PHOW Lab report: Computer Vision

Emilio Botero
Universidad de los Andes
e.botero10@uniandes.edu.co

1. Introduction

Image classification is a procedure that, through a set of image features, produces a class label for the image. Classifiers are built by taking a set of labeled examples and using them to come up with a rule that assigns a label to any new example. As such, we have a training dataset (x_i, y_i) where each of the feature vectors x_i consists of measurements of the properties of different types of objects (filter bank responses for pixels, for instance) and the y_i are labels giving the image's *classes* (typically objects, animals, etc).

The difficulty in classifying images is, then, choosing or building good image features. The goal is to build features that expose *between-class variation*, which is the reason classes look different from one another, and suppress *within-class variation*, which accounts for objects of the same class that look different [1]. In particular, a good way to build features is using the SIFT descriptor: we divide a given image into $16x16$ patches (centered at given keypoints), each of which is also divided into $4x4$ subpatches. Within each subpatch, gradient magnitudes and directions are computed: the orientations are put in an 8 bin (for 8 different gradient orientations) histogram according to each point's magnitude. However, we want gradient magnitudes further away from the subpatches' center to have a lower impact in our histogram, so we filter each subpatch with a gaussian function. Finally, we normalize the histogram. In this way, we generate a $4x4x8 = 128$ dimensional vector for each subpatch [2].

For all training images, then, SIFT histograms around keypoints are computed, and ultimately concatenated, building a $128xN$ array, where N is given by the number of training images multiplied by the number of keypoints within each image. This array is clustered using k-means in order to build a "visual vocabulary" of database content. Finally, pyramids of visual word histograms [3] are built by dividing a training image into regions, computing the visual word histograms for each region and concatenating all histograms into one final descriptor. Thus, we compute pyramids of visual word histograms for each training image and feed them into SVMs with a χ^2 kernel for each category in the dataset, which then classifies a new input image as to whether corresponding to a particular category or not. The image is classified according to the class with greatest margin.

2. Dataset and experiments

The ImageNet database [4] is an image database organized according to a hierarchy of categories, wherein a node in the hierarchy gives a general class for an object, and child nodes correspond to further divisions and types of the object (e.g a node might be "dog" and its child nodes "syberian husky", "cocker spaniel", etc.). We utilized a subset of the ImageNet dataset composed of 996 categories, with 100 color images of size 256x256 for each category in both the test and train subsets. In Caltech-101 [5], 102 classes were classified using a visual vocabulary of 600 words. Images were obtained by web-search, and given the image availability at the time of database construction, most images were "catalogue-like", with objects in the center of the image, white backgrounds, etc. We experimented with various visual word numbers, and decided to use a visual vocabulary of 600 words too, since using more tended to overfit and decrease classifier accuracy. Using less visual words also tended to decrease classifier accuracy.

We also used pyramids to train our SVMs. Visual word histograms were considered for 4 levels, in each of which the image is subdivided into more regions: each region's histogram of SIFT features is concatenated into a final histogram.

Given the size of the database and time constraints, however, we performed our experiments using at most 300 categories. Classifier accuracy and runtime was evaluated by increasing number of categories in steps of 5, and number of training images in steps of 15. 5 pyramid levels were considered.

3. Results

The following graphs illustrate the classifier's performance as a function of the number of classes, number of training images and number of pyramid levels in the problem:

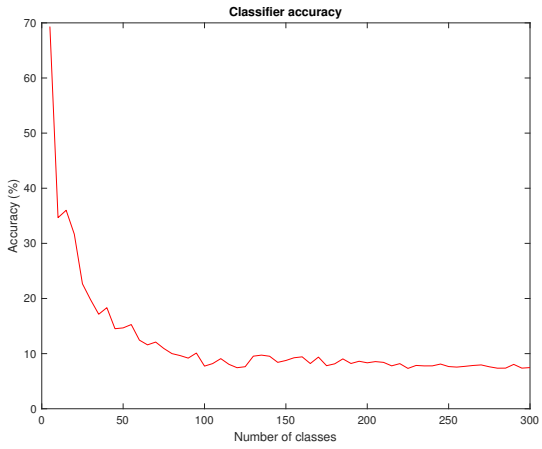

Figure 1. Classifier performance as a function of number of classes

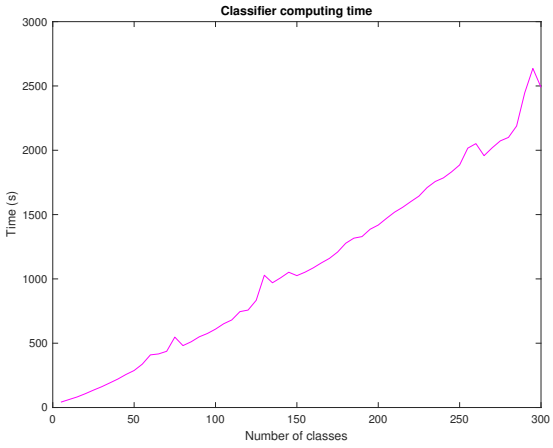

Figure 2. Classifier computing time as a function of number of classes

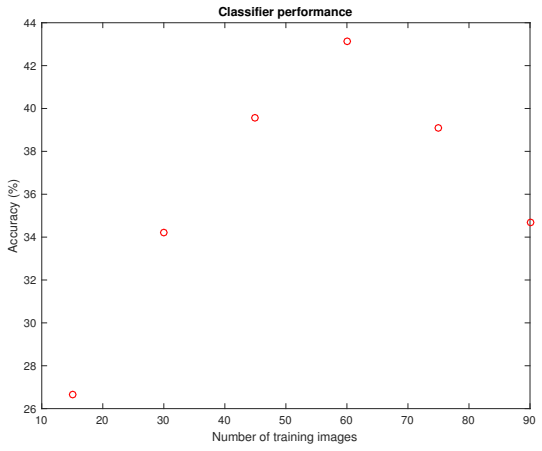

Figure 3. Classifier performance as a function of number of training images

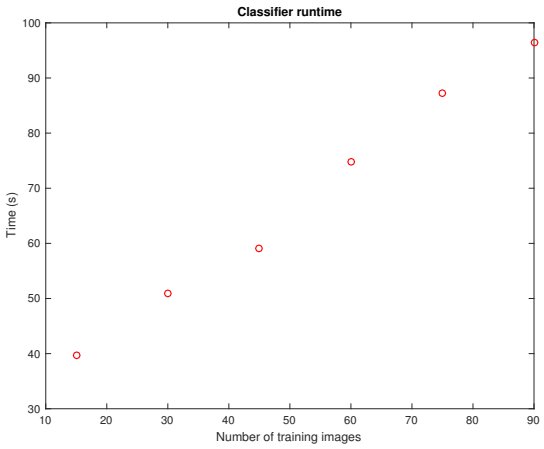

Figure 4. Classifier computing time as a function of number of training images

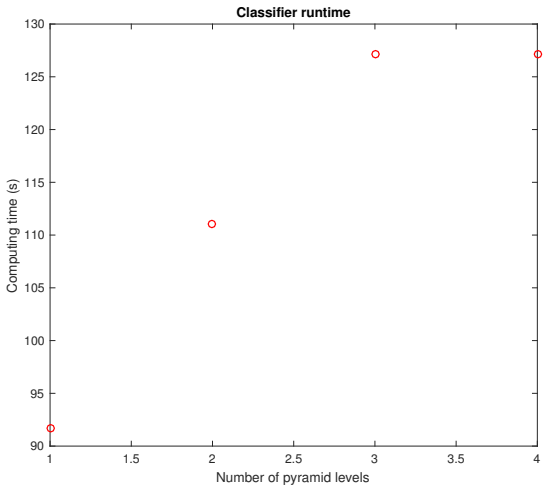

Figure 5. Classifier performance as a function of pyramid levels

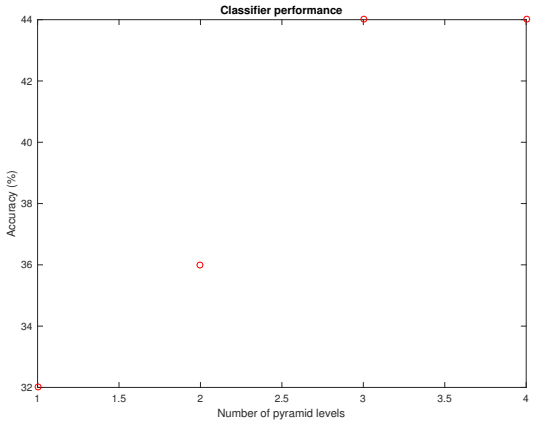

Figure 6. Classifier computing time as a function of pyramid levels

Additionally, sample results for 5 categories and 15 test and training images with only 1 pyramid level in the Caltech-101 database and the ImageNet database are shown below:

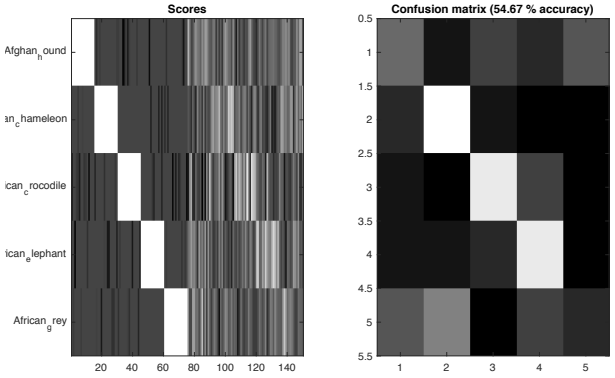

Figure 7. Performance in ImageNet subset

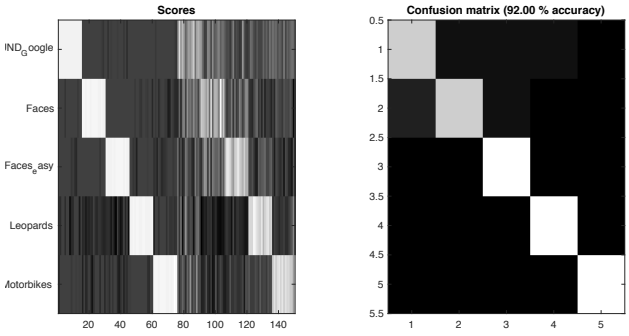

Figure 8. Performance in Caltech-101

It is of note, then, that classifier accuracy decreases almost exponentially with the number of classes included until it reaches a certain accuracy value. Similarly, classifier computing time increases almost linearly with the number of classes included. Increasing the number of training images results in an up and down pattern, reaching a peak at 60 training images and decreasing sharply after that. Runtime also increases linearly with the number of training images.

On the other hand, increasing the number of pyramid levels both increases accuracy and runtime linearly up to a point.

Finally, there is a sharp contrast between the ImageNet and Caltech-101 database results: a decrease in nearly 40% accuracy.

4. Discussion

As can be seen in figure 1, the classifier's performance sharply decreases when increasing the number of categories. First off, there is a vast difference between both databases, as can be seen in the figure below:

Figure 9. Visual comparison between both databases

The top row corresponds to images taken from the ImageNet database binoculars category, while the bottom row corresponds to the Caltech-101 binoculars category. In the Caltech-101 database, images are "catalogue-like": the binoculars are in the center of the image, often with white backgrounds, and generally very dominant in the image. In the ImageNet database, however, the objects are present in the image but the image includes other objects, points of view, etc. This is why the performance of these classifiers drops nearly 40% from Caltech-101 to ImageNet.

When the number of categories is increased, there is a higher chance that a given image gets wrongly classified simply because, given the complexities of images in the ImageNet database, it is easy to imagine how an SVM might produce a better margin for another class (with images with similar feature descriptors). This might be because our features aren't enough to correctly identify the between-class variation in this database. On the other hand, perhaps the number of visual words isn't high enough to capture significant class variance. Eventually, performance evens out as an asymptote at nearly 10% accuracy. This tells us that there are at least some categories for which the classifier learns and predicts correctly, which tells us we could attempt to increase the number of visual words used and see if the overall accuracy improves.

Runtime increases linearly with the number of classes

included. This is to be expected since each time we add a class, we add a fixed number of images to train and test into the algorithm. If training and testing for 300 categories takes about 2500 seconds, we can roughly estimate 8300 seconds for running the program (that is, building a dictionary, clustering, training and testing) in the entire subset of the ImageNet database. This would correspond to training and testing our classifier in 2.3 hours for 84660 images (70 training images and 15 test images for each category) : this is actually not an unreasonable runtime for such a number of samples.

Increasing the number of training images resulted in an up-and-down pattern of accuracy: it reaches a peak at 60 training images and then sharply decreases. This might be due to overfitting induced by the excessive number of training images used. Runtime increases linearly: each time 15 additional images are used and this accounts for the additional time.

Increasing the number of pyramid levels resulted in better accuracy. This is because splitting the image with pyramids encodes image information in multiple resolutions, resulting in a higher-dimensional representations (histograms) that preserve more information. There is a point, however, upon which increasing this resolution does not yield additional, significant information; rather, it may lead to redundancy in histograms since a perfectly uniform region might be getting split without need. Again, runtime increases almost linearly: more features need to be computed each time, which accounts for the increased runtime.

5. Limitations and improvements

Evidently, the SIFT descriptors fail at capturing the complexity of the ImageNet database. SIFT classifies nicely in simple datasets, but real world images are composed of a wide variety of objects, textures and regions. A possible solution to this might be to use different classifiers since at such a number of categories, SVMs might not be the best solution with our features. Random Forests could be attempted. Another possible approach would be to build SIFT descriptors for each of the color channels or use a different set of features such as GIST or HOG.

References

[1] D. Forsyth and J. Ponce, "Computer vision." Prentice Hall, 2012, pp. 483–491.

[2] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[3] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 2 (CVPR'06)*.

[4] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, and M. e. a. Bernstein, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[5] L. Fei-Fei, R. Fergus, and P. Perona, "Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories," *Computer Vision and Image Understanding*, vol. 106, no. 1, pp. 59–70, 2007.