Computer Vision - Final Project: Part One

Ioannis Gatopoulos 12141666, Philipp Ollendorff 11734078,
Konstantin Todorov 12402559, Vincent Roest 10904816

August 31, 2019

Introduction

This report is an amalgamation of the findings and methods of the previous assignments in order to build a fully-fledged image classifier. The classifier will be based on the Visual Bag-of-Words (BoW) pipeline. After cleaning and preparing the Stanford Image Dataset, this pipeline can be roughly divided into five steps, all of which will be outlined in the first section. Alongside their theoretical details, their intermediary results will be discussed. These five building blocks eventually constitute a rather effective image classifier that distinguishes between five labeled classes: airplanes, birds, ships, horses and cars. The validation of the classifier will also be discussed after a quick examination of the BoW pipeline, as well as the impact of hyperparameter choices on evaluation metrics. The experiment itself will also answer validation questions based on hyperparameters choices. For instance, it will examine the relation between the number of clusters - the size of the visual vocabulary - and classification accuracy. Furthermore, two sampling strategies in the first step in the pipeline sampling will be contrasted - dense and key point sampling. Finally, the impact of different SIFT descriptors on the mean average precision, the main evaluation metric used in this experiment, will be examined.

1 Visual Bag of Words Pipeline

This section provides a short overview of the 5 steps this image classifier takes in order to make predictions. We will go into more detail in the Experiments section. Firstly, feature extractors and descriptors will be generated from images using three variations of SIFT, which was discussed in Assignment 4. Secondly, a visual vocabulary - analogous to a textual word vocabulary - will be constructed from the set of training images through clustering, in our case with VLFeat's k-means function. The number of clusters, a hyperparameter in this classifier's implementation, will be varied to explore its best setting by comparing it with the evaluation metric. Subsequently, the images will be represented in terms of this visual dictionary. As such, the SIFT descriptors are calculated for each segment of the image and assigned to the closest center (a visual word) in our vocabulary. The fourth step then quantifies the distribution of words in an image by constructing a normalized Probability Density Function (PDF). Finally, in the classification step, each image in a particular class can be discriminated from another by training five binary SVM classifiers (one-vs-all) on these distributions. It is important to note that the construction of the visual vocabulary should strictly be done with training images to prevent polluting the evaluation metrics. The demo code also includes code to produce visualization of the visual words and we provide a representative sample of every class. We picked those which are closer to human understanding, in order to illustrate a clear correlation between them and the classes that they related the most.

2 Implementation

We assume that the training of visual vocabulary should include all training images. We therefore include 500 images per class in this phase. Since the given data set includes ten classes, but we only classify five of them we exclude all images of unused classes. This reduces the training data set to 2500 images. The relevant classes are: 1 - airplane, 2 - bird, 3 - car, 7 - horse, 9 - ship (Figure 1). All images are colored and of equal size: 96×96 pixels.

For the testing phase, we include all classes. Inspired by real-world applications, where user does not use an application properly (`user error`; user tries to classify objects that the algorithm was not trained on), we

included also the images where none of the objects which the classifier was trained on appear(i.e. the remaining 5 classes which were not used for training). This leaves us with the full 8000 images as testing data. Note that half of the classes have not been used as negative examples during training. However, we assume that leaving them when testing should not alter the end result - the top results should still remain the ones that are correctly classified.

(a) Airplane (b) Bird (c) Car (d) Horse (e) Ship

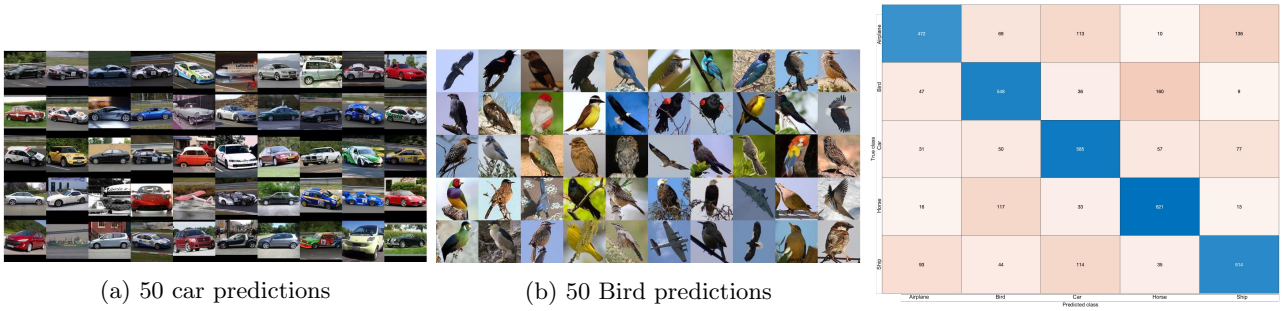
Figure 1: Example images of each class

3 Experiments

This section revolves around the questions presented in the introduction. There are at least three hyper parameters that can be varied:

- Vocabulary Size K, with values in [400, 1000, 4000]
- Sampling Strategy, which is either SIFT descriptors based on densely sampled regions, or key points.
- SIFT Descriptor Type, which is either "RGB", "grayscale" or "opponent".

