UNIVERSITY OF AMSTERDAM

LAB PROJECT PART 1

Image Classification using Bag-of-Words

October 18, 2021

Students:

Piyush Bagad
13677640

Mark Alence
13771272

Ankit
13608568

Group:
Group 38

Course:
Computer Vision 1

Abstract

Image classification has traditionally been a canonical problem in visual understanding. It is an effortlessly simple tasks for human beings yet remains an important benchmark for Computer Vision algorithms. In this part of the lab project, we focus on a very traditional method for image classification that leverages visual bag-of-words representation of images based on SIFT-like descriptors and performs classification via classical models like SVM.

1 Introduction

Computer vision has come a long way: from Marvin Minsky at MIT assigning a summer project to an undergraduate to enable computers to see, to modern-day deep-learning based architectures that have achieved incredible results on a variety of high-level vision tasks like classification, detection, segmentation, 3D reconstruction to name a few. Through the course of its history, classifying an image into a predefined set of (object) categories has remained a vital problem in vision. A large amount of classical vision literature has been devoted towards image classification. In this project, we consider the approach of treating an image as a bag of visual words that it consists. More concretely, we consider SIFT/HoG features of an image and construct a vocabulary of visual words that could be used to predict the class for a new image based on frequency counts.

2 Dataset

For this project, we use the STL-10 dataset [1] that was proposed in 2011. It has 10 classes with annotated images chosen from ImageNet-1000. It has 500 samples (size: $96 \times 96 \times 3$) in each class in the training set while 800 per class in the test set. In this project, we only consider the given five classes: `airplanes, birds, ships, horses, cars`. A randomly selected subset of samples is shown in Figure 1 consisting of two images per class.

3 Method

3.1 Feature Extraction and Description

We use standard SIFT features (keypoints and corresponding descriptors) that we detect in each image in the training set. The detected SIFT key-points for a sample of two images per class

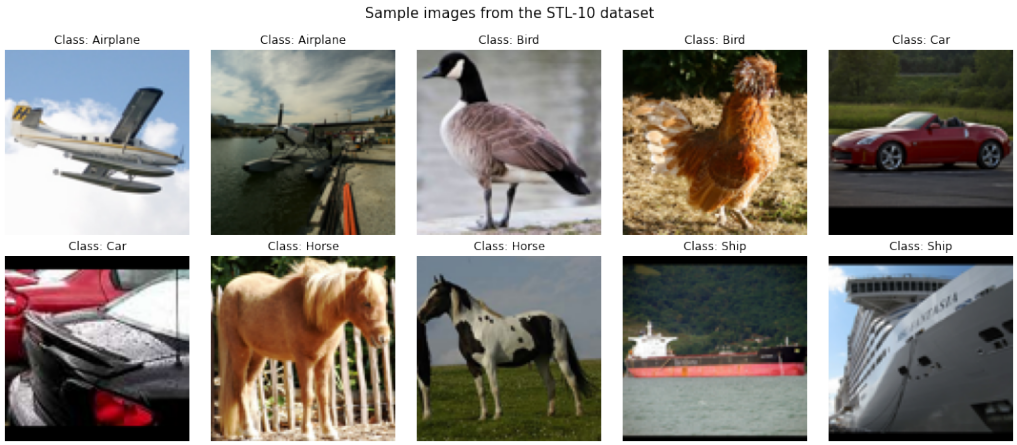


Figure 1: Randomly selected subset of samples from the STL-10 dataset.

