Assignment 3: Face Recognition

Daniele Giannuzzi r0876755

1 Face recognition setting

Face recognition is a well known biometric system, easily implemented and deployed, also for the smallest devices, since you just need a frontal camera on those. The purpose of the work is implementing a facial recognition system, evaluating the performances, criticalities and strengths of various strategies of face features extraction.

1.1 Dataset

The dataset we use is the $CALTECH\ Faces$ dataset, composed of 450 images of 27 unique people, with different conditions of lighting, background and face expression. Each image is 896 pixels wide and 592 pixels high. Nevertheless we are concretely working with 445 images e 26 classes, since 5 images of the dataset are wrongly labeled. In Figure 1 we can observe the grey images of the first person present in the dataset. They are converted in the greyscale since the HAAR-cascade detector (explained in the next paragraph), requires it to work properly.

Figure 1: Raw face images of one specific person from CALTECH Faces dataset

2 Face detection

In the context of face-recognition, face detection is a tool to extract from the images only the faces, in order to focus only on the relevant part of the image. This can be done quite easily by loading a pretrained HAAR cascade classifier model to recognize faces. It is a computational efficient algorithm comprised of weak learners that looks for objects, in this case a face, in a part of the image. An accurate classifier is then obtained by taking a weighted average of the decisions (face in the sub-window or not) made by the weak learners. In Figure 2 we can see the result of the face detection process applied on the same images we reported in Figure 1. As we can observe all the faces of this specific person are correctly detected, even if the light conditions for some of the images make the task tricky.

Figure 2: Cropped face images after HAAR face detection

However, this does not mean that HAAR-cascade detector is foolproof. On the contrary, despite various trials in a proper tuning of the hyperparameters, it showed some problem in:

- detecting only one face: sometimes when there is for instance some posters in the background, it detects also the faces in the posters. Therefore, it can happens that we have an extra face deriving from an image,

with a wrong associated label. One solution to mitigate this problem was increasing the $minSize$ parameter and $minNeighbors$

- detecting at least one face: poor light conditions does not always allow to recognize all the faces in the dataset, especially if we need to increase the $minNeighbors$ parameter for the problem of before.

Because of these problems, the detector recognized only 434 faces, of which 3 are outliers in the sense that are detected from the background of the image. The hyperparameters used to have these results, which seemed to be the better ones, are: $scaleFactor = 1.1$, $minNeighbors = 13$, $minSize = (120, 120)$.

3 Feature extraction

Before the advent of Deep Learning for computer vision tasks, many useful strategies to extract features for face recognition were proposed. Here we briefly describe some of them along with the deep metric learning.

3.1 PCA: Principal Components Analysis (Eigenfaces)

Principal Component Analysis (PCA) is an unsupervised technique used for dimensionality reduction. Eigenvectors and eigenvalues are computed from the covariance matrix to determine the principal components of the data. The greater the eigenvalue, the more useful the principal component vector. Therefore, each face vector can be represented as the linear of combination of the best K eigenvectors. These are called eigenfaces. The dimensionality into which data are projected depends on how many eigenfaces we retain. The number of retained eigenvectors can be manually selected. In this case the number of components is set to 35. For the K eigenfaces, we can find K dot-product for any given test face picture. We can present the result as weights of this face picture with respect to the eigenfaces.

3.2 LDA: Linear Discriminant Analysis (Fisherfaces)

When instead of using PCA, we use the Linear Discriminant Analysis (LDA) to find the subspace representation of a set of face images, we call the resulting basis vectors defining that space "Fisherfaces". LDA is very similar to PCA because both try to find the best combination of features that can explain the data. Nevertheless, while the LDA is based on maximize the separability between the groups, PCA focuses on capturing the direction of maximum variation in the dataset. As a result Fisherfaces are robust against lighting conditions, while Eigenfaces do not. The number of components used by LDA are in this case 25 (i.e. number of classes - 1).

3.3 LBP: Local Binary Pattern

Local Binary Patterns (LBP), are a texture descriptor The first step in constructing a LBP is to take the 8 pixel neighborhood surrounding a center pixel and threshold it to construct a set of 8 binary digits. We can then convert this binary string of length 8, to a decimal value, which is the value of that pixel in the LBP codes representation. This process is repeated for each pixel of the image. Since we have 8 binary digits string, we have also $2^8 = 256$ possible decimal values for each pixel. Therefore, we can construct a 256-bin histogram of LBP codes as our final image feature vector. Since using 8 neighbors for each pixels, leads to capture only fine-grained details (which is the main drawback of the algorithm), extensions to this method were proposed to extend the range of neighbors considered. In the implementation only the original version of the algorithm has been considered.