With the initial settings, meaning a vocabulary size of 1000, a dense sampling strategy and a RGB SIFT descriptor, decent mean average precision was achieved. Namely, 73% for airplanes, 71% for birds, 75% for cars, 78% for horses and 72% for ships. It is important to note that this was done with image filtering to boost our results, thereby cherry picking from the test images only these 5 classes. In contrast, the experiments performed below do not violate this assumption for the reason mentioned in section 2. Consequently, Table 1 shows drastically lower mean average precisions (mAP), with a highscore of 46.8% precision. Figure 2 shows the class-wise top 50 predictions based on confidence of the classifiers, as well as a confusion matrix that highlights troublesome classes for the classifier. Moving on now to consider the various hyperparameter settings and their effect on performance, this section will compare different settings, while keeping the others fixed. Finally, the section will include 18 figures comprising the 5 most and 5 least confident images, per class and per hyperparameter configuration. This means there will be `10x5x18` small images, which are grouped together in small detail, due to space constraints in Figure 5. The best mAP was achieved using the "default" settings of 1000 clusters, dense sampling strategy and RGB descriptors.



(a) 50 car predictions (b) 50 Bird predictions (c) Confusion Matrix

Figure 2: Top 50 most confident predictions (from left to right) from the classifier on the filtered test set using the standard settings and a general confusion matrix

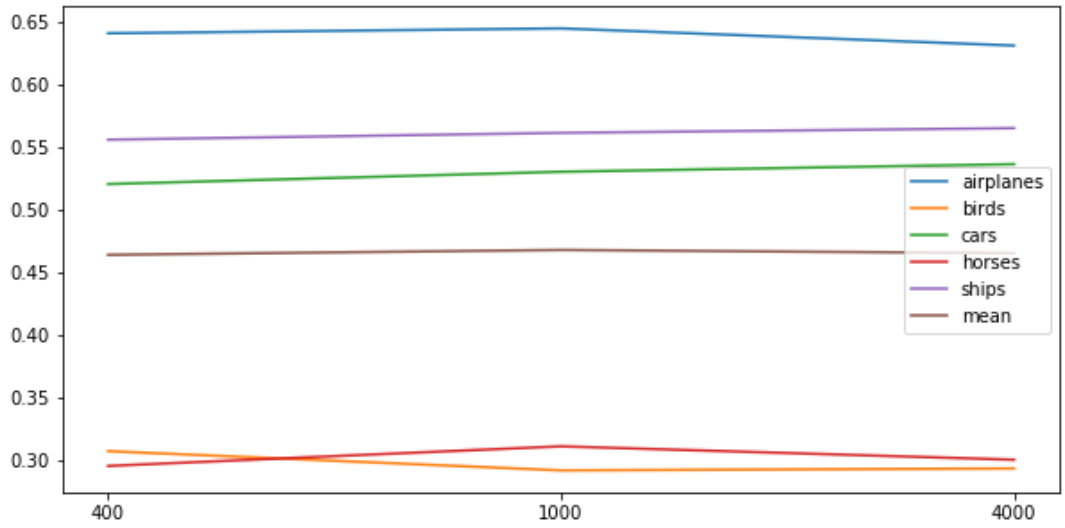


Figure 3: Impact of vocabulary sizes (k-means clusters) on class-wise accuracy

3.1 Vocabulary Size

The amount of clusters is pre-defined for K-Means Clustering and therefore the vocabulary size is set as well. We experiment with three values: 400, 1000, 4000. Figure 3 shows that the impact of vocabulary size on the actual mean precision is rather limited for this configuration. It also shows that some classes are definitely more difficult to predict than others. It appears that airplanes are rather easily detectable in this setting, which stands in contrast with the confusion matrix of the filtered image set depicted in Figure 2c, where they appeared to be one of the most difficult classes. Table 1 also shows that in terms of vocabulary size, there seems to be an according slight increase in performance, but not significantly so. Table 1 also hints that vocabulary of 4000, if it is compared to that of size 1000, looks more like an overfit than an improvement. Especially using the best configuration, that of dense sampling, RGB descriptor, it appears that differences are marginal.