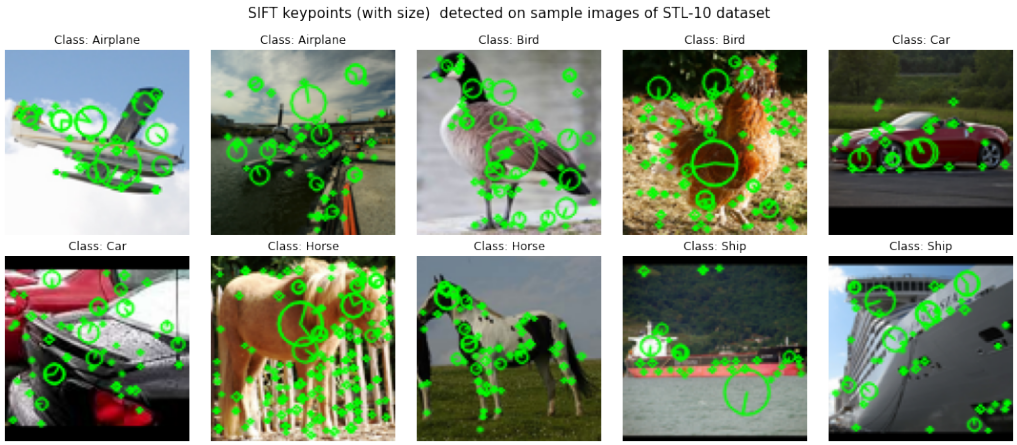


Figure 2: Detected SIFT key-points (with size) for a sample set of images in the dataset.

is shown in Figure 2. Notice that the keypoints look quite reasonable for various object classes. For example, on airplane, it detects some keypoints on the wings, on the body, for car, it detects keypoints around the wheels, doors or the back part and so on. Note that we also experiment with HoG features and talk about it in Section 4.

3.2 Building Visual Vocabulary

In order to build the visual vocabulary, we select 50% of the training images, compute the SIFT features and perform k-Means clustering over the SIFT descriptors with $K \in \{500, 1000, 2000\}$. Practically, it is infeasible to visualize these high-dimensional clusters. We work around this and randomly select 25 clusters and 10k descriptors to visualize these 25 clusters as two-dimensional TSNE embeddings in Figure 3.

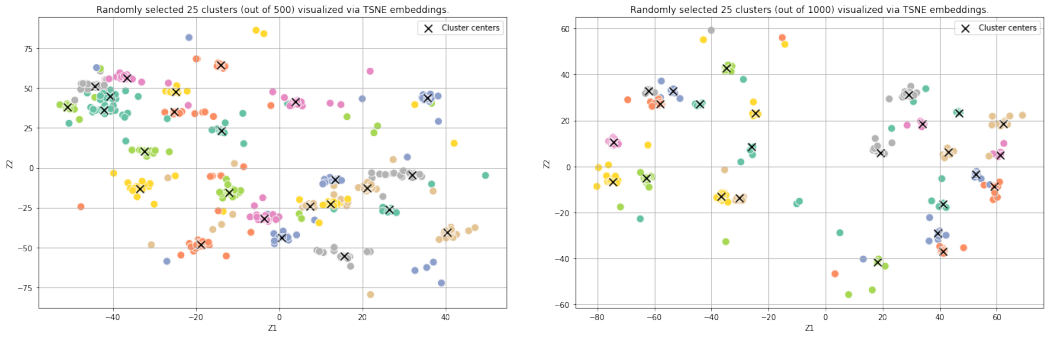
3.3 Encoding Features Using Visual Vocabulary

Once we have a visual vocabulary, we can represent each image as a collection of visual words. The process has been described in the following section.

3.4 Representing images by frequencies of visual words

Once the visual vocabulary is created, we encode a given image as follows: first, we compute the descriptors via SIFT or similar method. Next, for each descriptor we find the closest cluster center

Lab Project Part 1



(a) Total clusters $k = 500$ (b) Total clusters $k = 1000$

Figure 3: Visualizing randomly selected 25 clusters out of $k(= 500, 1000)$ clusters with `cross` denoting the cluster centers.

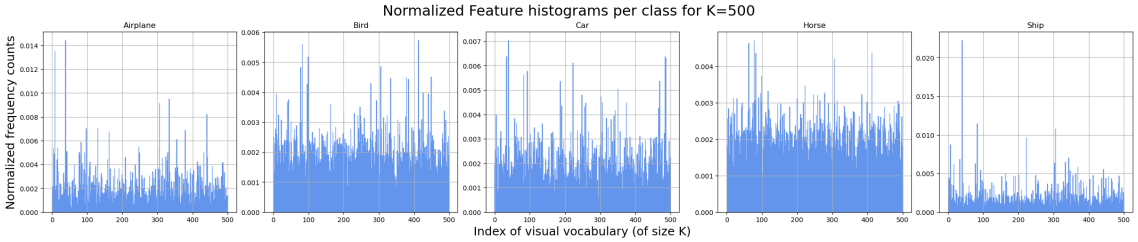


Figure 4: Feature histograms per class for the training set images.