3.4 Deep metric learning

Deep learning can also be used to get feature vectors from the images. With metric learning based methods, we train the network, then taking a projection of the input data on a low-dimensional vector (embedding).

As neural network we use the Siamese model, that is useful in such cases when one does not have many data points for each class. Siamese networks is made of two symmetrical neural networks both sharing the same weights and architecture and both joined together at the end using an energy function. The objective of the Siamese model is to learn whether two input values are similar or dissimilar.

4 Distance-based scoring

Since Biometrics is based on generating pairwise matching scores, we need to calculate these scores which can easily be computed from the feature vectors obtained with the four strategies previously described. We just need to apply a simple distance metric between the feature vectors such as the euclidean or the chi-square distance. Specifically, the euclidean distance has been applied for the PCA, LDA and DL strategies, while the chi-square distance for the LBP feature extraction strategy. In particular, the Chi-square distance is one of the distance measures that can be used as a measure of dissimilarity between two histograms.

5 Evaluation

To evaluate our 4 systems, we first need to convert our distances to similarities matrices. To do this, we convert our distances in similarities using this formula: $s = \frac{1}{1+d}$, then normalizing the scores between 0 and 1.

5.1 Validation as verification system

Given the similarity matrix, before evaluating our systems with different performance metrics, we just need to use the scores we have and a threshold which varies between 0 and 1 (because the scores were normalized) to classify our face images. The samples with the similarity score greater than the threshold are classified as Genuines(1), while if the contrary happens they are considered Impostors(0).

5.1.1 Accuracy and F1-score

The first performance metrics we consider are Accuracy and F1-score. As one can observe from Figure 3 only the systems based on LDA and DL feature extraction reach values near to 1 for both the metrics. Especially on F1-score, PCA and LBP strategies perform significantly worse.

Even if we can imagine we can take the threshold of the F1-score maximum value for each strategy as the optimal operational point, actually it really depends on the use we have to make of the system. If we are building a system for forensic purposes, we can set a threshold that is greater than the one for which the F1-score reach the maximum, since we could want to recognize all the possible impostors. Vice versa, if we are a company accessible by employees via the biometric system, we could want to recognize all the employees (genuines), therefore setting the threshold a bit lower.

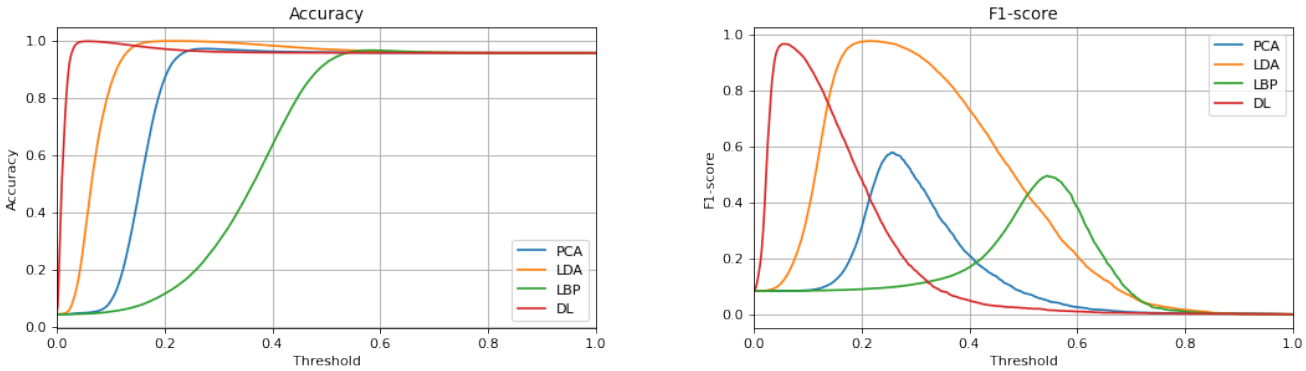

Figure 3: Accuracy and F1-score for the four strategies of feature extraction

5.1.2 Genuine and Impostor scores distribution

Analyzing the Genuine and Impostor scores distribution reported in Figure 4, we can clearly observe that the best performing systems are those based on LDA and DL feature extraction, since we can notice a small overlapping the the two distributions, while the contrary happens for the PCA and LBP feature extraction. We can suppose that for these strategies also others performance metrics we are going to analyze will show bad results.

