Learned Low Dimensional Image Representations for Facial Recognition

Sukrit Arora
sukrit.arora@berkeley.edu

Rafael Calleja
rafael.calleja@berkeley.edu

Abstract

In this paper, we propose a modification of the dictionary-based Sparse Representation-based Classification (SRC) facial recognition algorithm using different low dimensional representations of input images. We sampled data from the Extended Yale B Database (cropped) that include grayscale images of 38 subjects under various lighting conditions. The analysis was performed for five different low representational methods: naive decimation, filtered downsampling, Haar Wavelet downsampling, Principle Component Analysis, and a convolutional autoencoder. Evaluations for each method demonstrated 3% improvement to the test accuracies of the algorithm over the standard low dimensional representation (naive decimation). From this, it was concluded that incorporating more frequency content through the use of the Haar Wavelet downsampling method greatly increase performance. Furthermore, research into the convolutional autoencoder demonstrated its ability to compensate for lighting conditions which may open up new avenues through which to solve more complex problems.

1 Introduction

As a basis for our work, we looked at the Sparse Representation-based Classification [1] (SRC) facial recognition algorithm. This algorithm takes a dictionary-based approach to solve the basic object recognition problem of using labeled training samples to correctly determine the class to which a new test sample belongs. For a recognition of n_i training samples from the i-th class and $w \times h$ grayscale images, the training dictionary manifests itself as $\mathbf{A}_i = [\mathbf{v}_{i,1}, \mathbf{v}_{i,2}, ..., \mathbf{v}_{i,n_i}] \in \mathbb{R}^{m \times n}$ where $m = w \cdot h$.

Given a large $n_i \; \forall i$, any new sample $\mathbf{y} \in \mathbb{R}^m$ should lie approximately in the linear span of the dictionary. This encourages us to find a sparse solution to $\mathbf{y} = \mathbf{A}\mathbf{x}$. A sparse solution necessitates an underdetermined system and so depending on the image size ($w \times h$) and the number of training samples n, a solution to such a problem would require some low dimensional representation.

In this paper, we look at how different methods of achieving this low dimensional representation, of both the training and test data for the SRC algorithm, affects the efficacy of the algorithm as a whole. We analyze five methods: naive decimation, filtered downsampling, Haar Wavelet downsampling, Principle Component Analysis, and a convolutional autoencoder.

2 Sources

We used the Extended Yale B Database (cropped) which includes 2414, 192×168 ($w \times h$) grayscale images of 38 subjects under various lighting conditions. Each image was captured with the light source at various azimuths and elevations yielding a diverse portfolio of portraits per subject.

Preprint. Under review.

3 Methods

3.1 SRC Algorithm

Algorithm 1 Sparse Representation-based Classification (SRC) [1]

```
Input: matrix of training samples $\mathbf{A} = [\mathbf{A}_1, \mathbf{A}_2, ..., \mathbf{A}_L] \in \mathbb{R}^{m \times n}$; test sample $\mathbf{y} \in \mathbb{R}^m$
Normalize the columns of $\mathbf{A}$ and $\mathbf{y}$ to have unit $l_2$-norm
Solve the $l_1$-minimization problem:
    $\hat{\mathbf{x}} = \arg\min_x ||\mathbf{x}||_1 \quad \text{s.t.} \quad \mathbf{Ax} = \mathbf{y}$
Compute the residuals $r_i(\mathbf{y}) = ||\mathbf{y} - \mathbf{A}\delta_i(\hat{x})||_2$ for $i = 1, ..., L$
Output: identity($\mathbf{y}$) = arg $\min_i r_i(\mathbf{y}) \in [0, ..., L-1]$
```

The foundation of this algorithm lies upon the L_1-minimization. To solve this optimization problem we used `sklearn.linear_model.Lasso` which reframed the above problem as:

$$\hat{\mathbf{x}} = \arg\min_x ||\mathbf{y} - \mathbf{Ax}||_2^2 + \lambda||\mathbf{x}||_1$$

In order to get this problem to converge, we needed to perform some hyperparameter tuning for λ.

For the purposes of this paper, we look at how the inputs, $\mathbf{A}$ and $\mathbf{y}$, to the SRC algorithm can be modified to yield better results.

