UNIVERSITEIT VAN AMSTERDAM

COMPUTER VISION

ASSIGNMENT 5

IMAGE CLASSIFICATION

Ahmet Taskale
Student Number:12344087
ahmet.taskale@student.uva.nl

Andreea Teodora Patra
Student Number:13365169
andreea.patra@student.uva.nl

Jim Wagemans
Student Number: 11912286
jimwagemans@gmail.com

November 22, 2020

1 Introduction

In this assignment we will implement and compare two different methods for image classification. The first method is the bag of words approach, where we use a support vector machine for classification. The second approach is a convolutional neural network. Both approaches will be used to classify images in the STL10 database

2 Image classification with Bag of Words

In this part, image classification using bag of words will be performed on the images from STL10 dataset, but we will only keep 5 classes from this, such as 1: airplanes, 2: birds, 3: ships, 4: horses, 5: cars. The training set will include 500 images per class and the testing set will include 800 images per class.

2.1 Feature extraction and Descriptors

To be able to implement bag of words, we need to extract from each image their descriptors. For this task we will implement initially the SIFT algorithm. Later on, other types of descriptors will be used for comparison. We are interested in keypoints that are scale and rotation invariant so that we can predict with higher accuracy the class of an image based on its features. SIFT algorithm follows 4 steps to be able to detect the keypoints: scale-space extrema detection, keypoint localisation, orientation assignment and keypoint descriptor. To make the detection scale-invariant, the images are computed are different scales. Therefore, in the scale-space extrema detection, SIFT used Difference of Gaussians to find the local extrema over scale and space. In this way, the same corners can be detected even if the images are scaled.

Moreover, in the following images in Figure 1, we have plotted the keypoints on images from different classes to visualise which features are kept for each class.

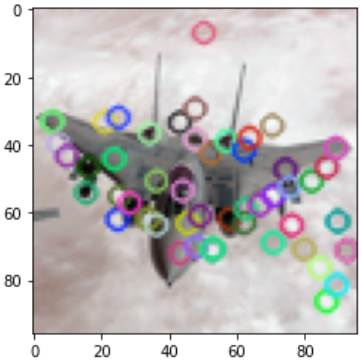
As shown in Figure 1, not only the objects representing the classes are recognized as keypoints, but the background noise as well. This could possibly lead to misclassification later on.

2.2 Building visual vocabulary and the histograms

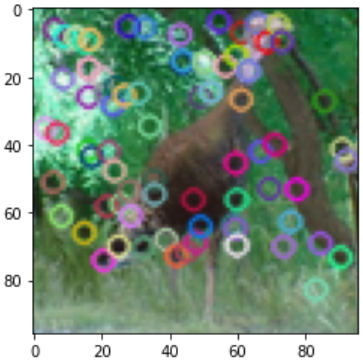
After the descriptors are extracted from each images, the next step is clustering. We are taking at random half of images from each class and we feed their descriptors to a k-means algorithm. Firstly, k-means randomly assigns the centroids to each cluster. Using the Euclidean distance, we assign the other points to the closest centroid. Then, we update the centroids by calculating the mean of the values from the cluster. The model will stop when it will converge, meaning that the centroids will stop changing. The main property that needs to be satisfied is that all points from a cluster should be similar to each other. We assess this by calculating the within sum of squared or inertia $\sum_{j=1}^{k} \sum || x_i - \mu_j ||_2^2$, where the first sum calculates the total distance from the centroid within one cluster. Therefore, our goal is the minimize the inertia which is exactly what the k-means does through the iterations. Our main goal is to reduce the feature space and we can do that by putting the descriptors in different groups and then keeping the cluster center. The cluster center is one descriptor that can be used as point of reference for all the descriptors in the group. By having this cluster center, we can construct a visual vocabulary and histograms for each image. After the clustering on one part of the images, the visual vocabulary is constructed on the rest by predicting the cluster they belong to based on k-means. Having these predictions, we can create a feature matrix/histogram, by counting for each image how many of their descriptors belong to each cluster respectively. Therefore, the final feature matrix will be of size $1250 \times cluster\_size$. Different cluster sizes (400, 1000, 4000) will be tested for comparison.

Figure 2 shows the histograms that have been created with a cluster size of 400 and SIFT descriptors.

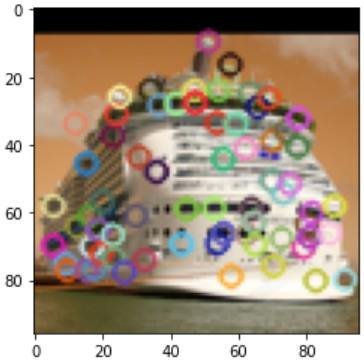
First of all, we can see that in all classes, predominantly there are values of 1 for each cluster center. Moreover, images from classes 1,3,4 and 5 are represented by more cluster centers. Therefore, the images that contain birds do not have as many descriptors in common with the rest. We would expect to find more corners in an image with a ship or car than in an image with a bird. Also, ships and cars have similar clusters with values of ones. We can also see that there are significant differences between images from each class. However, classes 1, 4 and 5 distinguish themselves by having more density in distribution compared to the other classes. An explanation for this might be that the images representing classes 1, 4 and 5 have more feature-rich areas.



