Visual Recognition
Week 2

●●●

Team 1
Eric López
Gerard Martí
Sergio Sancho
Adriana Fernández

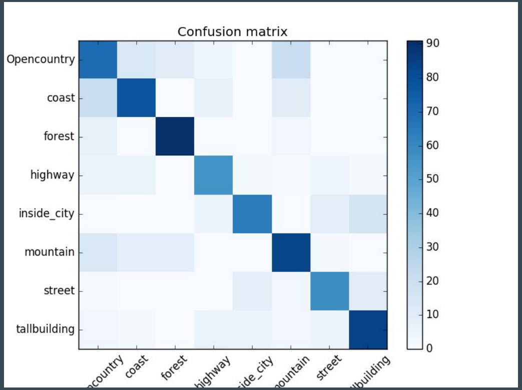
Improvements from week 1

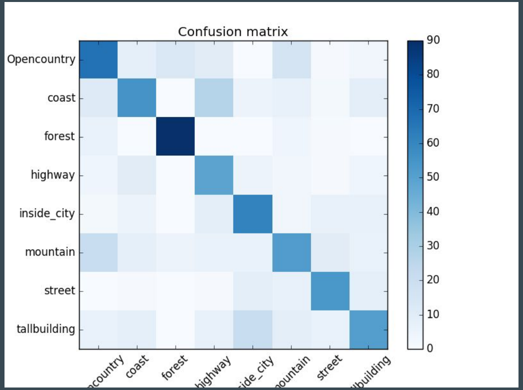
- Based on the feedback obtained in the follow-up session last week, we have made several improvements on our work from the past week to include said feedback:
    1. Fixed a bug that made the size of our dictionary really small, hindering our results.
    2. Fixed another bug in our ROC curve implementation.
    3. Parallelize some parts of the code (still working on this)
    4. Cluster usage for more testing (but some outdated libraries makes working with cluster really difficult).
- In the next slide you will find our improved results from past week, as well as some comments about them.

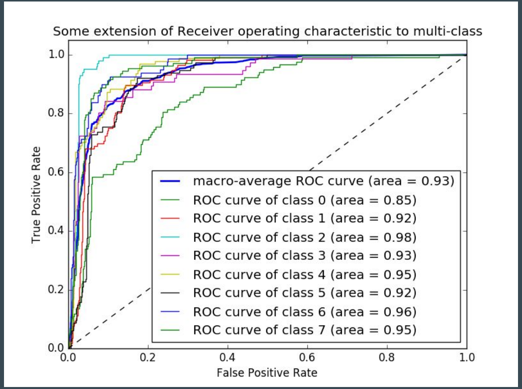
Improved from week 1

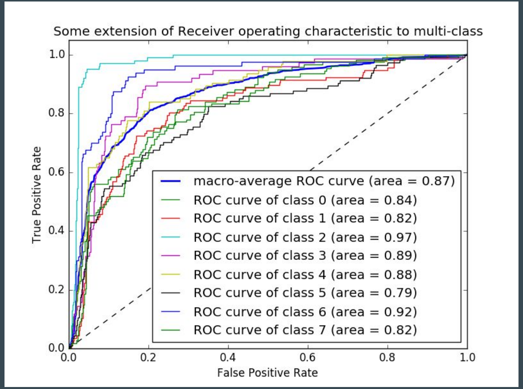
SIFT - k = 5000, nsamples = 50000
Accuracy: 0.7422

HOG - k = 5000, nsamples = 50000, blocksize = (16,16)
Accuracy: 0.68









- By only changing the value of the number of samples and the number of centroids we are able to improve the results from past week, from 0.64 to 0.73 - A big improvement from previous weeks.
- HOG is also improved by changing the window size from (64,64) to (16,16), as it seems that a smaller window is able to generalise a better model.
- With the parallelization of the cross-validation function, we have been able to reduce the computation time by **61%.** This allows us to perform more tests.
- The tests are performed using a SVM OvR Cross-validation with linear kernel and nfolds = 5.

