Computer Vision: Image Classification Part 1

Sylke Gosen, Victor Zuanazzi, Simon Passenheim, David Speck

April 5, 2019

1 Introduction

For part 1 of this assignment, we will be training a Visual Bag of Words image classifier on the STL-10 dataset. In short, we will extract SIFT descriptors for each image in the training data, cluster half of those to create a vocabulary of visual words and then create histograms for the other half of the training data based on how close their descriptors are to each of the words in the vocabulary. Then we will train a support vector machine classifier using these histograms and the provided image labels to learn binary classifiers that can predict for each image label based on a histogram over visual words. We will explore various hyperparameter settings for this setup to find out what works best.

2 Experimental setup

We will only classify between 5 of the 10 provided classes: airplanes, birds, ships, horses, and cars. We will use all 2500 training images. Half of those will be used for creating the vocabulary and half will be used for training the 5 support vector machines. For the initial exploration, we will consider three hyperparameters. Color space (Gray, RGB or Opponent), vocabulary size (400, 1000 or 4000) and sampling strategy (key points or dense). We will run a grid search over all possible variations and report mean average precision over the 5 classes for each.

For dense sampling, we had to adjust the default step size and size parameters to make it computationally feasible to run. We chose to use a step size of 5 and a size of 21 which results in a total of 61250 descriptors for the entire training set. This is similar in scale to the 39036 descriptors for the key point sampling strategy.

Because we train a separate binary support vector machine for each class label, the data we train these on is unbalanced with 20% belonging to the positive class and 80% belonging to the negative class. To make the support vector machines train properly using unbalanced data we use a cost function that specifies a penalty for miss-classification for the minority class that is 4 times as high. Through ad-hoc experimentation, we found that the RBF kernel seemed to perform better than the linear or Gaussian kernels.

Note that, because of the stochastic nature of the algorithm, it would be a good idea to run this setup multiple times to get an estimate for the significance of any differences we might find, but due to computational limitations we have decided to stick to running it only once as an exploratory experiment.

3 Results

Table 1: mAP Scores for key point sampling.

Color space \ Vocabulary size	400	1000	4000
Gray	0.498	0.496	0.484
RGB	0.498	0.504	0.498
Opponent	0.533	0.536	0.525

Table 2: mAP Scores for dense sampling.

Color space \ Vocabulary size	400	1000	4000
Gray	0.795	0.799	0.786
RGB	0.792	0.813	0.790
Opponent	0.755	0.761	0.734

In table 1 and 2 we can see a summary of the results. The most striking finding is that dense sampling works way better than key point sampling. Due to the small difference in total descriptors for both methods in our setup, this was not what we expected. One would expect descriptors for key points to be more salient than descriptors on an even grid. There are several possible explanations. Either the difference in descriptors is still big enough to cause the difference in mAP score,

the size of the dense patches is more suited for this task than the size of features found by the key point algorithm or sampling on a grid is better than sampling key points for this application. To test these hypotheses we ran a few more tests for dense sampling with different parameter settings. We used a gray color space and a vocabulary size of 400 for these tests.

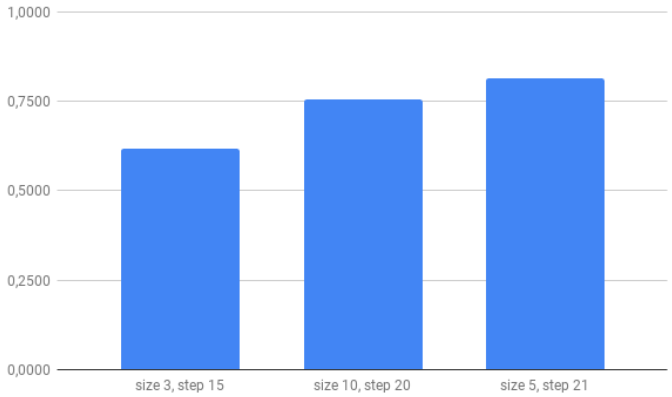


Figure 1: mAP scores for different dense sampling configurations

A default size of 3 combined with a step size of 15 results in 45000 total descriptors and an mAP score of 0.616. A size of 10 combined with a step size of 20 results in 20000 total descriptors and an mAP score of 0.756, as shown in figure 1. This seems to suggest that the total number of descriptors does not matter a lot and it is mostly the size parameter that strongly affects the performance. Bigger sizes seem to perform better, presumably because they use more information and the scale is more meaningful for this image classification task. We speculate that the increased performance for dense is mostly caused by the fact that key point tries to learn descriptors at multiple scales at once, while dense only has to consider descriptors at a single scale. This means that key point could be more powerful, but it is much harder to learn. If the scale for dense is optimized for a particular task, such as we did here, it will easily outperform key point unless the key point is provided with much more information to actually learn all scales at once.

For vocabulary size, we expect there to be an optimum where there are enough different words to capture differences between images, but not so many that each descriptor will be its own word with 1 occurrence each. It turns out that a vocabulary size of 1000 gets the best result in all cases except when combined with key point sampling and a gray color space where it just barely loses out to a vocabulary size of 400. The differences are rather small, but the fact that 1000 consistently outperforms the other sizes for all parameter settings is strong evidence that the vocabulary size of 1000 does perform significantly better.

