A weighted image reconstruction based on PCA for pedestrian detection

Guilherme V. Carvalho, Lailson B. Moraes, George D. C. Cavalcanti, Tsang I. R.
Centro de Informática (CIn)
Universidade Federal de Pernambuco (UFPE)
Recife, Pernambuco, Brasil
`http://cin.ufpe.br/~viisar`
`{gvc,lbm4,gdcc,tir}@cin.ufpe.br`

***Abstract*—Pedestrian detection is a task associated with security and surveillance systems. Its inherently complex nature makes it a hard challenge to develop a detection system. In this article we present an analysis of a pedestrian detection model based on PCA reconstruction errors. We find out that some components of the original classifier are not strictly necessary and can be eliminated to reduce the detection times. We then propose a weighted version of the classifier, that achieves a better classification accuracy.**

***Keywords*-pedestrian detection; pattern recognition; principal component analysis**

I. Introduction

Pedestrian detection systems have been widely used and developed throughout the computer vision history. Ranging from still image detection to automated car breaking systems, it is becoming an usual task in our lives.

In this paper we describe a pedestrian detection proposed by Malangón-Borja and Fuentes [1]. It classifies an image based on how much its reconstruction is different from the original one, that is, it classifies based on the image reconstruction error. We analyze how it works and how its original performance can be improved. The proposed improvements enhances the system's accuracy using weights on the detection errors and significantly reduces execution times by decreasing the number of calculations made by the algorithm.

This paper is divided as follows. Section 2 describes an image reconstruction technique using PCA. Section 3 describes how to use image reconstruction to classify images. Section 4 presents our experiments and discussions. Finally, in Section 5 conclusions are presented.

II. Image reconstruction with PCA

The idea of principal component analysis (PCA) seems to have been first proposed in 1901 [2] and was later developed in many works during the last century. It was popularized in computer vision by the eigenfaces method [3], a face recognition model still relevant and used in current days.

Given a set of samples, PCA yields a set of orthonormal vectors that can be used to linearly project the samples into a new space. This space maximizes the projected samples variance and minimizes their least mean square error, that is, it minimizes the difference between the projected sample and its reconstruction back to the original space [4]. This characteristic makes PCA suitable for data compression and dimensionality reduction. The data can come from many kinds of sources, but in this work we are concerned only with images.

The formulation of PCA is as follows. Consider a set of m images, each with r pixels of width and c pixels of height (i.e. of size $r \times c$ pixels). Each image I_i is represented by a column vector v_i of length rc. The set mean μ and the covariance matrix C are given by

$$\mu = \frac{1}{m}\sum_{i=1}^{m} v_i \tag{1}$$

$$C = \sum_{i=1}^{m} (v_i - \mu)(v_i - \mu)^T. \tag{2}$$

PCA applies eigen-decomposition on C and keeps the k eigenvectors ($1 \leq k \leq rc$) corresponding to the k largest eigenvalues. These eigenvectors are called the *principal components* (PCs) of C. It is proven that a projection onto the space defined by these eigenvectors provides the optimal reduced representation of the data, minimizing information loss [4].

Let P be the matrix whose columns are the first k principal components of C. The projection p of an image u into this eigenspace is given by

$$p = P^T(u - \mu). \tag{3}$$

As the projection is reversible, we can recover the original version of a previously projected image. The *reconstructed image* u' of a projection p is

$$u' = Pp + \mu = PP^T(u - \mu) + \mu. \tag{4}$$

This process can be interpreted as a form of lossy compression and decompression: the reconstructed image is generally not equal to the original one, being just an approximation of it. For this reason, each reconstruction has an associated *reconstruction error*, which can be measured by

$$d = |u - u'| = \sqrt{\sum (u_i - u_i')^2}. \qquad (5)$$

The more principal components we use to obtain a projection, the less information loss we have, what allows a more accurate reconstruction. Also, the more similar u is to the images that were used to produce P, the better the reconstruction should be for a fixed number of eigenvectors.

III. Classification using reconstruction

By definition, PCA looks for the set of PCs that best describe the distribution of the data being analyzed. Therefore, these PCs are going to preserve better the information of the images from which PCA was performed, or of those that are similar to them [1]. For example, consider a set of PCs obtained only from images that contains pedestrians. They must reconstruct better images of other pedestrians than any other type of images. Conversely, if we have a set of PCs obtained from images of anything except pedestrians, the reconstruction of pedestrian images must be poor.