Below we dive into what methods we chose to achieve a 132-length vector, low dimensional representation of the images in our dataset, how this affected the accuracy of the SRC algorithm, and why we think each representation yielded the results they did.

3.2 Low Dimensional Representations

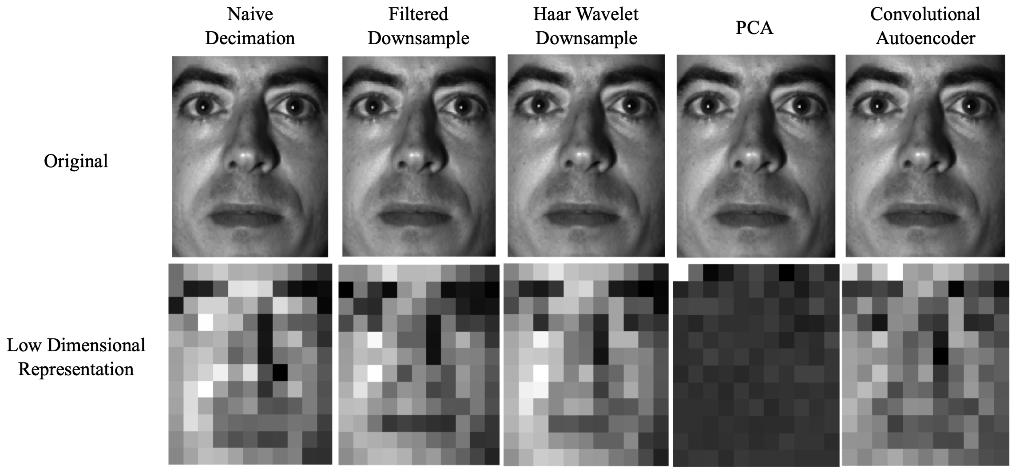

Figure 1: Table of original image and its corresponding low dimensional representation for each method

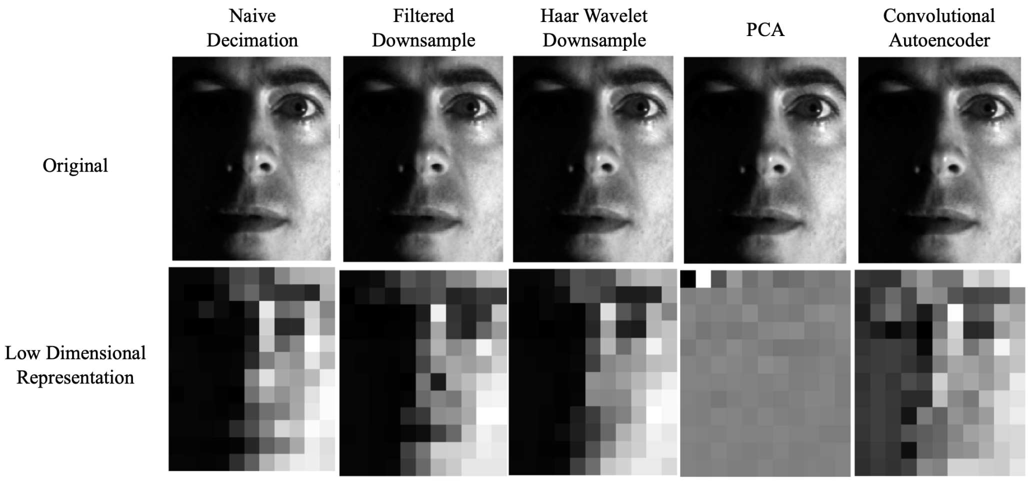

Figure 2: Table of original shaded image and its corresponding low dimensional representation for each method

3.2.1 Naive Decimation

The naive decimation method is the representation used in the original testing of this algorithm and is what we decided to use as the standard for which to compare other representations against.

Given an input image I of size $w \times h$ and a downsampling factor d, take every d pixel value in both the width and height of the image and save it into a new image I_{new}.

The naive decimation method is by far the simplest algorithm to implement and is therefore a good starting point to ensure SRC is working properly. However, unsurprisingly, throwing away so much information in between pixels tends to have negative effects on the algorithm's ability to identify the subject. An example of naive decimation's undesirable effects can be seen when comparing the eyes of the naive decimation example in Figure 1 to that of the Haar Wavelet downsampling.