3.2 Sampling Strategy

The experiments contrast SIFT descriptors based on densely sampled regions and key points. Keypoint sampling implies that SIFT identifies "key points" of interest using some filtering method, such as difference of Gaussians, from which the histograms - in our implementation probability density functions - are generated. Densely sampling region simply implies that the image is divided in to cells (a grid), after which the histograms are computed. The former focus more on salient parts of the image while the latter have the ability to draw an even number of sampled patches per image, on any place in the image. Table 1 shows a huge discrepancy in performance between the two methods. It appears that Keypoint sampling does not provide enough descriptive features from the (small) images, whereas the small images seem to lend themselves better to considering all aspects of the image. Keypoint sampling might, for example, not identify the blue sky as a key feature as it contains no edges or interesting points, whereas dense sampling will definitely regard this visual word as indicative of an airplane (or ship). Figure 5 show that blue skies are very often not indicative of cars and horses, for example, as they appear on the right side of the images in those classes. Edges of airplanes, cars and ships might also not be stereotypical for a particular class as they rather look alike in the small images. Finally, dense sampling maintains spatial relations between features [Tuytelaars, 2010]. On the other hand, key point sampling is much more robust when dealing with invariances in terms of view points or illumination (interesting key points). However, this dataset comprises mostly of standardized images with very similar shapes and scales, rendering these merits as less significant. For example, the airplanes in Figure 5a demonstrate a striking resemblance in terms of illumation, scale and color, and likewise do the birds. Spatial dense sampling might consequently yield better performances. On Figure 4 is illustrated a sample of key images patches, produced under the default setting (number of clusters: 1000, densely RGB sample). These patches lie close to the the center of clusters, or visual words, and are responsible for the shape of the histogram of every image, hence the classification of an image. We have to note that these were picked because they are more representatives to

humans and help the reader indicate some key feature-areas per class. For example, for the cars - a detection of a wheel plays a major role in classifying it as one.

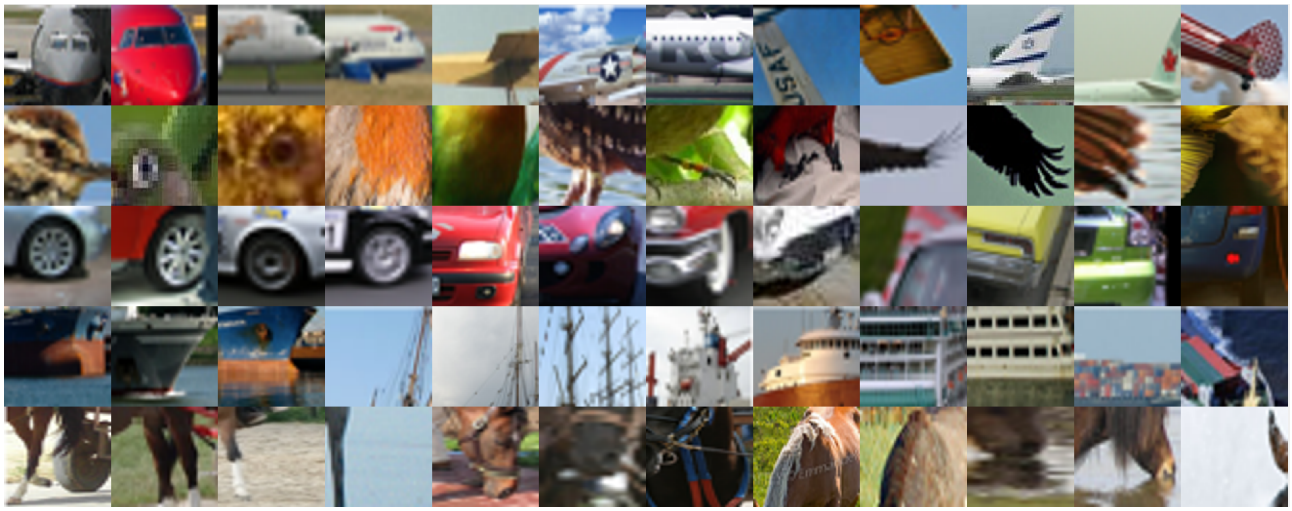


Figure 4: Examples of visual words per class which have meaningful interpretation for humans. Every row represents a different class. From the top to the bottom we illustrate airplane, bird, car, ship and horse classes.