Based on this fact, Malag'on-Borja and Fuentes [1] proposed a classifier for pedestrian detection that uses reconstruction errors as the classification criteria. In addition to the grayscale images, the classifier also uses the corresponding computed edge images, since they characterize better the pedestrian silhouette and eliminate background and clothing variations.

In the training phase, PCA must be performed separately for four sets of images, what results in the following sets of PCs:

- The principal components P_{gp} and the mean μ_{gp} from the set of pedestrian grayscale images;
- The principal components P_{ep} and the mean μ_{ep} from the set of pedestrian edge images;
- The principal components P_{gn} and the mean μ_{gn} from the set of non-pedestrian grayscale images;
- The principal components P_{en} and the mean μ_{en} from the set of non-pedestrian edge images.

In the classification phase, when a new grayscale image has to be classified, the first step is to obtain its edge image and perform four reconstructions: one for each set of PCs. Then, the reconstruction errors must be calculated and combined to produce the total reconstruction error, that is the final classification score. If the total error is greater than or equal to zero, the image is classified as a pedestrian; otherwise, it is classifier as a non-pedestrian. The detailed procedure is described below.

1) Obtain the edge image e from grayscale image g using the Sobel operator;
2) Perform four image reconstructions:
   a) $r_{gp} = P_{gp}P_{gp}^T(g - u) + \mu_{gp}$
   b) $r_{ep} = P_{ep}P_{ep}^T(e - u) + \mu_{ep}$
   c) $r_{gn} = P_{gn}P_{gn}^T(g - u) + \mu_{gn}$
   d) $r_{en} = P_{en}P_{en}^T(e - u) + \mu_{en}$
3) Calculate the corresponding reconstruction errors:
   a) $d_{gp} = |r_{gp} - g|$
   b) $d_{ep} = |r_{ep} - e|$
   c) $d_{gn} = |r_{gp} - g|$
   d) $d_{en} = |r_{en} - e|$
4) Compute the *total reconstruction error*, given by

$$d_t = d_{gn} + d_{en} - d_{gp} - d_{ep} \qquad (6)$$

5) Classify the image according to:

$$\text{class}(g) = \begin{cases} \text{Pedestrian}, & d_t \geq 0 \\ \text{Non-pedestrian}, & d_t < 0 \end{cases} \qquad (7)$$

When we analyze how this classifier works, it is easy to understand the role of the PCs generated by the positive samples. As the samples contains the recurrent pattern of the target object (pedestrians in this case), PCA is able to capture it and better reconstruct similar images, generating smaller reconstruction errors. On the other hand, the role of negative PCs is not so clear. Since the negative images contain just random objects and does not represent a specific concept, what pattern is there to capture? We initially thought that the errors of reconstructions obtained from the negative PCs had a negligible importance and therefore it would be possible to eliminate them without impacting the classification accuracy. However, our experiments show that is not the case: it turns out that these errors have a major importance on classification.

We found that the value of the negative PCs is not on the pattern PCA captures, but instead it lies on what pattern it does *not* capture. In this way, the negative reconstruction errors tend to be high for images that contains the target object while for the same images the positive reconstruction errors tend to be small, contributing to a high total error, what classifies the image correctly. For images that does not contains the target object, it is exactly the opposite. Thus, both components of total error (from positive PCs and from negative PCs) work together for classification and that is why it is not worth to construct a classifier that uses only positive reconstruction errors.

However, this does not mean that all the four components have the same relevance to classification. In the reconstruction error classifier, each component contributes equally to the total error, but our experiments show that this relation is not true. Some components have more importance than others and therefore have to be adjusted accordingly. Thus, we propose the *weighted total reconstruction error*, given by

$$d_w = w_{gn}d_{gn} + w_{en}d_{en} - w_{gp}d_{gp} - w_{ep}d_{ep} \qquad (8)$$

where w_{gn}, w_{en}, w_{gp}, w_{ep} are weights that adjust the relevance of each error to the classification (more relevant errors should have larger weights). They can be set manually or be determined by some optimization algorithm in relation to the training set.