Experiments and Implementations for week 2

For this week, we will focus on using Early fusion with several different descriptors and detectors, implementing PCA to reduce dimensionality and finding the best parameters for it, analyzing the keypoint density and computation time of the different detectors, and find the best combination of descriptors/detectors, color or shape, to be able to obtain the best results.

In the following slides we will explain each experiment we have performed and the results obtained.

Experiments - Keypoint density and computation time

- For each pair of detector and descriptor, we have their performance, their computation time and the number of keypoints generated. The code used for comparison is the same across each test, and it is done without PCA. The descriptor is SIFT for all cases for an equal comparison, and the parameters of the bag of words are nsamples = 50000 and k = 5000., with an SVM using cross-validation, with 5 folds. The total computation time is the extraction of local descriptors, the computation of the codebook and the extraction of the codewords, for the training set:

Detector	'SIFT'	'Dense'	'SURF'	'ORB'	'FAST'	'harris-laplace'
keypoints per image (avg)	506	1849	551	400	2157	501
Computation time	116.25s 32.85s 36.2s	17.08s 29.93s 120.93s	707.251s 29.98s 39.69s	443.98s 26.55s 33.37s	368.701s 28.760s 154.119s	32.035s 49.236s 64.892s
Total time	185.3060	167.9530	776.9220s	503.9120s	551.5820	146.16s
Accuracy	0.74721	0.79553	0.80173	0.71375	0.86245	0.7496

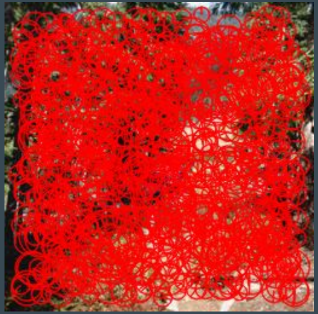
- In the following slide we could see the keypoints on the image, for the two best results, and the conclusions about said experiment.

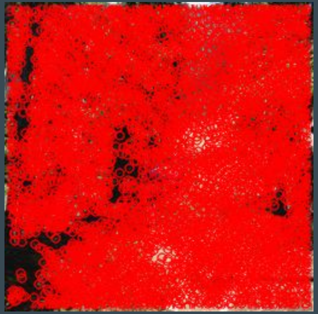
Experiments - Keypoint density and computation time

We can extract several conclusions from the results presented in the previous slide:

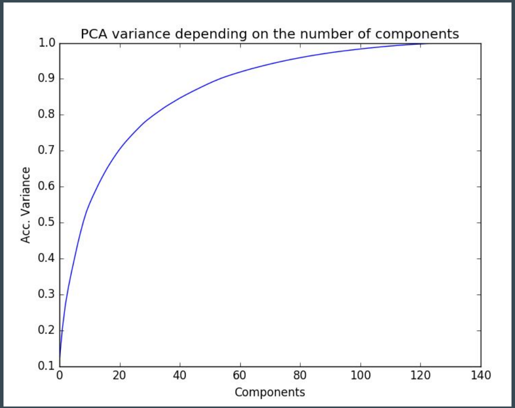
- The computation time for Harris-laplace, SIFT and Dense is shorter, but they also do not present the best results. Also, the methods that detect more keypoints per image seem to spent more time doing the extraction of visual word representations. Dense and Harris-laplace are really fast in computing the Local Descriptors.
- The best results are obtained for SURF and FAST. We will try to focus on those two detectors for experiments with early fusion.
- We have to mention that this computation was done without using dimensionality reduction. We will implement it for the early fusion.