5.1.3 ROC curve, DET curve and EER point

We analyze in Figure 5 the behaviour of the strategies on ROC curve expressed with a logarithmic scale for the FPR, to better notice the difference of the DL and LDA curves for low threshold values and the behaviour (which is the same since the DET curve is the ROC mirrored on the y-axis) on the DET curve for which also the Equal Error Rate (EER) points are shown. EER is the operating point where $fpr = fnr$. We can see how the LDA and DL curve are very similar with an appreciable performance also for low fpr values. Specifically, the DL method seems to outperform the one based on LDA until a threshold value of $1x10^{-}4$. From that value onwards, the LDA becomes the best strategy. The same can not be said for the other two curves, which are very poor in performance, even if the PCA is better than LBP until the threshold value of 0.1, when they reverse their roles.

From the Table 1 we can also observe how the EER point is very low for LDA and DL in order and very high for PCA and LBP. Even if it is not the most interesting operational point, it is quite clear also from this single number how the two strategies are better than the others.

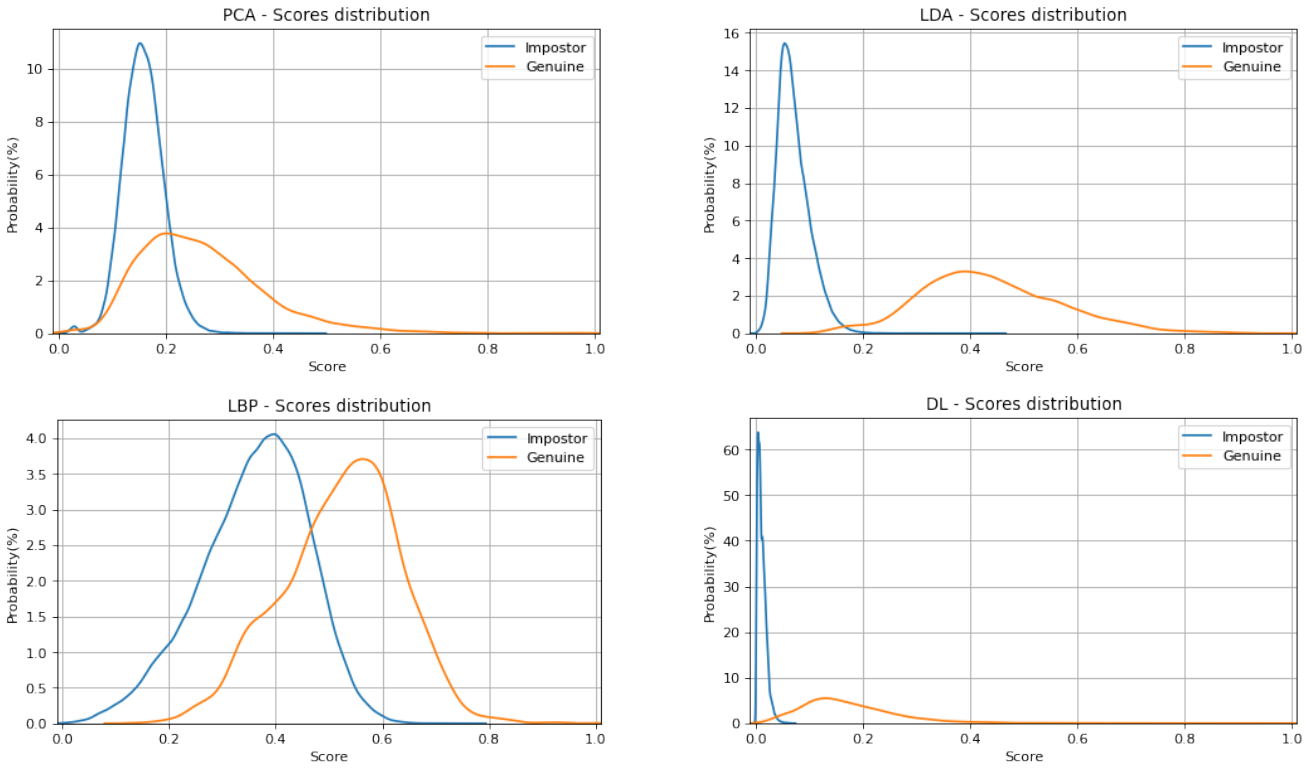

Figure 4: Genuine and Impostor score distributions for PCA, LDA, LBP, DL methods of feature extraction