The opponent color space performs slightly better for key point sampling and the RGB color space is part of the best performing parameter setting together with vocabulary size 1000 and dense sampling, but overall the color space also has little effect. This corresponds to the intuition that color information is not really needed to classify objects in images. Humans are perfectly able to classify objects in grayscale images so the color information is really not necessary and apparently does not even add much.

In figure 2 and figure 3 we can see the top 5 and bottom 5 images according to SVM score for each class for the best performing hyperparameter setting (Dense, RGB, 1000). There are two mistakes: The top 5 for both airplanes and birds include a ship.

(a) Airplane

(b) Bird

(c) Car

(d) Horse

(e) Ship

Figure 2: Top 5 images for each class with the best hyperparameters: RGB, dense sampling and vocabulary size 1000.

(a) Airplane

(b) Bird

(c) Car

(d) Horse

(e) Ship

Figure 3: Bottom 5 images for each class with the best hyperparameters: RGB, dense sampling and vocabulary size 1000.

For comparison, we also show the retrieved images of the worst model in the image 4. The model makes some interesting mistakes. For instance, the picture the model retrieved as most likely an airplane is actually a boat. Similar for birds, the most likely bird is actually an airplane. That is probably due to similar context information, large smooth areas around the object and similar shape. Both airplanes and birds have wings.

(a) Airplane

(b) Bird

(c) Car

(d) Horse

(e) Ship

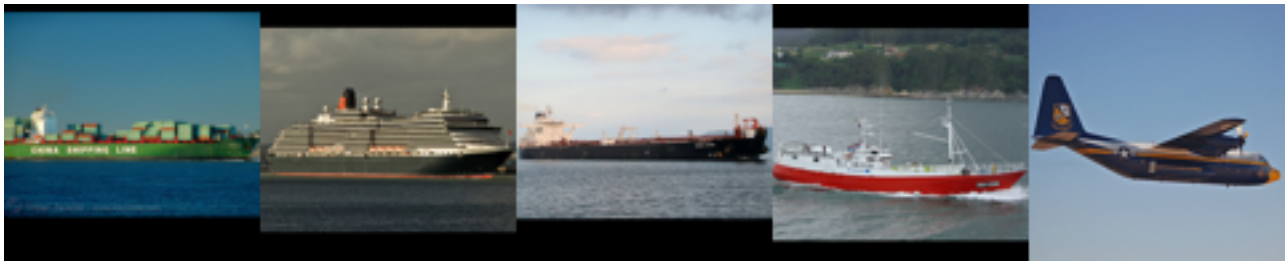


Figure 4: Top 5 images for each class with the best hyperparameters: gray scale, key point and vocabulary size 4000.

4 Extra Exploration

We experimented with several other hyperparameters. Trying out all different variations using a grid search would take too much computing power so any additional comparisons were done, unless specified otherwise, using the best hyperparameters we found above: Dense sampling, RGB color space and a vocabulary size of 1000.

4.1 Less training data

To find out how this approach scales with the number of training images available we trained a model with half of the training data and a model with a quarter of the training data. In all cases, half of the data was used to build the vocabulary and half to train the SVM classifier.

This results in an mAP score of 0.754 when using half of the training data and an mAP score of 0.714 when using a quarter of the training data. As shown in figure 5. As expected, the performance drops when less training data is used, but the classifier still works quite well even with a quarter of the training data.

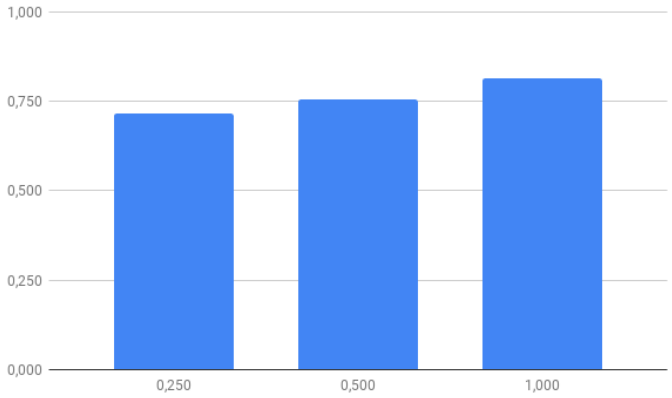


Figure 5: mAP scores for using 25%, 50% and 100% of the training set.

4.2 Different SVM kernels

To justify our choice for using an RBF kernel, we also experimented with linear and Gaussian kernels.

This results in an mAP score of 0.799 and 0.803 for a linear and Gaussian kernel respectively, as shown in figure 6. So an RBF kernel does indeed give the best performance which is not entirely unexpected because it can model the most complex relationships. This increased complexity also comes with the risk of overfitting, but that does not seem to cause any problems here.