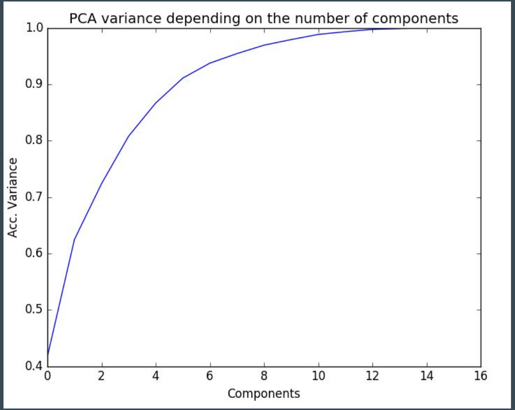
Original Image

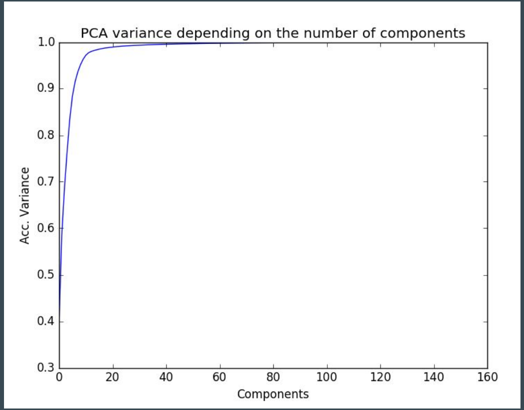

SURF Keypoints


FAST Keypoints

Experiments - PCA



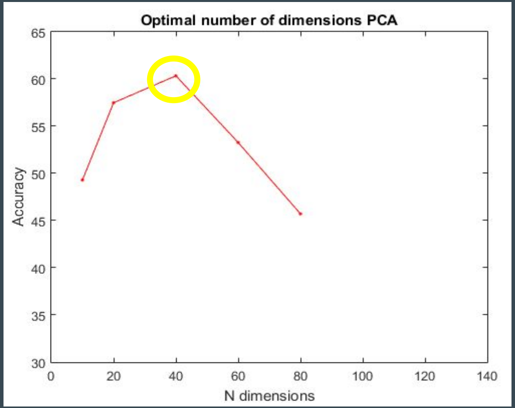


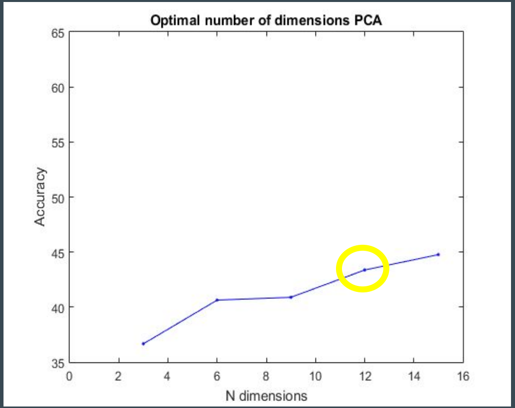


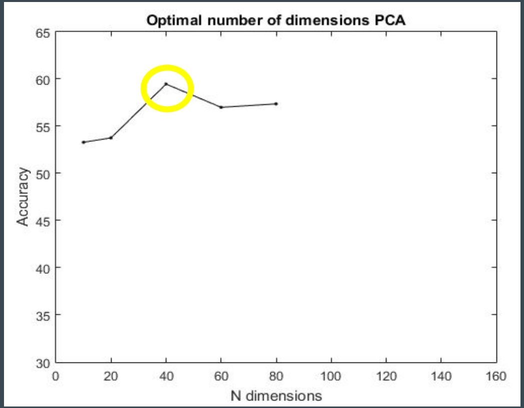
SIFT descriptor

Color descriptor

Early Fusion descriptor







Experiments - PCA

When a descriptor is computed not all the dimensions that we obtain are useful and representatives of our data. We only have interest in the dimensions that represents more variance of our data. PCA is an algorithm that allows as to know the relevance of each dimensions with respect to the variance that they represents. We have to choose the appropriate dimensionality of our descriptor taking into account that we don't want to lose enough information. In the previous slide we show two differents types of graphics. In the superior figures we are showing the variance that represents each dimension using different descriptors (SIFT, Color histogram, Fusion (SIFT + Color)). For example, if we see the early fusion descriptor we can see how only using 20 dimensions (of the total of 160) we are representing approx. the 100% of the variance.
In the inferior graphics we are computing the accuracy varying the number of dimensions selected with the different descriptors, fixing SIFT as detector and K = 5000 clusters for k-means. The optimal number of dimensions of each descriptor is shown in the next table.