It is also worth noting that the classifier can be made more flexible if we introduce a threshold parameter and compare the classification score (i.e. the total error) agains it, instead of keep it always fixed to zero (as shown in eq. 7). In this way, the users can adjust the specificity, depending on their particular application.

IV. Experiments

The classifier was tested in the context of the pedestrian detection task, that is, given an unlabeled image, the classifier must determine if it depicts or not a pedestrian. To train the classifier, we need a set of pedestrians images and a set of non-pedestrian images. This training phase consists of computing image edges and applying PCA to the four image sets separately, what produces four sets of PCs. They are used to perform the reconstructions on classification time.

The pedestrian images were obtained from the MIT CBCL pedestrian database [5]. It contains 924 color images of pedestrians in frontal or rear views, under different scene conditions. These images were converted to grayscale and had part of the background cropped on the border. The final samples had 45×105 pixels. This reduces the non-relevant background variation among the images.

The non-pedestrian images were extracted from 120 assorted pictures that did not contain pedestrians. Each one was chopped into slices of 45×105 pixels and 5.000 slices were randomly chosen to compose the negative dataset. This dataset is more than four times larger than the positive dataset; that is related with the representation capacity of each one. The object detection problem is naturally asymmetric [6], since the positive examples represents a specific type of object while the negative examples represent the "rest of the world". That is why we need more negative examples than positive ones.

To train the classifier, we selected 75% of the pedestrian images (693 samples) and the remaining images (231 samples) were used for testing. From the non-pedestrian images, we selected 80% of the images (4.000 samples) for training the classifier and the remaining images (1.000 samples) were used for testing. This particular scheme was chosen in an empirical way; it yielded the best results among the distributions we tested. The edge images are computed with x and y Sobel filters [7], which we combine to generate a single edge image. To implement the classifier and the auxiliary procedures, we used the MATLAB 2009b environment. The resulting code is available for download at http://github.com/lailsonbm/pca_reconstruction/.

We start our analysis by investigating the impact of the reconstruction errors from the positive PCs and from the negative PCs to the classification. As discussed before, we initially questioned the role of negative PCs, since they are computed from completely uncorrelated images and do not represent a particular object. However, when we classify test samples using reconstruction errors only from the positive PCs or only from the negative PCs (but still using grayscale and edge images), it becomes clear that neither one can perform a satisfactory classification; both classifiers yield a poor result. But when the errors are combined accordingly to the total reconstruction error, we have a good accuracy. The receiver operating characteristic (ROC) curves of the three classifiers are shown in Figure 1 for 100 PCs, i.e., $k = 100$. We get a similar pattern for all values of k that were tested.

To understand why this happens, it is interesting to see how the reconstruction errors are distributed. In Figure 2, we plot the reconstruction error of each individual test sample in the three situations previously described. It is possible to see that the samples are mixed on the two first plots, indicating that these criteria do not have discriminatory power to correctly separate the samples. Conversely, when the errors are combined, the distribution changes completely and we can notice that samples from different classes are in fact grouped, what makes possible the high classification accuracy we got. Hence, the reconstruction errors from both positive and negative PCs contribute in a significant way to the classification and none of them can be removed.

On the other hand, we get very similar ROC curves when the classifier considers the reconstruction errors from grayscale PCs and from edge PCs separately (now always using the errors from positive PCs and from negative PCs

