RDFIA Practical work 1

Giulia Prosio and Alexander Hölzl

October 14, 2022

1 1-ab: Images Descriptors with SIFT and Bag of Words

1.1 Summary

The purpose of the practical work 1-ab was to produce an algorithm able to classify images using visual descriptors.
In order to do so we proceeded by steps, the first being the realization of a SIFT descriptor for each 16x16 pixel patch of each picture contained in the dataset 12-Scenes.
The first tool presented throughout the Practical Work was the Scale-Invariant Feature Transform (SIFT) - which is a local descriptor widely used in Image Classification as it is a feature extraction method that is invariant to scale and orientation of the images and robust to fluctuations of illumination, noise and minor viewpoint changes in the images.
Among the key-points in the computation of the SIFT descriptor the first one is to calculate the gradient of each pixel composing the patch selected.
After having then computed the weighted gradient using a Gaussian mask and discretizing the orientation of the gradient to 8 possible values (having 0 for the NORTH direction, 1 for NORTH-EAST, and so on), it is possible to build an histogram of the gradients for each region. The histograms can finally be concatenated to compose the SIFT descriptor of the patch.
The second step of the practical work was to compute a Visual Dictionary, so that it is possible to find frequent patterns and patches with similar SIFT descriptors can be grouped together. In order to solve this problem we used a K-Means algorithm, determining K centers of the SIFT space and minimizing the distance from the actual SIFTs.
The final step of this first practical work was to obtain a numerical representation of each image, so that it can be easily classified. To do so we used a Bag of Words (BoW) algorithm, with which we were able to summarize all the local SIFT descriptors of the image into a global descriptor, with the help of the visual dictionary.

1.2 Answers

1.2.1 Question 1

The kernels M_x and M_y can be separated in to two vectors h_y and h_x with:

$$h_x = \begin{pmatrix}1\\2\\1\end{pmatrix} \qquad\qquad h_y = \begin{pmatrix}-1\\0\\1\end{pmatrix}$$

With that we get:

$$M_x = \begin{pmatrix}1\\2\\1\end{pmatrix} \times \begin{pmatrix}-1 & 0 & 1\end{pmatrix} = \begin{pmatrix}-1 & 0 & 1\\-2 & 0 & 2\\-1 & 0 & 1\end{pmatrix}$$

and

$$M_x = \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix} \times \begin{pmatrix} 1 & 2 & 1 \end{pmatrix} = \begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{pmatrix}$$

1.2.2 Question 2

The advantage of separating the convolution kernel is simply that instead of having to do 9 multiplications to apply convolution, one can do only 6 multiplications. This reduces the computational effort and speeds up the calculation.

1.2.3 Question 3

When computing the SIFT descriptor for the patch of an image, the goal of weighting the Gradient Matrix with the Gaussian Mask is to produce a more accurate representation of the gradient of each pixel by taking into account the bordering pixels with a different relevance depending by how distant they are to the selected one. This way it is possible to obtain a result smoothed of noises as it takes into account also the pixel's surroundings and is stronger to little changes and variations. It is also relevant because while doing so it is also robust to elements that could bias the result such as the padding pixels around the image.

1.2.4 Question 4

The discretization of the directions plays an important role in the computation of the SIFT descriptor as it reduces the little variability and fluctuations that there would be if we considered the unique direction of each pixel. This approximation is necessary to build an histogram of frequency that is able to well describe the whole image.

1.2.5 Question 5

The interest of using different post-processing steps is given by the fact that we want to create a robust SIFT descriptor, that can resist to small changes. The first step is to bring to zero any SIFT with a norm value less than 0.2, as there is not enough variation in the gradient (and contrast) of the picture. Then we normalize the remaining SIFTs, this way we can compare patches between themselves. Finally we clump values less than 0.2 again, to have invariance to lighting and other disturbances and noises in the image. We then normalize the remaining values again to be able to compare the patches again. In this way we are able to eliminate irrelevant descriptors that do not carry enough variability from the study of the patches.