	SIFT	Color	Early Fusion
N Components	40	40	12

Experiments - Early Fusion

We experimented with early fusion (to compute two or more descriptors and fuse them before computing the codebook). To to this, we added the following modifications to the code:

- Added a normalization step after each descriptor, in order to put them on the same level.
- Added PCA dimensionality reduction to each descriptor to reduce the dimensionality and reduce computation time, while minimizing the loss of information. For this, we try to use always an optimal PCA component number, whether it is by computing it or by using the automatic choice of Dimensionality for PCA option that its available in the sklearn implementation of PCA.
- Parallelization and usage of cross validation.
- Used the color naming descriptor and adapted it to our code.

We tried to do early fusion with both the original color descriptor and the color naming descriptor, by fusing them to a descriptor and with a descriptor that gave us good results. Looking at the slide 5, we decided to focus on FAST, SURF and Dense as the detectors, and SIFT and SURF as the descriptors.

On the next slide we will present the results obtained.

Experiments - Results

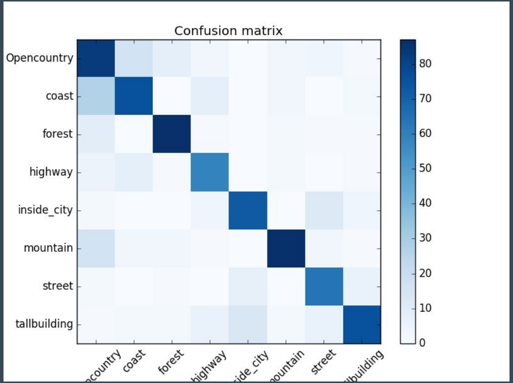
Detector - Descriptor	ColorDescriptor	Accuracy (D-C-F) (%)
Dense - SIFT	ColorHistogram	80 - 30.73 - 81.78
Dense - SIFT	ColorNaming	79.6 - 43 - 80.02
SURF - SURF	ColorHistogram	37 - 44 - 44
SURF - SURF	ColorNaming	36 - 52 - 42
FAST -SIFT	Color Histogram	84 - 40 - 84
FAST -SIFT	ColorNaming	66 - 53 - 64
SIFT - SIFT	ColorHistogram	68 - 45 - 49

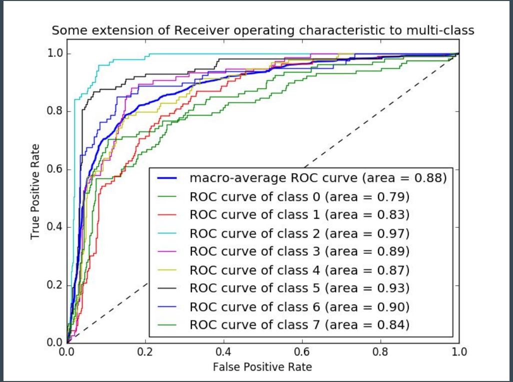
Note: For Accuracy, (D-C-F) means Descriptor - Color - Fusion, and references the accuracy that a Linear SVM model achieves using said descriptors.

On the table, all the experiments that we performed are presented. We can observe several things:

- In some of the fusion, we obtained a poorer performance than when using the single descriptor (mainly when using the ColorHistogram descriptor). Sometimes though, the fusion was beneficial (for example, for Dense-SIFT, with a great 80% accuracy).
- The PCA reduction did help us to improve our performance, but it also made us lose some accuracy. This may have been too due to a bad choice of number of components.
- In general, ColorNaming tends to give us a better performance.
- The computation times on some descriptors (mainly FAST) were prohibitive to work with (and even more since the Cluster was not an option, as it has a lot of out-of-date modules and our code would not work in it.

Final results and Conclusions





We present, to the left, the best results we obtained during this week experiments. The figures to the left correspond to the FAST-SIFT descriptor, with PCA, normalized and using a Linear SVM with C = 0.01, and with a codebook with nsamples = 50000 and k = 5000. We have a really good Confusion matrix, although the ROC curve values are not that better than other results. Moving onto next weeks, we have several points we can tackle to improve the results:

- As before, keep doing experiments and try different parameters to reach a better performance. A great focus could be the PCA number of components and to try other types of fusion.
- As we mentioned before, the cluster is not available to work, as when we tried it our code would not run due to incompatibilities. That is something we will have to overcome by using more computation time in our computers.