CHAPTER

21 Face detection and recognition: the impact of deep learning

Face recognition is highly important in modern society and crucial in security applications. Similarly, detection of faces in scenes is a starting point in many practical situations. This chapter covers both areas, using contrasting methods such as principal components analysis (PCA) to obtain "eigenfaces" and "boosting" to locate faces extremely rapidly. However, these early methods are becoming marginalized, e.g., by the frontalization approach and by deep learning so that computers can attain human levels of successful recognition.

Look out for:

- What can be achieved using simple techniques for efficient face detection
- Detection of facial features such as eyes, nose, and mouth
- The Viola–Jones boosting face detector
- The eigenface approach to face recognition
- The use of frontalization for transforming faces to a standard frontal form
- How deep learning may be applied to face detection and recognition (FDR)
- How faces may be considered as parts of 3-D objects
- Use of invariance for initiating face recognition.

Although this chapter is especially involved with FDR, it is forced to consider the use of modern deep learning architectures. Although the latter was dealt with mainly in Chapter 15, Deep Learning Networks, this chapter develops the ideas further and can be regarded as highlighting the practical issues: Indeed, by the end of the chapter, the impact of deep learning emerges much more clearly.

21.1 INTRODUCTION

Face detection has become important in many modern situations, ranging from crowd control to monitoring people entering a building. In many cases, face detection alone will be insufficient, and faces will need to be recognized or in

Computer Vision. DOI: http://dx.doi.org/10.1016/B978-0-12-809284-2.00021-6

some cases verified, as when entering a bank vault or logging into a computer. It can be argued that the most basic need is that of face detection, as that can not only permit people to be counted but also initiate recognition. It can also be argued that the act of recognition necessarily includes detection; and that verification is an instance of recognition where there is only one person to be recognized. In the overall scheme of things, face detection is the simplest of these tasks. In principle, it can be automated by applying a matched filter based on an "average" face, which could, as implied, be obtained by averaging a large number of faces available in a database. However, huge complications would be involved in achieving this, because face images are liable to be taken in widely different lighting conditions, and because faces are unlikely to taken frontally. Indeed, heads will have different positions and poses; so, even a fully frontal view may display heads differing in roll or pitch. As for ships, roll, pitch, and yaw are the three important angles to be controlled or allowed for when attempting face detection or recognition. Of course, there are situations in which face pose is controlled, such as when a passport or driving license photograph is being taken, though such cases must be regarded as exceptional. Finally, it must not be forgotten that faces are flexible objects: Not only can the jaws be moved, and the mouth and eyes opened or closed, but also very large numbers of facial expressions (and thus emotions) can be instigated. All these inter- and intraface variations make facial analysis and recognition a highly complex business, which is further complicated by varieties of facial and cranial hair, and by glasses, hats, and other apparel. Needless to say that all these factors have to be dealt with in efforts to recognize terrorists.

In this context, it is instructive to look at the many "faces in the wild" that have been extracted from Internet pictures. We shall examine some of these below—in particular, using examples from the well-known set of George W. Bush (Fig. 21.1). We now move on to see how face detection can be attempted using skin tones as the basic means of proceeding.

21.2 A SIMPLE APPROACH TO FACE DETECTION

The widely used "labeled faces in the wild" (LFW) datasets contain pictures that have been collected from the Internet and approximately centered and sized. For the most part, they show frontal faces with a relatively small range of 3-D roll, pitch, and yaw orientations—all typically within the range $\pm 30^\circ$ (see e.g., Fig. 21.1). In this section, we consider how face detection can be performed with reasonable speed and efficiency. One obvious approach is to aim to detect skin tones using both color and intensity controls. In Fig. 21.1, we have attempted this using a simple hue and intensity range test: *hue* $+180^\circ$ has to be in the range $180^\circ \pm 20^\circ$, and intensity has to be in the range 140 ± 50 (in an overall intensity range $0-255$). This range was obtained empirically: In a complete practical

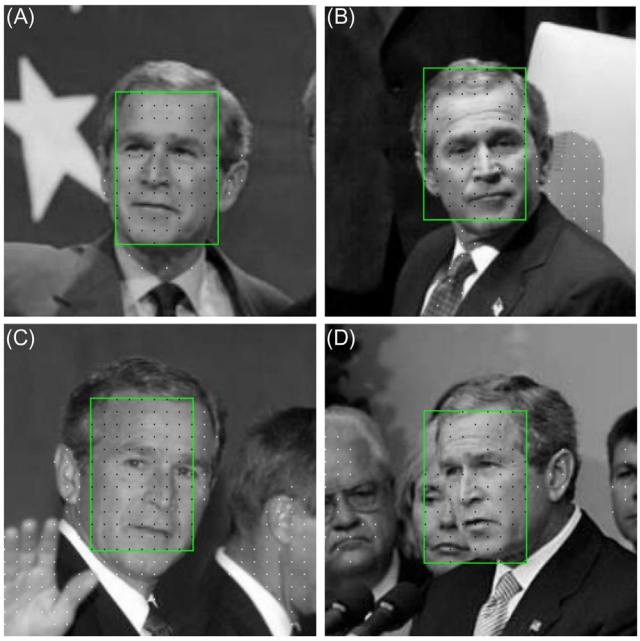


FIGURE 21.1

Face detection using a simple sampling approach. Here, four Bush faces (A—D) from the LFW database are detected using the simple sampling approach. First, skin tones are detected and recorded as white dots. Then, the best fit 2:3 aspect ratio bounding box is used to find the most likely face location. Within each best-fit box, white dots are changed to (inlier) black dots, and the remaining white dots indicate outliers. In cases (B) and (C), significant numbers of white dots remain, and in (C), they exceed the number of black dots. Note that the method actually indicates the best fit to skin tone regions rather than to faces as such: This is because the method does not include any specific within-face feature detection.

system, it would be determined by rigorous training. Here, it seemed appropriate to show what is possible with a simple schema.

In Fig. 21.1, processing was speeded up by sampling every 10th pixel in each direction. All pixels that have a skin tone signaled by the above test are marked with white dots. Next, a box with approximately 2:3 aspect ratio, corresponding roughly to LFW facial dimensions, is applied and the locations containing the greatest numbers of skin-tone signals are found: At this stage, the inlier dots are relabeled black, and the remaining white dots are regarded as outliers. In all four cases, the true faces are correctly found and labeled, and the decisions are not confused by the remaining outliers—even where they correspond to partial faces or other bright patches in the background. Clearly, the approach is robust and reasonably accurate: It is also very fast in operation as only every 100th pixel is

visited. The fact that the skin-tone detector is relatively crude and is not adapted to the individual faces seems not to matter. In fact, the most important parameter should be the box size. Oddly, this parameter is actually noncritical: It has been applied to a fair number of the LFW faces without any omissions occurring. Overall, the main problem is that this face detector will always find the object most akin to a face, even if there isn't one present. In this situation, the most obvious way forward would appear to be to find suitable facial feature detectors—such as eyes, ears, nose, or mouth—to verify that any face that is, found is actually a face. Interestingly, though eyes are probably the features that are most widely used for this purpose, they are relatively indistinct in the pictures presented in Fig. 21.2, except perhaps for the last three. This makes it problematic to decide quite what features should be sought, and how accurately they could be detected.

21.3 FACIAL FEATURE DETECTION

Detection of faces via their skin tones can be unreliable, because (1) hands and other regions of the body can mimic faces, (2) clothes can have similar tones and colors, and (3) even background regions composed of sand or other materials can have similar tones and colors. Hence, alternative methods of face detection based on facial features or facial shapes can be used in place of those based simply on skin tones. In addition, they can be used *as well as* skin-tone region detection, to ensure that the overall approach is sufficiently robust.

Prominent amongst relevant facial features are eyes, nose, mouth, and ears—and their subfeatures such as corners. These can be detected by correlation using trained templates, as shown in Fig. 21.3 for eye detection. For reliability, such templates need to be quite small, to offset the problems of variability that plague face detection. As shown in Fig. 21.3, eyes can be detected separately, and the results combined adaptively, if they appear to correspond within sufficiently accurate limits. Clearly, combining detections of several features can be carried out not just for eyes but also for multiple features, though a priori information on how to achieve this sufficiently robustly may be lacking, especially in cases where degrees of roll, pitch, and yaw are large and unknown. Overall, perhaps the worst aspect of the multifeature approach is that a number of detectors need to be designed and trained and also that the fusion of the feature data can be even more difficult to manage and train. Note also that handling all these factors and complications can involve considerable computation. Thus, it is not too surprising that in recent years, this rather complex approach has become less evident in the literature. It was in this context that Viola and Jones (VJ) looked at the situation again and arrived at their innovative Haar filter-based approach, which is outlined in the following section.

