University of Córdoba

Computer engineering
Computer Science Engineering Degree
3rd course

Fundaments of Computer Vision

Lab Assignment 7: Histopathology classification

Valentín Gabriel Avram Aenachioei
03524931C
p92avavv@uco.es

Academic year 2022-2023
Córdoba, June 9, 2023

Contents

In this paper, I will set out how I have done the seventh lab assignment, the histopathology classification, focusing on different experiments, trying different combinations of classifiers, descriptors and hyperparameters for those classifiers, and analysing the obtained results.

1 Descriptors

For this lab assignment, 3 different descriptors have been used: the descriptor given as a guideline, a descriptor based on the SIFT (Scale-Invariant Feature Transform) algorithm, and a descriptor based on the SURF (Speeded-Up Robust Features) algorithm.

The first descriptor will not be explained, because it is already given. For both SURF and SIFT descriptors, exhaustive information can be founded in the OpenCV documentation.

1.1 SIFT descriptor

This own-made descriptor is based on the SIFT algorithm, already implement as a non-free class in OpenCV.

The algorithm finds the coordinates of the key points in a determined scale, giving a orientation to each key point. The computation of those key points in the image is based ont the difference of Gaussian for finding scale-space, based on small corners of the image.

Once the key points are calculated, a 16x16 neighbourhood around the keypoint is taken, and sub-divided into blocks. Several measures are taken to achieve robustness against illumination changes, rotation etc.

The code implementation is based on those concepts.

A SURF features detector is created, it is used to detect the key points, which are stored into a vector.

Then, we use those key points to compute a descriptor. In this case, the descriptor will not have a size bigger than 64x64, so the shape is adjusted, filling the empty spaces with the mean values of the descriptors.

1.2 SURF descriptor

This own-made descriptor is based on the SURF algorithm, based on the SURF algorithm, already implement as a non-free class in OpenCV.

The process is similar to the SIFT algorithm. First of all, it detects the key points, using square-shaped filters as an approximation of Gaussian smoothing. To speed up the process of the key point detection, the integral image is used:

$S(x, y) = \sum_{i=0}^{x} \sum_{j=0}^{y} I(i, j)$

The rest of the process is quite similar to the SIFT algorithm. To describe the region around the point, a square region is extracted, centered on the interest point and oriented along the orientation as selected above.

The code implementation is again similar to the previous case. The difference are obviously the feature detector used, a SURF detector in this case, and just to try to change the functionality, the descriptor empty spaces are filled to zero, instead of the mean of the descriptor.

2 Experiments and results

First, it should be made clear that only the different tests and their results will be cited in this section. The analysis of these results will be explained in the following section.

For every experiment, 20% of the dataset has been used for validation. Also, the default hyperparameters as been used for each classifier, the reason will be explained in the **Conclusion** section.

In this case, the Random Forest has been implemented in code, but it is not working properly, so it will not be tested.

First of all, testing each descriptor with the different combination of image normalization and classifiers. The table 1 shows the results of validation using the simple gray descriptor.

Image Normalization	**Classifier**	**Mean Recognition Rate**	**Acuraccy**
None	KNN	*25%*	*34.91%*
None	SVM	*25%*	*27.63%*
MinMax	KNN	*25%*	*34.91%*
MinMax	SVM	*25%*	*27.63%*
Mean & Standard deviation	KNN	*25%*	*34.91%*
Mean & Standard deviation	SVM	*25%*	*27.63%*

Table 1: Gray descriptor in validation

The table 2 shows the results of validation using the SURF descriptor.

Image Normalization	**Classifier**	**Mean Recognition Rate**	**Acuraccy**
None	KNN	*25%*	*27.63%*
None	SVM	*25%*	*27.63%*
MinMax	KNN	*25%*	*27.63%*
MinMax	SVM	*25%*	*27.63%*
Mean & Standard deviation	KNN	*25%*	*27.63%*
Mean & Standard deviation	SVM	*25%*	*27.63%*

Table 2: SURF descriptor in validation

The table 3 shows the results of validation using the SIFT descriptor.

Image Normalization	**Classifier**	**Mean Recognition Rate**	**Acuraccy**
None	KNN	*25%*	*16.51%*
None	SVM	*25%*	*27.63%*
MinMax	KNN	*25%*	*16.51%*
MinMax	SVM	*25%*	*27.63%*
Mean & Standard deviation	KNN	*25%*	*16.51%*
Mean & Standard deviation	SVM	*25%*	*27.63%*

Table 3: SIFT descriptor in validation

Once we have created some models for each posible combination of descriptor, image normalization and classifier, we can tese those models through the testing option.