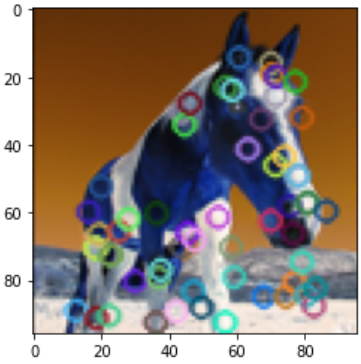
(a) keypoints airplane



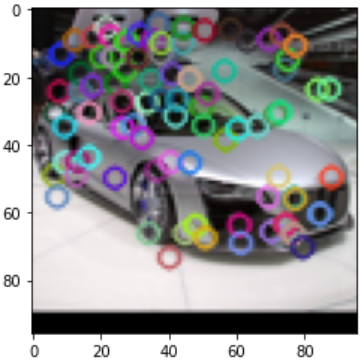
(b) keypoints bird



(c) keypoints ship

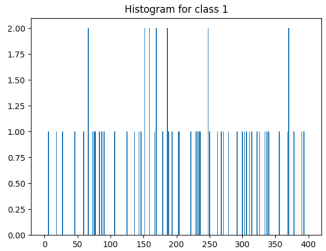


(d) keypoints horse

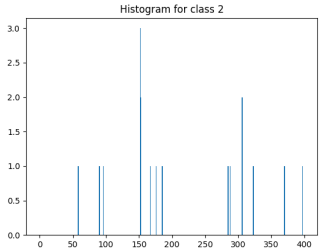


(e) keypoints car

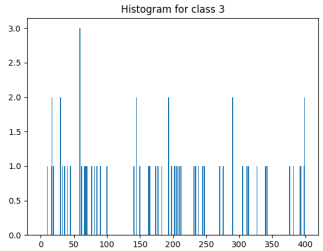
Figure 1. Keypoints drawn with circles for five images representing each class



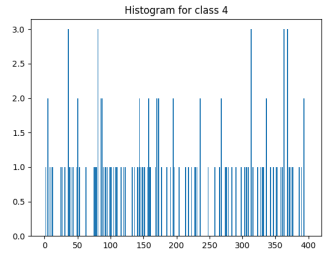
(a) Histogram for class 1 airplanes



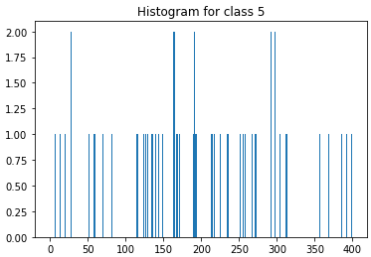
(b) Histogram for class 2 birds



(c) Histogram for class 3 ships



(d) Histogram for class 4 horses



(e) Histogram for class 5 cars

Figure 2. Histograms representing a single image from each class

The histograms of size 1000 for each class are shown in the appendix. Again, images representing classes 1, 3, 4 and 5 contain more cluster centers similar to the histograms of size 400. Remarkable is that only class 3 shows a high density in distribution, whereas the histograms of size 400 representing classes 1, 3, 4 and 5 had the same feature. Indeed, it is noteworthy to remind that this evaluation is based on the histograms of single images per class. As for histograms of size 4000

2.3 Classification

The next step is training the classifier. As discussed, we are using the other half of images that have not been used for KMeans training. For classification purposes, Support Vector Machine model will be used. The model tries to find the hyperplane that created the biggest margin between training points from 2 classes. Therefore, it is transformed into an optimisation problem $min \mid\mid \beta \mid\mid$ subject to $\xi_i \geq 0, y_i(x_i^T\beta + \beta_0) \geq 1$. However, in most cases, an overlap of the feature space is encountered. To deal with it, the algorithm allows for some points to be on the wrong side of the margin which can be written as $min\left(\frac{||\beta||^2}{2} + C\sum_{i=1}^{N}\xi_i\right)$ subject to $\xi_i \geq 0, y_i(x_i^T\beta + \beta_0) \geq 1 - \xi_i$ where ξ_i represents the proportional amount by which the prediction $f(x_i) = x_i^T\beta + \beta_0$ is on the wrong side of its margin and $\mid\mid \beta \mid\mid$ is the slope of the function f . Essentially, we can also think of C as $\sum_{i=1}^{N}\xi_i \leq C$, bounds the total number of misclassifications at C. A Gaussian RBF kernel will be used to be able to add similarity features. We will use one-vs-rest method, such as 4 binary classifiers will be trained for each class.

2.4 Evaluation

For evaluation, the average precision score and the accuracy will be calculated for each model. Both are these are implemented from the sklearn module.

