Image classification based on texture local descriptors

Alejandro Pardo
Universidad de los Andes
la.pardo2014@uniandes.edu.co

Lina Barbosa
Universidad de los Andes
lm.barbosa1099@uniandes.edu.co

Abstract

We propose an image classification method based on texture local descriptors. The algorithm generates an image representation based on a texton dictionary created after clustering a 32 filter bank responses. Random forest and Nearest Neighbor classifiers where trained and tested using Ponce group texture dataset [1] getting an overall precision of 66.8%. For the nearest neighbor classifier the intersection of histograms and the chi square distances were used to compare histograms.

1. Introduction

Image classification is one of the most important topics in computer vision and has been an are of intense research in the last years. This task consist of extracting and analysis numerical properties of images features and organize data in different categories. Texture analysis has been used as an important tool for image classification for the cases where specific textures can be recognized, and also can be associated with an specific object. In this work we propose a method for image classification based on texture local descriptors. This descriptor were formed by the response of a filter bank created conveniently to describe the different textures present in the target database. The algorithm was trained and evaluated using the Texture Database from Ponce Group [1]. To a better approach to the problem, we proposed 4 variation of the method by changing the classifier, its parameters, and the way of training the algorithm.

2. Proposed Method

We proposed a method to classify images based on its appearance and texture, which is divided into a learning and a classification stage. In the learning stage, training images response to the convolution of a bank of filters where obtained and then clustered using K-means generating different clusters or textons whose centroids will form a general dictionary that could be used to create a general representation of each input image. After the textons dictionary was created it was necessary to use it to learn different models that could represent each of the texture categories, so given an input image, its corresponding model is created based on the response of its convolution with the bank of filters and then labelling each of these responses which the centroid (texton) closest to it. After the image is represented using the dictionary of textons, it is possible to create an histogram of textons associated with this representation which measures the frequency with each texton occurs in the labelling. In the classification stage, the same process is developed but in this case the images used correspond to those in the test set ot the database. When the histogram of each of the input images is calculated it is compared with each of the training images to generate its corresponding label and category.

2.1. Database

To evaluate the proposed method we used the Ponce Group texture [1] which features 25 texture classes with 40 samples, so the hole database was formed by 1000 samples divided into train and test sets. In the train set there were 30 samples of each category (a total of 750 samples) and in the test set, 10 samples of each category (a total 0f 250 samples). All of the images were in grayscale JPG format with size of 480x640 pixels.

2.2. Bank of filters

The bank of filters was constructed to recreate some texture patterns that can be recognized in the images that will be trained and tested. To do this we used a multi scale, multi orientation filter bank with 32 filters. It consist of even and odd symmetric filters, both with 8 orientations and 2 scales. The even symmetric filter was generated with a Gaussian second derivative and the odd symmetric filter with its Hilbert transform and both occur at scales of $\sigma = 1,\sqrt{2}$. In the Fig. 1 it is shown the bank filter used by the algorithm at a scale of $\sigma = 1$ (16 filters)

The filters that discriminate the most were selected using the variance of the mean responses of each filter on each image. This measure was selected because it gives information about the variation of the filter response between the differ-

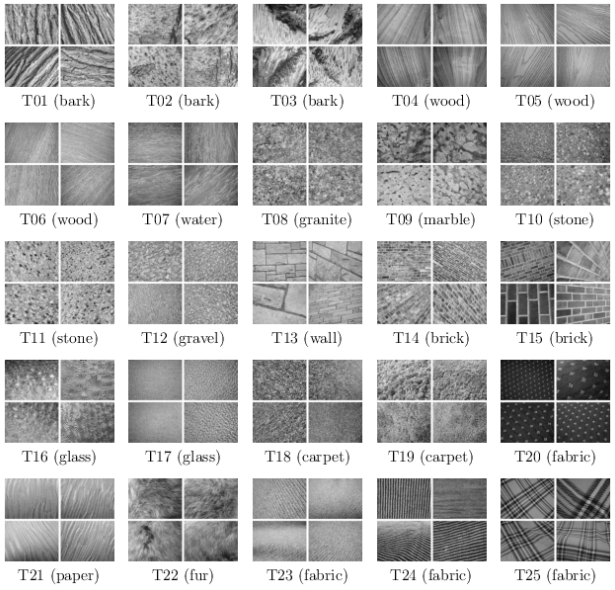

Figure 1. Four samples from each of the 25 categories of texture database from Ponce Group