3.3 SIFT Descriptors

Finally, three different SIFT descriptor extraction algorithms: grayscale SIFT, RGB-SIFT and opponent-SIFT are compared. Across the board, it is evident from Table 1 that RGB has an edge over the two other methods. These findings contrast the results from Van der Sande et al.'s seminal paper [Van De Sande et al., 2010], who recommend oppponent-SIFT when no prior information is available. It is also apparent that opponent-SIFT and grayscale SIFT, according to this experiment, do not significantly differ in performance.

4 Results

In our experiments we found a positive relationship between cluster size and mean average precision values for all classes. This means that cluster size 4000 outperformed all other cluster sizes, but we note this might be due to overfit. In terms of SIFT descriptors we found higher MAP values for RGB-SIFT and the worst values for grayscale-SIFT. In general descriptors extracted from densely sampled regions are better than those extracted from key points.

5 Conclusion

In this report we implemented five binary classifiers based on a bag of words approach using various SIFT features. The qualitative results do show that the classifiers discriminate very well between several classes, and even indicate that some classes are absolutely nothing like others. For example, it seems that birds and ships

	Dense			**Keypoints**		
	Gray	**OOP**	**RGB**	**Gray**	**OOP**	**RGB**
400	0.422 (a)	0.414 (b)	0.464 (c)	0.356 (d)	0.344 (e)	0.379 (f)
1000	0.437 (g)	0.425 (h)	**0.468** (i)	0.359 (j)	0.350 (k)	0.382 (l)
4000	0.430 (m)	0.429 (n)	0.465 (o)	0.314 (p)	0.316 (q)	0.356 (r)

Table 1: Mean Average Precisions for the different configurations K (400, 1000, 4000), SIFT descriptors (Gray, OOP and RGB) and sampling strategy (Dense and Keypoint). The values between parentheses indicate the corresponding classifier depicted in Figure 5

(a) K=400, F=Dense, T=GRAY (b) K=400,F=Dense, T=OOP (c) K=400, F=Dense, T=RGB

(d) K=400, F=Key, T=GRAY (e) K=400,F=Key, T=OOP (f) K=400, F=Key, T=RGB

(g) K=1000, F=Dense, T=GRAY (h) K=1000,F=Dense, T=OOP (i) **K=1000, F=Dense, T=RGB**

(j) K=1000, F=Key, T=GRAY (k) K=1000,F=Key, T=OOP (l) K=1000, F=Key, T=RGB

(m) K=4000, F=Dense, T=GRAY (n) K=4000,F=Dense, T=OOP (o) K=4000, F=Dense, T=RGB

(p) K=4000, F=Key, T=GRAY (q) K=4000,F=Key, T=OOP (r) K=4000, F=Key, T=RGB

Figure 5: Qualitative Evaluation of the classifiers. The rows represent - from top to bottom - the classes: airplanes, birds, cars, horses and ships. The left 5 columns show the most confident predictions, the right 5 columns the least confident. The captions below the figures show that particular configuration with K being the vocabulary size, F the sampling strategy and T the SIFT descriptor type. Ideally, the left half of the image would be filled completely with the correct predictions, and the right with pictures looking nothing like that class. These predictors' average precision correspond to those in Table 1, and are also arranged according to the ordering of this Table. Training and test set settings were exactly the same across all these experiments to ensure consistency in the results.

have no visual features in common, regularly being shown as opposite classes as seen in the second rows in most figures of Figure 5. On the other hand, some difficulty is visible with images that have a lot of sky and therefore share many visual features such as planes and boats.

The dataset itself is usually used for (semi-)supervised learning, because it lacks sufficient training and test examples for modern ConvNets to be trained on. Therefore, it is hard to compare the results of this experiment to modern-day baselines. However, the qualitative evaluation shows that these classifiers are definitely capable of classifying pictures correctly. Moreover computational costs are marginal compared to deep ConvNets.

Our experiments were performed using the experiment function included in the code. The associated .zip file will also include every other function that can be used to reproduce the results, including visualization.

References

[Tuytelaars, 2010] Tuytelaars, T. (2010). Dense interest points. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2281–2288. IEEE.

[Van De Sande et al., 2010] Van De Sande, K., Gevers, T., and Snoek, C. (2010). Evaluating color descriptors for object and scene recognition. *IEEE transactions on pattern analysis and machine intelligence*, 32(9):1582–1596.