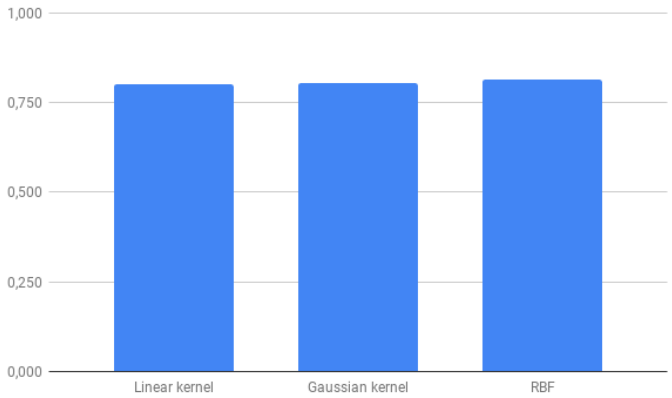


Figure 6: mAP scores for using linear, Gaussian and RBF kernel types.

4.3 LIOP descriptor

The VLFeat library also includes the Local Intensity Order Pattern descriptor so we decided to try this instead of SIFT. There is no dense implementation for this descriptor so we will be comparing the two descriptor types using our best hyperparameters for key point sampling: Opponent color space and a vocabulary size of 1000. We will also use the same key points for both descriptors. When doing this we find an mAP score of 0.484 for LIOP which is considerably lower than the 0.536 we found for SIFT, as shown in 7.

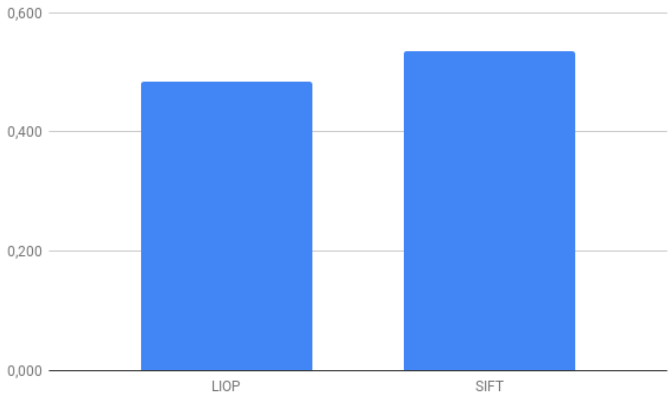


Figure 7: mAP scores comparing LIOP and SIFT descriptors.

4.4 K-medoids

We also tried a different clustering algorithm, namely k-medoids which uses actual data points as cluster centers instead of the mean of all elements of a cluster. Using k-medoids with key point sampling, opponent color space and a vocabulary size of 1000 gives an mAP score of 0.527 which is slightly worse than 0.536 by using k-means, as shown in figure 8.

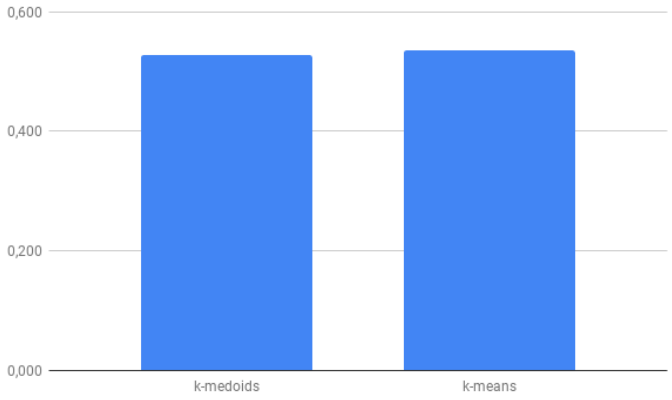


Figure 8: mAP scores comparing k-medoids and k-means as clustering techniques.

4.5 Using all training data for vocabulary and SVM training

The assignment is set up such that half of the training data is used for creating the vocabulary and half is used for training the support vector machines. This also seems to be the modus operandi in published research. We were not entirely convinced by the arguments against using all training data for both phases so we decided to experiment with that.

For the best hyperparameter setting, we end up with an mAP score of 0.814 which is almost the exact same score we get when splitting the data.

If we instead use key point sampling, gray color space and a vocabulary size of 400 we get an mAP score of 0.533 which is a considerable improvement. This seems to imply that adding the effective extra data for our best parameter setting adds no meaningful information, but using all data can yield improvement for subpar parameter settings. Possibly because

key point sampling scales better with more data due to having to learn multiple scales at once. In any case, we found no evidence for the claim that using the same data for creating the vocabulary and training the support vector machines can be problematic. At worst, the performance will be the same.

Figure 9: mAP scores comparing the best and worst set of hyperparameters and the influence of using the whole data for vocabulary building and training as opposite of splitting it in two halves.

5 Conclusion

Out of all our experiments, we found that the best parameter setting is using sift descriptors, dense sampling with size 21 and stepsize 5, RGB color space, a vocabulary size of 1000, kmeans clustering and an RBF kernel for the SVM. We also found no reason, other than computational limitations, not to use all training data for creating the vocabulary as well as training the support vector machines.