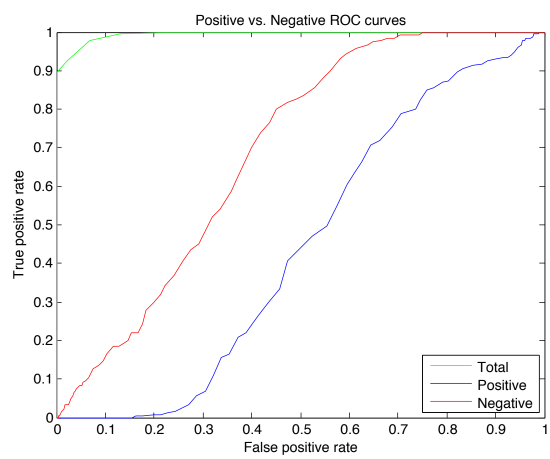

Figure 1. Comparison of positive and negative ROC curves. From left to right: the first curve (green) refers to a classifier that uses the total reconstruction error (eq. 6) and has an AUC of 0.995; the second curve (red) refers to a classifier that uses only the reconstruction errors from negative PCs (d_{gn} and d_{en}) and has an AUC of 0.690; and the third curve (blue) refers to a classifier that uses only the errors from positive PCs (d_{gp} and d_{ep}) and has an AUC of 0.444. On the three classifiers, only the first 100 principal components are being used, that is, $k = 100$. Both positive and negative classifiers yields a poor result, far from the full classifier performance. It is surprising to see that the negative classifier even got a better result than the positive one.

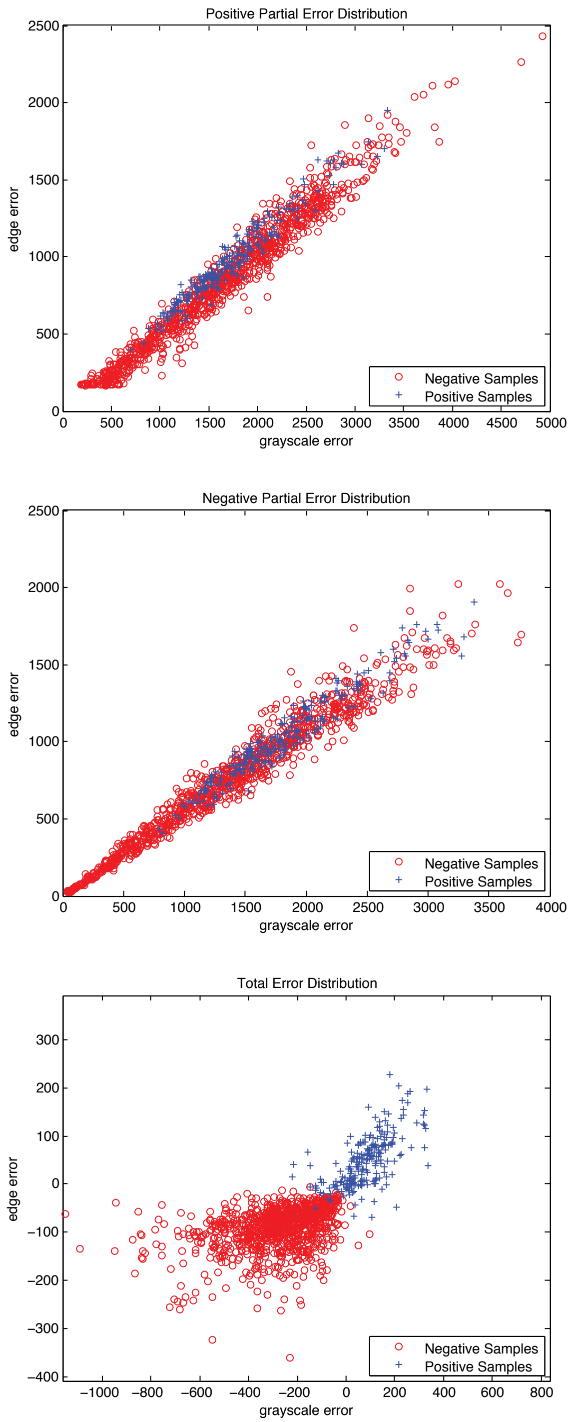

Figure 2. Comparison of reconstruction errors distributions. The first chart plots the errors of test samples when only the reconstructions from positive PCs are being considered. The second chart plots the errors only for the negative reconstructions. And the third chart plots the distribution of the total reconstruction error. Note that the errors are mixed on the first two charts; classify the samples based on them is a hard task. On the other hand, the third chart show well defined clusters of errors, what makes possible a good classification. This explains the curves of Figure 1.