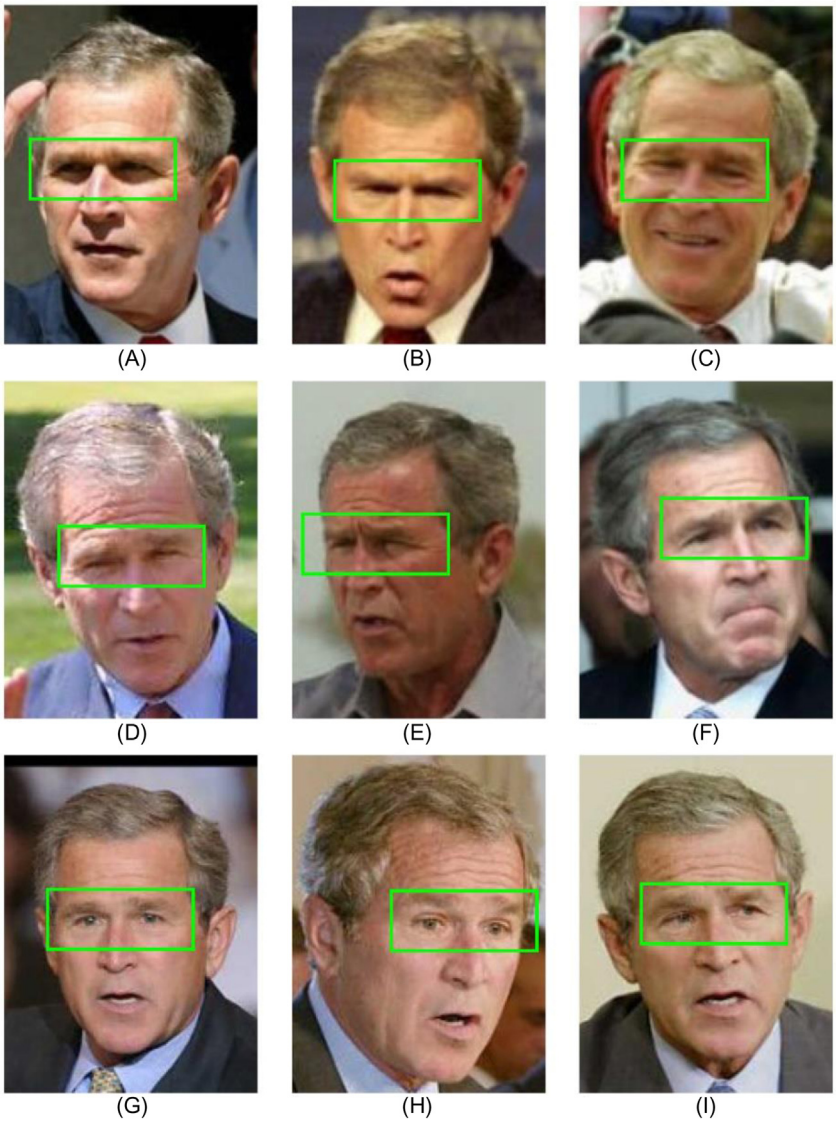


FIGURE 21.2

Variations in the eye regions for Bush LFW faces. In cases (A)–(E), there is strong shadow; so, it is impossible to apply obvious eye detectors—such as Hough transforms—based on iris detection. However, this starts becoming possible in cases (F)–(I), though it should be noted that the visible boundary of the iris in each of these cases is quite low (significantly less than 50%): At such low resolution, this becomes a problem. In addition, the well-known dark region around the eyes is almost completely absent in cases (G)–(I), making it difficult to locate the eyes reliably: In such cases, feature detectors for nose, mouth, or other features would also have to be employed.

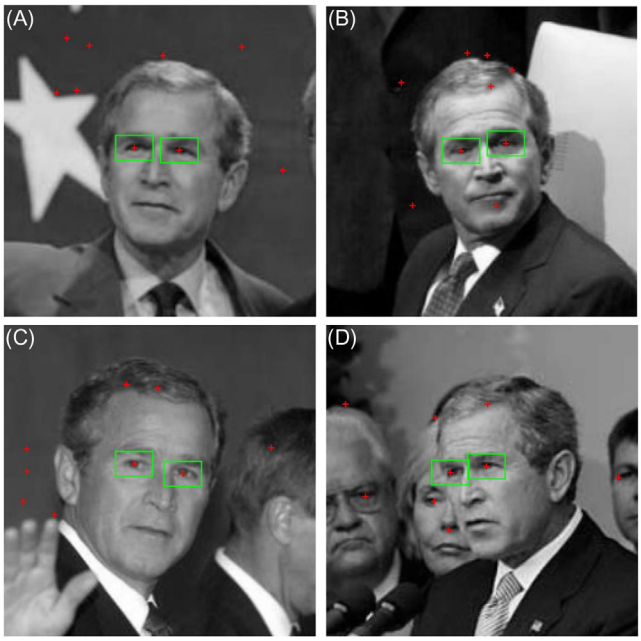


FIGURE 21.3

Detection of eye features. In these images (A–D), potential eye features are located using correlation with a trained eye template, the number recorded being limited to the nine most likely feature locations, these being marked with red crosses in each image. Then, a search is made for *pairs* of eye features that are within expected horizontal and vertical distance bounds (29 to 42 and ± 8 pixels, respectively), and the results are marked with green boxes. The box sizes indicate the size of the feature template used for eye detection. Note that many of the outliers arise from hidden likenesses in the hair and background, though in case (D), they also indicate isolated eyes from other people. Interestingly, the method copes well with faces exhibiting up to 20° roll and 30° yaw.

21.4 THE VIOLA–JONES APPROACH TO RAPID FACE DETECTION

The face detection problem outlined in the last section was analyzed in depth by Viola and Jones (2001). They decided on a totally new approach. First, they eschewed concentration on "obvious" features such as those mentioned above—viz. eyes, ears, nose, and mouth; in addition, they eschewed color and skin-tone detection, preferring to analyze faces in terms of intensity profile characteristics: Finally, they made no direct analysis of shape. What they sought above all was a set of features that worked in practice as a result of careful, sufficiently general

training on known datasets. For this reason, they decided to train the software system using Haar-like basis functions as general-purpose features. These had the possibility of locating relatively dark regions of the face: Indeed, it is often the case that the part of the face just below the eyebrows, which contains the eyes, is relatively shadowed and is significantly darker than the forehead or the region containing the nose and cheeks. Similarly, the part between the eyes is often significantly lighter than the nose region. If and when such intensity variations are actually present, Haar-like filters would be expected to show them up.

It should first be explained that a typical Haar filter consists of two adjacent, touching rectangles of equal size, but with opposite weights (normally ± 1): Hence, the sum of the weights will be zero, making the filter insensitive to the background illumination level. Such a filter will approximate to a differential edge detector. Other Haar filters will consist of three adjacent, touching rectangles, and if these are equal in area, their weights will have to be in the ratio $-1{:}2{:}-1$. Yet other Haar filters will consist of four adjacent, touching rectangles, whose weights alternate between rows: -1, 1 and 1, -1. These basic filters are illustrated in Fig. 21.4. The reasons for using such simple filters are (1) that very many of them may be constructed and applied easily, and (2) that their use involves minimal computation.

In Viola and Jones' (2001) work, the face images used for training were 24×24 pixels in size, and it was assumed that relevant features could arise anywhere within each image. Furthermore, relevant features could have any size within this area. They developed the idea that each feature could appear anywhere within an image and at any scale, and its discriminatory power should be determined during training.

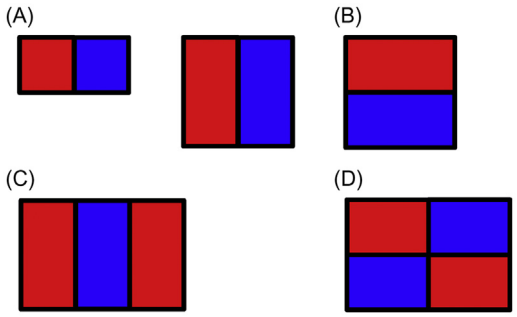


FIGURE 21.4

Typical Haar filters. (A) Basic two-element differential edge detector type filters. (B) Two-element differential edge detector operating in the vertical direction. (C) Three-element 1-D Laplacian type filter. (D) Four-element Roberts cross-type filter. If the red and blue elements have respective weights $+1$ and -1, the filters will be zero sum, though in case (C), the blue element must be assigned double the weight.

Considering a general Haar filter as being defined by an enclosing rectangle, it is straightforward to calculate the total number of such filters: First, the vertical boundaries of the filter can be chosen in ${}^{25}C_2 = 300$ ways, and similarly, for the horizontal boundaries: This makes the total number of features under consideration as being $300^2 = 90{,}000$. However, if the features are internally asymmetrical, e.g., containing two horizontally adjacent rectangles of opposite weights, there will in principle be twice this number, viz., 180,000. In fact, internal structures also act to limit the numbers of possibilities, as the overall width of the rectangle will then (as in the last case) have to contain an even number of pixels, leading to the total number of features of this type being $6 \times 24 \times 300 = 43{,}200$, or, including horizontally and vertically adjacent rectangles, a total of 86,400 features. Clearly, if we add the numbers of triple and quadruple rectangle features, and all other possible combinations, the total number will approach the total of 180,000 cited by Viola and Jones (2001). This is a very large number and far exceeds the number ($24^2 = 576$) required for a complete set of basis functions for images of size 24×24—i.e., it is many times "overcomplete." However, what is needed is a set of features that is, easily sufficient to accurately and compactly describe all the faces that can arise in practice. Of course, it would be impossible to include such large numbers of features in a practical face detector. Nevertheless, it is necessary to test a significant proportion of them during training, so as to make an accurate, rapidly operating face detector.

The approach adopted by VJ was to use a boosted classifier, based on Adaboost (see Chapter 14: Machine Learning: Probabilistic Methods), in which each feature would act as a weak classifier. A large proportion of these weak classifiers (features) would prove ineffective and would be rejected, and those that were retained would be assigned an optimum threshold and a parity value giving the sign of the filter (i.e., which way around the filter should be applied). At each stage, the weak learner selected would be the one that gives the best discrimination between faces and nonfaces: Note that, contrary to the statement given above that the training set consists of face images, it also has to contain many images that contain no faces. In VJ case, there were 4916 face images, plus their vertical mirror images, totaling 9832 images, and 10,000 nonface images of the same size. Making proper use of the latter is essential for training the classifier to eliminate false positives.

One of the factors that make the VJ detector operate rapidly is the fact that at each stage a good many negative subwindows are eliminated while retaining almost all the positive instances. This means that the false negative rate is kept close to zero. Indeed, this happens progressively, in that simpler more rapidly operating classifiers eliminate the majority of the subwindows, leaving it to later more complex classifiers to gradually reduce the false positive rate. The overall process is best described as a cascade of classifiers (Fig. 21.5): Each classifier in the sequence eliminates negative subwindows but virtually guarantees retaining and passing on all positive subwindows. Such sequential processing carries risks, because once a positive subwindow is rejected, it can't be recovered, and the error

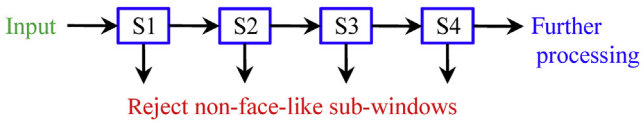


FIGURE 21.5

Viola–Jones cascade detector. This figure shows only the first four stages, S1–S4. All the subwindows stream into the input, and nonface-like subwindows are filtered out, leaving the remainder to undergo further processing. The complete detector contained 38 stages. By the very end, all the windows look very face-like, though exactly how many are actually false positives or false negatives must depend on the quality of the training.

rate necessarily rises. The overall classifier obtained by the VJ detector contained 38 layers and included a total of 6061 features. In fact, the first five layers contained, respectively, 1, 10, 25, 25, and 50 features, reflecting the increasing complexity of the stages of the cascade.