1.2.6 Question 6

SIFT is a good descriptor for the patch of an image when doing image analysis because the general purpose of a descriptor for an image is to somehow summarize it around detected keypoints.
The SIFT descriptor works on a 16x16 patch around the keypoints that uses image gradients pooled into orientation bins to capture the most interesting information in each patch for more reliable matching, while being robust to small misalignments, illumination changes and viewpoint changes.
this allows it to be a more abstract descriptor, more focused on the key points of the image than on the individual image itself.
In this way he can find similar elements and make matches even between images which present differences in illumination, scale or rotation.

1.2.7 Question 7

In this first part of the proposed practical work, we began to familiarize ourselves with the expressive power of the SIFT descriptor by working on some initial images not belonging to the Image Detection Dataset. The results we got for this section are visualized in figure (1).
Image (a) is the input image, while (b) plots the gradients of the image. As can be seen we calculate gradient norm G_n and discretized orientations G_o for the whole image. Image (d), where the gradient orientations have been discretized, show the importance of this step in the evaluation of the image's

gradients. Comparing it to image (c), where the gradients are not discretized, it can be observed how without a discretization each pixel has its own unique gradient that does not let us have a defined visualization of the situation.
The image is then split into 16x16 patches and we use G_n and G_o to compute a histogram of gradient directions for each 4x4 region of a patch.
Those histograms will then be used to construct the actual sift descriptors. The results of this process can be seen very well in figure (1e). The resulting sift descriptor has 8 peaks which correspond to the changes in gradients at the corners of the image.

1.2.8 Question 8

The visual dictionary is a useful tool needed for image recognition as it finds and displays patches of different images with similar SIFTs in order to create a robust and efficient list of similar patches that will help better classify and understand the different regions that compose each image

1.2.9 Question 9

Statement: ”The cluster’s center that minimizes the dispersion is its points’ barycenter”.
When we talk about a cluster we define a group of points, the only positions we can talk about are the points themselves.
There is no concept of average when simply talking about points: the way to do this is to define a ”clustroid,” a point ”nearest” to all the other points that form the cluster.
There may be several approaches to the meaning of ”nearest” point, and the one proposed, the smallest sum of the squares of the distance to the other points, is one of them.
Thus, when considering many points belonging to the same cluster, it is easier to think of a geometric shape that contains all the points.
By geometric definition, the barycenter of a shape is the point of the shape with the minimal the distance to its perimeter.
We can so apply this reasoning to the cluster algorithm.

1.2.10 Question 10

One of the most used algorithms to choose the optimal number of clusters is the Elbow Method. The method is based on calculating the Within-Cluster-Sum of Squared Errors (WSS) for different number of clusters (k) and selecting the k for which change in WSS first starts to diminish.

1.2.11 Question 11

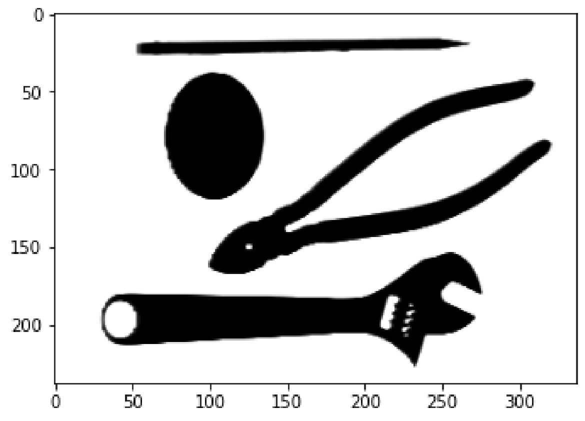
We create a visual dictionary on the SIFTs and not directly on the patches of raw image pixels because this way we are able to better see the similarities in patches without taking into account variation in illumination, noise or angle.
Considering the raw image and not its discretized gradients would introduce too much variability and the system would be too sensible to little changes.

