Computer Vision 1 - Final Lab Report

Bag-of-Words based Image Classification and Convolutional Neural Networks for Image Classification

Michelle Appel (10170359)

Nils Hulzebosch (10749411)

GitHub repository: `https://github.com/MichelleAppel/CV1_final`

1 Introduction

This paper discusses the results of different methods of image classification, the task of identifying whether or not a specific object is present within the image. Two methods are presented: a Bag-of-Words (BoW) based method and a Convolutional Neural Networks (CNNs) based method.

Furthermore, four object types are used to train (using 500 images per class) and test (using 50 images per class) both models: airplanes, motorbikes, faces, and cars.

The report is divided into three sections: 1) BoW methods and results, 2) CNN methods and results, and 3) conclusion and discussion.

2 Bag-of-Words based image classification

The Bag-of-Words approach uses a 'bag of features' to represent an image, which is in turn used to classify an object. The approach consists of feature extraction, building a visual vocabulary which is used to create the bag of features, representing images as feature frequencies, and training a Support Vector Machine (SVM) to classify objects.

2.1 Feature extraction

For feature extraction, points of interest and their descriptors are found using SIFT. Four color spaces are used to create descriptors to discover the impact of color space on classification performance. These are grayscale, RGB, normalized RGB, and opponent color space.

Furthermore, two methods of SIFT are used to extract descriptors: SIFT and dense SIFT. For SIFT, descriptors are extracted from points of interest (corners, edges, blobs), resulting in roughly 300 descriptors per image. For dense SIFT, descriptors are extracted using dense sampling, with a step size of 15 pixels, and block sizes of 4, 6, 8, 10 pixels, resulting in roughly 2000 descriptors per image. The combination of these methods yields eight different sets of features, that are all separately used to create visual vocabularies.

The creation of a visual vocabulary is done in the following way. First, all feature descriptors for a subset of training images are concatenated to a single matrix. Then k-means is used to find appropriate cluster centroids, which denotes a visual word. The set of cluster centroids is the extracted visual vocabulary, which is used to represent a bag of features for an image during training and testing the SVM. Four different vocabulary sizes (amount of clusters for kmeans) have been used to create features: 50, 200, 400, and 1000. Using four color spaces, two sift methods, four vocabulary sizes, and four classes, 128 SVM models are trained and evaluated. The combined results (per four classes) of 32 models are shown in the next subsection.

Then, the visual vocabulary is used to represent an image by making a histogram of how often each visual feature appears in the image, which is the resulting bag of features. Using multiple labeled

training images from all four classes, four binary classifiers are trained using SVM with a linear kernel. Eventually, using the test data, the performance of the four binary classifiers is measured using Mean Average Precision (MAP).

The following table shows the division of vocabulary data, training data, and test data of all classes.

	Airplanes	Cars	Faces	Motorbikes	Total
Vocabulary	125	115	100	125	465
Training	375	350	300	375	1400
Testing	50	50	50	50	200
Total	550	515	450	550	2065

Table 1: Division of vocabulary data, training data, and test data of all four classes.

2.2 Quantitative analysis

This subsection describes the results of all models, measured by Mean Average Precision, which is the mean of the Average Precision over the four classes. Tables 2, 3, 4, and 5 show the results for vocabulary sizes 50, 200, 400, and 1000 respectively. Furthermore, each table differentiates between sift method and color space, and displays the corresponding Mean Average Precision, which is calculated on the test set.

	gray	RGB	(norm) rgb	opponent
SIFT	0.59	0.64	0.69	0.64
DSIFT	0.37	0.87	0.79	0.88

Table 2: Mean Average Precision of several models using vocabulary size = 50.

	gray	RGB	(norm) rgb	opponent
SIFT	0.68	0.72	0.61	0.61
DSIFT	0.37	0.90	0.65	0.82

Table 3: Mean Average Precision of several models using vocabulary size = 200.

	gray	RGB	(norm) rgb	opponent
SIFT	0.62	0.73	0.53	0.56
DSIFT	0.36	0.91	0.53	0.70

Table 4: Mean Average Precision of several models using vocabulary size = 400.

	gray	RGB	(norm) rgb	opponent
SIFT	0.66	0.76	0.48	0.54
DSIFT	0.33	0.89	0.42	0.52

Table 5: Mean Average Precision of several models using vocabulary size = 1000.