Another factor that made the VJ detector considerably more rapid and efficient was the use of the integral image method (see Section 6.7.5 and Fig. 6.15) for implementing every feature. The Haar filters are all composed of rectangles, and this made the integral image method a particularly natural way forward. Indeed, rectangular Haar filters are not especially ideal as feature detectors. However, when implemented using the integral image method, the process is so fast that a great many similar features can be added efficiently and their outputs combined together optimally, easily overcoming any losses in optimality of individual filters. In any case, it should be remembered that the Haar features form a (complete) basis set that is, guaranteed to be able to generate all possible shapes.

Overall, the VJ detector was about 15 times faster than the best previous detector (Rowley et al., 1998) and amounted to a breakthrough whose possibility had not previously been envisaged. In fact, it set the scene for more learning-based systems that rely less on conventional vision algorithm design, albeit relying far more on the use of specially constructed datasets for training. Interestingly, the fact that shape, color, and skin-tone analysis were eschewed early in the design, in favor of pure gray-scale processing and intensity profile analysis, is salutary, though in the end, it must be limiting—particularly if recognition rather than mere detection is involved.

Finally, we compare the VJ detector with the simple sampling detector described in the previous section. Both methods found all the faces in the image of Fig. 21.6, but the sampling detector had two specific faults which the VJ detector did not succumb to: One was that it was more susceptible to false positives, because it did not have the additional information about the eye regions being relatively dark; and the other was that, for the same reason, it sometimes biased the faces toward locations where there were larger patches of skin tone, rather than ensuring that the eyes were both present in the final boxes: In fact, this problem only occurred when the faces were subject to in-plane rotation (i.e., roll).

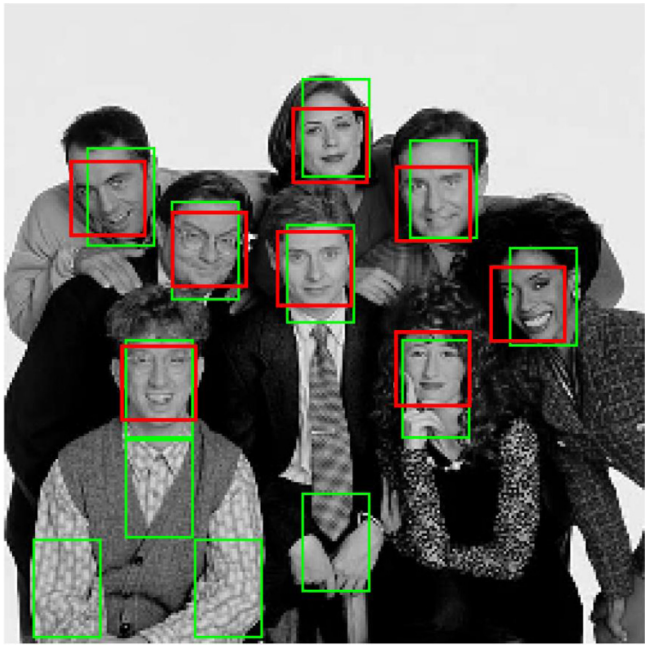


FIGURE 21.6

Detection of faces using two methods. Here, the green bounding boxes show the results obtained using the simple sampling method of Fig. 21.1; the red bounding boxes indicate the approximate locations found by the Viola–Jones detector. As might be expected, the latter finds the faces more accurately, without the bias toward skin-tone regions exhibited by the sampling method—though when properly combined with an eye detector, the latter method should be able to overcome this limitation. Note also that the sampling method is sometimes fooled by light regions—in this case containing hands and/or light clothing. In fact, the false positives could in this case be eliminated by ignoring the lower regions of the image, where faces are unlikely. Note that, to make this test, the simple sampling algorithm had to be modified to search sequentially for multiple nonoverlapping faces.

21.5 THE EIGENFACE APPROACH TO FACE RECOGNITION

The eigenface method of face recognition was invented by Sirovich and Kirby (1987) and developed into a successful practical technique by Turk and Pentland (1991). The basic idea was to take a set of face images as vectors in a face space and to perform PCA to form a basis set of standard face vectors. Faces could then be represented as linear combinations of this basis set, and each face could therefore be represented by the numerical weights in its own linear combination. The advantage of representing faces in this way is that each face would be represented

by a limited set of coefficients rather than by the large number of pixels in the original image. This means that any test face image can be compared with the training set images and classified by applying the nearest neighbor algorithm to determine the closest match.

As so far stated, there is a serious flaw in this approach—that the training set may be so large that the number of coefficients to be compared when testing could exceed the number of pixels in each image—in which case, the recognition problem will have been made worse rather than better. However, one of the main advantages of PCA is that each eigenvector is also assigned an eigenvalue, which effectively expresses its relative importance—at least in regard to the particular training set that has been employed. By arranging the eigenvalues in descending numerical order, it is possible to curtail the list of eigenvalues (and the corresponding eigenvectors) at a point where they become so small that the eigenvectors represent noise rather than useful features and scarcely contribute to the recognition process: for example, to include 95% of the total variation of the face images in the Extended Yale Face Database B of 16,128 images, the first 43 eigenfaces must be retained. This is a general point that has already been treated in Section 14.5. Thus, we start with M training vectors and reduce this to a basis set of N. It is now the number N that has to be compared with the number of pixels P in each face image. Note that all the face images used in the training set must have the same dimensions, which we shall take as $r \times c$, or typically as 100×100. Fortunately, to span a space of M faces, a lot can be achieved with a relatively small basis set of size N, and it is generally reasonable to assume that $N \ll P = rc$: Indeed, N is typically 50, and P is typically $\sim 10{,}000$, so the assumption is more or less guaranteed.

To use the Eigenface system, the training set must first be set up. Note that it is important to ensure that all the faces in the training set are as similar as possible in form: The pictures must be taken under the same lighting conditions; they must be normalized and cropped as necessary so that the eyes and mouth are aligned in all the images; they must be resampled to the same pixel resolution; they must be converted from their initial 2-D (pixel) format into a 1-D vector of standard length P; finally, they must be concatenated into a single training matrix $\mathbf{T}$ of width M. Another important factor is that, to proceed with PCA, each input image (i.e., each column of $\mathbf{T}$) must be converted to zero-mean by subtracting its mean value.

At this point, PCA is carried out to determine the eigenvalues and eigenvectors of the covariance matrix $\mathbf{S}$ (in the face recognition, scenario eigenvectors are commonly called "eigenfaces"). After this, the M eigenvalues are placed in descending order and the smallest ones discarded, leaving a basis set of N, as discussed above.

Unfortunately, PCA is highly computation intensive, and it must be remembered that it involves diagonalizing the covariance matrix $\mathbf{S}$ which is defined for pairs of images, each containing P elements: This means that $\mathbf{S}$ is typically a $10{,}000 \times 10{,}000$ element matrix. This is what makes PCA so computation intensive. However, when the number of training images M is smaller than the number

of pixels P in each image, it is simpler to compute the principal components as indicated below. First, we consider the usual way of computing the principal components, i.e., in terms of the following equation:

$$\mathbf{S}v_i = \lambda_i v_i \tag{21.1}$$

Next, we represent the covariance matrix in terms of $\mathbf{T}$:

$$\mathbf{S} = \mathbf{T}\mathbf{T}^{\mathrm{T}} \tag{21.2}$$

Substituting for $\mathbf{S}$, we obtain

$$\mathbf{T}\mathbf{T}^{\mathrm{T}}v_i = \lambda_i v_i \tag{21.3}$$

As $\mathbf{T}\mathbf{T}^{\mathrm{T}}$ is a large matrix, we consider what happens when applying the smaller matrix $\mathbf{T}^{\mathrm{T}}\mathbf{T}$:

$$\mathbf{T}^{\mathrm{T}}\mathbf{T}u_j = \lambda_j u_j \tag{21.4}$$

Premultiplying by $\mathbf{T}$, we obtain:

$$\mathbf{T}\mathbf{T}^{\mathrm{T}}\mathbf{T}u_j = \lambda_j \mathbf{T}u_j \tag{21.5}$$

This shows that if u_j is an eigenvector of $\mathbf{T}^{\mathrm{T}}\mathbf{T}$, $v_j = \mathbf{T}u_j$ is an eigenvector of $\mathbf{S}$.

As $\mathbf{T}\mathbf{T}^{\mathrm{T}}$ is a large matrix (typically $\sim 10{,}000 \times 10{,}000$), and $\mathbf{T}^{\mathrm{T}}\mathbf{T}$ is a much smaller matrix (e.g., 200×200), it makes very good sense to compute the eigenvalues and eigenvectors from the smaller matrix, to obtain results for the larger matrix. The main problem with this approach is that the resulting vectors v_i are not normalized, and normalization has to be carried out later, if necessary.

Note that the coefficients used to record the projections of any face image against the eigenfaces are specific to the particular image rather than to the subject appearing in the image. This is a severe limiting factor of the method, as the weights for a subject seen under two sorts of lighting (e.g., left and right lighting) may differ radically even for the same subject.

Interestingly, in spite of the warnings given about how to prepare training set images, it is usually the case that the first three eigenfaces in the dataset have to be discarded. This is because they normally arise from variations in illumination rather than from differences in the faces themselves. Hence, eliminating them will tend to boost recognition accuracy (Belhumeur et al., 1997). However, it should be asked why the first *three* eigenfaces should be eliminated. First, note that mean image intensity has already been eliminated from the PCA analysis, so we have to look for further possibilities. The first is that of contrast: Two other candidates are linear intensity variations in two perpendicular directions. Note that once these have been eliminated, all possible linear intensity variations will have been covered. Higher order variations probably always have to be retained as it would be difficult for the computer to distinguish between variations due to lighting illumination and those due to facial characteristics.