6 Appendix

(a) Airplane

(b) Bird

(c) Car

(d) Horse

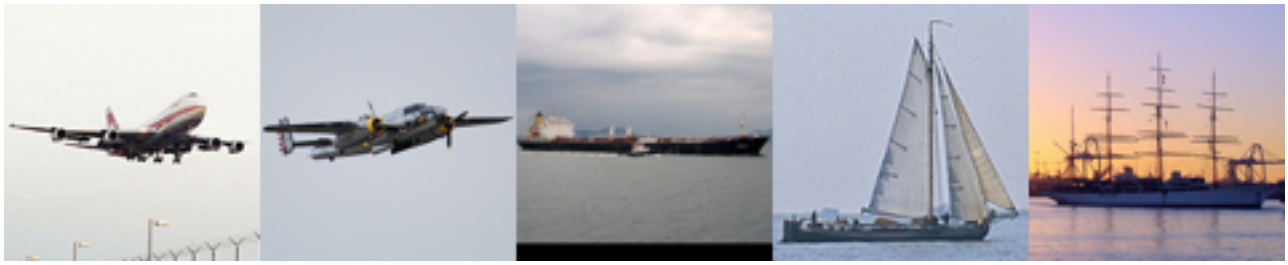
(e) Ship

Figure 10: Top 5 images for each class with hyperparameters: RGB, key point sampling and vocabulary size 400.

(a) Airplane

(b) Bird

(c) Car



(d) Horse

(e) Ship

Figure 11: Bottom 5 images for each class with hyperparameters: RGB, key point sampling and vocabulary size 400.

(a) Airplane

(b) Bird

(c) Car

(d) Horse

(e) Ship

Figure 12: Top 5 images for each class with hyperparameters: RGB, key point sampling and vocabulary size 1000.

(a) Airplane

(b) Bird

(c) Car

(d) Horse

(e) Ship

Figure 13: Bottom 5 images for each class with hyperparameters: RGB, key point sampling and vocabulary size 1000.

(a) Airplane

(b) Bird

(c) Car

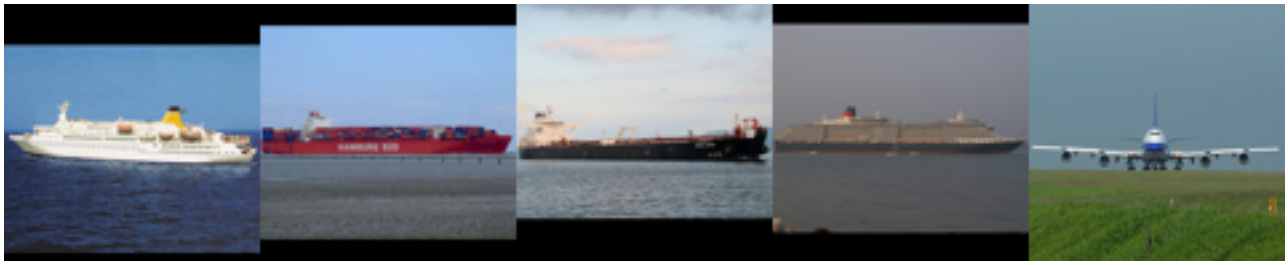
(d) Horse

(e) Ship

Figure 14: Top 5 images for each class with hyperparameters: RGB, key point sampling and vocabulary size 4000.

(a) Airplane

(b) Bird



(c) Car

(d) Horse

(e) Ship

Figure 15: Bottom 5 images for each class with hyperparameters: RGB, key point sampling and vocabulary size 4000.

(a) Airplane

(b) Bird

(c) Car

(d) Horse

(e) Ship

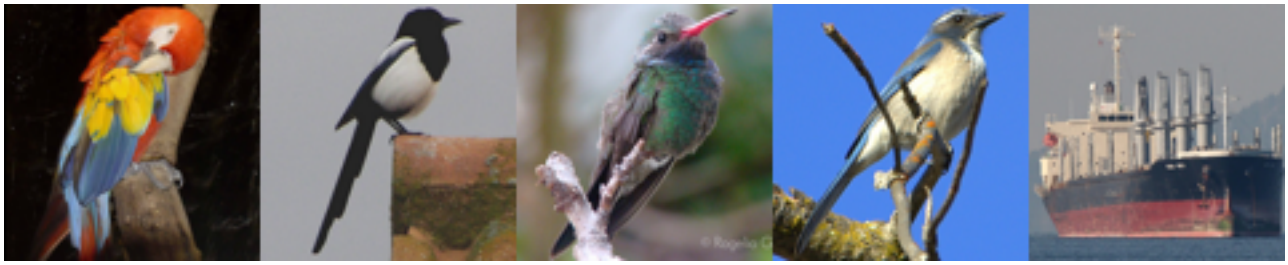
Figure 16: Top 5 images for each class with hyperparameters: RGB, dense sampling and vocabulary size 400.

(a) Airplane

(b) Bird

(c) Car

(d) Horse



(e) Ship

Figure 17: Bottom 5 images for each class with hyperparameters: RGB, dense sampling and vocabulary size 400.

(a) Airplane

(b) Bird

(c) Car

(d) Horse

(e) Ship

Figure 18: Top 5 images for each class with hyperparameters: RGB, dense sampling and vocabulary size 1000.