1.2.12 Question 12

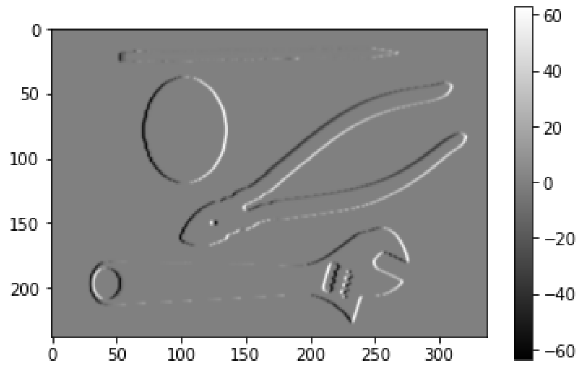
A typical example of a feature from our visual dictionary can be seen in figure (2). We can observe first hand how the features put together are similar, sharing similar gradients. Here the features seen to share a grid-like structure and might be coming from images depicting a part of a building structure. We can appreciate from these examples (add others), what we have studied about the SIFT descriptor - the patches presenting the grid-like structure are very different in terms of illumination, angle and scale, but nevertheless it is recognized their similarity.

1.2.13 Question 13

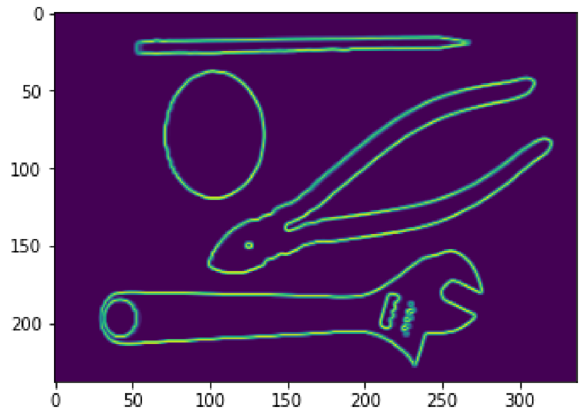
The vector z represents which descriptors from the visual dictionary are the ones with the higher correspondence to the descriptors of the image, putting in relation the image classificator built with the ”real” images