As shown in the tables, dense sift outperforms regular sift in general. This could be due the large amount of features extracted by dense sift, which is rougly 10 times more than the features extracted by regular sift.

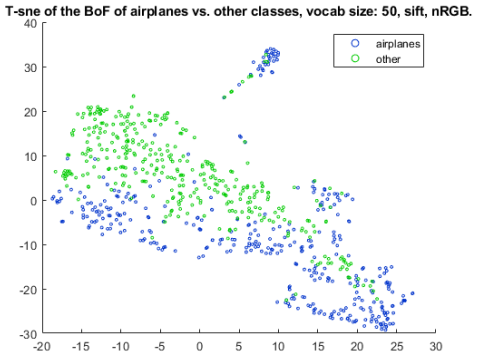
Secondly, the larger the vocabulary size, the lower the performance. This might be due to the fact that similar features are not assigned to the same cluster due to the higher amount of clusters. In the next subsection, some features will be visualized, which could be helpful to explain some of these aspects.

Moreover, the color space used for extracting features also has a large impact on the performance. As expected, features from coloured images (non-grayscale) outperform grayscale in most cases. This is probably due to the extra information that these features encode. Overall, RGB outperforms normalized RGB and opponent.

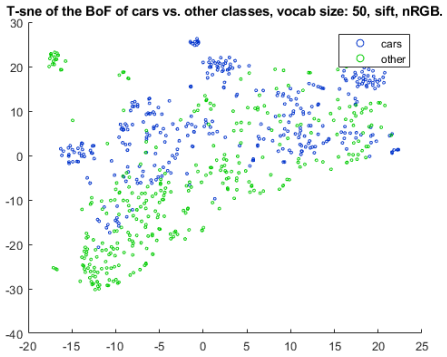
Lastly, dense sift using grayscale features performs much worse than the other dense sift and normal sift models: in each of the four tables, it has the lowest score. It might be due to the fact that the features are less informative (they only have 1 information channel). However, when the MAP are compared, for all vocabularies it is the case that the MAPs of airplanes, cars, and faces are very low (around 0.3) and the MAP of motorbikes is reasonable (around 0.5). In the next subsection, some image orders will be visualized for a better understanding.

2.3 Qualitative analysis

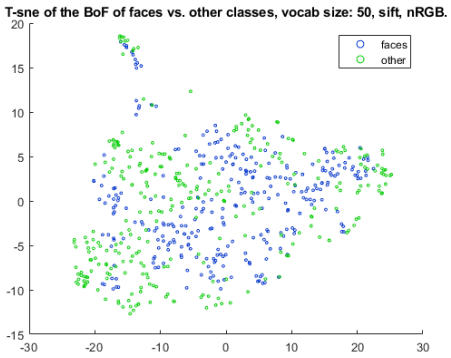
In this subsection, some features will be visualized. Furthermore, a few examples of top classified images for some models will be given.