2.4.1 SIFT with cluster size of 400

When investigating the descriptors created by SIFT, we noticed that different classes will have different numbers of keypoints found. To have a balanced number across classes, we have changed the following parameters per class for the SIFT algorithm:

- class 5 and 4 - contrastThreshold = 0.03
- class 1 - contrastThreshold = 0.02
- class 3 - contrastThreshold = 0.005 and edgeThreshold = 14
- class 2 - contrastThreshold = 0.06 and edgeThreshold = 8

The default value for the contrastThreshold is 0.04 and for edgeThreshold is 10. We are decreasing the value of contrastThreshold as we want more descriptors to be kept and we decrease the value of edgeThreshold when we want to discard more descriptors.

```
>>> print(class_Report)
              precision    recall  f1-score   support

         1.0       0.66      0.22      0.33       800
         2.0       0.66      0.25      0.36       800
         3.0       0.90      0.37      0.52       800
         4.0       0.71      0.21      0.32       800
         5.0       0.28      0.99      0.43       800

    accuracy                           0.41      4000
   macro avg       0.64      0.41      0.39      4000
weighted avg       0.64      0.41      0.39      4000
```

(a) Class report

	0	1	2	3	4
0	174	51	25	11	539
1	39	199	0	42	520
2	34	1	294	14	457
3	16	48	6	167	563
4	1	2	1	1	795

(b) Confusion matrix

Figure 3. The class report and the confusion matrix for the classification obtained with SIFT and cluster size 400

The accuracy score that resulted is 40.725 and the mAP score is 0.6427. We can see from the confusion matrix in Figure 3 that most of the misclassification have been performed as class 5. Most of the images have been misclassified

as being from class 5 and the model has also a recall of 0.99 for this class. On the other hand, the class with the highest precision is class 3. Therefore, not as many images are misclassified as being from this class. Even if we do not manage to recognize all the images with ships, we are mostly confident that we made a correct prediction. As expected, for the misclassifications for class 4, most of them come from class 2, birds as they belong to the same class, animals compared to airplanes and cars. We can also see that some airplanes are misclassified as birds as they tend to have similar aerodynamics. Class 4 is the only class that has as a misclassification amongst its top 5 and that belongs to an image from class 2. For bottom 5, we are mostly interested to see if there are patterns to describe why model was not confident in classifying an image that belongs to that class. Classes 5 and 1 have as bottom 5 images from other classes. The top 5 and bottom 5 images for all classes obtained with SIFT and a cluster size of 400 are shown in Figure 14, 15 and 16 in the appendix.

2.4.2 SIFT with cluster size 1000

To have a balanced number across classes and to increase the number of predictions per class, we have changed the following parameters per class for the SIFT algorithm:

- class 1 - contrastThreshold = 0.01
- class 5 - contrastThreshold = 0.03
- class 2 and 4 - contrastThreshold = 0.06 and edgeThreshold = 8
- class 3 - contrastThreshold = 0.005 and edgeThreshold = 14

	precision	recall	f1-score	support
1.0	0.37	0.63	0.47	800
2.0	0.29	0.91	0.44	800
3.0	1.00	0.02	0.03	800
4.0	1.00	0.00	0.01	800
5.0	0.37	0.06	0.10	800
accuracy			0.32	4000
macro avg	0.61	0.32	0.21	4000
weighted avg	0.61	0.32	0.21	4000

(a) Class report

	0	1	2	3	4
0	501	286	0	0	13
1	69	730	0	0	1
2	536	186	13	0	65
3	53	742	0	3	2
4	183	569	0	0	48

(b) Confusion matrix

Figure 4. The class report and the confusion matrix for the classification obtained with SIFT and cluster size 1000

Results showed that the accuracy and mAP corresponded to values of 32.375 and 0.5696. As shown in Figure 4, most of the images being misclassified belong to class 2 followed by class 1. The top 5 and bottom 5 images obtained from the SIFT with cluster size 1000 are shown in Figures 17, 18 and 19 in the appendix. Taking a closer look at the ranked images, class 1 consists only of correctly classified images. Class 2 contains one misclassification, whereas class 3 has no misclassification. One remarkable feature is that class 4 only contains three correct images. As for class 5, there seem to be three misclassifications belonging to class 3 ships.

2.4.3 SIFT with cluster size 4000

For this model, only class 1 seems to be recognized as shown below in Figure 5. Furthermore, values of 20.0 and 0.5423 were obtained for accuracy and mAP, respectively. In this specific case, it is safe to assume that SIFT with cluster size 4000 doesn't work optimal. The ranked images of class 1 is shown in Figure 20 in the appendix.