Figure 2. Bank filter at a scale of $\sigma = 1$

ent test images, so it means that those with higher variation will be more discriminating because it could be useful to distinguish between distinct images due to the variation on its response. On the other hand, if the filter has a similar response in all of the images it could not be used to discriminate between categories and images. Based on this, the filters that discriminate the most are shown in Fig. 3

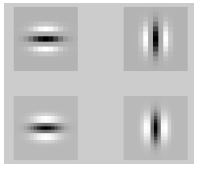

Figure 3. Filters that discriminate the most

2.3. Textons Dictionary

The textons dictionary was created based on a selection of 3 images from the training set for each texture class of the database. For each of the images the 36 filters responses were concatenated and then grouped into 50 clusters using K-means. The centroid of each of the 50 clusters represents a texton and contributes to the generation of the textons dictionary, which will be formed by 50 textons. The number of textons was selected based on the 25 categories giving place to the database because all of its samples appear to have an homogeneous texture. Based on that, the first approximation was to set the number of clusters in 25; nevertheless, this won't be enough because in spite of the apparent homogeneity of the texture in all the categories, there were common patterns between pairs of categories, so we preferred to double the number of textons to increase the selectivity of the classification method.

2.4. Classifiers

To develop the classification stage we used to different classifiers. The first one was a nearest neighbor classifier based on the comparison between the textons histogram of the whole training set and an input test image. Based on this, the label that will be assigned to the input image will be the same of the train image whose texton histogram is closer to the one of the input image. To estimate the distance between both histograms we used as the kernel intersection of histograms which can be calculated with the following expression:

$$d(H_1, H_2) = \sum_{i=1}^{n} min(H_1(i), H_2(i))$$

Where n corresponds to the number of bins of the histograms which is determined by the number of textons. This distance is based of the bin to bin comparison of both of the histograms. Because all of the histograms must be normalized (the sum of all its bins equals 1) the distance given by this classifier for two identical histograms must be one and the one for two completely different histograms will be zero. Based on this, the classifier will give to the input image the label of the image whose histogram has the maximum distance to it.

the second method was developed using the Chi-square distance which can be obtained using the following expression:

$$d(H_1, H_2) = \sum_{i=1}^{n} \frac{(H_1(i) - H_2(i))^2}{H_1(i)}$$

We can observe that the selection parameter oppose the one in kernel intersection because in this case for two identical histograms we will obtain a distance of 0, so the classifier will give to the input image the label of the image whose histogram has the minimum distance to it.

The second kind of classifier employed was random forest, which was trained with the texton histogram of the whole training set of images using a total of 50 trees to generate the label of test input image.

3. Results

In order to obtain the best possible performance, We implemented four experiments that allow us to establish the

best combination of the different parameters. The principal parameters that we consider was: the number of images which were used for the creation of the texton dictionary, the number of clusters k (number of textures) used in k means and the classifier used (nearest neighbor with different distances and random forest with different number of trees). Below is shown a table with the parameter of the four performed experiments:

Alternative	Dictionary Images	Textures	Trees
1	1	30	20
2	3	30	20
3	3	50	20
4	3	50	50

For alternative one, we used only one image from each category to create texton dictionary. We also used 30 textures and 20 trees for the random forest. For the second one, we increase the number of images to 3 for category and rest of the parameters remain constant. Third alternative was performed with same parameters as second, but increasing the number of textures from 30 to 50. Finally, the last alternative was developed thinking on the improvement random forest by increasing from 20 to 50 the number of trees used.

The results are shown below:

Alternative	N.N.-Chi	N.N.-K	Forest
1	0.0360	0.0520	0.0480
2	0.6160	0.5560	0.6040
3	0.6320	0.6120	0.6600
4	0.6320	0.6120	0.6680

Previous table show the performance for the different experiments. The first column of the table is the number of the experiment, the second column is showing Nearest Neighbor with Chi square distance, and third column shows results for k-Nearest Neighbor. Finally, last column shows results for random forest classification. We can observe in red the best Overall Classification Accuracy.