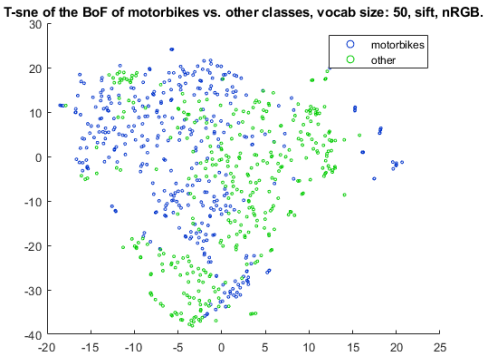
(a) Airplanes



(b) Cars

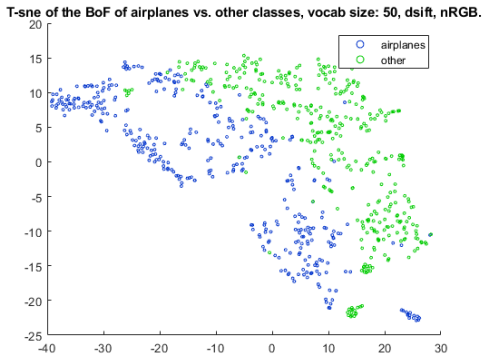


(c) Faces

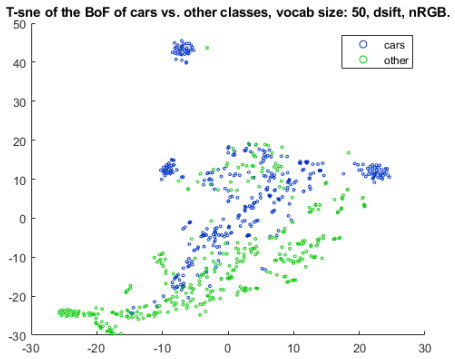


(d) Motorbikes

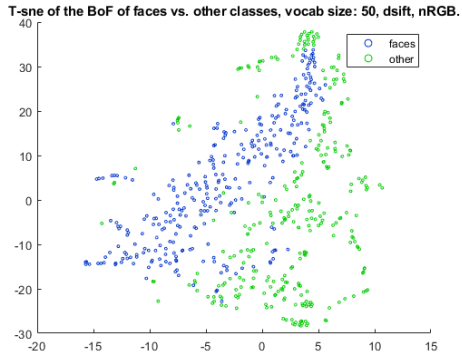
Figure 1: T-sne of the features with vocabulary size = 50, normalized RGB colorspace and sift. Blue denotes the class, while green denotes the other classes.



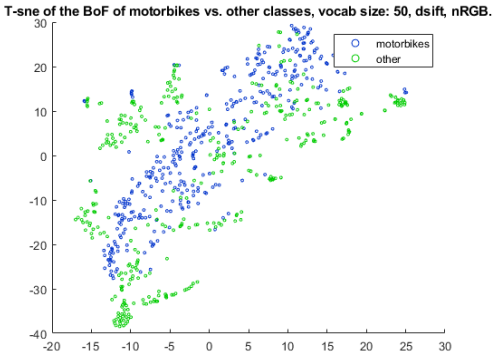
(a) Airplanes



(b) Cars



(c) Faces



(d) Motorbikes

Figure 2: T-sne of the features with vocabulary size = 50, normalized RGB colorspace and dense sift. Blue denotes the class, while green denotes the other classes.

Besides the T-sne features, several visualizations of the top five predictions are shown in the following (figures 3, 4, 5, 6). Four models are chosen: the best performing sift model, the worst performing sift model, the best performing dense sift model, and the worst performing dense sift model, based on the Mean Average Precisions in the previous subsection.

Figure 3: Visualization of first five predictions for each of the classes for the best sift method (using vocabulary size 1000 and RGB descriptors).

