Textons

Johana M. Ramirez Borda
Los Andes University
jm.ramirez11@uniandes.edu.co

Francisco A. Rozo Forero
Los Andes University
fa.rozo1843@uniandes.edu.co

Abstract

A texton is the representation of a small texture patch, for example a collection of filter bank responses, that is usually expressed in terms of a frequency histogram. These histograms are used as trainning data in classifiers and then the test images are classified. So, the present project used the ponce group database in order to represent those images using textons and then train and evaluate two classifiers (K nearest neighbour KNN and random forest RF) based on the texton representation. It was found out that $k = 125$, $\#trees = 50$, *cityblock distance and a proportion of 70/30 in train/test images was the best combination in order to improve the performance of both classifiers. Moreover, random forest algorithm performed better than the KNN algorithm once the number of trees used is greater than a certain value.*

1. Introduction

Texture analysis is a common task in computer vision and refers to the characterization of regions in an image by their texture content. In this sense, the texture is related to roughness, bumpiness, smoothness... which quantitatively refers to variations in the gray levels of an image. This analysis is used in numerous applications such as medical image processing, automated inspection, remote sensing and texture segmentation. Even more, texture analysis is used when trying to identify objects in an image that are more characterized by their texture than by intensity [2].

A very frequent approach to texture analysis is based on the study of the responses of the images to a filter bank. So, within this context, the concept of *textons* emerged. A texton is the representation of a small texture patch, for example a collection of filter bank responses, that is usually expressed in terms of a frequency histogram. This histogram measures the relative frequency by which textons from the codebook apper in the texture [3]. So, once aqcuired the frequency histogram of textons, texture classification is performed by classifiers such as nearest neighbour (NN), suport vector machines and random forest (RF).

Considering this, the objective of the present project is to represent images using textons and then train and evaluate two classifiers (KNN and RF) based on the texton representation.

2. Materials and methods

For this lab, the software implemented was MATLAB R2016b. The inside functions implemented were the following:

- fbCreate: This function creates a filter bank with 16*2 filters.
- fbRun: This function convolves the train images with the filter bank.
- ComputeTextons: This function creates the textons dictionary from a template set of concatenated images. Also, the number of textons is set as a parameter K (Number of centroids in teh K-means algorithm).
- AssignTextons: This function finds the response of the image to the library of textons.
- Fitcknn: Function to compute K nearest neighbours algorithm so a classifier can be trained.
- Confusionmat: This function creates the confusion matrix when the inputs are the predictions and the groundtruth.
- Treebagger: Function to compute the random forest algorithm so a classifier can be trained. The parameters for this function are the number of trees and the training set with their respective annotations.
- predict: this functions assigns a class for an input, based on a classifier.

Moreover, functions like *distSqr, isum, padReflect* were provided in order to succesfully crete the filter bank, texton library and textons assignment.

2.1. Database

The database used in this project was provided by the Ponce group and was downloaded from [1]. This is a texture database which contains 25 texture classes, 40 samples each. All images are in grayscale, .jpg format and 640x480 pixels. The creation of this database was supported by the National Science Foundation, the European project LAVA, the UIUC-CNRS Research Collaboration Agreement, the UIUC Campus Research Board, and the Beckman Institute. This database was divided using a code created –lo que sea– and trying 3 different combinations. First, 50-50 train and test images, second 30-70 and third 70-30.

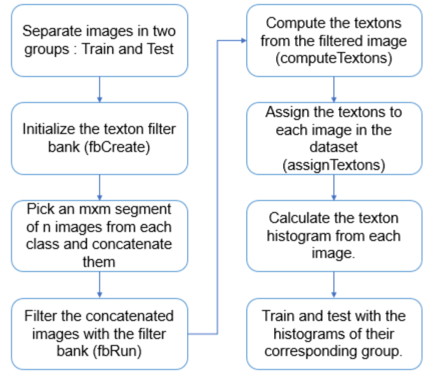
2.2. Image Pre-processing

Before the texton processing and assignation, a set of images from the database was chosen to serve as the base to make the texton library. From each category, were picked at random two images so they could represent their set. However, as the textons requires an unique image that contains all the concatenated images, using the entire image would had made an image too big for processing (1280x12000). Then, a small portion of each of the chosen images was concatenated. A window of 40x40 from the centre of each of the images was used, because most of them contain more characteristics from the images than in the border.

2.3. Textons processing

The texton representation of an image allows to identify and quantify it by its textures. In addition, this representation is useful when trying to classify images that are more characterized by texture than intensity. To do so, a textons library is created. Then, this library is used to acquire the frequency histogram which gives information about how many times a texton is present in an image. This histogram is the called ”feature” of the image and is used to train the classifier. So, in both cases (KNN and RF), the histograms of the training images are provided as well as their annotations. Then, when tryingt to classify a test image, the classifier will compare its histogram to those saved and will assign a category to the test image.