(a) Airplane

(b) Bird

(c) Car

(d) Horse

(e) Ship

Figure 19: Bottom 5 images for each class with hyperparameters: RGB, dense sampling and vocabulary size 1000.

(a) Airplane

(b) Bird

(c) Car

(d) Horse

(e) Ship

Figure 20: Top 5 images for each class with hyperparameters: RGB, dense sampling and vocabulary size 4000.

(a) Airplane

(b) Bird

(c) Car

(d) Horse

(e) Ship

Figure 21: Bottom 5 images for each class with hyperparameters: RGB, dense sampling and vocabulary size 4000.

(a) Airplane

(b) Bird

(c) Car

(d) Horse

(e) Ship

Figure 22: Top 5 images for each class with hyperparameters: gray scale, key point sampling and vocabulary size 400.

(a) Airplane

(b) Bird

(c) Car

(d) Horse

(e) Ship

Figure 23: Bottom 5 images for each class with hyperparameters: gray scale, key point sampling and vocabulary size 400.

(a) Airplane

(b) Bird

(c) Car

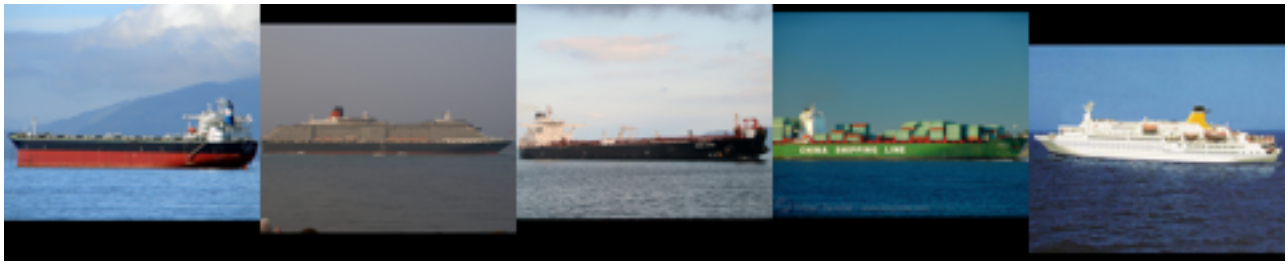
(d) Horse

(e) Ship

Figure 24: Top 5 images for each class with hyperparameters: gray scale, key point sampling and vocabulary size 1000.

(a) Airplane

(b) Bird



(c) Car

(d) Horse

(e) Ship

Figure 25: Bottom 5 images for each class with hyperparameters: gray scale, key point sampling and vocabulary size 1000.

(a) Airplane

(b) Bird

(c) Car

(d) Horse

(e) Ship

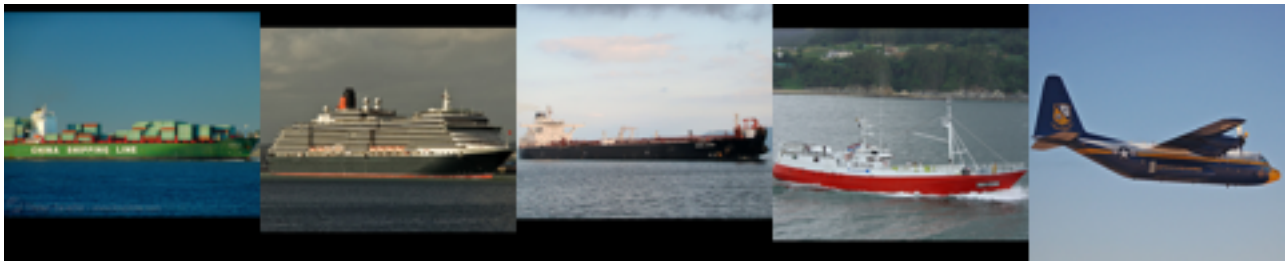


Figure 26: Top 5 images for each class with hyperparameters: gray scale, key point sampling and vocabulary size 4000.

(a) Airplane

(b) Bird

(c) Car

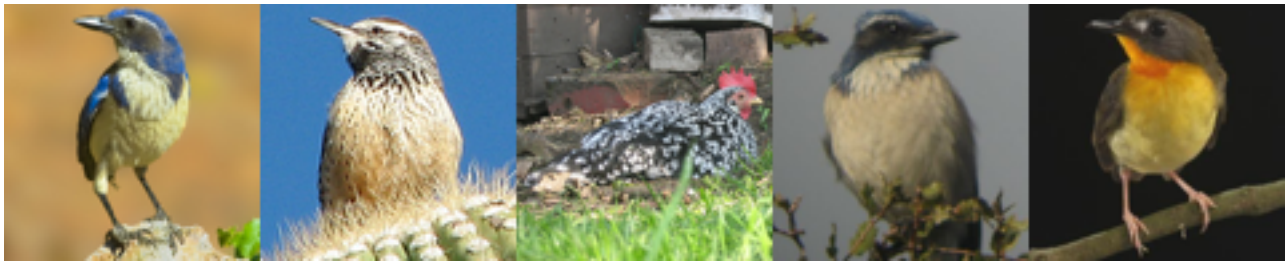
(d) Horse

(e) Ship

Figure 27: Bottom 5 images for each class with hyperparameters: gray scale, key point sampling and vocabulary size 4000.