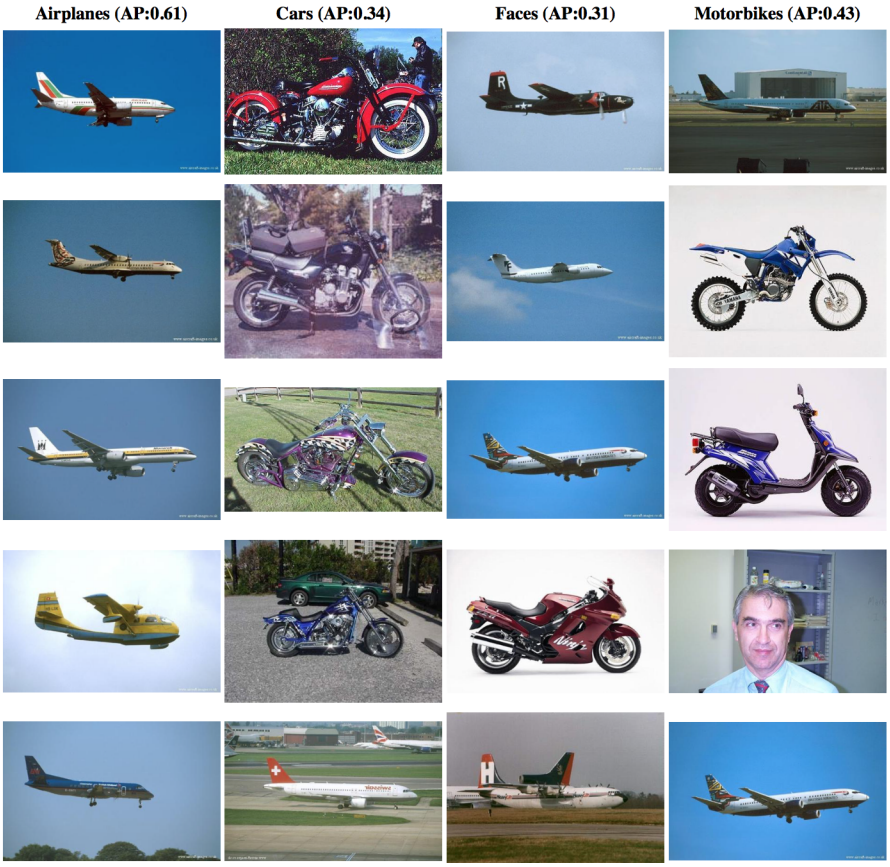


Figure 4: Visualization of first five predictions for each of the classes for the worst sift method (using vocabulary size 1000 and normalized RGB descriptors).

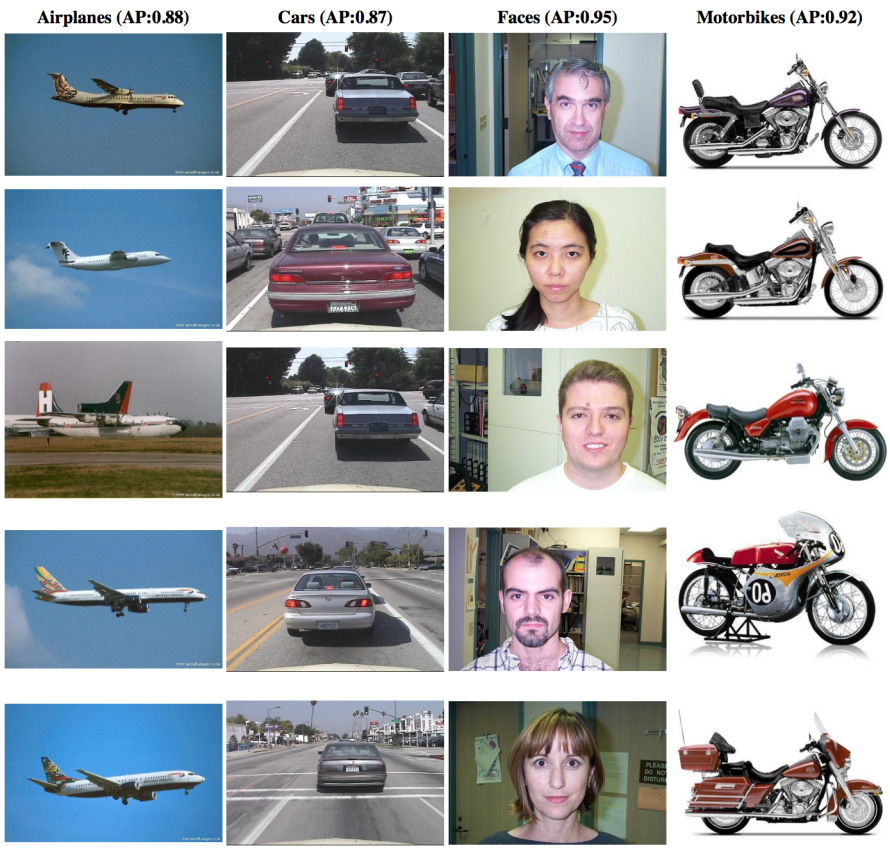


Figure 5: Visualization of first five predictions for each of the classes for the best dense sift method (using vocabulary size 400 and RGB descriptors).