It is also worth pointing out that the eigenfaces themselves bear little pictorial relation to real faces: Indeed, they can be regarded as potential incremental

adjustments to existing weighted sums of eigenfaces applied so as to model real faces better: At most, they will only add minor details to the models. Another point is that PCA is more an efficient mathematically based *representation* procedure than one designed for optimum classification: i.e., it provides a descriptor rather than an ideal decision surface. However, it is useful in the realm of face recognition for the purpose of dimensionality reduction.

Another area in which the eigenface approach ought perhaps to be useful is in determining modes of variation, such as degrees of opening of the eyes or mouth, and even in representing differing facial expressions. However, it seems that this is too hopeful in an approach that is, more orientated toward face recognition. For analysis of facial expression, active shape and appearance models (which employ PCA internally) are tools that are more overtly adapted to this purpose.

Finally, "Fisherfaces" constitutes another approach that has been successful in this area. Invented in 1997 by Belhumeur et al., it was based on linear discriminant analysis and included labeled data to retain more class information during dimension reduction; as a result, it was rather more successful than the eigenface approach: In particular, it was less sensitive to lighting variations, and thereby attained higher recognition accuracy. In fact, its classification error rates were around 7.3% compared with 24.4% for eigenfaces and 15.3% for eigenfaces with the first three principal components excluded (Belhumeur et al., 1997).

21.6 MORE ON THE DIFFICULTIES OF FACE RECOGNITION

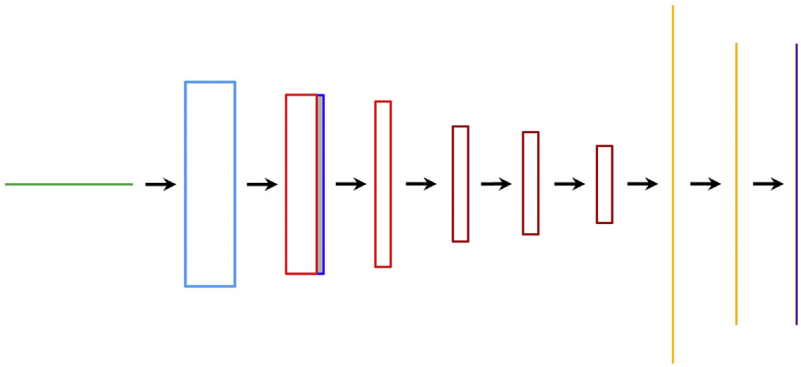
By now, it will have become clear that faces are exceptionally variable objects. Here, we clarify the reasons for this. First, faces come in many sizes, shapes, colors, albedos, and expressions—not to mention the possibility of partial self-occlusion, and occlusion by glasses, hair, hats, and even disguises; age is also a serious factor when considering face variations of given individuals. Furthermore, the original 3-D shapes largely determine the appearance in 2-D images. This means that the full pose of the original shape may have to be taken into account in analyzing the 2-D image. Indeed, the three generally unknown orientations—roll, pitch, and yaw—form a substantial part of what makes face recognition difficult. However, another factor is also of vital importance, and that is, ambient illumination—and in particular, which direction or directions the face is lit from. Indeed, this factor is responsible for something like half the difficulty of managing face recognition. Not surprisingly, as a result of all the variations mentioned above, early work of face recognition concentrated on controlling the viewpoint and the lighting, insisting that faces were viewed from directly in front. However, the systems that were produced in this way were necessarily severely limited in their application and accuracy. Nevertheless, by arranging to view faces from several directions during training, comparisons with test faces could be improved, each training image only having to cope with angles over a range $\sim 20^\circ$ instead

of the whole range. In fact, numerous schemes were devised for managing face recognition across pose: In particular, 2-D statistically based algorithms and 3-D algorithms have often been used to generate a 3-D model of each individual's head (e.g., Blanz and Vetter, 2003; Gross et al., 2004; Yan et al., 2007). In such cases, each test image had to be compared with a rerendered image derived from the 3-D model. In fact, the 3-D-based algorithms were significantly more computation intensive, and it was difficult to arrange that up to 10 images were available per individual in order to synthesize sound 3-D models. Overall, by 2007–2008, such approaches yielded performance accuracies of up to 87%, though still being limited to overall pose variations of up to $\sim 30^{\circ}$: In some cases, the results were shown to exceed human performance—though it gradually became clear that such conclusions were misleading because of the artificial conditions under which the photographs had been taken.

At that stage, a shift took place from controlled to uncontrolled environments, and the LFW database was introduced (Huang et al., 2007), so that researchers could utilize realistic sets of images for training. At that point, the performance achievable by available algorithms plummeted—though in very few years, performance had again risen markedly (e.g., Wolf et al., 2009 claimed 89.5% classification accuracy).

In 2014, Taigman et al. published a paper on their "DeepFace" approach to face recognition. This used a deep learning architecture (Fig. 21.7) which brought the performance level up to 97.35%: They compared it with the performance obtained by humans, for which the corresponding figure was 97.53%, giving little doubt—at least with the LFW dataset—that human performance was extremely close to being matched. To some extent, it came as no surprise that a deep learning network could achieve this. However, a closer look at the architecture reveals (1) that it employed an initial nonneural "frontalization" procedure (see Section 21.7) and (2) that the deep learning network employed a two-layer convolutional network (these were separated by a max-pool layer), followed by a three-layer locally connected section followed in turn by two fully connected layers (Fig. 21.7). The two convolutional layers, respectively, contained 32 filters of size $11 \times 11 \times 3$ and 16 filters of size $9 \times 9 \times 16$; the max-pool layer had an input field of 3×3 and a stride of 2; and the three locally connected layers each had 16 filters with respective sizes of $9 \times 9 \times 16$, $7 \times 7 \times 16$, and $5 \times 5 \times 16$. (Note that in Fig. 21.7, the first two figures in the sizes are listed in the row labeled $r \times r$, whereas the last appears in the row labeled N.)

The reason for having locally connected layers was to retain localization of the data from the input image. In face analysis, we are far from the situation where any object has to be detected anywhere in the image: In fact, eyes will be predominantly in one place and mouth and nose predominantly in others; in addition, they will usually be in similar relative positions. Nevertheless, the additional numbers of parameters required for localization can only be afforded when a large labeled dataset is available for training. Finally, the second fully connected layer is fed to a K-way softmax layer which produces a probability distribution over all



Layer	Image	Front	C1	C2	L3	L4	L5	F6	F7	Output
N	3	3	32×3	16×32	16×16	16×16	16×16	4096	4030	4030
$n\times n$	152×152	152×152	142×142	63×63	55×55	25×25	21×21	1	1	1
$r\times r$			11×11	9×9	9×9	7×7	5×5	1	1	1
$s\times s$			1×1	1×1	1×1	2×2	1×1	1	1	1

Layer	S1
N	32×32
$n\times n$	71×71
$r\times r$	3×3
$s\times s$	2×2

FIGURE 21.7

Schematic of Taigman et al.'s DeepFace face recognition architecture. The previous architecture that this most strongly resembles is that of ZFNet in Fig. 15.10. However, it differs in including the "Front" nonneural frontalization module (see Section 21.7), reducing the six convolution layers to just two, reducing the three sampling (max-pool) layers to one, adding three locally connected layers L3–L5, and maintaining higher layer dimensions ($n \times n$) before moving on to the fully connected layers. The reason for having locally connected layers is to retain localization of the data from the input image, bearing in mind that face features such as eyes, nose, and mouth are likely to be similarly placed within each image and also need to be assessed as quite large individual features.

the class outputs. For the Facebook Social Face Classification dataset, $K = 4030$; for the LFW dataset, $K = 5749$; and for the YouTube Faces video dataset, $K = 1595$: all three of these datasets were used to test the DeepFace system. (In fact, the last two values of K need to be interpreted carefully, because the images and videos were used to compare *pairs* of faces.)

21.7 FRONTALIZATION

The idea of frontalization needs to be considered in some detail. The basic concept is that if all face images can be converted to a standardized frontal view,

comparison between test and training images will become far easier and much more accurate. The problem is quite how to achieve this: In the past few years, several approaches have been devised for the purpose—as we will see below. First, in spite of the fact that 3-D models had "fallen out of favor," Taigman et al. (2014) decided to proceed by incorporating fiducial points to guide the modeling process. They identified six fiducial points—at the centers of the eyes, tip of the nose, ends of the mouth, and center of the lower lip (Fig. 21.8A); starting with these locations, they carried out 2-D similarity transformations to move them to a set of anchor points. In fact, this operation did not allow for out-of-plane rotation, so 3-D modeling had to be carried out as well. Here, a further 67 fiducial points were *manually* placed on the 3-D model and fitted by a least squares procedure—a process involving Delaunay triangulation derived from the 67 fiducial points. Once this had been achieved, obtaining a frontalized image of the face became trivial. Note that an affine camera model was used in the calculation, so full perspective projection was *not* taken into account, and the result was stated to be "only an approximation"—though the results achieved using the model were nonetheless impressive, as indicated above.