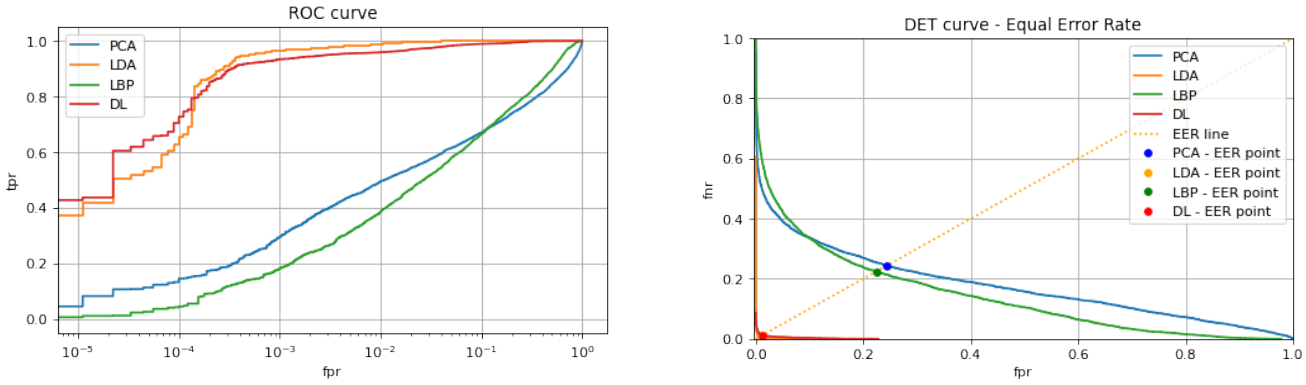

Figure 5: ROC curve and DET curve with EER point for the four strategies of feature extraction

Facial representation	Equal Error Rate
PCA	0.243
LDA	0.010
LBP	0.224
LD	0.026

Table 1: EER values for the four different facial representation strategies

5.2 Precision-Recall curve

The Precision-Recall curve in Figure 6 shows how LDA is performing a little bit better than DL strategy for high values of Recall. While PCA and LBP curve are completely overlapped, showing as before not so good performances.

In Table 2 are the Area Under the Precision Recall curve values for each strategy that confirms the statements we did previously, and its approximation that is the Average Precision, which shows different values for LBP that is considered the worst strategy by this approximation.

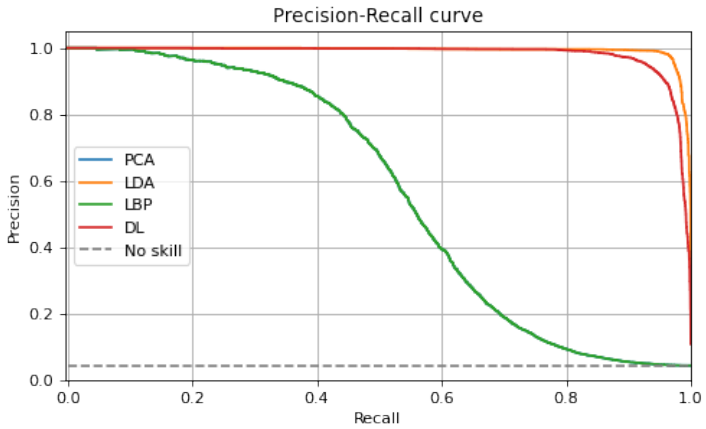

Figure 6: Precision-Recall curve for the four different strategies of feature extraction

Facial representation	Area Under PR curve	Average Precision
PCA	0.566	0.566
LDA	0.992	0.992
LBP	0.566	0.488
DL	0.975	0.975

Table 2: Area Under the Precision Recall curve values for the four strategies

5.3 Validation as identification system

For the identification scenario, we focus on the Cumulative Matching Curve, presented in Figure 7. We can see how in this scenario the PCA can be surely considered the worse performing, while the best performing for the first percentage of ranks is the DL-based curve. Indeed only for a small percentage range of ranks the LDA-based one seems to be better. While, increasing the number of considered individuals after the first 10% of the rank, DL and LDA based strategies perform equally good.