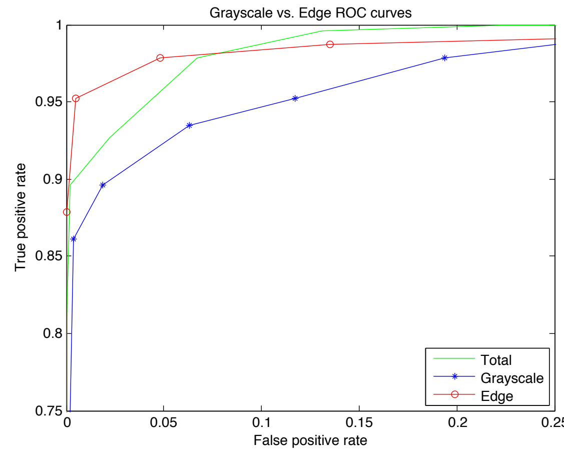

Figure 3. Comparison of grayscale and edge ROC curves. The green curve refers to a classifier that uses the total reconstruction error (eq. 6) and has an AUC of 0.995; the red curve refers to a classifier that uses only the reconstruction errors from edge PCs (d_{ep} and d_{en}) and has an AUC of 0.994; and the blue curve uses only the errors from grayscale PCs (d_{gp} and d_{gn}) and has an AUC of 0.985. On the three classifiers, only the first 100 principal components are being used, that is, $k = 100$. Note that the scale is different of that from Figure 1; we approximated the upper left corner to better show the small difference between the curves.

together). This in shown in Figure 3, which exhibit three ROC curves: one from the classifier that uses only grayscale errors, one from the classifier that uses only edge errors and one from the classifier that considers both (i.e. the total reconstruction error). Note that the ROC curves from the edge error classifier and from the total error classifier are nearly indistinguishable. In fact, their area under the curve (AUC) are practically equal.

Again, we plot the reconstruction error distributions, but now segregating errors from grayscale PCs and from edges PCs. These plots are shown in Figure 4, that also displays a plot of the total reconstruction error. Now, we get a different scenario. All the three plots exhibit a clear boundary that separate samples from different classes. This indicate that there is a lot of redundancy between errors from grayscale PCs and from edge PCs and one of them can be safely removed from the classifier with just a minor impact on the classification accuracy. Our results indicate that the errors from grayscale PCs generally yields a slightly worse result, as shown in the Table I. Therefore, it is possible to compose a classifier only with the edge errors and speed up the classification significantly (since two reconstructions will not have to be computed anymore). In general, such classifier drops the classification accuracy in less than 1% when compared with a classifier that uses the total error. Actually, the classifier based only on the edge errors sometimes performs even better than the full classifier in our experiments (when $k >= 200$), as also shown in Table I.

These findings point out that, among the four reconstruction errors, some are more relevant to the classification than