Sagonas et al. (2015) developed a totally different approach to frontalization. They started with the idea that, by collecting sets of frontal images, all that was necessary was to reexpress any new image in terms of these, i.e., we use the set

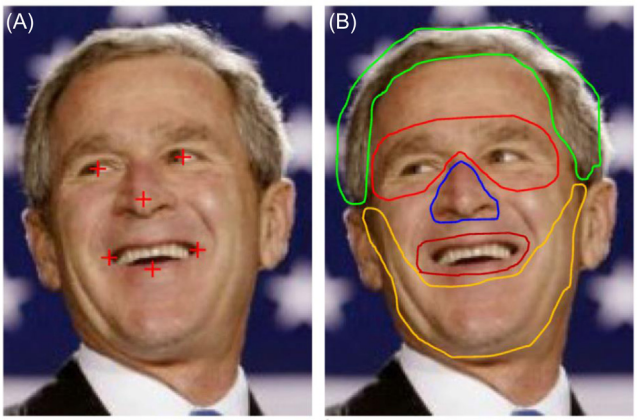


FIGURE 21.8

Facial features sought by several prominent methods. (A) Shows the six landmark points used by Taigman et al.'s Deepface recognition method—namely, the centers of the eyes, tip of the nose, ends of the mouth, and center of the lower lip. (B) Shows the five facial regions used by Yang et al.'s Faceness-Net detection system. Note that Sun et al.'s DeepID face recognition scheme started by detecting all the landmark points in (A) *except* for the center of the lower lip. Finally, note that the eye centers marked with red crosses in (A) do not coincide with iris centers, and this has implications about how the resulting landmarks are used in practice.

of frontal images as a basis set spanning a generic frontal face subspace, and then warp any new nonfrontal image to express it optimally as a linear combination of images in this basis set. In fact, it is clear that as a face is rotated in front of a camera, there will be one point at which an optimal match with an appropriate frontal image will occur, and at this point, the "nuclear norm" has been reached. The full calculation expresses this idea. However, it also has to express another fact: that when a face is viewed nonfrontally, part of it may well not be visible from the camera. This arises because even at quite small angles, the nose tends to partially obscure a small part of the cheek behind it (Fig. 21.9), whereas for larger rotations, an entire cheek, and also an ear, is likely to be occluded; in addition, a substantial region immediately under the chin will often be occluded. These factors play havoc with the optimization problem as so far described, because there is no possible single-valued warping that can cover the situation. In fact, part of the nonfrontal image has to be cut away. This immediately means that least squares fitting on its own will not work: Instead, it has to be replaced by, or

FIGURE 21.9

Face frontalization occlusion problem. Here, the author's face is viewed frontally, but the illumination is mainly from a light source about 40° to the right. The shadow to the left of the nose indicates the region that will *not* be visible from a camera placed at the light source position. Hence, if face frontalization is carried out for such a camera, the occluded region will have to be replaced by using either a transformed and reflected version of the face or by a trained map from a set of frontally viewed faces. The latter possibility is employed by the FAR technique devised by Sagonas et al. (2015).

combined with, l_1 norm analysis. Sagonas et al. developed a systematic iterative procedure for achieving this: It automatically eliminated occlusions in the form of large-magnitude sparse errors. The fact that the algorithm typically took ~ 100 iterations to achieve this meant that it could successfully manage progressively larger occlusions—of the sort that can occur for head rotations of up to $\sim 30^\circ$. The authors compared the effectiveness of their "Face frontalization for Alignment and Recognition" (FAR) technique with the 3-D modeling approach of Taigman et al., and found that the average r.m.s. error for FAR and DeepFace were 0.082 and 0.103, respectively. They explained their improved performance as due to not using any kind of 3-D modeling in their analysis. Overall, the FAR approach is superior to many previous approaches for producing statistical models of images: This is because it uses only a few hundred frontal images, which are simply obtained, rather than many thousands of labeled faces of many orientations.

Remembering that this is a fast-moving subject, it is worth drawing attention to a further approach to face frontalization, in this case, by Hassner et al. (2015). This method reverts to the 3-D modeling approach. However, it uses a standard 3-D surface and makes all images fit it accurately: It also extracts frontal face images from it by projection. In each case, a 3×4-projection matrix is used for the purpose, no attempt being made to use full perspective projection. It can be said that the method uses the standard 3-D surface as a proxy for moving face images around. To achieve all this, the authors use a total of 49 facial features, avoiding points along the jawline as these are not as distinct or as accurately defined as many others. In fact, the features used are all points lying close to the 3-D plane at the front of the face. Because of this, there is less reliance on the exact shape of the 3-D surface.

As is bound to happen, the nose and head as a whole can occlude parts of the face, and this is analyzed using the 3-D model surface. It soon becomes clear when this is happening, and in this situation, the visibility of each pixel in the frontalized picture is determined, using a clearly defined formula. Intensities of poorly visible pixels are replaced not merely by those of the corresponding pixels on the other side of the face but by a suitably weighted mean of the two intensities. The paper goes on to describe cases where this strategy does not work well—e.g., when the occlusion is due to other objects than the face itself (e.g., a hand or a microphone), when the face has an asymmetric expression, or when an eye is occluded: In this last case, a cross-eyed effect can result. There will also be problems when a person has a bandage or eye patch on one side of the face, or is wearing a monocle.

The advantage of this approach (in contrast with other 3-D methods) is that it results in retaining crisp details and sharp edges, and above all, it results in "aggressive alignment" and high accuracy. Even if the adherence to the 3-D model surface is slightly artificial and can result in absolute errors (e.g., potentially, a narrowed or widened face), it results in exceptional consistency. Essentially, the frontalizations lose negligible identifiable detail and remain

highly aligned. Remarkably, the lines on President Bush's face remain so well aligned from one derived frontal image to another that they are retained accurately and are recognizable in each case.

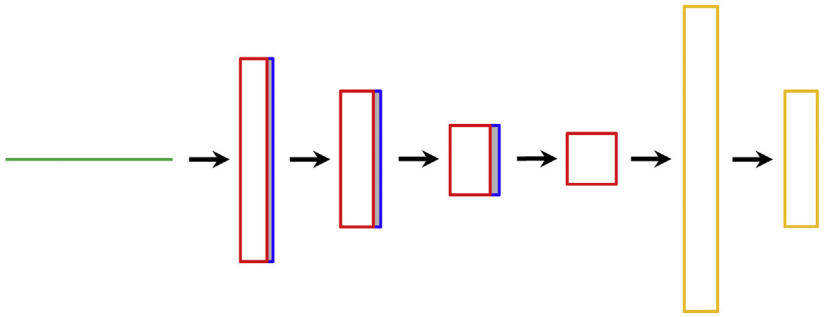
The authors tested the method using the "Image-Restricted, Label-Free Outside Data" (IRLFOD) protocol, developed in 2014 by Huang and Learned-Miller—two of the authors of the original LFW dataset (Huang et al., 2007). The reason that IRLFOD was produced was to act as even more rigorous and stringent test of algorithm performance. In fact, the authors' method gave the highest score (91.7%) achieved "to date" (i.e., 2015) reported in the IRLFOD category, attesting its quality as a high-performance approach: Numerically, its performance exceeded that of Cao et al. (2013) by $\sim 2\%$.

Of the three methods listed described above, the last probably requires the least computation and gives the highest alignment accuracy, though the various checks that have to be made to overcome problems that may have been introduced by the symmetry operation would appear to add a significant overhead. Interestingly, the method of Sagonas et al. should not have any such problems, as it takes standard frontal views as a basis set and automatically reverts to these (via the l_1 norm) when a fault due to self-occlusion arises. However, its optimization algorithm probably has a higher computational load. The Taigman et al. approach to frontalization seems to require excessive labor to set up (it involves the manual placement of 67 anchor points), and its reliance on a model using Delaunay triangulation would appear to cut down the achievable accuracy—as noted by both Sagonas et al. and Hassner et al. This has to be weighed against the close-to-human performance which their overall system achieved. It remains to be seen which schemes win in the end, as datasets such as LFW come with ever more stringent protocols setting out how to make fair comparisons using them. In this respect, Huang and Learned-Miller (2014) point out the need for care with "outside data" that is, not part of the dataset in question: Adding additional outside data to a given dataset can give clues to a classifier about the ground truth, thereby preventing a fair comparison with other classifiers.

21.8 THE SUN ET AL. DEEPID FACE REPRESENTATION SYSTEM

In Section 21.6, we looked closely at Taigman et al.'s approach to face recognition: At this point, it is worth comparing it with another recent method, which is claimed to be superior, but which follows a rather different route forward. That is, the method of Sun et al. (2014a,b). In fact, their DeepID face recognition scheme relies strongly on their face-point detector, which was published the previous year (2013). We shall consider the latter first.

Fig. 21.10 shows the F1 whole-face network for the detection of eyes, nose, and mouth: For this purpose, the specific features sought are the eye centers, the



Layer	Image	C1	C2	C3	C4	F5	F6
N	1	20	40	60	80	120	10
$n \times n$	39×39	36×36	16×16	6×6	2×2	1	1
$r \times r$		4×4	3×3	3×3	2×2	1	1
$p \times q$		2×2	2×2	3×3	2×2		

Layer	S1	S2	S3
N	20	40	60
$n \times n$	18×18	8×8	3×3
$r \times r$	2×2	2×2	2×2

FIGURE 21.10

Schematic of Sun et al.'s CNN facial point detection architecture. This diagram shows the F1 whole-face network for detection of eyes, nose, and mouth. Similar networks are used to detect eyes and nose (EN1) and nose and mouth (NM1). The three networks form the first level of a cascade, the next two quite similar levels being targeted at refining the location of the five features. In all layers, the stride s is kept at 1. Parameters p and q indicate that the convolutional layers are divided into $p \times q$ equal-sized patches within each of which the convolutional weights are shared. Note that for EN1 and NM1, the values of p and q are different from those listed above for F1. For cascade Levels 2 and 3, $p = q = 1$; so, the convolutional layers remain undivided, which is best for accurately locating low-level features. Note also that cascade Levels 2 and 3 have only two convolutional layers and one max-pooling layer. Overall, there are three networks in cascade Level 1, 10 in cascade Level 2, and 10 in cascade Level 3: For further details, see Sun et al. (2013).

tip of the nose, and the corners of the mouth (Fig. 21.8A). Similar networks are used to detect eyes and nose (EN1) and nose and mouth (NM1): The predictions of the three networks are averaged to increase location accuracy. The three networks form the first level of a cascade, the next two quite similar levels being targeted at refining the location of the five features: Carefully constructed rules are employed to ensure that the last two layers of the cascade don't pull the point locations too far. In all layers, the stride s is kept at 1. Parameters p and q indicate

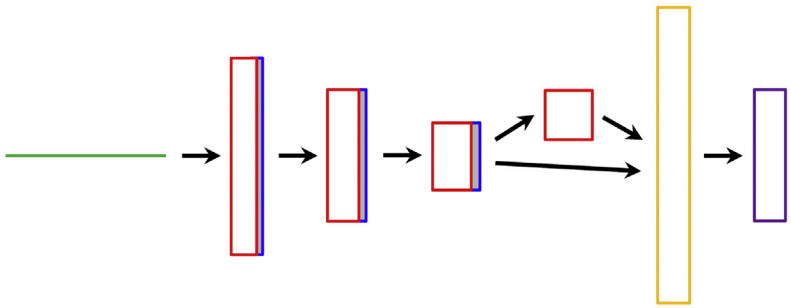
that the convolutional layers are divided into $p \times q$ equal-sized patches within each of which the convolutional weights are shared. Note that for EN1 and NM1, the values of p and q are different from those listed above for F1. For cascade Levels 2 and 3, $p = q = 1$, so the convolutional layers remain undivided, which is best for accurately locating low-level features. Note also that cascade Levels 2 and 3 have only two convolutional layers and two max-pooling layers, though both retain two fully connected layers. Overall, there are three networks in cascade Level 1, 10 in cascade Level 2 and 10 in cascade Level 3—all 23 of which are similar to, and in many cases, simpler than, that shown in Fig. 21.10: For further details, see Sun et al. (2013).