(word in the visual vocabulary) and increment its count. This gives us a K-dimensional count vector which we normalize by the number of visual words found in the given image. We do this for each of the 1250 training images that were not used in the creation of the vocabulary. In Figure 4, we show the mean of histograms of all images for a given class, for $K = 500$.

We note the following observations:

1. In general, the number of visual words in an image of class `airplanes`, `ships` is much lesser than those from others classes.
2. The peaks and near-peaks are different for different classes - this makes sense because this intuitively implies that, for instance, images from `airplanes` class will have more visual words that correspond to, say, airplane-wings or the front of the plane and so on.
3. All the class-histograms seem to be multi-modal impliying that there are multiple "cannonical cues" that the algorithm finds in trying to describe images of a given class.

3.5 Classification

We use Support Vector Machines (SVMs) and perform one-vs-rest classification for each of the classes. The training set is constructed based on samples that were *not* used in the creation of visual vocabulary. The test set consists of 800 samples per class, and thus, has $800 \times 5 = 4000$ samples in total. Table 1 shows dataset statistics like train and test set size.

Split / step	k-Means	SVM	Total
Train	250 (1250)	250 (1250)	2500
Test	N.A.	800 (4000)	4000

Table 1: Dataset statistics for STL-10 dataset. Number in braces show total across classes.

Descriptor	K	$AP_{airplanes}$	AP_{birds}	AP_{cars}	AP_{horses}	AP_{ships}	**Mean (mAP)**
SIFT	500	0.624	0.474	0.619	0.607	0.599	**0.585**
	1000	0.625	0.473	0.596	0.609	0.577	0.576
	2000	0.607	0.454	0.589	0.594	0.552	0.559
HoG	500	0.642	0.634	0.753	0.688	0.698	**0.683**
	1000	0.621	0.623	0.753	0.675	0.697	0.674
	2000	0.631	0.617	0.760	0.670	0.672	0.670

Table 2: Average precision (AP) per class and mean average precision (mAP) for varying K (number of clusters) and feature descriptor methods.

4 Experimental Evaluation

4.1 Qualitative evaluation

To visualize our results qualitatively, for every class, we show the top-5 and worst-5 predictions based on the prediction scores obtained by SVMs. For brevity, we only show results for all classes for the case $K = 500$ within the main report in Figure 5. For other cases (e.g. $K = 1000, 2000$), we refer the reader to the appendix section.

4.2 Quantitative evaluation

While our primary evaluation metric is Mean Average Precision (`mAP`), we also compute accuracy for better understanding. We perform various experiments with the key axes of variations being: size of visual vocabulary K, method of extracting feature descriptors and hyper-parameters of SVM.

4.2.1 Size of visual vocabulary

We vary the size of visual vocabulary (i.e. number of clusters) $K \in \{500, 1000, 2000\}$ for feature descriptors SIFT and HoG keeping hyper-parameters for SVM fixed and report the results in Table 2. With both feature descriptor methods, we see that $K = 500$ provides the best results in terms of mAP and it decreases as K increases. This may be because K controls amount of regularization in some sense. Because the larger the vocabulary, the higher is the number of words picked from the train set and that may mean that a test image may contain very many of these with uniform counts making its encoded feature vector less discriminatory.

4.2.2 SIFT vs HoG

The BoW classifier pipeline seems to perform better with HoG-based descriptors as against SIFT features (Table 2). For HoG, we use the parameters pixels per cell as 16 and cells per block as 4. We get an output of 1152-dimensional vector which can be seen as a $(12 \times 12) \times 8$ descriptors. Although HoG is more prone to overfitting, its better performance maybe attributed to the similarity of the test set with that of train set.

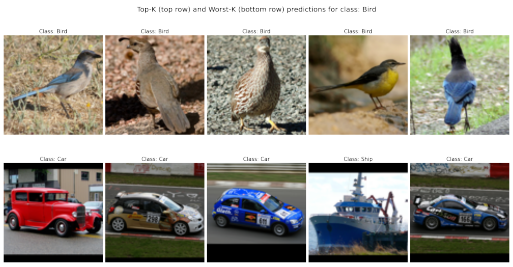
4.2.3 Impact of hyper-parameters of SVM

We experiment with two key hyper-parameters of SVM: the regularization parameter $C \in \{0.1, 1.0, 10.0\}$, and the kernel function $\kappa \in \{\texttt{rbf}, \texttt{poly}, \texttt{linear}\}$. In this experiment, we keep $K = 500$ as constant and use SIFT features. The results are tabulated in Table 3. We note that we obtain the best performance using the default values of $C = 1.0$ and $\texttt{kernel} = \texttt{rbf}$.