4. Discussion

Following the results showed previously, We can observe that the best performance is reached with the random forest classifier, as We expected. Random Forest works as a non-linear classifier, while Nearest Neighbor is linear. This fact is the breaking point between both classifiers, because the nonlinearity can explain better the distribution of the data. However, table shows that increment the number of trees in the forest (an increment of 150%) was not an important parameter because the difference between 20 trees to 50 trees was only from 0.660 to 0.668, an insignificant 0,8% of improvement. In contrast, We observe that the number of images used to create the dictionary was the most important parameter, we only used three by category due to the limitation of the computational requirements. But We would expect that the more images you use to create the dictionary you will get better results. Another important parameter, but not as transcendent as the previous one, is the number of clusters used to create the textons (the number of textons used), We can see in the table the difference between alternative 3 and alternative 2, which is a 6% of improvement by changing the number of textons from 30 to 50.

The most confusion was caused by category 12. With the best of the alternatives the error in this category was 70% which means that the method used was unable to describe this texture. However, there are other categories which error is over 50%, categories: 2,7,10,12,14,19,22,24. As you can see below, We show you the confusion matrix obtained for alternative 4, that illustrates better the performance of each category:

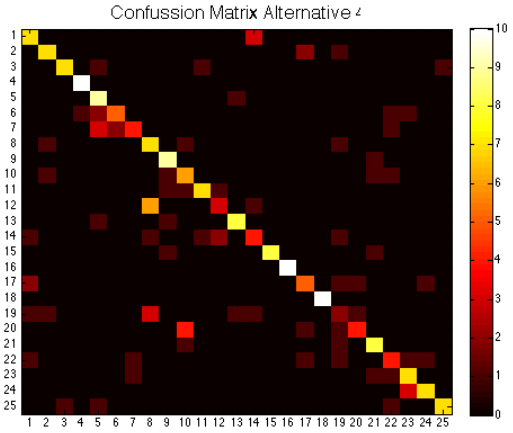

Figure 4. Confusion Matrix for best Result

This matrix show us as the performance of the method. We can see that in the trace of the matrix the values are bigger that in the rest of the matrix, which explain the number o 0,668 obtained. However, we can also see the categories with more problems as dark tones in its trace, e.g. category 12 show us a really dark value in its trace, which means that we have problems with the classification of this category. In the same way we can see the dark values in categories mentioned previously which have more than 50% of failure. The best categories can be seen in white and yellow, and We can conclude that with our method, this categories can be classified with a really high accuracy.

5. Improvements

To propose some improvements to this works, first we have to see the principal limitations. Firstly, method implemented during this works had a really high computational

cost and it would not escalate good with other problems with more amount of data. Furthermore, for a good improvement using this method we need more computational cost because, as we mention earlier, the breaking point of this method is the number of images used to create the dictionary. However, add just one image increase significantly the cost.

Besides, this kind of methods of texture analysis are not invariant to changes of perspective or geometric transformations. We can have the same texture image with different perspective and the method will show different results for both images, despite they are both the same.

Another important limitation is that this method works for recognition of textures, however textures may not be the better descriptor to other more common problems of artificial vision. So this method may works quite good with databases which images are formed by only one object (because one object have its own texture), however with images within several objects may not be a good approach.

In addition to the limitation of the method, we can also find limitation of the database. Images that formed the database are textures only, and in real-life problems it would be really difficult to find images which are formed only by one kind of textures without background clutter and other information. Real life images are formed by many objects each one with its own texture, so the data base could not be a good approximation to solve real- classification problems.

After clarify the limitations of the method, improvements seems to be quite obvious. Firstly, the method have to improve its scaling to larger problems, this can be done by getting textures representations with simpler descriptors, in order to decrease the computational cost. Another improvement is to include in the method another kind of information like possible objects or a light segmentation, and with this previous knowledge get the textures by regions, in order to generalize the problem. Now, the perspective problem could be solved by the representation of one object by several images with different points of view. With this we would have more information to match textures that are in different perspectives. Finally, the database should be more general, with categories differentiated by textures but with images with more information like background, closer to real-life images.

References

[1] S. Lazebnik, C. Schmid, and J. Ponce. A sparse texture representation using local affine regions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1265–1278, 2005.