The original idea of using convolution networks was to save computation and to ensure positional invariance when locating objects. Here, the reason for using patches which internally share parameters is to specialize location of objects such as eyes, NM1 to various subregions. This was also seen in Taigman et al.'s DeepFace architecture, where layers L3−L5 were labeled as local. In the Sun face-point detector, there are tensions between reliable detection of the five types of feature and accuracy of location. Their architecture achieves this by several means: Using small patches for sharing weights, keeping detection of the five types of feature separate as far as possible, and progressively refining their locations (via the cascade levels). The type of nonlinearity used is the tanh function followed by an abs function (an empirical study showed that the latter led to improved performance).

Oddly, the paper says little about how sets of five keypoints were marked on the 10,000 images used in training and testing, which were obtained from the web or from the LFW dataset. However, the approach gave higher detection reliability and location accuracy than previous face-point detectors, including those of Belhumeur et al. (2011) and Cao et al. (2012), when tested on LFPW (labeled face parts in the wild). On the BioID test set, the new method gave detection failure rates approaching zero—better than five earlier methods. Interestingly, the method gave reliable detections under large variations in pose, illumination, and facial expression and also gave accurate predictions under near occlusion, e.g., when an eye was closed or the head rotated so far that an eye was barely visible.

We can now move to the DeepID face recognition scheme (Sun et al., 2014a,b), which uses the above face-point detector. In fact, it starts by detecting the five facial landmark points (eye centers, nose tip, and mouth corners) and aligns the face globally using a similarity transform. As the latter involves only in-plane translation, rotation, and scaling, by following the two eye centers and the midpoint of the mouth corners, the face is only weakly aligned. Such a similarity transform will convert a square into a square: By comparison, an affine transform will convert a square into a parallelogram (see Section 6.7.1), but a perspective transformation is needed to contort a square into a (convex) quadrilateral: This carries immediate lessons for realistic 2-D face representation. Nevertheless, as we shall see, the authors manage to get the recognition system to work impressively on this basis.

After aligning a face image, 10 rectangular patches are extracted from it: Five of these are global regions, and five are local, the latter being centered around the five landmark features. All 10 patches are scaled by three factors ~ 0.75, 1.0, ~ 1.2 and expressed (1) in gray scale and (2) in color, giving a total of 60 patches. Finally, 60 DeepID convents (Fig. 21.11) are trained to extract two 160-dimensional DeepID vectors using 10,000 original faces and leading to 10,000 identity classes. (The process can be called identification, verification, or recognition, depending on one's mind-set.) Notice that the total length of the DeepID vector is $160 \times 2 \times 60$, factor 2 signifying that the face has been flipped horizontally (though the patches around the eye centers and mouth corners are dealt with by flipping the opposite ones!).



Layer	Image	C1	C2	C3	C4	F5	Output
N	1	20	40	60	80	160	10000
$n \times n$	39×31	36×28	16×12	6×4	2×1	1	1
$r \times r$		4×4	3×3	3×3	2×2	1	1
$p \times q$		1×1	1×1	2×2	2×1		

Layer	S1	S2	S3
N	20	40	60
$n \times n$	18×14	8×6	3×2
$r \times r$	2×2	2×2	2×2

FIGURE 21.11

Schematic of Sun et al.'s DeepID face recognition architecture. This architecture starts with much the same form as that of Fig. 21.10, but in one crucial detail, it is different—in that layer, F5 is fed from C3 both directly and indirectly via C4. This prevents the small C4 from becoming a bottleneck in the information flow to F5—C4 nevertheless being important for capturing multiscale features. In all layers, the stride s is kept at 1. Parameters p and q indicate that the convolutional layers are divided into $p \times q$ equal-sized patches within each of which the convolutional weights are shared, though these operate mainly in the higher levels of the network: Weights in higher convolutional levels are locally shared to ensure that they can learn *different* higher level features in different regions.

Next, it ought to be emphasized that the locations of the five landmark points are only brought into the learning system via the patches that are centered on them. The system has to learn for itself what the meanings of the individual patches are, and the fact that half of them are global patches and half are landmark-centered.

An even more important point is that, when the network has learned to recognize the 10,000 face identities under quite variable conditions, it has a highly accurate knowledge of the compact identity-related features residing in its topmost layers—principally the 60 F5 layers. Indeed, after so much training, the system can be described as having "overlearnt" the features characteristic of a human face. We can say that the 60 F5 layers contain complementary sets of vectors and form an overcomplete representation. Also, because the DeepID output vector is so large, the learned features are far from being overfitted to the data, but on the contrary should be well generalized to faces not seen during training. Above all, the number of neurons in the last hidden layer is much smaller than the number in the output layer: This forces the last hidden layer to learn "shared hidden representations" (Sun et al., 2014a,b) for faces of different people, and to be both discriminative and compact.

Although it is easy to claim that the method is well generalized to faces not seen in training, and the paper does not expand on this, there is one way in which it is obvious that it has actually achieved this. Notice first that each face image is trained on CelebFaces + as well as LFW, so that the number of face images available is $\sim$200,000 and the number of identities (people) is $\sim$10,000, i.e., there are $\sim$20 faces per identity. Even with this number, the method is able to cope with the huge number of possible 3-D poses, which means it is successfully interpolating between them, and above all, it is doing so without making any overt 3-D models. The pure training and the method's intrinsic generalization capability is managing to achieve this. This is in direct contrast with Taigman et al.'s DeepFace approach. In fact, DeepID proved to be even more accurate than DeepFace, and to have classification performance of 97.45% compared with 97.25% for DeepFace, and 97.53% for human-level performance. Among the most valuable contribution to this performance came from the number of patches used per face image—namely 60—and amounted to an improvement of some 5.27%.

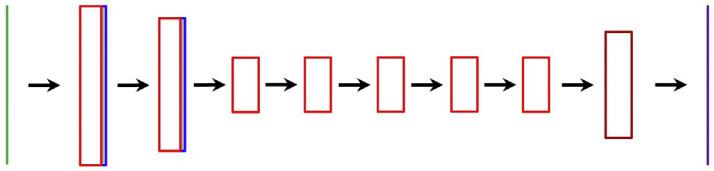
All this leads to a strong conclusion—that in spite of the obvious fact that faces are attached to a 3-D head that has considerable variation in expression (including joint articulation and variable degrees of opening of the eyes and mouth), high performance face recognition approaching that of the human can be achieved by systems with no built-in knowledge of 3-D and 3-D pose. One can ask whether the architecture itself has been constructed using human knowledge of 3-D. Looking at the architectures in Figs. 21.10 and 21.11, this does not seem to be the case. Admittedly, they have evolved to be sufficiently complex to achieve this, but in the main, the complexities are more to do with appreciation of possible low-to-high-level facial features than with anything to do with 3-D. One can even argue that the whole evolution of face recognition systems has only got where it has by ignoring 3-D aspects and freeing CNN-based learning systems to learn what they can in their own way.

21.9 FAST FACE DETECTION REVISITED

Having seen the considerable success that has been achieved in face classification, which in turn depends on facial feature detection and on face detection itself (face features only being found in face regions), it seems worth revisiting the problem of fast face detection. Following the ground-breaking work of Viola and Jones (2001), and the advances they made quite soon after (2004), Felzenszwalb et al. (2010) made a further big step forward by developing the concept of deformable parts models (DPM). These are based on the idea that faces can be considered collections of parts. Thus, to detect faces, it should only be necessary to locate the parts and examine their interrelationships. This can be carried out by identifying parts and their bounding boxes and then making proposals for combining them into larger bounding boxes representing (in Felzenszwalb et al.'s case) objects or in our case faces. Basically, once object or face bounding boxes have been found, these regions are protected from further analysis by nonmaximum suppression. In practice, this means giving each potential bounding box a score, keeping the highest scored and skipping any that overlap an already existing bounding box by a critical percentage, e.g., 50%. This highly successful approach achieved state-of-the-art results on the PASCAL VOC 2006, 2007, and 2008 benchmarks (Everingham et al., 2006, 2007, 2008) and "established itself as the de-facto standard for generic object detection" (Mathias et al., 2014).

Mathias et al. (2014) tested the DPM approach very thoroughly and showed that it could achieve top performance for face detection. They then went on to develop their own "HeadHunter" detector and showed that even if such a detector is based on *rigid* templates, it can also realize near-top performance: e.g., the two methods gave respective responses of 97.21% and 97.14% on the AFW test set: see Zhu and Ramanan (2012) for details of this test set. The conclusion they drew was that "parts are useful but not critical to reach top performance." The main problem with the rigid template approach was found to be its need for large quantities of training data.

Yang et al. (2015a,b) called HeadHunter "the state-of-the-art method": They then went on to show that their own new method outperformed it by 2.91% on the FDDB (face detection data set and benchmark) benchmark (defined by Jain and Learned-Miller, 2013). In fact, Yang et al.'s Faceness-Net was amongst the first face detectors to be developed along CNN lines. First, Yang et al. devised a CNN architecture for finding attributes of face images (Fig. 21.12). This was to be run forward to find the intrinsic face features and "in reverse"—by up-sampling—to regenerate localized face-part response maps: Here, they followed the methods set out by Zeiler and Fergus (2014), Simonyan et al. (2014), and Noh et al. (2015) on deconv nets, as outlined in Chapter 15, Deep Learning Networks. Then, face proposals were produced with the aid of a pipeline, using a faceness score to rank the resulting bounding boxes. Finally, nonmaximum suppression was used to identify the most reliable set of bounding box proposals.