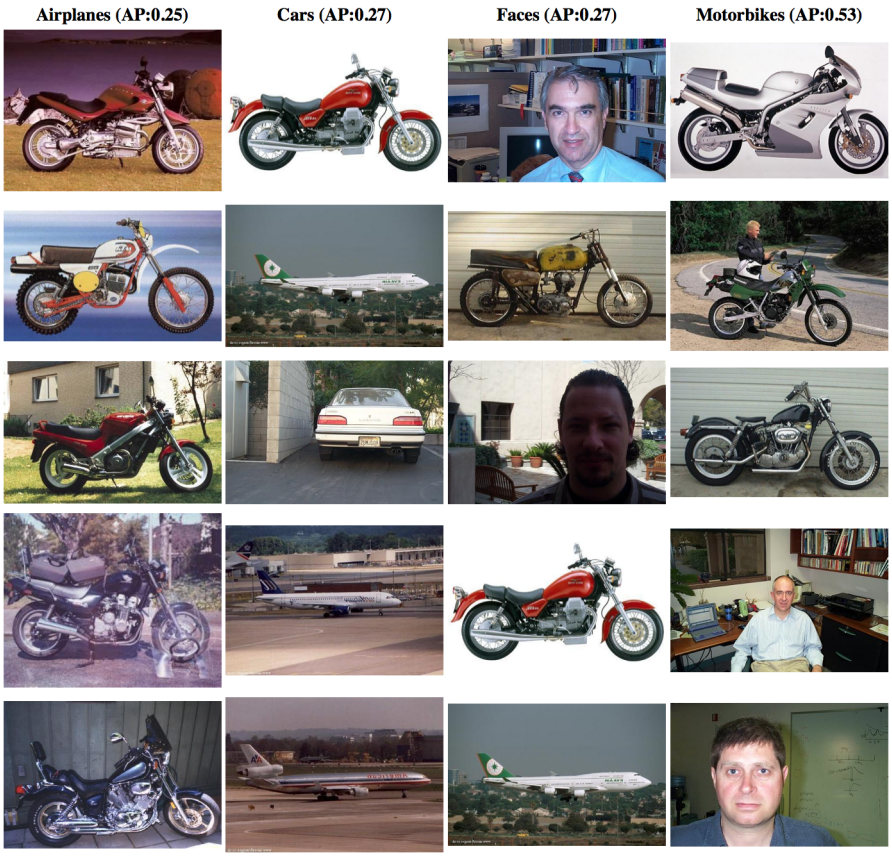


Figure 6: Visualization of first five predictions for each of the classes for the worst dense sift method (using vocabulary size 1000 and grayscale descriptors).

3 Convolutional Neural Networks for image classification

In the second part, the performance of a Convolutional Neural Network is tested. The main difference is that the CNN learns features and classification jointly, instead of sequentially with the first approach.

This section is divided into 4 subsections: 1) network architecture, 2) data preprocessing, 3) 4) training network and hyperparameter tuning, and 4) evaluation.

3.1 Network architecture

A schematic overview of the network is given in figure 7. As shown, its structure is build out of a repetition of a convolution layer, a pooling layer and a relu layer. Finally, the output is obtained by a softmax layer.

Convolutional layer 10 in figure 7 has the most parameters (4x4x64x64) and the biggest size (256KB).

layer	0	1	2	3	4	5	6	7	8	9	10	11	12	13
type	input	conv	mpool	relu	conv	relu	apool	conv	relu	apool	conv	relu	conv	softmxl
name	n/a	layer1	layer2	layer3	layer4	layer5	layer6	layer7	layer8	layer9	layer10	layer11	layer12	layer13
support	n/a	5	3	1	5	1	3	5	1	3	4	1	1	1
filt dim	n/a	3	n/a	n/a	32	n/a	n/a	32	n/a	n/a	64	n/a	64	n/a
filt dilat	n/a	1	n/a	n/a	1	n/a	n/a	1	n/a	n/a	1	n/a	1	n/a
num filts	n/a	32	n/a	n/a	32	n/a	n/a	64	n/a	n/a	64	n/a	10	n/a
stride	n/a	1	2	1	1	1	2	1	1	2	1	1	1	1
pad	n/a	2	0x1x0x1	0	2	0	0x1x0x1	2	0	0x1x0x1	0	0	0	0
rf size	n/a	5	7	7	15	15	19	35	35	43	67	67	67	67
rf offset	n/a	1	2	2	2	2	4	4	4	8	20	20	20	20
rf stride	n/a	1	2	2	2	2	4	4	4	8	8	8	8	8
data size	32	32	16	16	16	16	8	8	8	4	1	1	1	1
data depth	3	32	32	32	32	32	32	64	64	64	64	64	10	1
data num	1	1	1	1	1	1	1	1	1	1	1	1	1	1
data mem	12KB	128KB	32KB	32KB	32KB	32KB	8KB	16KB	16KB	4KB	256B	256B	40B	4B
param mem	n/a	10KB	0B	0B	100KB	0B	0B	200KB	0B	0B	256KB	0B	3KB	0B