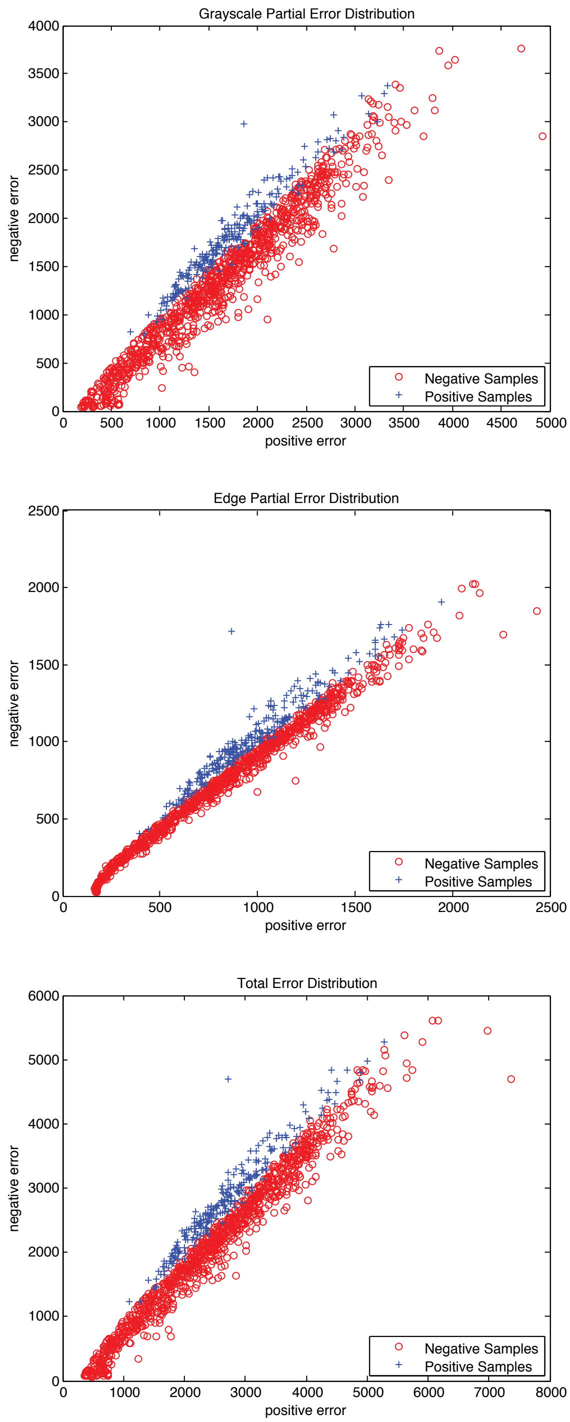

Figure 4. Comparison of reconstruction errors distributions. The first chart plots the errors of positive and negative test samples when only the reconstructions from grayscale PCs are being considered. The second chart plots the errors only for the edge reconstructions. And the third chart plots the distribution of the total reconstruction error. Note that the three charts show a clear boundary between errors from positive and from negative samples.

Table I
AUCs FOR GRAYSCALE, EDGE AND TOTAL ERRORS CLASSIFIERS

k	Gray	Edge	Total
400	0.9611	0.9863	0.9744
300	0.9694	0.9877	0.9826
200	0.9768	0.9905	0.9877
100	0.9851	0.9944	0.9948
50	0.9905	0.9956	0.9972
25	0.9929	0.9913	0.9984

others. However, on the total reconstruction error (eq. 6) each error contributes equally. Thus, we use the weighted total reconstruction error (eq. 8) to improve the classification accuracy. The problem with using weights is how to choose them. We can try some values and make adjustments manually, test after test. But we can also employ an optimization algorithm that looks for the values in an automated way. In this work, we use a genetic algorithm [8]. It initially chooses a set of random values and uses operations as crossover and mutation to improve them during many iterations. The values are adjusted in relation to a fitness function. In our case, this functions aims to correctly classify the largest number of training samples using a given set of weights. The function can be fine-tuned in order to give more importance in finding more pedestrians despite the false detections or to minimize the number of non-pedestrians classified as pedestrians.

As this algorithm performs a local optimization and therefore is not optimum, we runned it five times for each value of k. We then have chosen the set of weights that gets the best classification rates among these five. Finally, we classify the test samples considering the weighted total reconstruction error. The produced ROC curves always have a larger AUC than the corresponding versions without weights for all values of k we tested. This is shown in Figure 5. The difference is more pronounced for large values of k and becomes small when k diminishes. Still, the AUC of the weighted classifier is always larger, what indicates that the weighted error is a better classification criteria than the non-weighted one. Finally, we plotted the distribution of the weighted total reconstruction error, that can be seen in Figure 6. This time, we get a clear boundary between errors from samples of different classes.

V. Conclusion

In this work we analyzed a classifier that uses PCA image reconstruction for pedestrian detection. It uses four sets of principal components (PCs): two originated from grayscale and edge images that contains pedestrians, respectively, and two originated from grayscale and edge images that did not contain pedestrians, respectively. To classify an image, it uses these sets of PCs to perform four image reconstructions and compares the respective reconstruction errors to determine if the given image depicts or not a pedestrian.

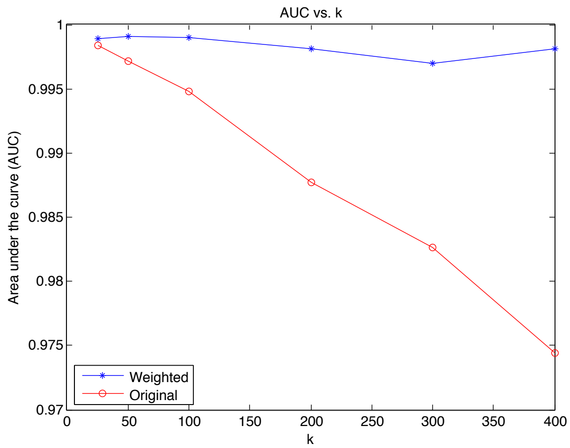

Figure 5. Comparison of weighted and original classifiers AUCs for different values of k. The AUCs of the weighted classifier are more stable when k varies.