2.4.4 RGBSIFT with cluster size 400

RGBSIFT descriptors are constructed by detecting the descriptors on each channel and concatenating them. To have a balanced number across classes, we have changed the following parameters per class for the SIFT algorithm:

- class 1 - contrastThreshold = 0.04
- class 3 - contrastThreshold = 0.01
- class 4 and 5 - contrastThreshold = 0.005 and edgeThreshold = 8
- class 2 - contrastThreshold = 0.07 and edgeThreshold = 6

	precision	recall	f1-score	support
1.0	0.20	1.00	0.33	800
2.0	0.00	0.00	0.00	800
3.0	0.00	0.00	0.00	800
4.0	0.00	0.00	0.00	800
5.0	0.00	0.00	0.00	800
accuracy			0.20	4000
macro avg	0.04	0.20	0.07	4000
weighted avg	0.04	0.20	0.07	4000

(a) Class report

	0	1	2	3	4
0	800	0	0	0	0
1	800	0	0	0	0
2	800	0	0	0	0
3	800	0	0	0	0
4	800	0	0	0	0

(b) Confusion matrix

Figure 5. The class report and the confusion matrix for the classification obtained with SIFT and cluster size 4000

Results showed that the accuracy had a value of 48.325 and the mAP 0.6436. Taking a look at the class report and the confusion matrix in Figure 6 below, it becomes clear that most of the images that have been misclassified belong to class 4. Class 2 seems to have the highest precision, which is not surprising since images from class 2 have the lowest rate of misclassification, as can be seen in the confusion matrix. All of the top 5 ranked images from all classes were classified correctly, except for one image in class 4 which belongs to class 2 birds. The top 5 and bottom 5 images are shown in Figures 21, 22 and 23 in the appendix.

	precision	recall	f1-score	support
1.0	0.78	0.23	0.35	800
2.0	0.86	0.04	0.08	800
3.0	0.53	0.96	0.69	800
4.0	0.38	0.91	0.53	800
5.0	0.63	0.28	0.39	800
accuracy			0.48	4000
macro avg	0.64	0.48	0.41	4000
weighted avg	0.64	0.48	0.41	4000

(a) Class report

	0	1	2	3	4
0	183	3	313	264	37
1	44	32	21	617	86
2	1	0	767	30	2
3	1	1	62	728	8
4	6	1	273	297	223

(b) Confusion matrix

Figure 6. The class report and the confusion matrix for the classification obtained with RGB SIFT and cluster size 400

2.4.5 ORB with cluster size 400

ORB detector or Oriented FAST and Rotated BRIEF is built up on the FAST and BRIEF descriptor. The keypoints that are detected by FAST gives us information of the location of the edges in an image, but this ones are not scale invariant. Therefore, ORB uses multiscale image pyramid at different resolutions to solve this issue. After locating the keypoints, it assigns orientation depending on the level of intensity change around that keypoint. Intensity change is detected by using intensity centroid, assuming that a corner's intensity is offset from its center. BRIEF will take all the keypoints found by FAST and convert them into a binary feature vector so that they can represent an image. As BRIEF is not rotation invariant, ORB will use a modified version called Rotation-aware BRIEF.

However, results showed that the accuracy and mAP had values corresponding to 19.85 and 0.2694. These values reflect the poor classification of images, as can be seen in Figure in the appendix. Most of the images in the top 5 of each class consist mostly of airplanes. These images are shown in Figures 24, 25 and 26 in the appendix

	precision	recall	f1-score	support
1.0	1.00	0.00	0.01	800
2.0	0.21	0.84	0.34	800
3.0	0.62	0.05	0.09	800
4.0	0.12	0.08	0.10	800
5.0	0.07	0.02	0.03	800
accuracy			0.20	4000
macro avg	0.40	0.20	0.11	4000
weighted avg	0.40	0.20	0.11	4000

(a) Class report

	0	1	2	3	4
0	3	497	19	209	72
1	0	670	0	118	12
2	0	513	40	109	138
3	0	734	0	64	2
4	0	730	6	47	17

(b) Confusion matrix

Figure 7. The class report and the confusion matrix for the classification obtained with ORB and cluster size 400

Taking a look at all the models computed, it is concluded that the RGB SIFT model with cluster size 400 provided the optimal results with values of 48.325 and 0.6436 corresponding to the accuracy and mAP, respectively. Furthermore, all of the top 5 ranked images from all classes were classified correctly, except for one image in class 4 which belongs to class 2 birds.