3.2.2 Filtered Downsampling

The filtered downsampling method is very similar to naive decimation, but with extra preproessing. Before the decimation takes place, a low pass filter is convolved first in one dimensions and then in the other to save some spatial information (and therefore frequency content) of the image.

The filtered downsampling method is implemented using `cv2.resize` and is therefore also very easy to implement. While more spatial information is saved with respect to the naive decimation method, because the low pass filters are applied independently, the 2 dimensional nature of the image is not leveraged.

3.2.3 Haar Wavelet Downsampling

For a subsampling factor of N, we take the $\log N$ level 2D Haar Wavelet transform of our image [2], and take the average of the lowest dimensional outputs. This is equivalent to taking the full frequency content of the lowest resolution images. A visualization of the transform can be seen in Figure 3.

The benefit of the Haar Wavelet transform is that, because it is both a two dimensional and multiresolutional, it better preserves the frequency content of the image as it downsamples it. The only downside to this method is that the image can only be downscaled by factors of 2.

3.2.4 Principle Component Analysis

For our Principle Component Analysis low dimensional representation, we take our data matrix **A** and find its singular value decomposition: $A = U\Sigma V^T$. Because our data lies on the columns of A,

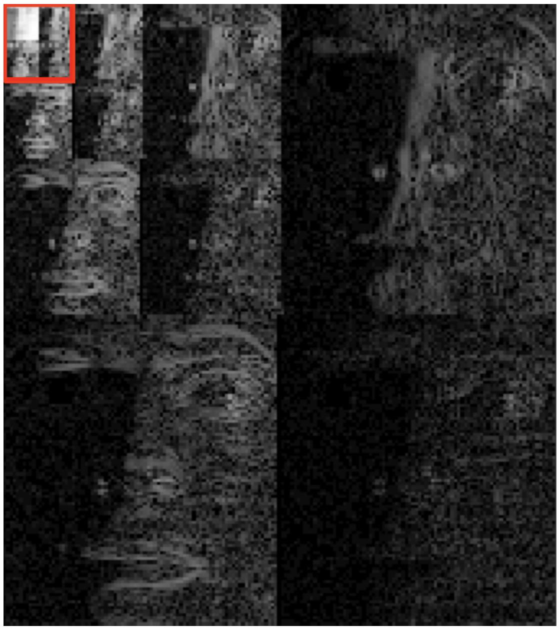

Figure 3: Visualization of the Level 4, 2D Haar Wavelet Transform of a face image. Low dimensional representation is average of red square.

we take the top 132 columns of U, and project our data matrix onto the subspace spanned by the selected columns of U: $A_{\text{ld}} = U_{1:132}^T A$. [3]

We then save this transform, and apply it to test time data in order to project those images into the same low dimensional space.

The benefit of PCA is that it can compact the data very well (i.e. make the problem very low dimensional), but it has several disadvantages. For one, PCA is the only method that works on the dataset as a whole, as opposed to on a per image basis. This means we need a certain number of training examples for our problem to remain undertermined, and therefore solvable using Lasso. Another downside to this method is that the low dimensional representation is not as readily interpretable when compared to the other methods.

3.2.5 Convolutional Autoencoder

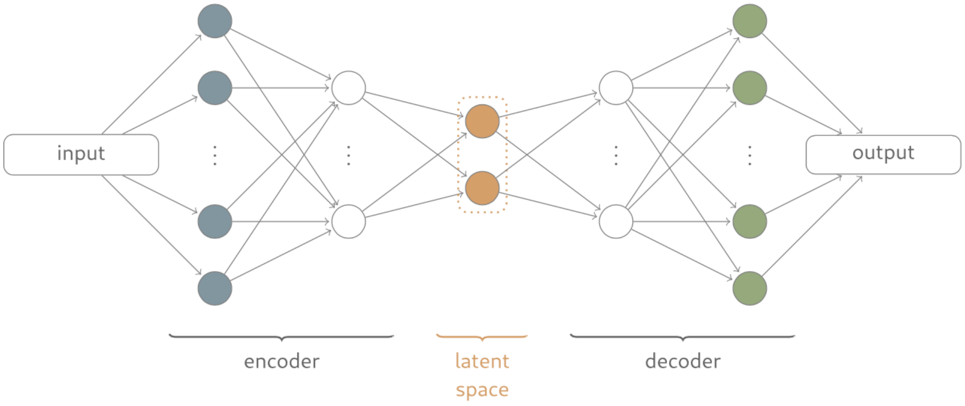

Figure 4: Example of an autoencoder architecture, where the latent space is the low dimensional representation used in our classification