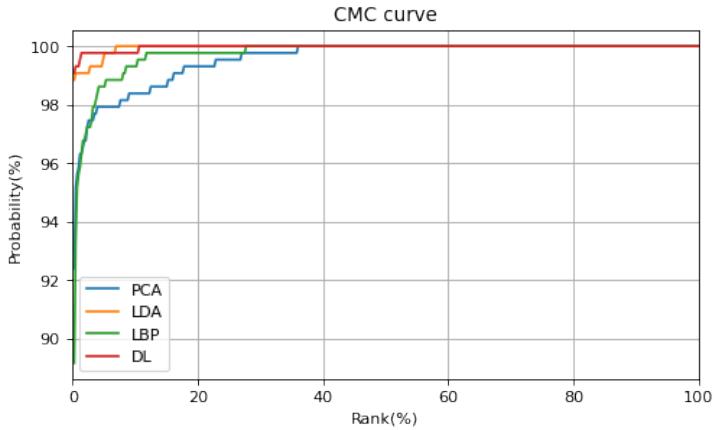

Figure 7: Cumulative Matching Curve for the four strategies of feature extraction

In case of Rank-1 identification scenario, as one can observe from Table 3, the DL-based strategy outperforms the others, being even better than the LDA-based one, while the LBP-based is clearly the worst.

Facial representation	Rank-1 recognition rate
PCA	0.923
LDA	0.988
LBP	0.891
DL	0.990

Table 3: Rank-1 recognition rate for the four strategies

5.3.1 Final consideration

For the verification scenario we could expect these results since the PCA-based strategy is sensitive to different light conditions, while the LDA is robust to that. Moreover, the LBP original strategy shows its drawbacks, being acceptable against the others only in the identification scenario if we consider at least the first 10% individuals in the rank.

6 Tasks of choice

Optional tasks can be found at the end of the jupyter notebook.

6.1 Task 3: classification-based scoring

In this task we want to build the similarity matrix, using the scores (i.e. normalized probabilities) produced by a classifier on the face feature vectors, rather than using a distance metric on those. To do that, we first leave one image out for each class to create the test set (made of n=26 samples). We then train the classifier of choice on the images feature vectors and produce the probabilities on the test set. The classifier chosen for this task is a SVM with a rbf kernel, since the mapping of the feature vectors is non-linear. While the feature representations chosen are the LBPs since it is not the best strategy and the performances can be improved. Finally we build the similarity matrix [nxn], in which each row corresponds to a test image and each column corresponds to an individual in the dataset.

Now we can evaluate this system by using the obtained similarity matrix in both validation and identification scenarios. We have chosen to compare it with the distance-based (i.e. Chi-squared based) version of LBP strategy, even if the similarity matrix has a different shape, just to get an idea if the SVM-based scoring improves the performances. The problems of comparing strategies using different similarity matrices, will be further explained in the next paragraph, where for the identification scenario problems arise.

6.1.1 Validation as verification system

In Figure 8 we can appreciate how the performance with the SVM-based scoring remains good for a higher range of threshold values, with respect to the distance-based strategy for which for example the F1-score reach the maximum for a smaller thresholds range.