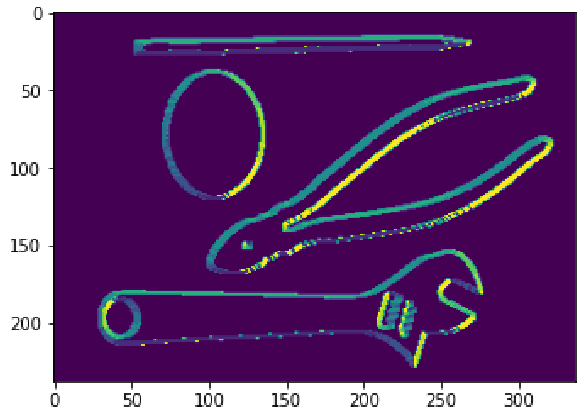
(a) The input image



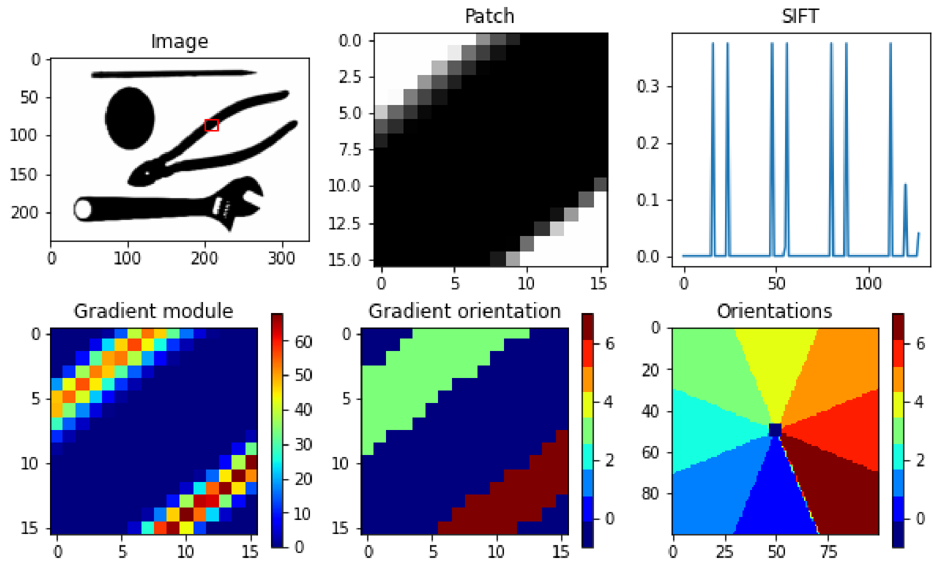
(b) The gradients



(c) Gradient norm



(d) Discretized gradient orientations



(e) Result of sift computation for a patch

Figure 1: Visualization of results from section 1

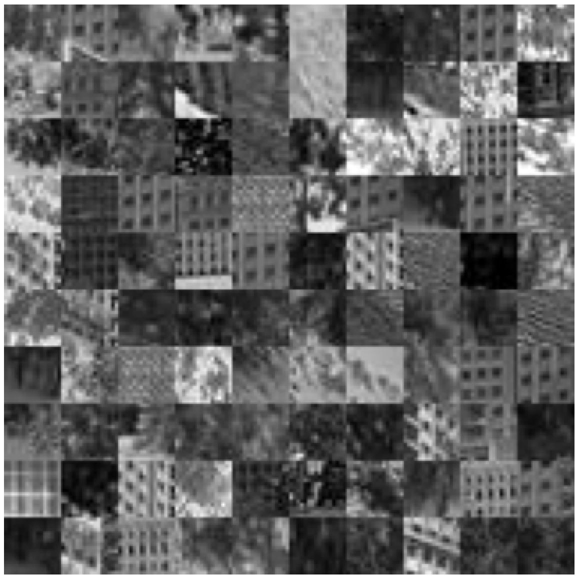
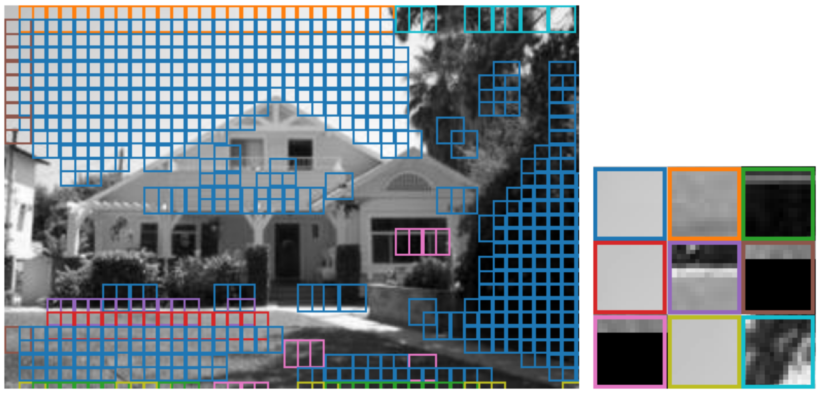


Figure 2: An example of a feature from the visual dictionary



(a) Matching of the features onto an image (b) Used features

Figure 3: Visual results of SIFT

1.2.14 Question 14

The visual results of our implementation can be seen in figure (3).
The picture does not present much contrast, and we can see how elements with a much different shade, but a similar gradient, have similar SIFT descriptors, such as the sky, the part of the house's facade without windows nor doors and the internal part of the bush on the right, very dark and so ”smooth”, like the former elements.

1.2.15 Question 15

The interest of Nearest Neighbour encoding is due to the fact that we want to find the closest determined cluster to each feature. Other methods could be Vectorial Quantization and Gaussian Mixture Models. The Vectorial Quantization is a lossy compression technique which allows us to represent each image by an histogram of visual words so that each feature must be discretized to a specific bin.
This way the different features of different images can be compared to one another.
This way we look at each visual word as a quantization function and we are able to translate the problem of image categorization to the viewpoint of statistical consistency.
Another possible approach is the Gaussian Mixture Model - to use this model we assume that there are a certain number of Gaussian distributions, and each of these distributions represent a cluster.

Hence, a Gaussian Mixture Model tends to group the data points belonging to a single distribution together and can be described as an iterative method to find maximum likelihood estimates of parameters, which works well with Pattern Recognition in Image Analysis.

1.2.16 Question 16

The interest into sum pooling is to create an image signature, its global index, and so the image likelihood to get each visual word.
Classic pooling strategies include, other than sum pooling, the max pooling strategies and average pooling.
Maximum pooling is a pooling operation that calculates the maximum value in each patch of each feature map. The results are down sampled or pooled feature maps that highlight the most present feature in the patch.
The average pooling strategy, on the other hand, calculates the average value of each patch in each feature map. This way the returned result takes into account all the features of the patch.