3 Image classification with CNNs

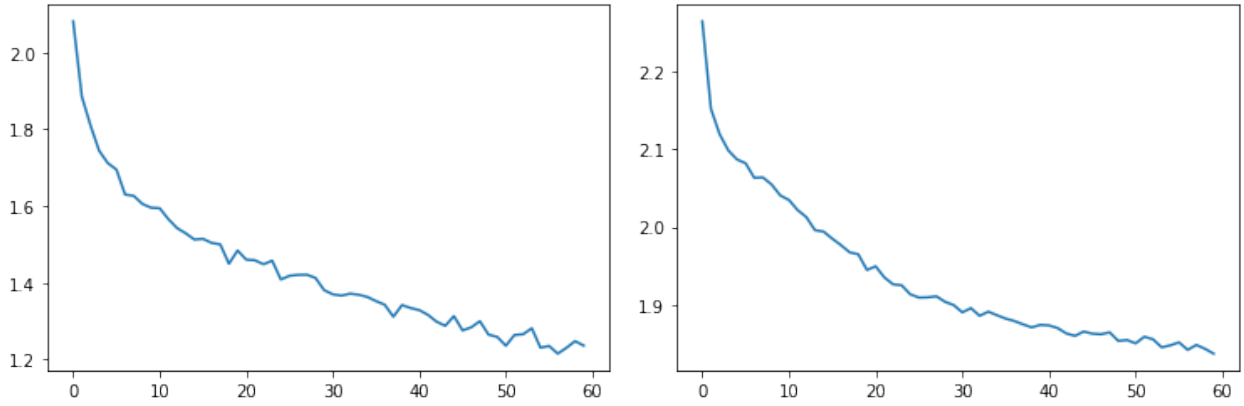
3.1 Introduction

In this section we will attempt to make use of convolutional neural networks to classify images. We will be working with 2 datasets. First we will train two different networks on the CIFAR10 dataset and later finetune one of these networks on the STL10 dataset. In figure 8 we see examples of the images in CIFAR 10, one image from the plane and car class and two from the boat class. CIFAR10 contains a total of 10 classes. The images are RGB and 32×32

Figure 8. Cifar 10 images

3.2 TwoLayerNet and LeNet5

We implemented two different types of networks. The first one is a standard two layer fully connected neural network. We know from the universal approximation theorem that two layers is sufficient to approximate any function (given a large enough hidden layer). The other network is a LeNet5 network [1], this network consists of a convolutional part, followed by a fully connected part. We trained both of these for 10 epochs (and computed the loss 6 times per epoch). The result can be seen in figure 9.



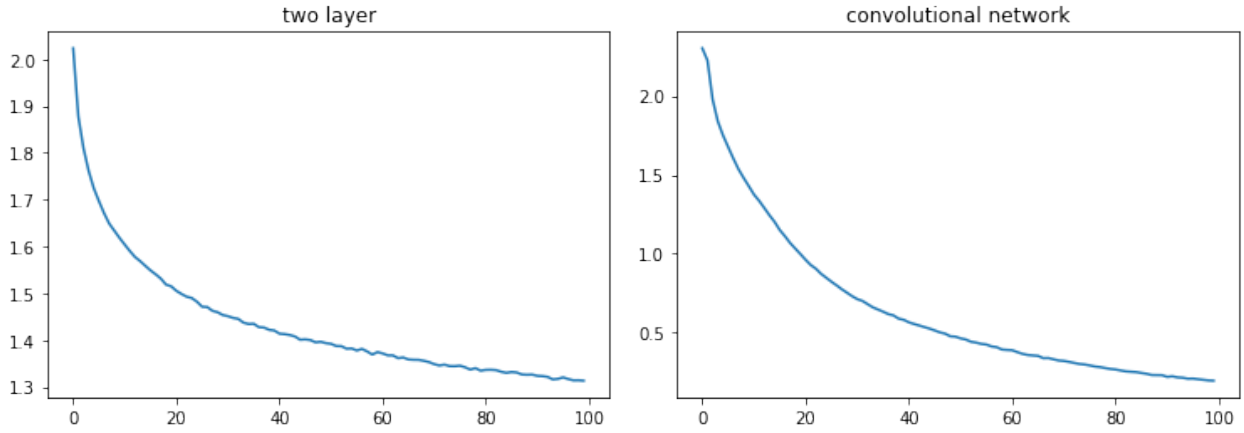
(a) Two layer net loss

(b) LeNet5 loss

Figure 9. Two layered and LeNet5 training losses

The two layer net has a learning rate of 0.001 and the LeNet5 of 0.01. The two layer net performs better here. Do note that our dataset is biased against convolutional neural networks, because our images are very small. The number of parameters increases way faster for fully connected NN, then convolutional ones as the size of the image increases.

[1] https://towardsdatascience.com/understanding-and-implementing-lenet-5-cnn-architecture-deep-learning-a2d531ebc342



(a) Two layer net loss (b) convnet3 loss

Figure 10. Two layered and LeNet5 training losses

class	ConvNet3	TwoLayerNet
average	84	53
plane	83	66
car	74	80
bird	69	35
cat	82	43
deer	77	34
dog	91	40
frog	85	62
horse	95	61
ship	95	69
truck	93	57

Table 1. Accuracy convnet3 and two layer network (as %)

3.3 Training preparation and hyper parameters

We implemented the CIFAR10 loader as well as the optimizer and transformer. The optimizer is quite standard as it just applies basic gradient descent. The transform augments the data by cropping and flipping randomly. This increases the different examples that our network is trained on, preventing overfitting. It also normalizes the data, we found a mean and standard deviation of 0.5 works well. The convolutional neural network also needed a higher learning rate 0.01 than the two layer net 0.05. We also changed the architecture of the convolutional neural network. We switched to ReLu activation functions and added dropout layers. These dropout layers randomly set values to 0 to prevent overfitting. This network we call convnet3. We can see the losses in figure 10. We also trained these for longer, a total of 100 epochs and recording the loss once every epoch. The convolutional neural network seems to be performing better this time. This might be because of the changes in architecture or because we let it train for longer. We can see the accuracy in table 1. We see that the convnet3 outperforms in every case. Interestingly mechanical objects seem to be easier to identify. Perhaps these classes have less internal variation.