(a) Airplane

(b) Bird



(c) Car

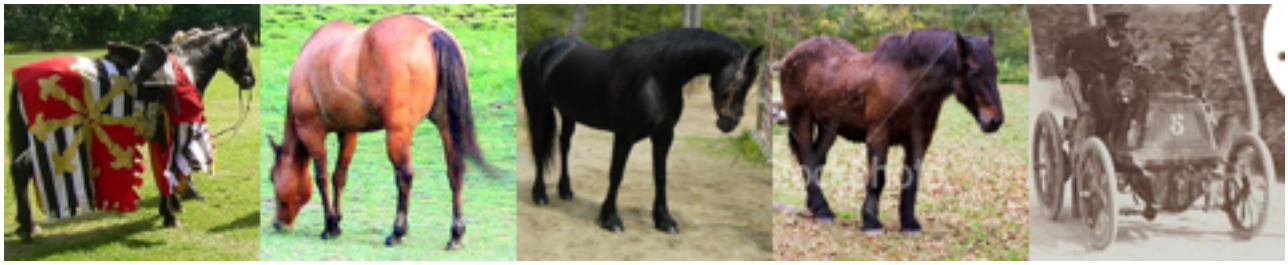
(d) Horse

(e) Ship

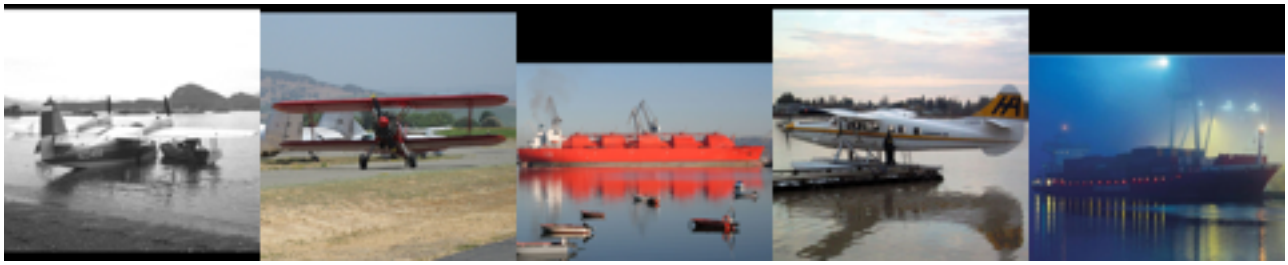
Figure 28: Top 5 images for each class with hyperparameters: gray scale, dense sampling and vocabulary size 400.

(a) Airplane

(b) Bird



(c) Car



(d) Horse

(e) Ship

Figure 29: Bottom 5 images for each class with hyperparameters: gray scale, dense sampling and vocabulary size 400.

(a) Airplane

(b) Bird

(c) Car

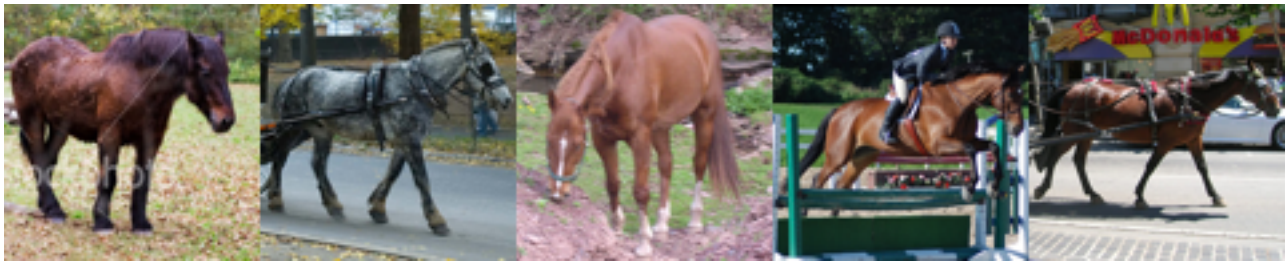
(d) Horse

(e) Ship

Figure 30: Top 5 images for each class with hyperparameters: gray scale, dense sampling and vocabulary size 1000.

(a) Airplane

(b) Bird



(c) Car

(d) Horse

(e) Ship

Figure 31: Bottom 5 images for each class with hyperparameters: gray scale, dense sampling and vocabulary size 1000.

(a) Airplane

(b) Bird

(c) Car

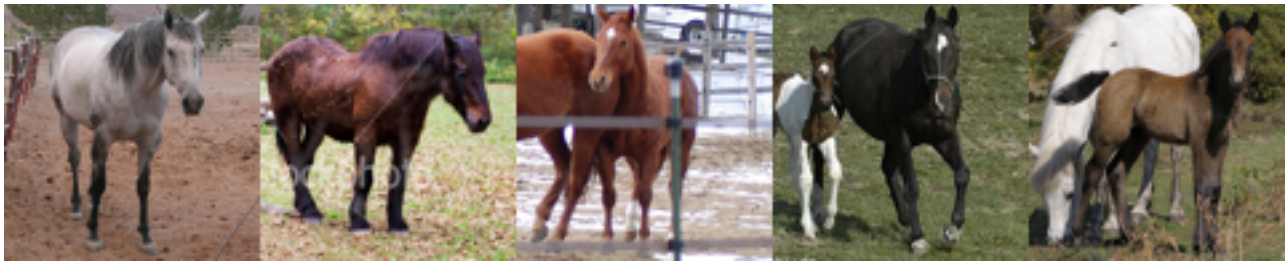
(d) Horse

(e) Ship

Figure 32: Top 5 images for each class with hyperparameters: gray scale, dense sampling and vocabulary size 4000.

