Presented at the *IEEE International Conference on Computer Vision*, Kerkyra, Greece, Sept. 20-25, 1999.

Transformed Component Analysis: Joint Estimation of Spatial Transformations and Image Components

Brendan J. Frey
Computer Science, University of Waterloo
Beckman Institute, University of Illinois

Nebojsa Jojic
Beckman Institute, University of Illinois
at Urbana-Champaign

Software available at www.cs.uwaterloo.ca/~frey

Abstract

A simple, effective way to model images is to represent each input pattern by a linear combination of "component" vectors, where the amplitudes of the vectors are modulated to match the input. This approach includes principal component analysis, independent component analysis and factor analysis. In practice, images are subjected to randomly selected transformations of a known nature, such as translation and rotation. Direct use of the above methods will lead to severely blurred components that tend to ignore the more interesting and useful structure. In previous work, we introduced a clustering algorithm that is invariant to transformations [1]. In this paper, we propose a method called *transformed component analysis*, which incorporates a discrete, hidden variable that accounts for transformations and uses the expectation maximization algorithm to jointly extract components and normalize for transformations. We illustrate the algorithm using a shading problem, facial expression modeling and written digit recognition.

1 Introduction

Many popular ways of modeling images use a linear combination of vectors to represent each input. Principal components analysis (PCA, a.k.a. "eigen-*whatever*") is a way of representing a class of images using a small set of component vectors in the vector space of image pixel intensities [2]. An image can be described by a linear combination of these components plus some distortion. The first principal component is the direction in which the projected training data has greatest variance, the second principal component is the direction of greatest variance *after* the first component is subtracted off, and so on.

A different technique that is gaining in popularity is independent component analysis (ICA), which tries to find components such that when the training data is projected on these components, the component activities are independent (not just uncorrelated) [3]. ICA has been used for blind separation and deconvolution.

Factor analysis (FA) [4] is a generative model that is similar in spirit to PCA. A *generative model* is trained to generate patterns that look similar to the training data and the representation of an input is the posterior distribution over some hidden variables. In a FA model, the distribution over a small set of real-valued hidden variables is a zero-mean unit-covariance Gaussian and the distribution over the inputs given the hidden variables is also Gaussian, with a diagonal covariance matrix and a mean given by a linear combination of the hidden variables. A factor analyzer can be fit to training data using the EM algorithm [5].

In practice, data is often subjected to randomly selected transformations of a known nature, such as translation and rotation in images. In these cases, direct application of the above methods will produce severely blurred components that mostly account for the transformations and ignore the more interesting and useful structure.

We propose a method called *transformed component analysis*, which incorporates a discrete, hidden variable that accounts for transformations and uses the expectation maximization algorithm to jointly extract components and normalize for transformations. After illustrating the algorithm using a toy problem and facial expression modeling, we give results on classifying images of written digits and compare the components found by transformed component analysis with the components found by PCA and FA.

2 The Transformed Component Analyzer (TCA)

A transformed component analyzer (TCA) is a probability model that specifies how the linear combination of a set of component vectors is transformed in different ways to model input patterns. The component activities (amplitudes) $\mathbf{y}$, which form a subspace representation of an image, are assumed to be independent and Gaussian:

$$p(\mathbf{y}) = \mathcal{N}(\mathbf{y}; \mathbf{0}, \mathbf{I}), \tag{1}$$

where $\mathbf{I}$ is the identity covariance matrix. A latent image $\mathbf{z}$ is produced by combining these components linearly using a "factor loading matrix" $\mathbf{\Lambda}$, offsetting the resulting image by an image mean $\boldsymbol{\mu}$ and then adding independent Gaussian noise to each pixel:

$$p(\mathbf{z}|\mathbf{y}) = \mathcal{N}(\mathbf{z}; \boldsymbol{\mu} + \mathbf{\Lambda}\mathbf{y}, \mathbf{\Phi}), \tag{2}$$

where $\mathbf{\Phi}$ is a diagonal covariance matrix.

