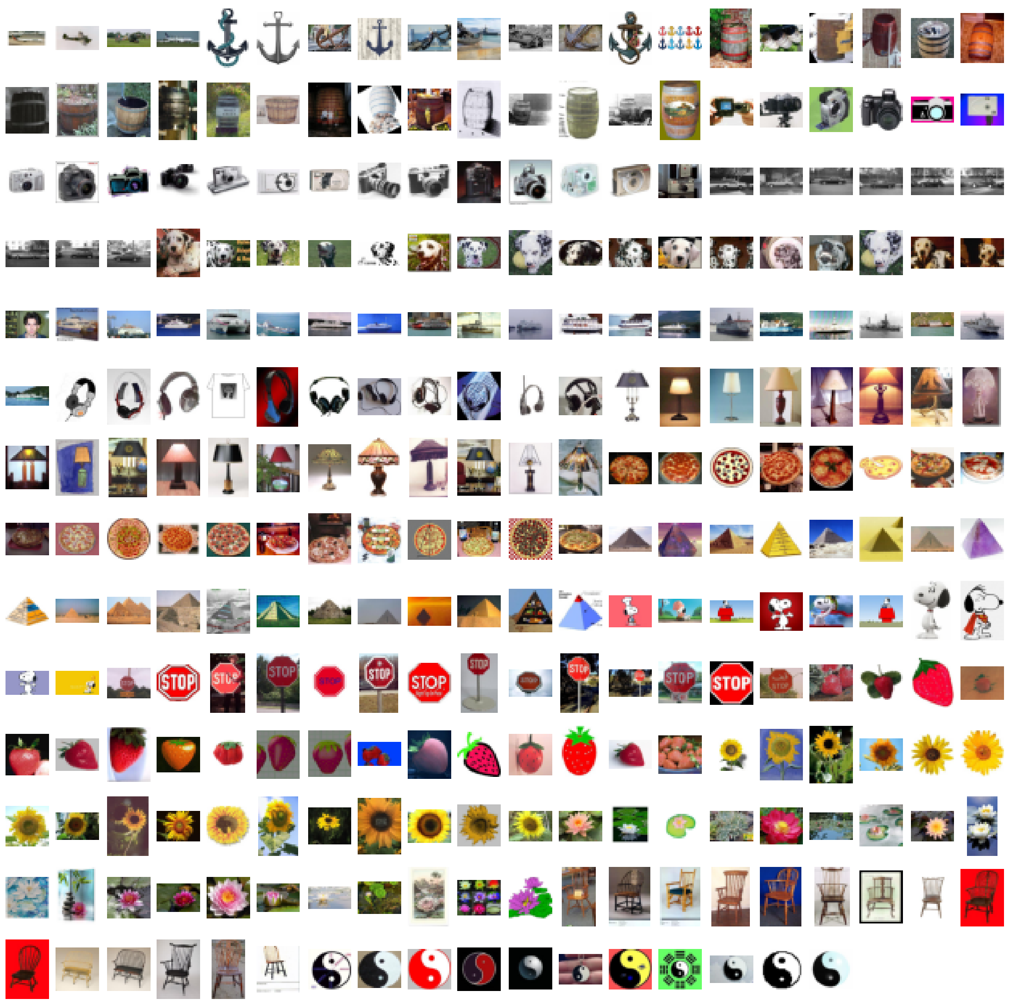

Figure 6: Wrongly Classified Images for ORB - Hierachical K-Means - Bag of Visual Words - SVM