3.4 Fine tuning

In this section we discuss neural network fine tuning. We want to use the convnet3 to classify images in the STL10 dataset (from bag of words). We could use the same architecture (except for the output layer) and train a new network, but we can do something smarter. We can use the network that we have trained for the CIFAR10 dataset, and train it on the STL10 dataset. For this we do need to modify the output layer. We can see the training of this in figure 11. The network achieved an accuracy of 76%.

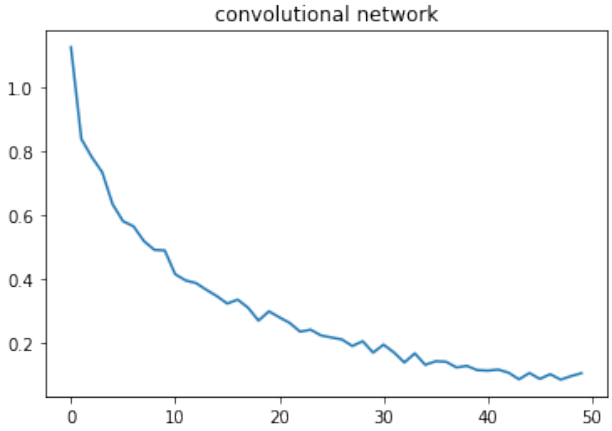
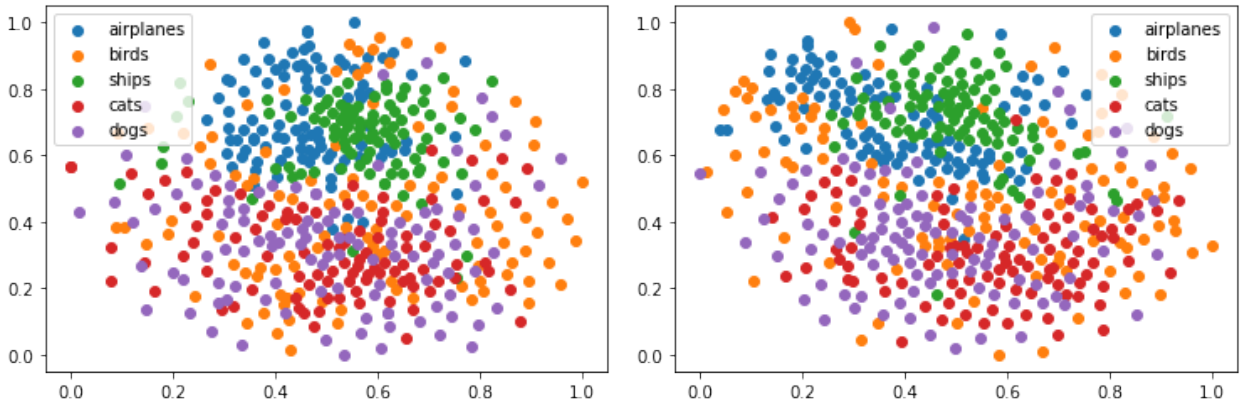


Figure 11. Fine tuning on convnet3



(a) Pre-tune

(b) Post-tune

Figure 12. feature visualization

Next need to visualize the difference between the pre-tuned network and the fine tuned network. For this we will look at the features that a network picks up on in an image. The network has a convolutional part that maps the input image to these features, followed by a fully connected part that classifies based on these features. The problem with visualizing the features is that they have high dimensionality. We will make use of a dimensionality reduction technique clalled t-SNE, we based our implementation on [2]. The idea is that we have N vectors $x_n \in \mathbb{R}^D$, and we wish to find N vector $\hat{x}_n \in \mathbb{R}^2$ such that, for some constant C, we have

$$\forall i, j \leq N \; \|x_i - x_j\| \approx C\|\hat{x}_i - \hat{x}_j\|$$

In normal words this means that we want $\hat{x}_i, \hat{x}_j$ to be close if and only if x_i, x_j are close. We plot the normalized feature vectors in figure 12. The colors correspond to image labels, we can see that there are several cluster, but they do overlap. It is then interesting to consider which clusters overlap. We see that the animals are clustered in the bottom and the vehicles are clustered in the top. These two sets of clusters overlap much internally than between the two. It is hard to observe a difference between the pre- and postune features. This would suggest that the tuning mainly changed the classification part of the network, rather than the feature extraction.

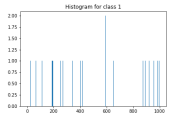
4 Conclusion

We have seen different approaches for image classification. We explored both the classification using support vector machines combined with bag of words and the use of convolutional neural networks. The results to indicate that the

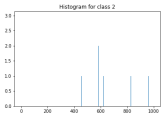
[2] https://www.learnopencv.com/t-sne-for-feature-visualization/

CNN performs better at image classification. The reason why this is is subject to speculation. One possible explanation is that we give fewer restriction we to the algorithms that we optimize over, when using neural networks. The CNN can learn which features to extract, whereas the other method has to use the bag of words.