We assume the image produced by the subspace model is further transformed to obtain the observed image. Although real-valued transformations (*e.g.*, rotation) are common, real-valued latent variables introduce complicated integrals (instead of summations) into the EM learning algorithm. So, we will assume there is a fixed set of possible transformations and that this set is specified beforehand. Also, the algorithm is simplified by assuming that the vector of pixel values for the transformed image is obtained by multiplying the vector of pixel values for the latent image by a *sparse* matrix. This permits a broad class of transformations, including translation, scale, in-plane rotation and out-of-plane rotation.

Let $\ell \in \{1, \ldots, L\}$ index the set of L transformations represented by matrices $\mathbf{G_1}, \ldots, \mathbf{G_L}$. The probability density of the vector of pixel values $\mathbf{x}$ for the image corresponding to transformation ℓ and latent image $\mathbf{z}$ is

$$p(\mathbf{x}|\ell, \mathbf{z}) = \mathcal{N}(\mathbf{x}; \mathbf{G}_\ell \mathbf{z}, \mathbf{\Psi}), \tag{3}$$

where $\mathbf{\Psi}$ is a diagonal covariance matrix that specifies the noise on the observed pixels.

The variances $\mathbf{\Phi}$ of the pixels in the latent image are quite different from the variances $\mathbf{\Psi}$ of the pixels in the observed image. The noise modeled by $\mathbf{\Phi}$ coheres to the latent image during transformations. In contrast, $\mathbf{\Psi}$ models noise in the observed pixels and this noise does not depend on the transformation.

The translation, scale, rotation, *etc.* corresponding to each ℓ are selected ahead of time so that the set of transformations effectively covers the range of possible transformations in the data and at the same time is small enough to keep the learning time reasonable. Transformation ℓ has a prior probability $P(\ell) = p_\ell$, which may be fixed to be uniform or learned from the training data.

The joint distribution over the observed image, the transformation index, the latent image and the subspace representation is

$$\begin{aligned} p(\mathbf{x}, \ell, \mathbf{z}, \mathbf{y}) &= p(\mathbf{y})P(\ell)p(\mathbf{z}|\mathbf{y})p(\mathbf{x}|\ell, \mathbf{z}) \\ &= p_\ell \mathcal{N}(\mathbf{y}; \mathbf{0}, \mathbf{I})\mathcal{N}(\mathbf{z}; \boldsymbol{\mu} + \mathbf{\Lambda}\mathbf{y}, \mathbf{\Phi})\mathcal{N}(\mathbf{x}; \mathbf{G}_\ell \mathbf{z}, \mathbf{\Psi}). \end{aligned} \tag{4}$$

In this generative model, the subspace representation, transformation index and latent image are unobserved variables. Generating an image from a TCA consists of drawing a subspace representation from $p(\mathbf{y})$, drawing a latent image from $p(\mathbf{z}|\mathbf{y})$, drawing a transformation index from $P(\ell)$ and then drawing an image from $p(\mathbf{x}|\ell, \mathbf{z})$.

The number of parameters in a TCA is roughly equal to the number of parameters in a standard factor analyzer and the number of "parameters" used in PCA. In Sec. 3, we show how the ML estimate of this model can be obtained using the EM algorithm.

2.1 Inferring the likelihood for an image and the responsibilities of the transformations

For a given image, it is useful to know how appropriate each transformation is for matching the image to the transformed components (the "responsibility" of the transformation) and overall how well the components match the image (the likelihood).

The responsibilities are the posterior probabilities of the transformation indices,

$$P(\ell|\mathbf{x}) = p(\mathbf{x}, \ell)/p(\mathbf{x}), \quad p(\mathbf{x}) = \sum_{\ell=1}^{L} p(\mathbf{x}, \ell), \tag{5}$$

where $p(\mathbf{x})$ is the likelihood. Both the responsibilities and the likelihood can be easily computed from $p(\mathbf{x}, \ell)$.