The table 4 shows the results of testing using the simple gray descriptor.

Image Normalization	**Classifier**	**Mean Recognition Rate**	**Acuraccy**
None	KNN	*25%*	*39.27%*
None	SVM	*25%*	*32.95%*
MinMax	KNN	*25%*	*39.27%*
MinMax	SVM	*25%*	*32.95%*
Mean & Standard deviation	KNN	*25%*	*39.27%*
Mean & Standard deviation	SVM	*25%*	*32.95%*

Table 4: Gray descriptor in testing

The table 5 shows the results of testing using the SURF descriptor.

Image Normalization	**Classifier**	**Mean Recognition Rate**	**Acuraccy**
None	KNN	*25%*	*32.95%*
None	SVM	*25%*	*39.27%*
MinMax	KNN	*25%*	*32.95%*
MinMax	SVM	*25%*	*39.27%*
Mean & Standard deviation	KNN	*25%*	*32.95%*
Mean & Standard deviation	SVM	*25%*	*39.27%*

Table 5: SURF descriptor in testing

The table 6 shows the results of testing using the simple gray descriptor.

Image Normalization	Classifier	Mean Recognition Rate	Acuraccy
None	KNN	*25%*	*32.95%*
None	SVM	*25%*	*32.95%*
MinMax	KNN	*25%*	*32.95%*
MinMax	SVM	*25%*	*32.95%*
Mean & Standard deviation	KNN	*25%*	*32.95%*
Mean & Standard deviation	SVM	*25%*	*32.95%*

Table 6: SIFT descriptor in testing

3 Conclusions

As a basic conclusion, it can be stated that these results are useless, as the programme is badly programmed, and is not functional.

For all three descriptors, we obtain the same or very similar results, changing the image normalization type or the classifier used. Those results are expected for the first descriptor, the simple gray descriptor, but meaningless for the other two descriptors. Since the descriptors are wrongly implemented, the results obtained are practically null, which makes that the change of classifiers and their respective hyperparameters does not affect the results. Because of that, only the default hyperparameters of the classifiers have been used.

We can see that, in validation, the best model obtained are using KNN as classifier, in this case because of the previously mentioned problems, with any of the descriptor. The best accuracy obtained is *34.91%*.

For testing, the results are improved, comparing to those obtained in validation in every case. The best accuracy obtained is *39.27%*.

While using the simple gray descriptor, we obtain slightly better results using KKN as classifier, instead of using SVMs, in validation. As it is expected, the results are slightly improved while testing, instead of validating, because a smaller set of samples is used.

While using the SURF descriptor, we obtain the same results, with every posible combination of image normalization and classifier. Again, the cause of this results is the bad implementation of the descriptor. At testing, the results are improved, obtaining better results using

While using the SIFT descriptor, we obtain the worst results so far. Those results are improved using SVM as classifier, instead of KNN. Once validated the model, the results obtained in testing are the same for every posible combination of classifier and image normalization.

For every case, in both testing and validation, there is no appreciable difference between using different types of image normalization. This is caused by the wrong implementation of the descriptors.

About classifiers, there is a significant difference while using KNN or SVM. This difference can be positive or negative towards the accuracy obtained, depending on the descriptor used.

In every single case, for both testing and validation, the Mean Recognition Rate obtained is *25%*. This can be explained because of, depending on the descriptor used, it classifies every single image into the same class, getting a perfect accuracy for that class, and a constant missclasification for the rest of classes.

As a personal conclusion, I tried to do my best. I tried to create my own descriptors based on the Canny edge detection, based on the gradient and magnitude, based on the already patented SURF, SIFT, BRIEF and ORB algorithms. None of it worked. I even tried to copy the descriptor made by other years students, and, of course, it did not worked.

My conclusion is that in every case, I am saving incorrectly the final descriptors. The classifiers are already implemented in OpenCV, the already patented algorithms work properly and are able to find key points in the images, but when comparing different descriptors, the classifier is not using properly the information.

Depending on the classifier used, the program classifies every single image into the same class, getting accuracy ratios close to 25%.

Some other charts and plots could be added, but there is no conclusion to obtain since the program doesn't work properly, so there is no reason to add more charts.

I really tried to improve it, to make it work properly, but I could not. I know that it does not work, but I really do not know how to solve it. I also tried to do an in-depth analysis of the results, but there were not any results to analyse. Im deeply sorry.