parameter memory|569KB (1.5e+05 parameters)|
data memory| 313KB (for batch size 1)|

Figure 7: Schematic overview of the CNN.

3.2 Data preprocessing

To make the dataset of images usable for the network, it is preprocessed into a specific format. In this format, all data is stored into a *struct*, including the images, labels, sets, and meta information.

3.3 Training network and hyperparameter tuning

Due the the small size of the image dataset, it is not feasible to train a complete network. Therefore, a pretrained network is used, and the last layer is relearned via transfer learning. With this adaption, the new input size and output size of the last layer are 64 and 4 respectively, namely the output of the previous layer (64) and the amount of classes (4).

Using test runs, good hyperparameters can be chosen before training the final model. In this part, several values have been experimented with for batch size (50 and 100) and number of epochs (40, 80, 120). However, due to some errors in our code, the resulting models were all equal. Figure 8 shows the errors during training using epochs = 120 and batch size = 50.

3.4 Evaluation

Unfortunately, due to errors in the network, it was not possible to get the accuracy of the SVM using fine-tuned features and fine-tuned CNN. The accuracy of the SVM using pretrained features is 94.5%.[1]

[1] Sorry for this anti-climax, but due to many hours of debugging part 1, we did not have much time left for part 2.

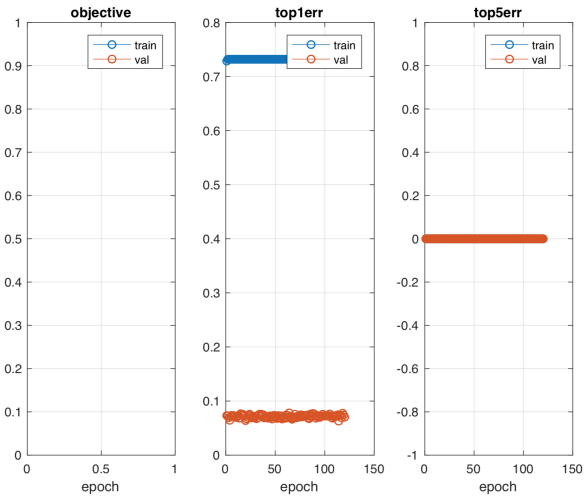


Figure 8: Errors during training using epochs = 120 and batch size = 50.

4 Conclusion and Discussion

In this report, two methods were tested for image classification, a Bag of Words model and a Convolutional Neural Network. This conclusion will only focus on the first, since the second did not yield proper results due to code errors.

Overall, SIFT can be used as a good method for extracting descriptors, which can be turned into features using kmeans. For all vocabulary sizes, dense SIFT outperforms normal SIFT, which could be due to the increase in descriptors, which might lead to better features. Descriptors from coloured images results in better features than those from grayscale image, where RGB descriptors results in the best performance (almost with every vocabulary size the highest Mean Average Precision). Furthermore, MAP seems to have good correspondence with the visualizations of predicted images.

It seems that a rotated image generates a different BoF than the same original image, as a result of SIFT generating a different set of descriptors for it. This may be due to MATLABs rotation method, which might create distortion in the rotated image. However, it is expected that SIFT generated the same set of descriptors of the same image, independent of its rotation and scaling. Obviously, it is desired to obtain the same BoF for the same image, independent of rotation and scaling, as the object must be recognized under all angles and scales.

When observing the t-sne representations, it is visible that the dense features are better separable than the sift features. However, the t-sne representations of the features imply that the data might be separated more effectively using a non-linear kernel, as the points that represent the features of one class are located between the other classes. This could be interesting for future research.