Therefore, using the functions provided and described before, the process used to obtain the dictionary is shown in Figure 1.



,

Figure 1. Algorithm used to create the dictionary.

When computing the textons, several values of k were tested (25,50,75,100,125,150,175,200 and 225). We decided to test more than one value in order to prove whether the number of textons affects the performace of the classifier. Moreover, all values are greater than the number of categories in order to improve the classification.

In the same way, three different proportions of train/test images were tested: 50/50, 30/70 and 70/30, in order to prove whether the proportion of trainning images over test images affects the performace of the classifier.

On the other hand, it is valid to say that some filters are more discriminative than others since each filter will have a better response to certain characteristics. Hence, it will provide more information about those images so it can be used for trainning and classification.

2.4. KNN classifier

K nearest neighbour algorithm finds a centroid for n categories based on a set of points; it clusters the information based on their distance. For the training, a random set of images from the database was filtered through textons and then the corresponding textons histograms were used as the data used for training.

Besides selecting the training images at random, and varying the proportion of images, the distance measurement method was modified to see which one fit the most to this dataset using the knn algorithm. Between these methods, was used:

2.4.1 Euclidean

$$d_{st}^2 = (x_s - y_t)(x_s - y_t)' \qquad (1)$$

2.4.2 Correlation

$$d_{st} = 1 - \frac{(x_s - \bar{x_s})(y_t - \bar{y_t})}{\sqrt{(x_s - \bar{x_s})(x_s - \bar{x_s})'}\sqrt{(y_t - \bar{y_t})(y_t - \bar{y_t})'}} \qquad (2)$$

2.4.3 Chebyshev

$$d_{st} = max_j\{\mid x_{sj} - y_{tj} \mid\} \tag{3}$$

2.4.4 Cosine

$$d_{st} = (1 - \frac{x_s y_t'}{\sqrt{(x_s x_s')(y_t y_t')}}) \tag{4}$$

2.4.5 City Block

$$d_{st} = \sum_{j=1}^{n} \mid x_{sj} - y_{tj} \mid \tag{5}$$

2.4.6 Minkowski

$$d_{st} = \sqrt[p]{\sum_{j=1}^{n} \mid x_{sj} - y_{tj} \mid^p} \tag{6}$$

2.5. Random Forest Classifier

Random forest is an algortih that by taking different characteristics of an input, it classifies it by using probabilities. The algorithm consists on taking different subsets of images making trees out of them, whose nodes are a characteristic and the arcs the probability of having it. Then, for the evaluation, an input passes through an initial node, then branching unitl it reaches a category. Finally, depending on where the input landed on each tree, it's category is chosen.

3. Results and discussion

In the first place, both classifiers were tested varying: the proportion between train/test images, the number of textons (k) and the number of trees in the random forest method. First, the confusion matrix was calculated using the function *confusionmat* and then, the ACA (the mean of the diagonal of the normalizated confusion matrix) was calculated too. The results are shown in Table 1.

Table 1. ACA obtained for three proportion of train/test images, k nearest neighbour (KNN) and random forest (RF) classifiers.

Parameters			**Results**			
Image division	**k**	**# trees**	**ACA Train KNN**	**ACA Test KNN**	**ACA Train RF**	**ACA Test RF**
50/50	64	5	1	0,6273	0,957	0,5892
50/50	128	5	1	0,6598	0,9601	0,6521
50/50	128	15	1	0,6798	1	0,6695
50/50	64	5	1	0,5855	0,9525	0,561
30/70	128	5	1	0,6151	0,9645	0,5202
30/70	128	15	1	0,5794	0,9592	0,5922
30/70	64	5	1	0,6595	0,9791	0,5825
70/30	128	5	1	**0,7165**	0,9771	0,6309
70/30	128	15	1	0,6726	0,9989	**0,7455**

As shown in Table 1, $k = 128$ and the $70/30$ proportion of train/test images improved the performance of both classifiers. On the contrary, $\#trees = 15$ improved only the RF method.

Second of all, different values of k were tested, and the results are shown in Table 2.