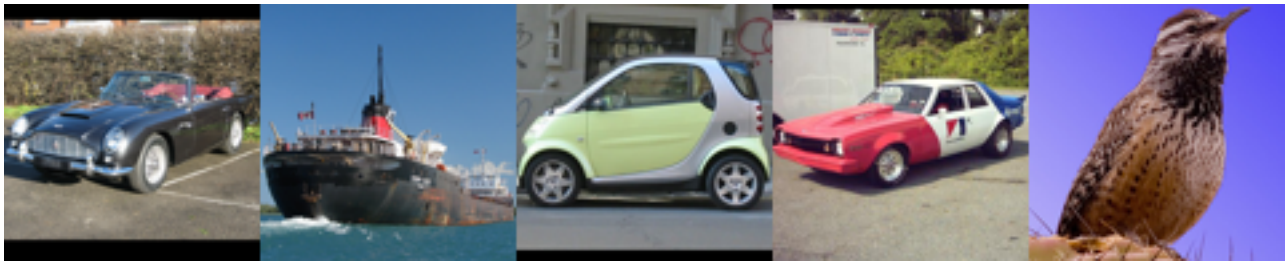
(a) Airplane

(b) Bird



(c) Car

(d) Horse



(e) Ship

Figure 33: Bottom 5 images for each class with hyperparameters: gray scale, dense sampling and vocabulary size 4000.

(a) Airplane

(b) Bird

(c) Car

(d) Horse

(e) Ship

Figure 34: Top 5 images for each class with hyperparameters: opponent, key point sampling and vocabulary size 400.

(a) Airplane

(b) Bird

(c) Car

(d) Horse

(e) Ship

Figure 35: Bottom 5 images for each class with hyperparameters: opponent, key point sampling and vocabulary size 400.

(a) Airplane

(b) Bird

(c) Car

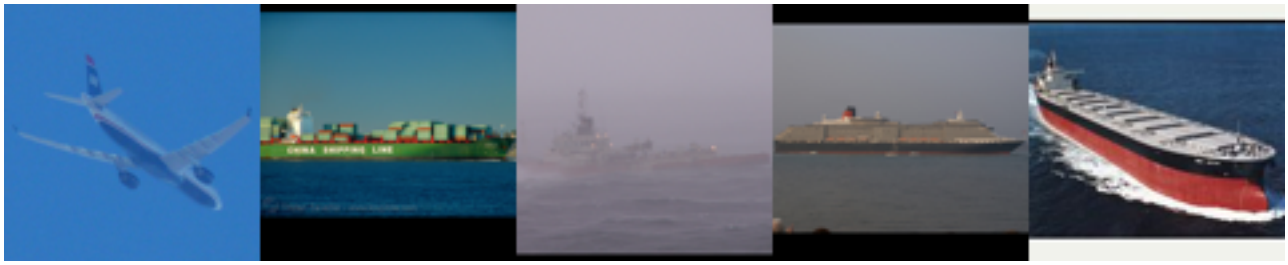
(d) Horse

(e) Ship

Figure 36: Top 5 images for each class with hyperparameters: opponent, key point sampling and vocabulary size 1000.

(a) Airplane

(b) Bird



(c) Car

(d) Horse

(e) Ship

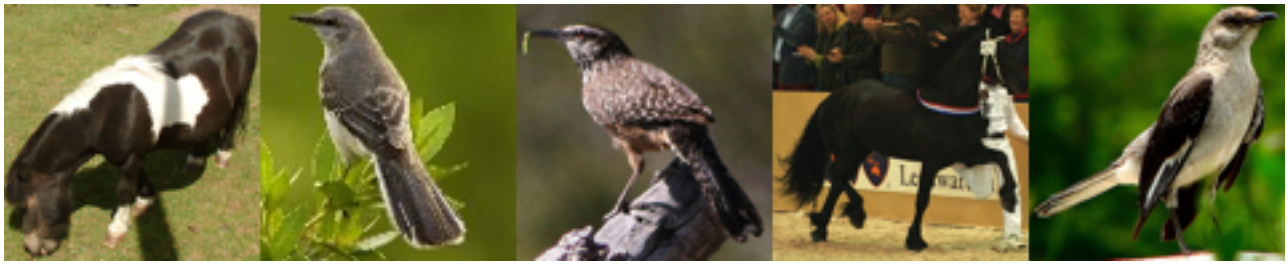


Figure 37: Bottom 5 images for each class with hyperparameters: opponent, key point sampling and vocabulary size 1000.