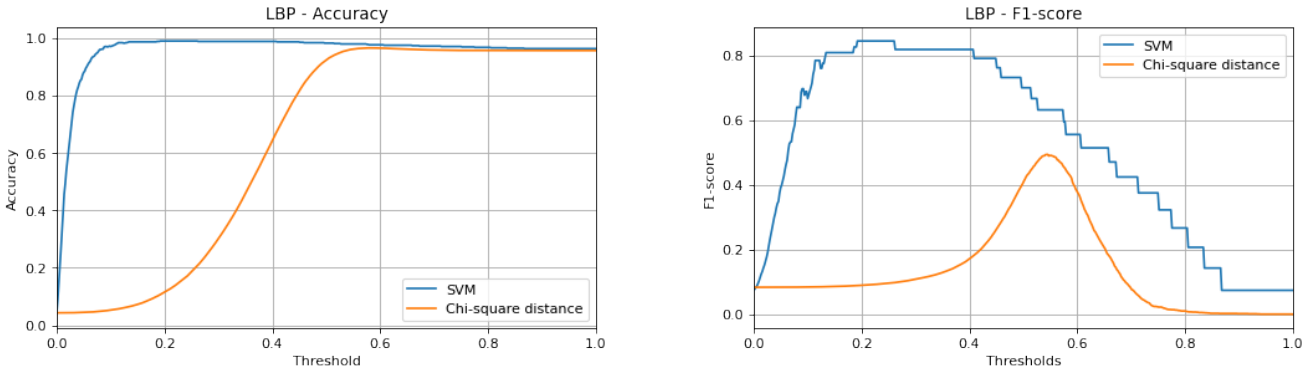

Figure 8: Accuracy and F1-score for LBP feature extraction using SVM-based scoring method and Chi-square distance

The distribution of scores can be observed in Figure 9. There is clearly less overlap than before between the Genuine and Impostor distributions, reinforcing the idea that using SVM-based scores can be beneficial in this case to get higher performance.

In Figure 10 we can see the ROC curve on the left and the DET curve on the right showing also the EER points. Also in this case the SVM-based system is outperforming the distance-based one. The value of the EER point is 0.106, much better with respect to the previous value of 0.224.

The Precision-Recall curve in Figure 11 makes the earlier claims even stronger. The SVM-based system outperforms the distance-based one for every Recall value. We can notice that the SVM curve is less smoothed than the one of before. This happened for the ROC, DET and F1-curve as well and it is simply due to the smaller similarity matrix that we are forced to use in this case.

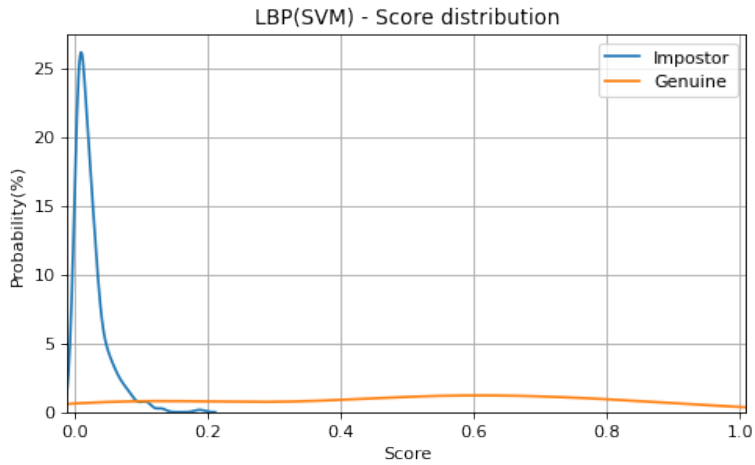

Figure 9: Genuine and Impostor distribution for LBP method based on SVM-scores

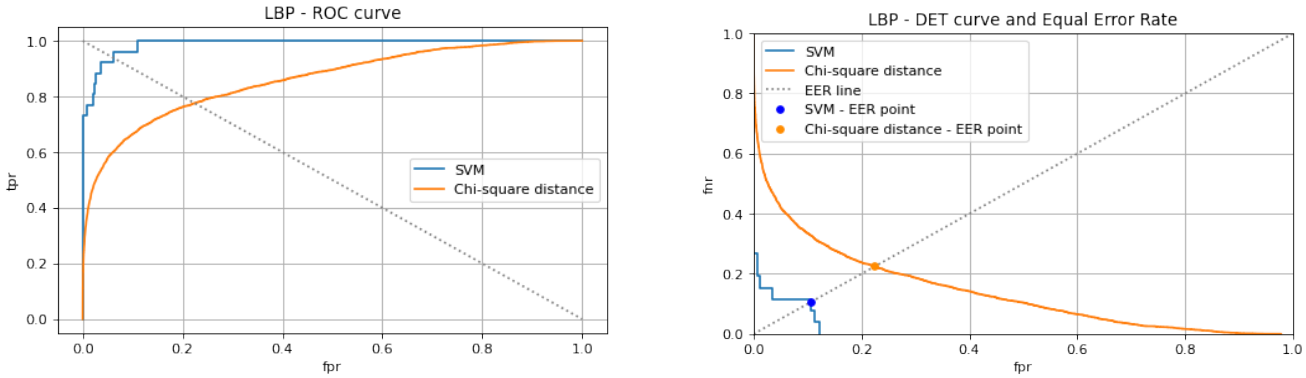

Figure 10: ROC and DET curve with EER point for LBP feature extraction using SVM-based and Chi-square distance scoring method