1.2.17 Question 17

The interest for the final normalization is to be able to better compare the final global indexes to one another.
The L2 norm calculates the distance of the vector coordinate from the origin of the vector space.
Another approach could be the Hamming Norm, which is the ratio of the Hamming distance of the patches to their dimensions.

2 1-c: SVM Classifier

2.1 Summary

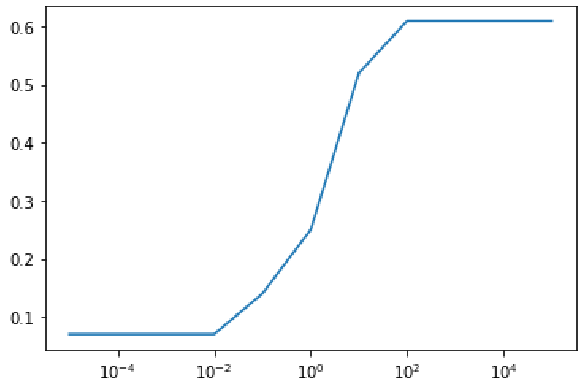
In section c we use the bag of words representation from the previous two sections to build an image classifier. This classifier is able to map the BoW representation to one of 15 classes to which the images belong.
In order to build this classifier we a so called Support Vector Machine (SVM).
A SVM is able to find a separating hyperplane which acts as a decision boundary to which is used to map a sample to its corresponding class. The closest distance from one of the training samples to the hyperplane is the so called support vector, and the goal of the SVM is to find a hyperplane which maximizes the distance — or the length of the support vectors — between the to classes.
As a SVM is only to distinguish between two classes, in order to be able achieve the desired the multi-class classification multiple SVMs need to be trained. Each of those SVMs compares one class against the rest. To actually classify a sample, each of the SVMs is used to find the class with the highest score.
For the programming exercises we used sklearns implementation of a SVM. We trained it on a given dataset and evaluated its performances using different hyperparameters.

2.2 Answers

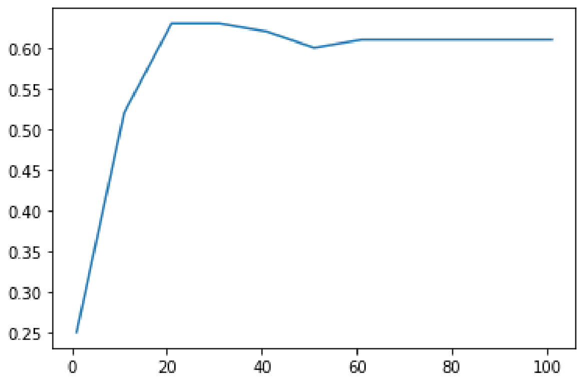
2.2.1 Question 1

Hyperparameter C
The influence of the hyperparameter C is visualized in figure 4. With a rising value of C the accuracy of the model rises until it is capped at a accuracy of about 0.6. It is intresting to note that we also observed a better performance with high values of C when evaluating on the test set. This was a bit confusing for us, as we would have expected that the good accuracy for high values of C would be a result of overfitting.

Due to good performance of higher values of C, we used C = 100 for all other hyperparamter evaluations.



(a) C on a log scale from 10^{-4} to 10^{4}



(b) C on a linear scale from 0 to 100

Figure 4: The influence of the Hyperparameter C on the Accuracy

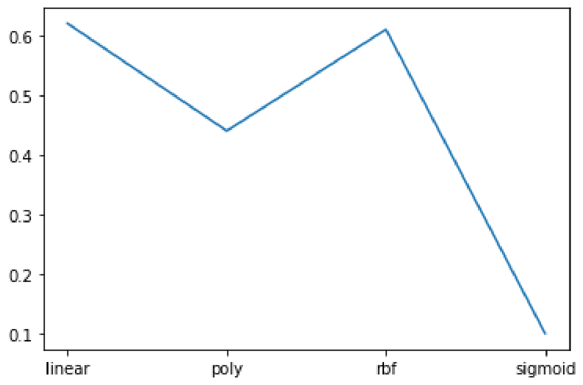


Figure 5: The influence of the Kernel on the Accuracy

Hyperparameter Kernel
The influence of the kernel can be seen in figure 5. As mentioned above we set C to 100 for this evaluation. All other parameters were set to sklearns default values. RBF, which is sklearns default, and the linear kernel are the best performing ones.