(a) Airplane

(b) Bird

(c) Car

(d) Horse

(e) Ship

Figure 38: Top 5 images for each class with hyperparameters: opponent, key point sampling and vocabulary size 4000.

(a) Airplane

(b) Bird

(c) Car

(d) Horse

(e) Ship

Figure 39: Bottom 5 images for each class with hyperparameters: opponent, key point sampling and vocabulary size 4000.

(a) Airplane

(b) Bird

(c) Car

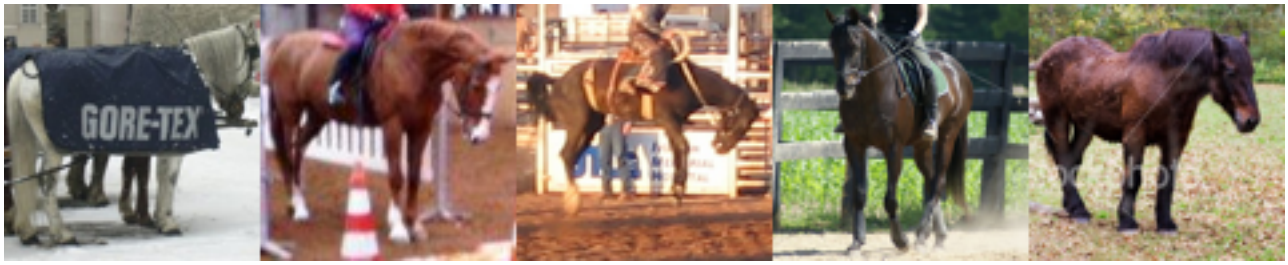
(d) Horse

(e) Ship

Figure 40: Top 5 images for each class with hyperparameters: opponent, dense sampling and vocabulary size 400.

(a) Airplane

(b) Bird



(c) Car

(d) Horse

(e) Ship

Figure 41: Bottom 5 images for each class with hyperparameters: opponent, dense sampling and vocabulary size 400.

(a) Airplane

(b) Bird

(c) Car

(d) Horse

(e) Ship

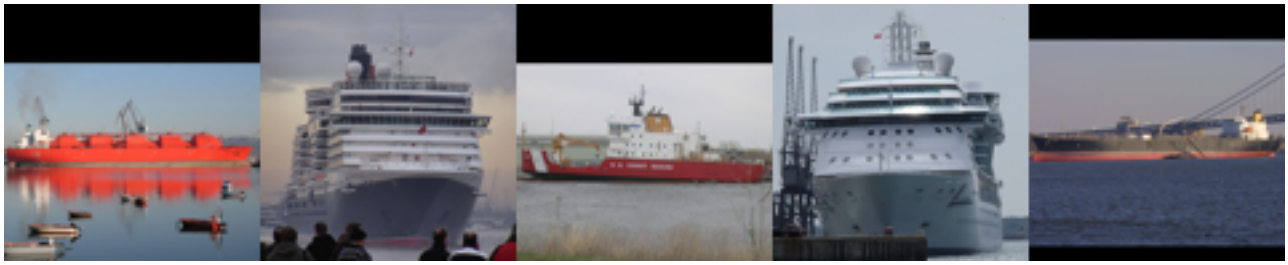


Figure 42: Top 5 images for each class with hyperparameters: opponent, dense sampling and vocabulary size 1000.

(a) Airplane

(b) Bird

(c) Car

(d) Horse

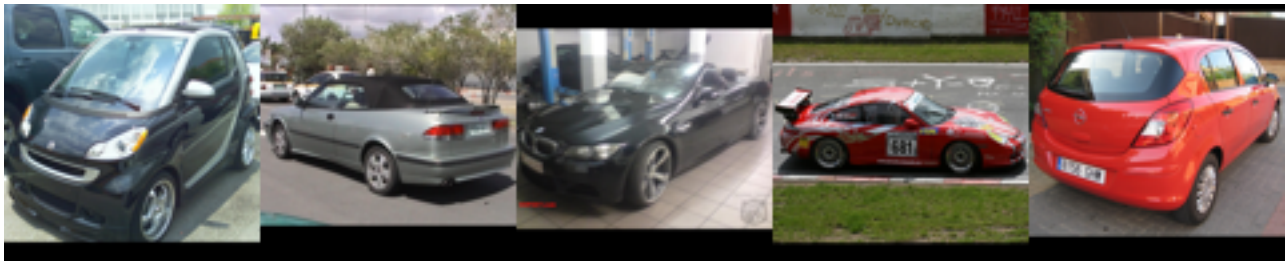
(e) Ship

Figure 43: Bottom 5 images for each class with hyperparameters: opponent, dense sampling and vocabulary size 1000.

(a) Airplane

(b) Bird

(c) Car



(d) Horse

(e) Ship

Figure 44: Top 5 images for each class with hyperparameters: opponent, dense sampling and vocabulary size 4000.

(a) Airplane

(b) Bird

(c) Car

(d) Horse

(e) Ship

Figure 45: Bottom 5 images for each class with hyperparameters: opponent, dense sampling and vocabulary size 4000.