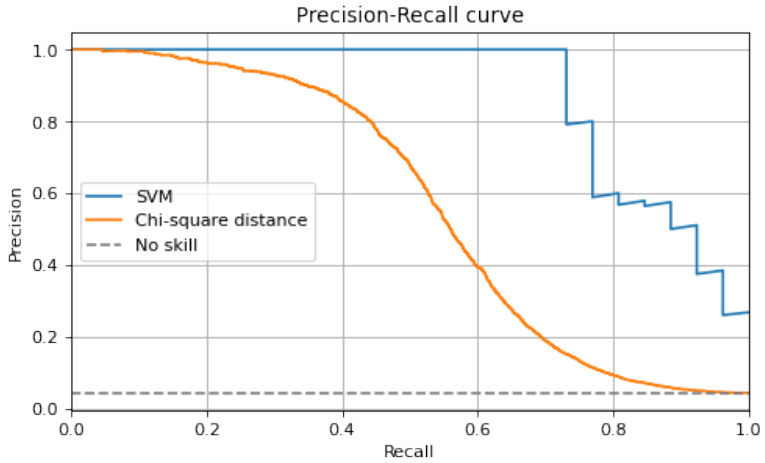

Figure 11: Precision-Recall curve for LBP feature extraction using SVM-based and Chi-square distance scoring method

6.1.2 Validation as identification system

For the identification scenario we prefer not to make a comparison between the curves based on classifier-based and distance-based scoring. Indeed, here we have a similarity matrix with shape [26x26], while for the distance-based system we use a similarity matrix with dimensions [434x434] that is bigger, leading to have 434 ranks instead of 26. Moreover, in the SVM-based system we have the similarity matrix with one class for each column, while in the other case we have with a number of columns for each individual that is not balanced among the different classes. The unbalanced distribution of the classes can be seen in Figure 12. Therefore even using a rank expressed in percentage, could lead to several problems of comparison.

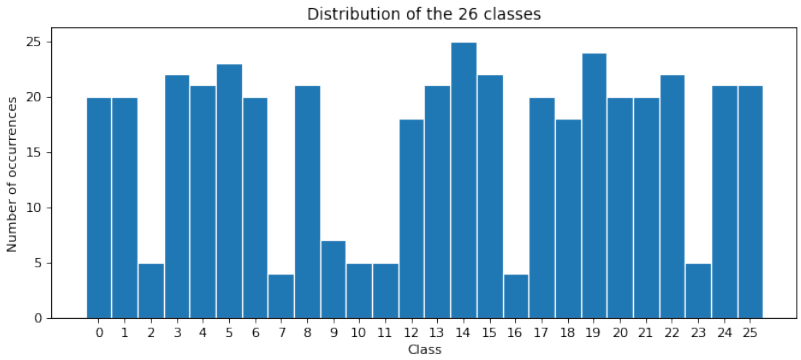

Figure 12: Distribution of the 26 classes of the dataset

In Figure 13 is reported the CMC curve for the SVM scoring based strategy. One can observe how the performance for the Rank-1 is not high, being 0.731. However looking for the identity in rank-9 best scoring identities, we can reach the 100% of matching for the 26 samples we have in the test set. However we can always find the proper identity of a sample (i.e. 100% matching), among the 26 we have in the test set, looking at the rank-9 best scoring identities.

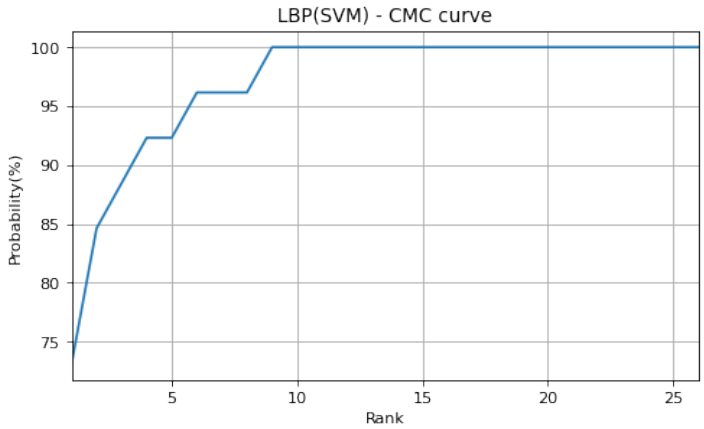

Figure 13: Cumulative Matching Curve for LBP feature extraction using SVM-based scoring method