Hyperparameter Degree for Polynomial Kernels
The influence of the degree when using a polynomial kernel can be seen in figure 6. It is interesting to note the polynomial kernel performs worse with rising degree and performs best with degree set to one. This also matches the results from the above section where the linear kernel outperforms the polynomial kernel. We repeated this experiment also with C = 1 which resulted in a reversed result — we observed a rising accuracy with rising degree altough the maximum observed accuracy of 0.28 at a degree of 10 is still much lower than 0.6 obsereved with C = 100.

Hyperparameter Decision Function
Then influence of the used decision function can be seen in figure 7. As is obvious from the graph the choice of the decision function has no impact on the accuracy performance. This is due to the fact that sklearn defaults to one-vs-one internally.

2.2.2 Question 2

Hyperparameter C
The hyperparameter C tells the SVM how much you want to avoid misclassifying training data. If C is very large, the SVM will be trained in a way that prefer choosing a hyperplane with a very small distance to the samples, if this hyperplane classifies more training samples correctly.

On the other hand if C is very small, a hyperplane with a distance margin will be chosen, even if that

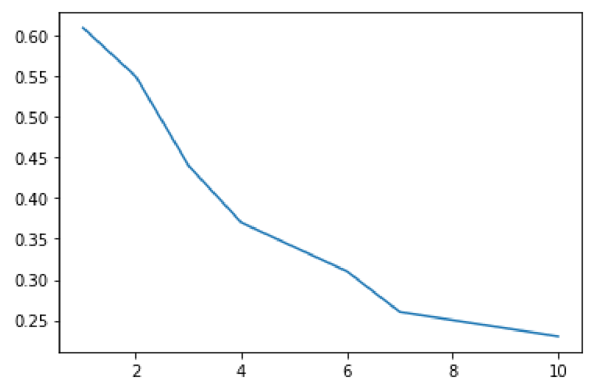


Figure 6: Influence of the Degree on the Accuracy when using Polynomial Kernels

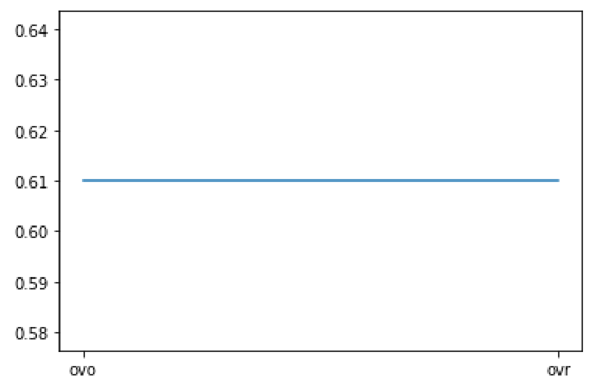


Figure 7: Influence of the Decision Function on the Accuracy

means that more training data is misclassified.

If we can assume that in our case training and validation data are very close to each other this would also explain the results from the previous section, where it was visible that with a higher value of C, the accuracy of our model rises.

Hyperparameter Kernel
The task of the kernel is to transform the training data to the so called feature space, which enables the classification of the data via a hyperplane. There are different kinds of kernels which achieve the transformation in different ways and have different strengths and weaknesses.

Hyperparameter Degree for Polynomial Kernels
A polynomial kernel is defined as $(x^T y + c)^d$. The degree is the parameter d.

Hyperparameter Decision Function
As a SVM can only do binary classification a special strategy to deal with multiclass classification is needed. There are two ways, one-vs-all where multiple SVMs are trained, each SVM can discriminate between one class and the rest of the other classes. The other ways to do it is one-vs-one where multiple SVMs are trained each discriminating between two of the classes. As, accoring to the documentation, sklearn is internally always falling back on one-vs-one classification it is not suprising that there is no visible difference in accuracy between the two strategies. .

2.2.3 Question 3

The validation set and test set have slightly different roles. The validation set is used to tune hyper-parameters and to monitor the progress of training. The test set is used to evaluate the model after training is complete.