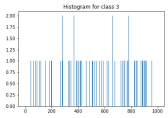
5 Appendix



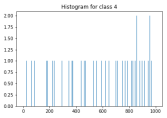
(a) Histogram for class 1 airplanes



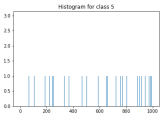
(b) Histogram for class 2 birds



(c) Histogram for class 3 ships



(d) Histogram for class 4 horses



(e) Histogram for class 5 cars

Figure 13. Histograms representing a single image from each class

(a) Top 5 images class 1

(b) Top 5 images class 1

(c) Top 5 images class 1

(d) Top 5 images class 1

(e) Top 5 images class 1

(f) bottom 5 images class 1

(g) bottom 5 images class 1

(h) bottom 5 images class 1

(i) bottom 5 images class 1

(j) bottom 5 images class 1

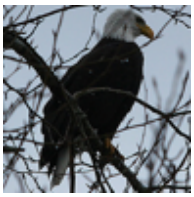
(k) Top 5 images class 2

(l) Top 5 images class 2

(m) Top 5 images class 2

(n) Top 5 images class 2

(o) Top 5 images class 2



(p) bottom 5 images class 2

(q) bottom 5 images class 2

(r) bottom 5 images class 2

(s) bottom 5 images class 2

(t) bottom 5 images class 2

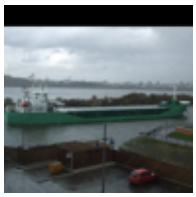
Figure 14. Top 5 and bottom 5 images for classes 1 and 2 using SIFT with a cluster size of 400

(a) Top 5 images class 3

(b) Top 5 images class 3

(c) Top 5 images class 3

(d) Top 5 images class 3



(e) Top 5 images class 3

(f) bottom 5 images class 3

(g) bottom 5 images class 3

(h) bottom 5 images class 3

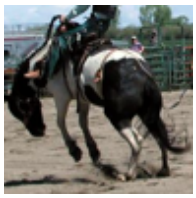
(i) bottom 5 images class 3

(j) bottom 5 images class 3

(k) Top 5 images class 4

(l) Top 5 images class 4

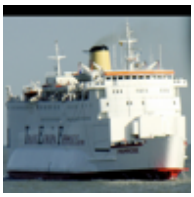
(m) Top 5 images class 4



(n) Top 5 images class 4

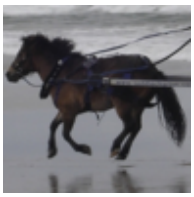
(o) Top 5 images class 4

(p) bottom 5 images class 4



(q) bottom 5 images class 4

(r) bottom 5 images class 4



(s) bottom 5 images class 4

(t) bottom 5 images class 4

Figure 15. Top 5 and bottom 5 images for classes 3 and 4 using SIFT with a cluster size of 400

(a) Top 5 images class 5

(b) Top 5 images class 5

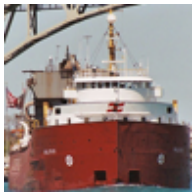
(c) Top 5 images class 5

(d) Top 5 images class 5

(e) Top 5 images class 5

(f) bottom 5 images class 5

(g) bottom 5 images class 5



(h) bottom 5 images class 5

(i) bottom 5 images class 1

(j) bottom 5 images class 5

Figure 16. Top 5 and bottom 5 images for class 5 using SIFT with a cluster size of 400

(a) Top 5 images class 1

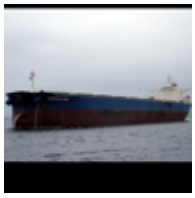
(b) Top 5 images class 1

(c) Top 5 images class 1

(d) Top 5 images class 1

(e) Top 5 images class 1

(f) bottom 5 images class 1



(g) bottom 5 images class 1

(h) bottom 5 images class 1

(i) bottom 5 images class 1

(j) bottom 5 images class 1

(k) Top 5 images class 2

(l) Top 5 images class 2

(m) Top 5 images class 2

(n) Top 5 images class 2

(o) Top 5 images class 2

(p) bottom 5 images class 2

(q) bottom 5 images class 2

(r) bottom 5 images class 2

(s) bottom 5 images class 2

(t) bottom 5 images class 2

Figure 17. Top 5 and bottom 5 images for classes 1 and 2 using SIFT with a cluster size of 1000

(a) Top 5 images class 3

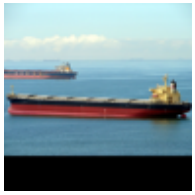
(b) Top 5 images class 3

(c) Top 5 images class 3

(d) Top 5 images class 3

(e) Top 5 images class 3

(f) bottom 5 images class 3



(g) bottom 5 images class 3

(h) bottom 5 images class 3

(i) bottom 5 images class 3

(j) bottom 5 images class 3

(k) Top 5 images class 4

(l) Top 5 images class 4

(m) Top 5 images class 4

(n) bottom 5 images class 4

(o) bottom 5 images class 4

(p) bottom 5 images class 4

Figure 18. Top 5 and bottom 5 images for classes 3 and 4 using SIFT with a cluster size of 1000