Layer	Image	C1	C2	C3	C4	C5	C6	C7	Deconv	Map
N	1	96	256	512	512	1024	512×5	512×5	5	1
n×*n*	256×256	84×84	42×42	21×21	21×21	21×21	21×21	21×21	256×256	256×256
r×*r*		7×7	5×5	3×3	3×3	3×3	3×3	3×3		
s×*s*		3×3	1×1	1×1	1×1	1×1	1×1	1×1		

Layer	S1	S2
N	96	256
n×*n*	42×42	21×21
r×*r*	3×3	2×2
s×*s*	2×2	2×2

FIGURE 21.12

Schematic of Yang et al.'s Faceness-Net face detection architecture. [This schematic is as close to the Faceness-Net architecture as can be achieved from the details given in the paper by Yang et al. (2015a,b), which is slightly incomplete, e.g., in regard to stride, padding, convolution dimensions, and image sizes. In addition, compressing the whole architecture into one diagram itself risks introducing inaccuracies in order to make the underlying structure clearer to the reader.] The basic structure is that of a standard convolution net, with seven convolutional layers C1–C7 and two max-pooling layers S1, S2. There follows a Deconv network that effectively reverses the previous Conv network by applying unpooling and up-sampling units: This can be construed as taking the strongest activations in C7 and tracking them forward to a full-size face-parts map (see column heading "Deconv"). Note that layers C1–C5 are shared and C6 and C7 are distributed over five times as many channels in order to lead to five different face attributes: See "$\times 5$" (twice) under row heading "*N*." These also lead to the five face-part results produced by the end of the Deconv operation. Finally, by optimally combining face-part bounding boxes, these are grouped together to produce a single faces map: See text for details.

Much of the success of this method was due to adequately defining the five categories of part faces and indicating their likely relative placement. The five collective categories were labeled as follows (moving downwards from the top of the face, as in Fig. 21.8B):

1. "Hair": Including possible colors, waviness, straightness, baldness, hairline, and fringe;
2. "Eye": Including eyebrow and eye attributes, bags under eyes, and glasses;
3. "Nose": Including size and shape;
4. "Mouth": Including size of lips, openness of mouth, and use of lipstick; and
5. "Beard": Including presence of beard, sideboards, goatee, mustache, and stubble.

Note that hair is categorized differently in the top, middle, and bottom half of the face. As for their likely spatial arrangements, the various attributes should appear in the top-to-bottom order given above—though allowing for any items that are missing, hidden, or occluded. The faceness score would be penalized for any inconsistencies. The scores would then be used to rank potential object proposals for whole faces. At each stage, a bounding box would be generated and (trained) bounding box regression used to predict the optimum position for each final face proposal.

The above approach owes much to the original VJ method in its use of a pipeline, to Zeiler and Fergus and others for the deconv net approach, and to Felzenszwalb et al. (2010) for their DPM methodology, including the use of non-maximum suppression.

In this context, it should be noted that Bai et al. (2016) have produced a very much simpler and faster design: This is a fully convolutional network working at multiple scales, with five shared convolutional layers followed by a branching out into two further convolutional layers—the latter, respectively, coping with (1) the multiple scales and (2) the sliding window effect needed to perform the final matching. The network makes no use of pooling, though a convolution layer with a stride of 2 is used following each of the first three layers. Bai et al. compared their method with what they regarded as the prior leading approaches—Faceness-Net (Yang et al., 2015a,b), HeadHunter (Mathias et al., 2014), DenseBox (Huang et al., 2015), and others. Their conclusion was that their method outperformed all but two of the state-of-the-art methods (see below), while (because of its greater simplicity and efficiency) keeping real-time performance. In particular, it achieved an average precision of 97.7% on the AFW dataset, compared with 97.2% for Yang et al.'s Faceness-Net detector, with an even wider margin for all the others: Arguably, this is because it is simply trained end-to-end rather than having to go through deconvolution with the possibility of introducing further uncertainties or inaccuracies. Furthermore, on the PASCAL face dataset (Yan et al., 2014), it performed slightly less well (91.8%) than Faceness-Net (92.1%) but outperformed all the other methods. And, on the FDDB dataset, it managed to beat all the tested *CNN-based* detectors other than DenseBox. In this context, note that DenseBox used three times as much training data as Bai et al.'s method. Overall, it seems that a simpler system that is trained end-to-end has a better chance of achieving superior performance.

21.9.1 EVEN MORE POWERFUL OBJECT DETECTION SCHEMES

In Section 21.9, we saw something of the power of the Felzenszwalb et al.'s (2010) DPM approach to object detection. Interestingly, this approach is already in the process of being ousted by alternative approaches, one of which is the "Regions with CNN features" (R-CNN) method (Girshick et al., 2014). The latter involves using region proposal methods to generate potential bounding boxes: These are then classified and refined before duplicate detections are eliminated and the remaining ones are rescored. Overall, this is a slow and complex process,

though it has been speeded up in later versions (Girshick, 2015; Ren et al., 2015; Lenc and Vedaldi, 2015). However, all these versions have been radically improved upon by the YOLO approach (Redmon et al., 2015). YOLO stands for "You only look once," which means that all the multiple pipeline processes of the previously mentioned methods are dealt with in just one pass through the convolution net: During this one pass, it notes all bounding box possibilities and gradually refines the judgments made about each possibility, until by the end of the single pass a set of mutually consistent decisions has been arrived at. Although this may initially seem like an impossibility, this merely shows that the approach is ahead of its time, and there is nothing impossible about it. That is, not to say that the method has no problems and no faults. Indeed, it has some difficulty in coping with multiplicities of small objects, and its main fault is that of inaccurate object localization. On the other hand, it makes far fewer background mistakes than other methods: Specifically, it makes far fewer false predictions of objects in background regions.

Another important factor is that the single-pass approach is far faster and results in immediate real-time performance, at rates $\sim$45 frames per second. The main feature of the architecture is that it contains 24 convolutional layers followed by two fully connected layers and ends very neatly with a $7 \times 7 \times 30$ tensor consisting of a 7×7 image grid containing the five parameters x, y, w, h, and *confidence* (all times two to allow for touching objects) + 20 conditional class probabilities, $Pr(Class_i|Object)$. It ought to be added that an even more impressive version of this approach has already been published (Redmon and Farhadi, 2016). This detects objects in 200 classes in more than 9000 different object categories and is still fast enough to run in real time. The many refinements described in the paper cannot be covered fully here, but it is interesting that the final architecture (called "Darknet-19") has 19 convolutional layers and 5 maxpooling layers and is able to perform joint classification and detection.

21.10 THE FACE AS PART OF A 3-D OBJECT

It has been noted several times in Sections 21.6–21.8 that workers have used a 2-D similarity transform—or at most an affine transform—for modeling facial features, in spite of the fact that the head is a 3-D object. They have done this (1) in the interests of simplicity and (2) to keep computation within reasonable limits. However, there is the possibility that it would sometimes be useful to apply more accurate methods so as to improve classification performance. Hence, we indicate below how the analysis might be carried out.

To proceed, note that we can define a plane Π containing the outer corners of the eyes and the mouth. To a very good approximation, it can also be assumed that the inner corners of the eyes will be in the same plane (Fig. 21.13A–C). The next step is to estimate the position of the vanishing point V for the three horizontal lines λ_1, λ_2, λ_3 joining these three pairs of features (Fig. 21.13D). Once this

has been done, it is possible to use the relevant cross-ratio invariants (see Chapter 18: Invariants and perspective) to determine the points that in 3-D lie mid-way between the two features of each pair: This will give the symmetry line λ_s of the face. It will also be possible to determine the horizontal orientation θ of

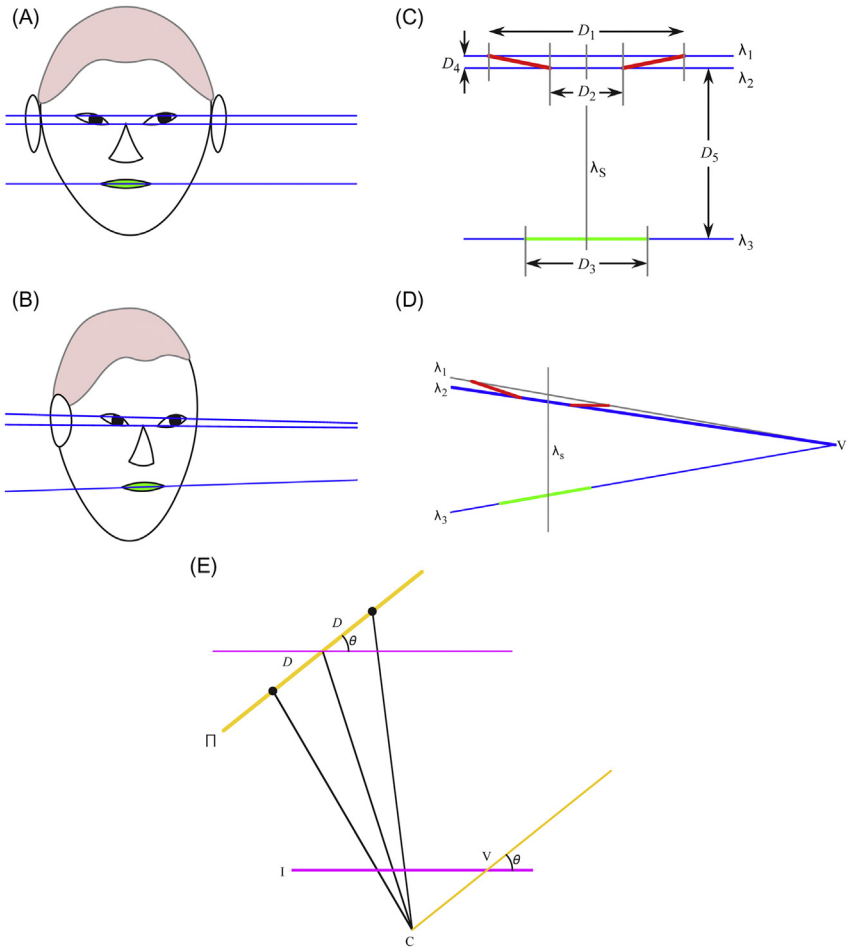


FIGURE 21.13

3-D analysis of facial parameters. (A) Front view of face. (B) Oblique view of face, showing perspective lines for corners of the eye and mouth. (C) Labeling of eye and mouth features and definition of five interfeature distance parameters. (D) Position of vanishing point V under an oblique view. (E) Positions of one pair of features and their midpoint on the facial plane Π. Note how the midpoint is no longer the midpoint in the image plane I when viewed under perspective projection. Note also how the vanishing point V gives the horizontal orientation of Π.

the facial plane Π, i.e., the (yaw) angle through which it has been rotated about a vertical axis, from a full frontal view. The geometry for these calculations is shown in Fig. 21.13E. Finally, it will be possible to convert the interfeature distances along λ_1, λ_2, λ_3 to the corresponding full frontal values, taking proper account of perspective, but not yet taking account of the vertical orientation φ of Π, which is still unknown. Note that the theory underlying these procedures is closely related to that of Section 18.8: see also Fig. 18.13.