6.2 Task 4: implementation of a different distance calculation layer in Siamese deep learning model

The implementation of the different distance calculation layer of the Siamese model can be find in the Assignment folder, in: siamese_manhattan.py file. Basically, the Siamese network we are training, uses a contrastive loss which accept as arguments: the ground truth labels and the prediction that is the distance between the embeddings of the image pairs we gave as inputs. The objective of the Siamese network is training in order to increase the similarity score of images with the same label and decrease it when the labels of the two input images do not match.

In the original implementation the euclidean distance layer has been used to calculate the distance of the embeddings. In this optional task manhattan distance has been considered to build that distance layer. Furthermore, to maintain integrity with respect to the fact that the Siamese model weights are learned to maximize or minimize the manhattan distance between images embeddings, we use the manhattan as a distance metric of embeddings to build the similarity matrix as well.

For the verification scenario, as one can observe from the ROC curve on the left in Figure 14, the system with manhattan-based Siamese model performs generally worse than the one based on the euclidean distance for very low values of threshold, while in the other cases it shows very similar performances. For this reason we avoid to show also all the other curves and performance metrics, since in all the other cases the two curves overlap. Therefore, also the Area Under the Precision-Recall curve (0.973), the Average Precision(0.973) and the EER point (0.030) have all almost identical values.

For the identification scenario, we can state that the euclidean-based Siamese works better than the euclidean one, at least for the first ranks of the CMC curve which are probably the most important ones, as we can observe on the right of Figure 14. Surprisingly, the rank-1 recognition rate using manhattan distance is better, reaching a great value: 0.993.

Finally, we can state that, with the exception of the last advantage analyzed, the euclidean distance layer is more suitable for the Siamese networks based systems. Indeed, generally manhattan distance works better than the euclidean one in those situations in which there is high dimensionality. In this case, since the embeddings on which the distances are computed have shape 32x128, probably we are not dealing with a case in which curse of dimensionality makes the euclidean distance less good. Perhaps, if the embeddings had been larger in dimensions, manhattan distance might have worked better.

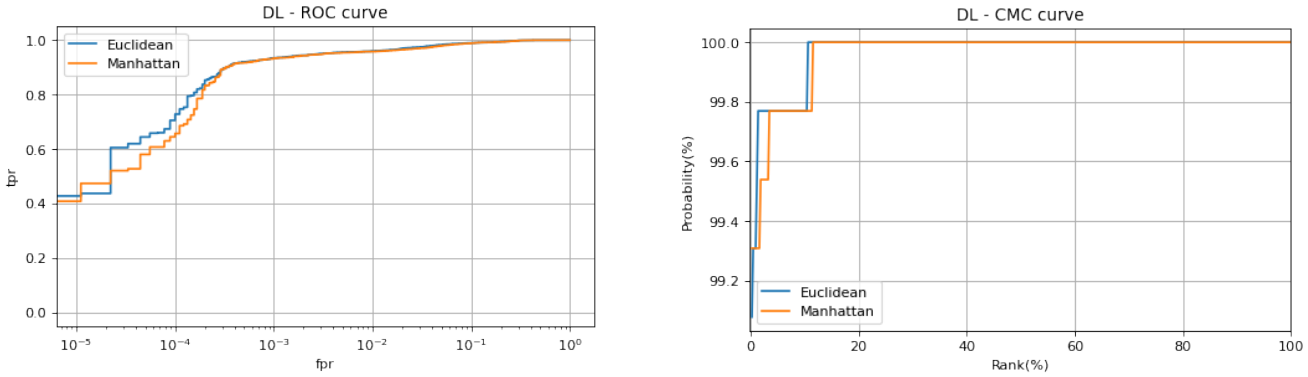

Figure 14: ROC in verification scenario and CMC in identification scenario with features extracted from Siamese model with manhattan or euclidean distance layer