5 Conclusion

In summary, we studied, implemented and experimented with a pipeline for Visual Bag-of-words' based classification based on image feature detectors like SIFT.

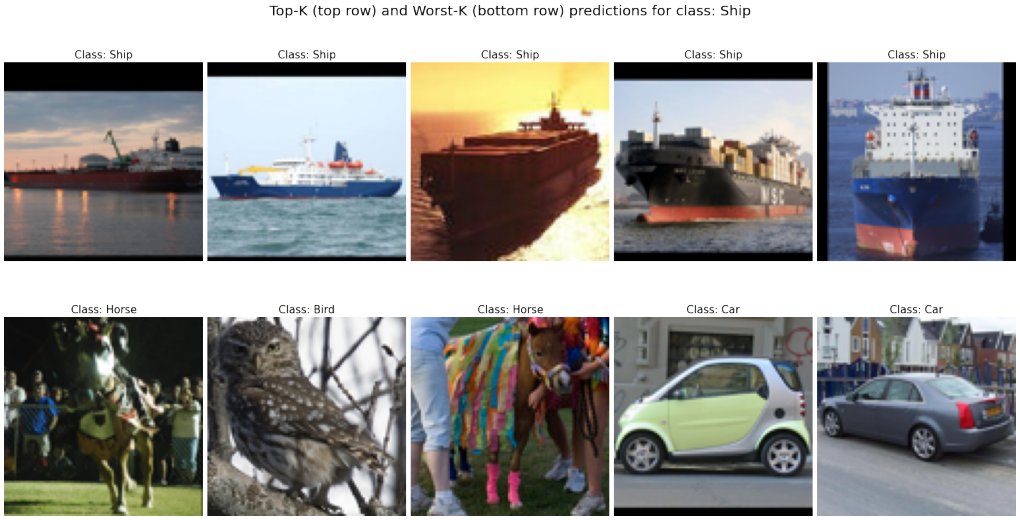
(a) Airplanes



(b) Birds

(c) Cars

(d) Horses

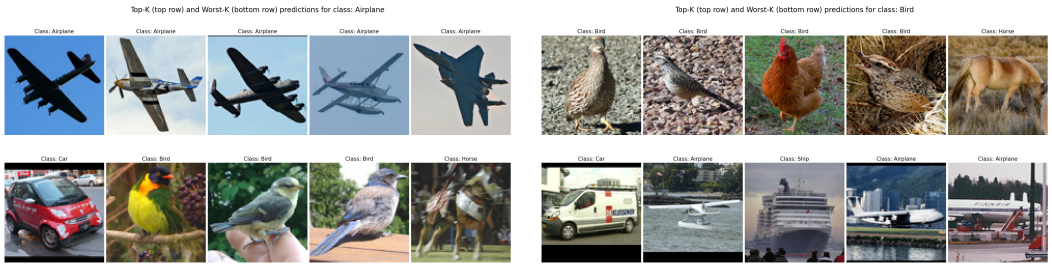


(e) Ships

Figure 5: Visualizing top-5 and worst-5 examples based on predicted score for $K = 500$. For each class, the top row shows top-5 retrieved images; the bottom row shows the worst-5.