We construct a convolutional autoencoder network that consists of convolutional layers for both downsampling and upsampling, and parameterized rectified linear units (PReLU) for nonlinearities. Figure – [4] details the specific structure for each downsampling/upsampling "unit."

The network can be regarded as a pair of networks: an encoder that embeds the image into a low dimensional space via a series of nonlinear, convolutional downsamples, and a decoder that aims to reconstruct the original image using the low dimensional representation via a symmetric series of nonlinear, convolutional upsamples. The channel dimensions for our encoder network were: $N = [32, 32, 64, 64, 64, 32, 16, 1]$, and our decoder network channel dimensions are the reverse.

The encoder can be interpreted as an analysis transform represented by $y = f_\theta(x)$, and the decoder a synthesis transform represented by $\hat{x} = g_\phi(y)$. In these equations, x corresponds to the original image, $\hat{x}$ to the reconstructed image, and y to the low dimensional representation that will be used for classification [4]. The parameters of the network, θ and ϕ, were trained using Adam for – epochs on our training data, evaluated with the following loss:

$$\begin{aligned} L(x; \theta, \phi) &= ||x - \hat{x}||^2 + ||y||^2 \\ &= ||x - g_\phi(f_\theta(x))||^2 + ||f_\theta(x)||^2 \end{aligned}$$

An example of an image and its embedded, low dimensional representation can be found in Figure –.

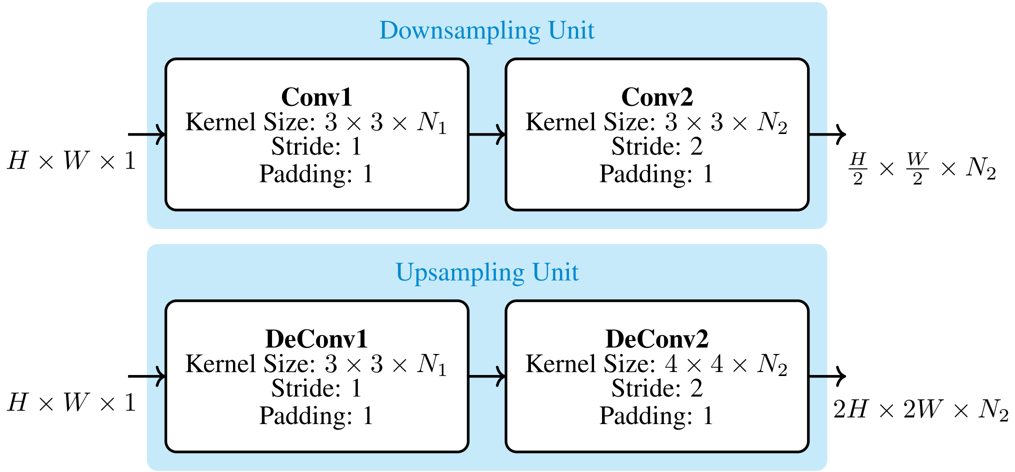
Downsampling Unit
$H \times W \times 1$
Conv1
Kernel Size: $3 \times 3 \times N_1$
Stride: 1
Padding: 1
Conv2
Kernel Size: $3 \times 3 \times N_2$
Stride: 2
Padding: 1
$\frac{H}{2} \times \frac{W}{2} \times N_2$
Upsampling Unit
$H \times W \times 1$
DeConv1
Kernel Size: $3 \times 3 \times N_1$
Stride: 1
Padding: 1
DeConv2
Kernel Size: $4 \times 4 \times N_2$
Stride: 2
Padding: 1
$2H \times 2W \times N_2$

Figure 5: Each downsampling unit of the encoder and upsampling unit of the decoder, respectively

The benefit of the convolutional autoencoder is that, because it has so many degrees of freedom in how it can express the low dimensional representation, it is very expressive. As we can see in Figure –, it is interesting to see that the autoencoder learns to compensate for lighting condition. The downside to this method is that it is data driven, and so its performance depends on the size of the dataset. It also takes much longer than other methods as it needs to be trained and the weights need to be learned.