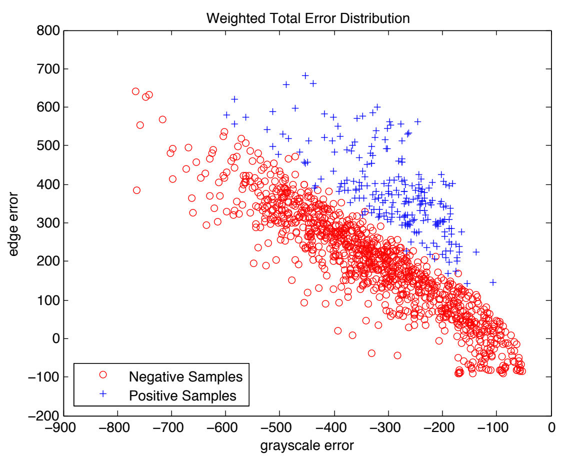

Figure 6. Distribution of the weighted total reconstruction errors for positive and for negative training samples, for $k = 100$. Compare it with the third chart of Figure 2. Here the boundary between errors from positive and from negative samples is much clearer.

To understand how the individual errors contribute to the overall performance, we compared two classifiers: one that uses only the reconstructions from the positive PCs and other that uses only the reconstructions from the negative PCs. In our experiments, both performed poorly when compared to the full classifier, indicating that positives and negatives reconstructions are indeed relevant to the classification. The crucial role of the reconstructions from negative PCs is particularly surprising, since non-pedestrian samples are not related and do not contain a specific pattern. Therefore, we conclude that the importance of the negative reconstructions is not on what they characterize, but instead on what they do not characterize. In this way, the negative reconstruction errors contribute to raise the total reconstruction error when the image does not contain a pedestrian, what makes possible a better classification.

We also investigated the relevance of grayscale and edge errors. In other experiment we compared the accuracy of a classifier that uses only grayscale reconstructions and other classifier that uses only edge reconstructions. This time, they performed very well, with a performance close to that of the total error classifier. This suggest that there is a lot of redundancy between grayscale and edge information. The classifier based only on edge images got better results, indicating that edge information is more relevant to this classifier. In fact, the edge classifier performed better than the total error classifier itself in some cases and, when the results were worse, the accuracy drop was less than 1%. For this reason, it is possible to speed up the classification time in a significant way by using only edge information and keep the classification accuracy virtually unchanged.

Since our experiments show that each error has a different relevance for classification, we propose a classifier that uses a weighted version of the total reconstruction error. The problem with weights, though, is how to find the best values. To get an approximation in an automated way, we used a genetic algorithm with a fitness function that aims to achieve the best classification rates for the training samples. This classified performed better than the original version in our experiments for all numbers of PCs we tested.

REFERENCES

[1] L. Malagón-Borja and O. Fuentes, "Object detection using image reconstruction with PCA," *Image and Vision Computing*, vol. 27, no. 1-2, pp. 2–9, 2009.

[2] L. Sirovich and M. Kirby, "Low-dimensional procedure for the characterization of human faces," *Journal of the Optical Society of America A*, vol. 4, no. 3, pp. 519–524, 1987.

[3] M. Turk and A. Pentland, "Eigenfaces for recognition," *Journal of cognitive neuroscience*, vol. 3, no. 1, pp. 71–86, 1991.

[4] J. Shlens, "A Tutorial on Principal Component Analysis," *Systems Neurobiology Laboratory*, 2009.

[5] Center for Biological and Computional Learning, "CBCL Pedestrian Database #1," 2000. [Online]. Available: http://cbcl.mit.edu/software-datasets/PedestrianData.html

[6] X. Jiang, "Asymmetric principal component and discriminant analyses for pattern classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 5, pp. 931–937, 2009.

[7] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*. Boston, MA, USA: Prentice Hall, 3rd edition, 2007.

[8] D. Goldberg, "Genetic Algorithms in Search Optimization and Machine Learning, 0201157675," 1989.