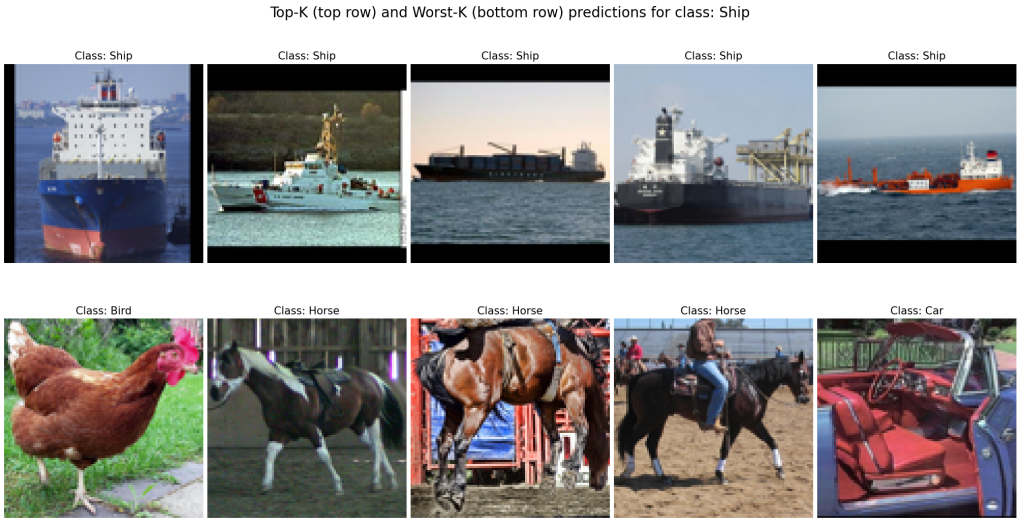
C / Kernel	poly	rbf	linear
0.1	0.396	0.583	0.492
1.0	0.410	**0.584**	0.494
5.0	0.449	0.567	0.499

Table 3: Impact of SVM hyper-parameters.



(a) Airplanes (b) Birds

(c) Cars (d) Horses



(e) Ships

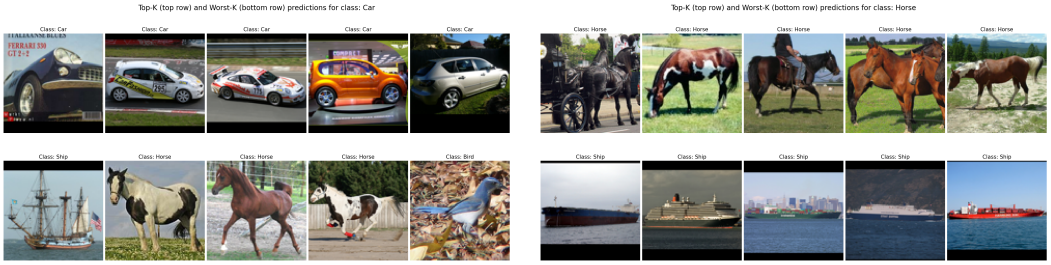
Figure 6: Visualizing top-5 and worst-5 examples based on predicted score for $K = 1000$.

References

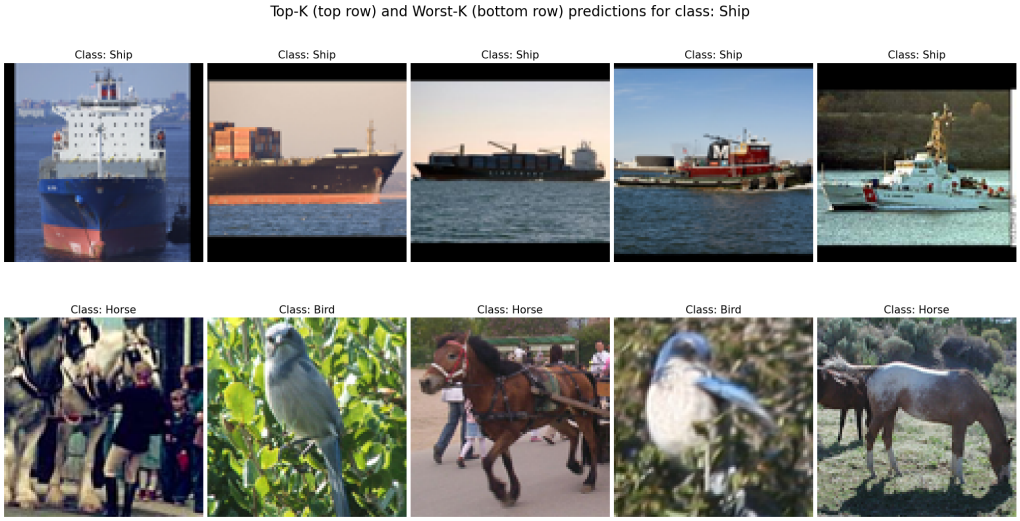
[1] Adam Coates, Andrew Ng, and Honglak Lee. "An Analysis of Single-Layer Networks in Unsupervised Feature Learning". In: *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*. Ed. by Geoffrey Gordon, David Dunson, and Miroslav Dudík. Vol. 15. Proceedings of Machine Learning Research. Fort Lauderdale, FL, USA: PMLR, 2011, pp. 215–223. URL: https://proceedings.mlr.press/v15/coates11a.html.