(a) Top 5 images class 5

(b) Top 5 images class 5

(c) Top 5 images class 5

(d) Top 5 images class 5

(e) Top 5 images class 5

(f) bottom 5 images class 3

(g) bottom 5 images class 5

(h) bottom 5 images class 5

(i) bottom 5 images class 5

(j) bottom 5 images class 5

Figure 19. Top 5 and bottom 5 images for class 5 using SIFT with a cluster size of 1000

(a) Top 5 images class 1

(b) Top 5 images class 1

(c) Top 5 images class 1

(d) Top 5 images class 1

(e) Top 5 images class 1

(f) bottom 5 images class 1

(g) bottom 5 images class 1

(h) bottom 5 images class 1

(i) bottom 5 images class 1

(j) bottom 5 images class 1

Figure 20. Top 5 and bottom 5 images for class 1 using SIFT with a cluster size of 4000

(a) Top 5 images class 1

(b) Top 5 images class 1

(c) Top 5 images class 1

(d) Top 5 images class 1

(e) Top 5 images class 1

(f) bottom 5 images class 1

(g) bottom 5 images class 1

(h) bottom 5 images class 1

(i) bottom 5 images class 1

(j) bottom 5 images class 1

(k) Top 5 images class 2

(l) Top 5 images class 2

(m) Top 5 images class 2

(n) Top 5 images class 2

(o) Top 5 images class 2

(p) bottom 5 images class 2

(q) bottom 5 images class 2

(r) bottom 5 images class 2

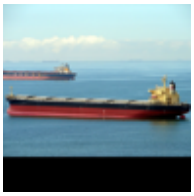
(s) bottom 5 images class 2

(t) bottom 5 images class 2

Figure 21. Top 5 and bottom 5 images for classes 1 and 2 using RGB SIFT with a cluster size of 400

(a) Top 5 images class 3

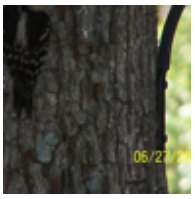
(b) Top 5 images class 3



(c) Top 5 images class 3

(d) Top 5 images class 3

(e) Top 5 images class 3



(f) bottom 5 images class 3

(g) bottom 5 images class 3

(h) bottom 5 images class 3

(i) bottom 5 images class 3

(j) bottom 5 images class 3

(k) Top 5 images class 4

(l) Top 5 images class 4

(m) Top 5 images class 4

(n) Top 5 images class 4

(o) Top 5 images class 4

(p) bottom 5 images class 2

(q) bottom 5 images class 4

(r) bottom 5 images class 4

(s) bottom 5 images class 4

(t) bottom 5 images class 4

Figure 22. Top 5 and bottom 5 images for classes 3 and 4 using RGB SIFT with a cluster size of 400

(a) Top 5 images class 5

(b) Top 5 images class 5

(c) Top 5 images class 5

(d) Top 5 images class 5

(e) Top 5 images class 5

(f) bottom 5 images class 5

(g) bottom 5 images class 5

(h) bottom 5 images class 5

(i) bottom 5 images class 5

(j) bottom 5 images class 5

Figure 23. Top 5 and bottom 5 images for class 5 using RGB SIFT with a cluster size of 400

(a) Top 5 images class 1

(b) Top 5 images class 1

(c) Top 5 images class 1

(d) Top 5 images class 1

(e) bottom 5 images class 1

(f) bottom 5 images class 1

(g) bottom 5 images class 1

(h) bottom 5 images class 1

(i) Top 5 images class 2

(j) Top 5 images class 2

(k) Top 5 images class 2

(l) Top 5 images class 2

(m) Top 5 images class 2

(n) bottom 5 images class 2

(o) bottom 5 images class 2

(p) bottom 5 images class 2

(q) bottom 5 images class 2

(r) bottom 5 images class 2

Figure 24. Top 5 and bottom 5 images for classes 1 and 2 using ORB with a cluster size of 400

(a) Top 5 images class 3

(b) Top 5 images class 3

(c) Top 5 images class 3

(d) Top 5 images class 3

(e) Top 5 images class 3

(f) bottom 5 images class 3

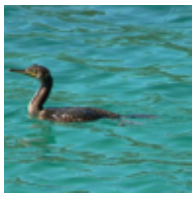
(g) bottom 5 images class 3

(h) bottom 5 images class 3

(i) bottom 5 images class 3

(j) bottom 5 images class 3

(k) Top 5 images class 4



(l) Top 5 images class 4

(m) Top 5 images class 4

(n) Top 5 images class 4

(o) Top 5 images class 4

(p) bottom 5 images class 2

(q) bottom 5 images class 4

(r) bottom 5 images class 4

(s) bottom 5 images class 4

(t) bottom 5 images class 4

Figure 25. Top 5 and bottom 5 images for classes 3 and 4 using ORB with a cluster size of 400

(a) Top 5 images class 5

(b) Top 5 images class 5

(c) Top 5 images class 5

(d) Top 5 images class 5

(e) Top 5 images class 5

(f) bottom 5 images class 5

(g) bottom 5 images class 5

(h) bottom 5 images class 5

(i) bottom 5 images class 5

(j) bottom 5 images class 5

Figure 26. Top 5 and bottom 5 images for class 5 using ORB with a cluster size of 400