Table 2. ACA obtained when varying k.

k	**ACA Train KNN**	**ACA Test KNN**
25	1	0,5262
50	1	0,6275
75	1	0,6216
100	1	0,6558
125	1	**0,7029**
150	1	0,675
175	1	0,6601
200	1	0,695
250	1	0,6854

,

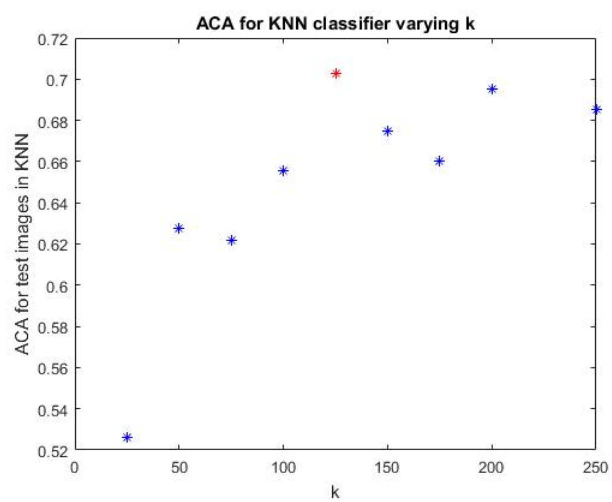


Figure 2. Values of ACA obtained when varying k.

It can be seen that the value of k that improves the performance of the classificator was 125.

3.1. KNN classifier

As mentioned before, for the Knn classifier was varied the method of measurement. So, the parameters K and proportion of images was set and the measure was changed to see which one was better, being cityblock the best method (see table 3).

Table 3. ACA obtained when varying the distance method in KNN.

Parameters			**Results**	
Image Proportion	**K**	**Distance**	**ACA Train Knn**	**ACA Test Knn**
50/50	128	Euclidean	1	0.6894
50/50	128	Euclidean	1	0.6424
50/50	128	Correlation	1	0.667
50/50	128	Cosine	1	0.6657
50/50	128	**Cityblock**	1	**0.7156**
50/50	128	**Cityblock**	1	**0.7232**
50/50	128	Correlation	1	0.6568
50/50	128	Minkowski	1	0.6711
50/50	128	Chebyshev	1	0.6212

3.2. Random forest classifier

In order to evaluate the random forest classifier, the number of trees was modified. So, like the evaluation of knn, the parameters K and proportion of images was set and the number of trees was changed to see which one was better. In this case, the best results were for a big number of trees, which was 300 and 500. However, using a similar amount of trees to the number of train images is overfitting. In consequence, the number of trees selected as the most adequate was 50, which still had one of the highest ACA(see table 4).

Table 4. Combination of parameters that improves the performance of both classifiers and their ACA.

Parameter			**Results**	
Image Proportion	**K**	**# Trees**	**ACA Train RF**	**ACA Test RF**
50/50	128	1	0.7919	0.4266
50/50	128	5	0.9694	0.5636
50/50	128	10	0.9981	0.6912
50/50	128	15	0.9962	0.7456
50/50	128	20	1	0.7392
50/50	128	**50**	1	**0.7915**
50/50	128	100	1	0.7495
50/50	128	**300**	1	**0.8045**
50/50	128	**500**	1	**0.8091**

,

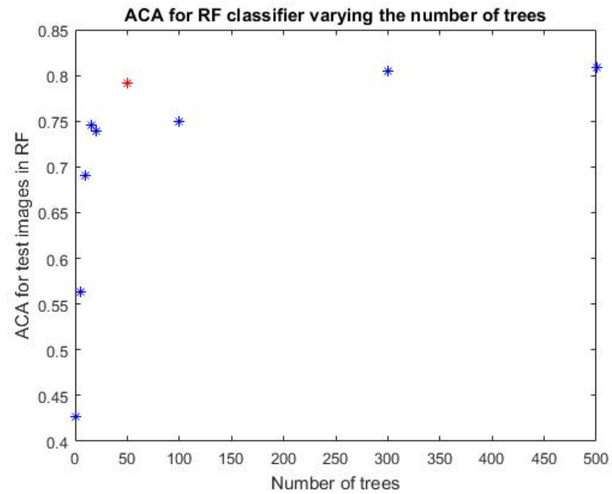


Figure 3. Values of ACA obtained when varying the number of trees.

3.3. Best Classificator

Once selected the best parameters, the program was runned again with those three times to evaluate its performance. As seen in table 5, the overall performance of both the knn and random forest methods was improved by combining them.

Table 5. My caption

Distance method	**# Trees**	**k**	**ACA Train KNN**	**ACA Test KNN**	**ACA Train RF**	**ACA Test RF**
Cityblock	50	125	1	0,753	1	0,7787
Cityblock	50	125	1	0,7334	1	0,8381
Cityblock	50	125	1	0,7513	1	0,8141

Finally, one of the main limitatons is that, as the texton dictionary is based on certain group of images, it is hard to find those train images that will be a good descriptor to classify all the test images.

4. Conclusions

According to the results, the random forest algorithm performed better than the knn algorithm once the number of trees used is greater than a certain value. However, when the amount of trees is too big, the training and testing time can be larger than KN

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.

[2] MathWorks. Texture analysis, 2017.

[3] L. van der Maaten and E. Postma. Texton-based texture classification. In *Proceedings of the Belgium-Netherlands Artifical Intelligence Conference*, 2007.

All the codes created and used are attached.