A More results for Retrieval

(a) Airplanes (b) Birds

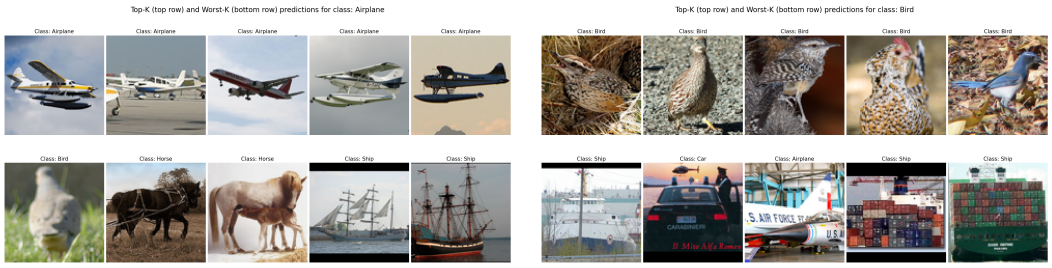


(c) Cars (d) Horses



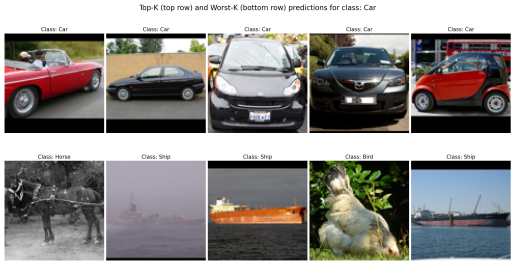
(e) Ships

Figure 7: Visualizing top-5 and worst-5 examples based on predicted score for $K = 2000$.

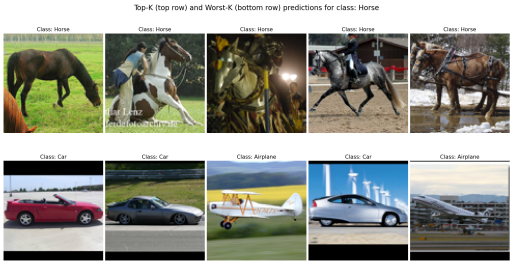


(a) Airplanes

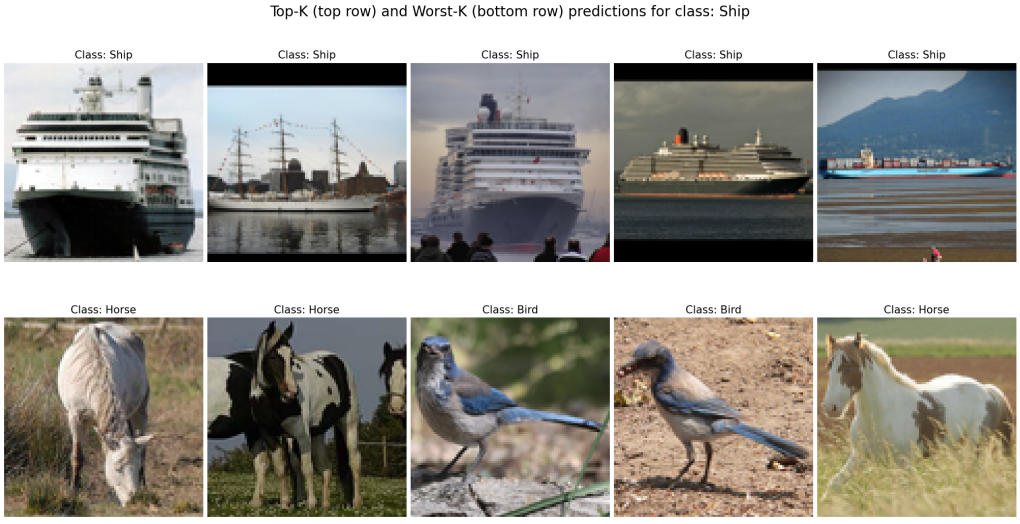
(b) Birds



(c) Cars



(d) Horses



(e) Ships

Figure 8: Visualizing top-5 and worst-5 examples based on predicted score for $K = 500$ and HoG features.