4 Results

Below is a table of Train and Test accuracies using each of the low dimensional representations detailed above:

	Naive Decimation	Filtered Downsampling	Haar Wavelet Downsampling	PCA	Convolutional Auto Encoder
Train Accuracy (%)	100	100	100	99.01	99.75
Test Accuracy (%)	90.47	90.8	**93.37**	90.31	92.46

5 Discussion

From the results in the table above, there is something to be seen from both the train accuracies and the test accuracies.

First, we note that the Haar Wavelet Downsampling method far outperforms naive decimation. This is most likely because of the way the Haar Wavelet transform preserves frequency content by leveraging the full frequency content of the lowest resolution images. While the fitered downsampling method is only two dimensional, the Haar Wavelet transform is multi-dimensional, breaking up the image into multiple frequency components.

The convolutional autoencoder method, while only being trained for 100 epochs, still performs significantly better than naive decimation probably because it has many degrees of freedom. Furthermore, because the neural net "sees" the entire training set, we note in Figure 2 that it seems to compensate for poor lighting conditions. Whereas naive decimation, filtered downsampling, and Haar Wavelet downsampling all are unable to represent where the subject's left eye is, the convolutional autoencoder is able to show some semblance of many obscured facial features. An interesting thread that may be followed is using the convolutional autoencoder to learn a transformation of poorly lit images to properly lit images which may lead to a solution of this problem without the use of the robust form of this algorithm.

PCA performs even worse than naive decimation despite being known as an algorithm that attempts to maximize variance within a dataset. We hypothesize that although PCA is attempting maximize the variance within the dataset, the maximum variance within the dataset is not found between classes, but rather between lighting conditions. This can be seen when comparing the low dimensional representations of Figures 1 and 2. For the projection of the subject's face in good lighting conditions, the projection onto the first principal component (the top left pixel) seems to be a very positive value. In comparison, the projection of the subject's face in poor lighting conditions onto the first principal component yields a very negative value.

One last thing to note is that the fact that neither PCA nor the convolutional autoencoder achieve 100% training accuracy. Of the five methods, these two are dataset dependent and so the learned representation for a given image may not be necessarily characteristic of that image.

6 Conclusion

From the analysis above, we have shown that the learned representation input into the SRC algorithm has a large impact on the overall efficacy of the algorithm. The Haar Wavelet transform, which leverages the full frequency content of the downsampled image, performs the best on a randomly sampled dataset with a 50-50 split. While it seems that relying on the inherent information embedded in an image yields the best results, the convolutional autoencoder, which attempts to learn a representation of the image through weighted convolutional layers, holds potential in outperforming natural filters. Furthermore, training the autoencoder on the randomly sampled, 50-50 split dataset demonstrates its ability to compensate for lighting conditions. This last point shows promise in potentially solving the facial recognition problem of classifying the faces of subjects in poor lighting using only a dictionary of images taken under proper lighting conditions without the use of the robust form of this algorithm.

References

[1] John Wright and Yi Ma. *High-Dimensional Data Analysis with Low-Dimensional Models: Principles, Computation, and Applications*. https://book-wright-ma.github.io/. Cambridge University Press, 2020.

[2] Piotr Porwik and Agnieszka Lisowska. "The Haar-wavelet transform in digital image processing: its status and achievements". In: 13 (Nov. 2004).

[3] Svante Wold, Kim Esbensen, and Paul Geladi. "Principal component analysis". In: *Chemometrics and Intelligent Laboratory Systems* 2.1 (1987). Proceedings of the Multivariate Statistical Workshop for Geologists and Geochemists, pp. 37–52. ISSN: 0169-7439. DOI: https://doi.org/10.1016/0169-7439(87)80084-9. URL: http://www.sciencedirect.com/science/article/pii/0169743987800849.

[4] Zhengxue Cheng et al. *Deep Convolutional AutoEncoder-based Lossy Image Compression*. 2018. arXiv: 1804.09535 [cs.CV].