Final Project Report

Ren Tu

MIDS W281: Computer Vision Spring 2022

Objective

This project is focused on an image classification task. We will have a dataset of approximately 2,200 images from 20 unique categories. We will create a variety of image features to distinguish between images and implement a classification model in Python.

Data Cleaning

Our image dataset will be split in the following way: approximately 1,500 initial images will be split into 80% training and 20% validation to help with creating features and tuning models. Another ~700 images will be held out as a final test set to evaluate our best model.

We found that across the dataset, the minimum number of rows for an image was 102 and the minimum number of columns for an image was 115. To ensure consistency of image sizes for this project, we decided crop every image to its 102x115 central pixels as the objects represented by the images are mostly located in the center portion of images.

Feature Extraction

The first type of feature we created was Histogram of Oriented Gradients (HOG), which has potentially useful signals on the appearance and shapes of objects within images. Using Python's skimage library, we tried various combinations of HOG parameters to find the combination with the highest explained variance of top 10 components when using principal components analysis (PCA):

	orientations	pixels_per_cell	explained_var
0	4	8	0.283889
1	4	9	0.281376
2	5	8	0.270032
3	5	9	0.267827
8	6	8	0.255726
9	6	9	0.254135
6	7	8	0.245201
7	7	9	0.244031
4	8	8	0.233596
5	8	9	0.232604

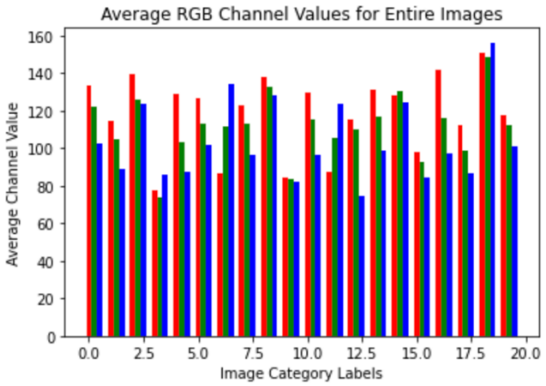
From the summary table above, we found that HOG parameters with orientation of 4 and pixels per cell of 8x8 was best. The below sample image of HOG features of an airplane shows that these parameters do make visual sense as the 102x115 image are split into enough cell blocks that each block contains unique signals on the image gradients.

The second type of feature we created was Harris Corners (HC), which has potentially distinguishing signals on the locations of corners within images. Using Python's skimage library, we tried many combinations of HC parameters to find the combination with the best explained variance of top 10 components when using PCA:

	min_dist	thres_rel	explained_var
2	1	0.0100	0.041929
5	2	0.0100	0.035526
8	3	0.0100	0.031690
1	1	0.0010	0.031673
11	4	0.0100	0.028335
4	2	0.0010	0.026964
14	5	0.0100	0.026218
0	1	0.0001	0.025438
7	3	0.0010	0.024577
3	2	0.0001	0.022362

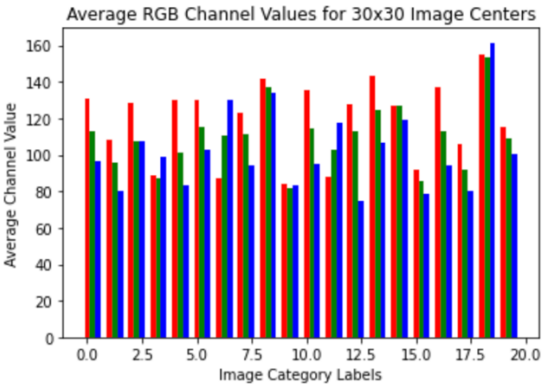
From the summary table above, we found that HC parameters with min_dist of 1 and thres_rel of 0.01 was best. The below sample image of HC features of an airplane shows that these parameters do make visual sense as HC recognizes many corners on the airplane's windows, wings and wheels.

The third feature type we created was the average RGB color channel values for an entire image. Here is what the average red, green and blue channel values across image categories look like:



The above plot indicates that there is enough dispersion for every color channel across image types for these color features to contain potentially helpful signals.

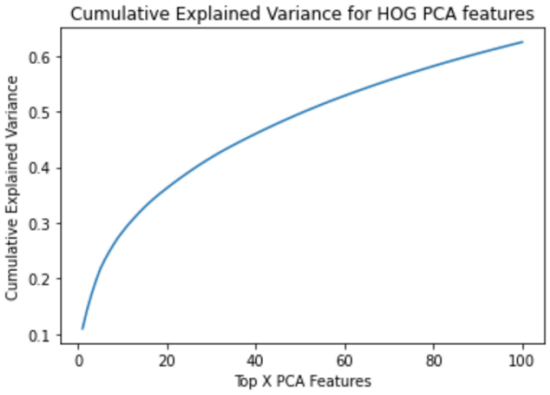
Finally, the fourth feature type we created was the average RGB color channel values for a smaller 30x30 central square with images. From the sample images we have seen above, the main object is mostly in the image center while being surrounded by other objects at the edges. Therefore, it may be useful to focus on a smaller central region to reduce the potential noise from surrounding objects. Here is the average red, green and blue channel values of central regions across image categories:

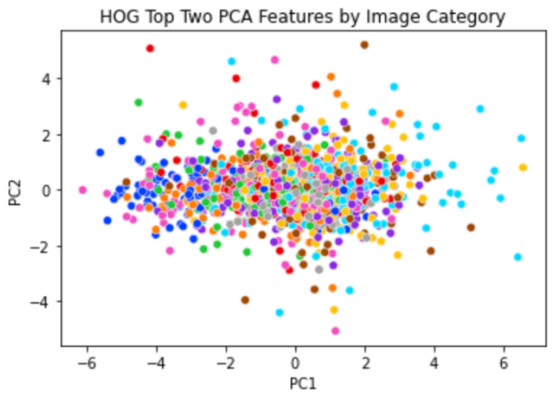


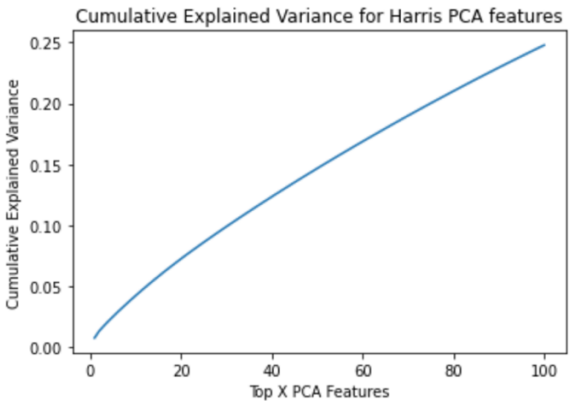
The above plot suggests that there is also enough dispersion for every color channel across image types for these color features on the smaller 30x30 central regions to contain potentially distinguishing signals.

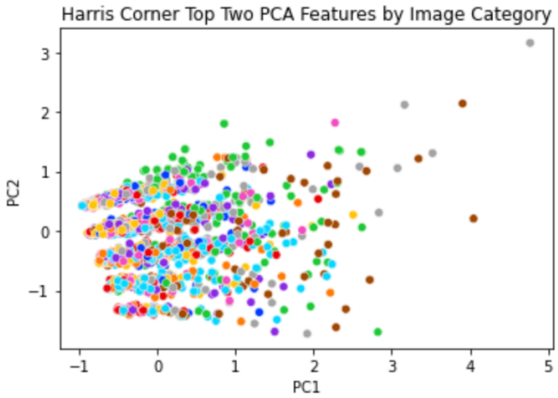
Feature Visualization

Two of the feature types we created, HOG and HC, are very high-dimensional. Therefore, we decided to use PCA as a dimensionality reduction technique to transform these features into fewer principal components. Here is what these features look like after PCA transformations:









The top two HOG plots demonstrate that the top 100 HOG PCA features explain ~60% of the variance while the top two features can already separate the 20 image categories into partially defined clusters. Meanwhile, the bottom two Harris Corner plots indicate that the top 100 HC PCA features only explain ~25% of the variance and the top two features cannot produce easily separable clusters among the 20 image categories. With the greater strength of the HOG features, we decided to use the top 30 PCA HOG features in modeling as that amount can explain a significant portion of variance while reducing to a small enough set of features to balance the ~2,200 image total dataset. For the Harris Corner features, we decided to use only the top 10 PCA HC features given the weaker strength seen from the plots.

Image Classification

Once the feature engineering was complete, we utilized our 46 features (30 HOG PCA features, 10 Harris Corner PCA features, 3 full-image average color channel features, 3 central-image average color channel features) from our training set (80% split, ~1,200 images) as inputs into three models: Logistic Regression, SVM and K-Nearest Neighbors. After hyperparameter tuning, we found that the best accuracy on our validation set (20% split, ~300 images) was the SVM model with 35% prediction accuracy. Logistic Regression and K-Nearest Neighbors lagged at 31% and 22% accuracy respectively. A more detailed view of classification accuracy for the SVM model at the image category level indicates that it can generalize to an unseen dataset across most image categories. 19 out of 20 image categories in the validation set saw accuracies greater than chance (>5%) with the only exception being category 10 (kangaroo), which had a 0% accuracy partially due to having the smallest support (11 images) from the random train/validation split.

Conclusions

Using our best SVM model, we made classification predictions on the held-out unseen test set of ~700 images (predictions included in separate CSV file). When random sampling 50 test set images and manually labeling them using our human eyes, our predictions were able to achieve a similar accuracy to the 35% accuracy seen in the validation set. Our SVM model performed well on objects with unique and well-defined shapes such as airplanes or well-defined colors such as leopards. On the other hand, our SVM model performed poorly on objects with overlapping shapes and/or colors such as bears/gorillas, killer-whales/dolphins, and goats/llamas. From these results, we learned that while features on object shapes, corners and colors are good starting points to recognize different objects, more sophisticated feature engineering is required to achieve even better accuracy in massively multi-class image classification tasks.