In fact, there is insufficient information to estimate the vertical orientation φ (the pitch) without making further assumptions: Ultimately, this is because the face has no horizontal axis of symmetry. If we can assume that φ is zero (i.e., the head is held neither up nor down, and the camera is on the same level), then we can gain some information on the relative vertical distances of the face, the raw measurements for these being obtained from the intercepts of λ_1, λ_2, λ_3 with the symmetry line λ_s. Alternatively, we can assume average values for the interfeature distances and deduce the vertical orientation of the face. A further alternative is to make other estimates based on the chin, nose, ears, or hairline, but as these are not guaranteed to be in the facial plane Π, the whole assessment of facial pose may not then be accurate and invariant to perspective effects.

Overall, we are moving toward measurements either of facial pose or of facial interfeature measurements, with the possibility of obtaining some information on both, even when perspective distortions have to be allowed for (Kamel et al., 1994, Wang et al., 2003). Of course, the analysis will be significantly simpler in the absence of perspective distortions, when the face is viewed from a distance, or when a full frontal view is guaranteed. Indeed, the bulk of the work on facial recognition and pose estimation to date has been done in the context of weak perspective, making the analysis altogether simpler. Even then, the possibility of wide varieties of facial expression brings in a great deal of complexity. Clearly, the face is not merely a rubber mask (or deformable template) which can be distorted "tidily": The capability for opening and closing the mouth and eyes creates additional nonlinear effects that are not modeled merely by stretching rubber masks.

21.11 CONCLUDING REMARKS

Over many years, face recognition has been an ongoing target for practitioners of computer vision. On a number of occasions, it has been claimed that the problem has been solved, only for criminologists to deny this and confirm that there is still some distance to go before reliable identification of people can be attained in *practical* situations. Not least, there are problems of hats, glasses, hairstyles, beards, degree of facial stubble, wildly variable facial expressions, and of course variations in lighting and shadow—and this does not even touch on the problem of deliberate disguise. Added to this, there is the all-too-obvious problem that the face is not flat, but part of a solid, albeit malleable object—the head—which can

appear in a variety of orientations and positions in space. Interestingly, it is part of the human psyche and mind-set that the face is representable as a flat photograph: humans are so good at understanding imagery that they don't perceive what they are really seeing, i.e., the process of seeing is a nonpassive one, and the 2-D input is confused with its 3-D interpretation.

Of course, criminologists are not the only workers who need face recognition algorithms and methodology. Indeed, there is also the need to measure facial expression, for reasons as diverse as determining whether a person is telling the truth, and finding how to mimic real people as accurately as possible in films (over the next few years, it is possible to anticipate that a good proportion of films will contain no human actors as this has the potential for making them quicker and cheaper to produce). In addition, medical diagnosis or facial reconstruction can also benefit from facial measurement procedures. Nevertheless, person verification for computers, banks, and other security applications is vital in many walks of life, though it needs to be carried out quickly and with negligible error. In the latter respect, efforts moved in the direction of identifying people highly accurately from their iris patterns (e.g., Daugman, 1993, 2003), and even more accurately from their retinal blood vessels, using the methods of retinal angiography. Here, some of the main signals are commercial rather than academic, though an important message is that it only becomes cost-effective to deal with the technical difficulties where there is need for the highest security: in that case, "the false acceptance rate for a correctly installed retina scan system falls below 0.0001 percent" (ru.computers.toshiba-europe.com: website accessed 19 May 2004). Although retinal methods are bound to be rather expensive to implement, the iris method is not, and much progress has been made in this direction.

In this chapter, we have only covered the beginnings of this subject but have successfully broached the topics of FDR, and of facial feature location: Note that the latter is a key to FDR but is also useful in its own right, e.g., for lip reading and for analyzing eye movement and attention patterns. The history of the subject has proceeded in distinct jumps, none being more ground breaking than the ultrafast yet highly effective Viola–Jones (2001) face detector. Subsequently, considerable advances were made by Mathias et al. (2014)—on the back of vital work by Felzenszwalb et al. (2010) and others, on object detection using trained part-based models. However, these advances were immediately followed by breakthroughs using deep neural networks: See, for example, Yang et al. (2015a,b).

Likewise, face recognition followed a similar path, with very substantial improvements in classification performance from $\sim$87% around 2007 to $\sim$97% subsequently—notably by Taigman et al. (2014), Sun et al. (2014a,b), Sagonas et al. (2015, 2016), and Bai et al. (2016)—all of these achieving close-to-human performance levels, but again obtaining their successes by means of convolutional networks. As already outlined in Chapter 15, Deep Learning Networks, CNNs only took off in 2012 (in the sense of suddenly being seen to be capable of outperforming standard approaches such as SVMs)—a factor that has already hit FDR hard. It has permitted training and testing on realistic databases such as

LFW and has necessitated improved protocols for use of these datasets. It seems clear that deep networks are here to stay, though just as the internal combustion engine survived the arrival of the jet engine, so will the standard vision algorithms from the pre-2012 era and probably new ones from the next swing of the pendulum. Meanwhile, face recognition is in a far more comfortable state than it was less than a decade ago.

The eigenface method was the earliest systematic approach to face analysis and held promise of coping with huge numbers of facial variations, though it was soon overtaken by the Fisherface method. Face detection was also of vital importance and, when the boosting-based method of VJ emerged, it was 15 times faster than previous methods. Meanwhile, face recognition was found to be vastly inferior to human recognition on faces "from the wild." Eventually, the real breakthrough came when applying deep learning to the task, using hundreds of thousands of faces and even more face patches, allowing "overlearning" to take place. By 2016, deep learning had also rerevolutionized face detection.

21.12 BIBLIOGRAPHICAL AND HISTORICAL NOTES

Amongst the earliest systematic approaches to face analysis and recognition was the eigenface method which was invented by Sirovich and Kirby (1987) and developed into a successful practical technique by Turk and Pentland (1991). The basic idea was to take a set of face images as vectors in a face space, and to perform PCA to form a basis set of standard face vectors. Much effort was placed on this approach as it held the promise of analyzing faces into types, together with their modes of variation—interfacial and intrafacial (the latter including facial expressions), and even including different facial poses and lighting conditions. However, all this was too much for a single method to handle, and the Fisherfaces approach of Belhumeur et al. (1997) evolved out of it, in particular being less sensitive to lighting conditions and not needing the arbitrary elimination of the first three principal components. Indeed, Fisherfaces were found to reduce classification error rates from $\sim 24\%$ to $\sim 7\%$—a notable success.

Another aspect of the face analysis problem was that of rapid detection. In 2001, VJ cut right across the existing methods and developed a highly novel approach using boosting to achieve the required speed and power. They also reduced the feature set to rudimentary rectangular-shaped Haar filters and achieved a 15-fold speed improvement over the best previous detector (that of Rowley et al., 1998): This amounted to a staggering breakthrough whose possibility had not hitherto been dreamed of.

By 2007, there was a contention between 2-D and 3-D-based face recognition algorithms, and performance was stuck at successful recognition rates $\sim 87\%$. At this point, it became apparent that face recognition was far from being a mature field as performance lagged well behind human performance on real images

"from the wild." Thus, attention shifted from controlled to uncontrolled environments, and the LFW database was introduced (Huang et al., 2007). Although this led to an immediate drop in performance from available algorithms, by 2009, performance had again risen markedly: For example, Wolf et al. (2009) claimed 89.5% classification accuracy. However, the tide turned markedly in 2014 when Taigman et al. published their "DeepFace" (deep learning) approach to face recognition. This brought the performance level up to 97.35%, compared with a human performance level of 97.53%. Key to their success was the initial "frontalization" technique aimed at standardizing faces to a symmetric frontal view. Sagonas et al. (2015, 2016) developed their own frontalization technique obtaining what can best be described as a trained eigenset of frontal images. At about the same time, Yang et al. (2015a,b) developed a high-performance Faceness-Net face detector using a CNN architecture for finding attributes of face images (see also Yang et al., 2017). This was run forward to find the intrinsic face features and "in reverse" to regenerate localized face-part response maps: Here, they followed the coding−decoding procedures set out by Zeiler and Fergus (2014) and others. Finally, Bai et al. (2016) produced a very much simpler and faster design: This is a fully convolutional network working at multiple scales, with five shared convolutional layers followed by a branching out into two further convolutional layers—the latter, respectively, coping with (1) the multiple scales and (2) the sliding window effect needed to perform the final matching. An interesting and actually quite powerful conclusion is that a simpler system that is trained end-to-end has a better chance of achieving superior performance than one that is put together using separately pretrained sections.

Overall, although face recognition is in a far more comfortable state than it was less than a decade ago, it is difficult to say that the subject has stabilized. To a large extent, we are now in an area where the controlling algorithms are completely trained, and—apart from principled approaches such as the new one based on frontalization—we are arguably in a situation wherein we have relatively little knowledge of what the computer recognition system is actually doing and how close to optimality it actually is. But at least the introduction of CNN-based systems has given the subject a distinct shakeup and moved it from the doldrums to broad, sunlit uplands (cf. Part of the speech made by Winston Spencer Churchill in June 1940, following the Battle of Britain; "... the life of the world may move forward into broad, sunlit uplands").