To obtain $p(\mathbf{x}, \ell)$, we integrate out the component activities $\mathbf{y}$ and the latent image $\mathbf{z}$:

$$\begin{aligned} p(\mathbf{x}, \ell) &= \int_{\mathbf{z}} \int_{\mathbf{y}} d\mathbf{z} d\mathbf{y} p(\mathbf{x}, \ell, \mathbf{z}, \mathbf{y}) \\ &= p_\ell \mathcal{N}(\mathbf{x}; \mathbf{G}_\ell \boldsymbol{\mu}, \mathbf{G}_\ell(\mathbf{\Lambda}\mathbf{\Lambda}' + \mathbf{\Phi})\mathbf{G}_\ell' + \mathbf{\Psi}), \end{aligned} \tag{6}$$

where " $'$ " indicates transpose. Each transformation ℓ has a corresponding mean image $\mathbf{G}_\ell \boldsymbol{\mu}$ and covariance matrix $\mathbf{G}_\ell(\mathbf{\Lambda}\mathbf{\Lambda}' + \mathbf{\Phi})\mathbf{G}_\ell' + \mathbf{\Psi}$.

For an N-pixel image, K components and L transformations, the responsibilities and the likelihood for an image can be computed in $\mathcal{O}(LKN)$ time, if we assume $|\mathbf{G}_\ell(\mathbf{\Lambda}\mathbf{\Lambda}' + \mathbf{\Phi})\mathbf{G}_\ell' + \mathbf{\Psi}| \approx |\mathbf{G}_\ell(\mathbf{\Lambda}\mathbf{\Lambda}' + \mathbf{\Phi})\mathbf{G}_\ell'|$. This corresponds to assuming the image noise is significantly lesser than the variability introduced by the transformed components (*i.e.*, the image structure).

2.2 Inferring the subspace representation

For a given image $\mathbf{x}$, the posterior distribution over the component activities is given by

$$p(\mathbf{y}|\mathbf{x}) = \sum_{\ell=1}^{L} p(\mathbf{y}|\mathbf{x}, \ell)P(\ell|\mathbf{x}), \tag{7}$$

where $P(\ell|\mathbf{x})$ is the responsibility of transformation ℓ from above. Given ℓ, the TCA is a multinomial Gaussian model, so $p(\mathbf{y}|\mathbf{x},\ell)$ is Gaussian and the above posterior is a mixture of Gaussians, where each Gaussian corresponds to a different transformation.

In contrast with the single point representation obtained from PCA and ICA, TCA gives a set of weighted points. The weight for point ℓ is $P(\ell|\mathbf{x})$ and the corresponding estimate is

$$\mathrm{E}[\mathbf{y}|\mathbf{x},\ell]=\boldsymbol{\beta}_\ell\boldsymbol{\Lambda}'\boldsymbol{\Phi}^{-1}[\boldsymbol{\Omega}_\ell\mathbf{G}'_\ell\boldsymbol{\Psi}^{-1}\mathbf{x}-(\mathbf{I}-\boldsymbol{\Omega}_\ell\boldsymbol{\Phi}^{-1})\boldsymbol{\mu}], \quad (8)$$

where

$$\boldsymbol{\Omega}_\ell = \mathrm{COV}(\mathbf{z}|\mathbf{x},\mathbf{y},\ell) = (\boldsymbol{\Phi}_c^{-1} + \mathbf{G}'_\ell\boldsymbol{\Psi}^{-1}\mathbf{G}_\ell)^{-1},$$
$$\boldsymbol{\beta}_\ell = (\mathbf{I} + \boldsymbol{\Lambda}'\boldsymbol{\Phi}^{-1}\boldsymbol{\Lambda} - \boldsymbol{\Lambda}'\boldsymbol{\Phi}^{-1}\boldsymbol{\Omega}_\ell\boldsymbol{\Phi}^{-1}\boldsymbol{\Lambda})^{-1}. \quad (9)$$

To compare the subspace representations of two images, the two corresponding sets of weighted points can be compared. Alternatively, the MAP point (the point with the highest weight) can be selected for each image and a Euclidean distortion can be measured.

3 Transformed Component Analysis using the EM Algorithm

In this section, we describe an iterative expectation maximization (EM) algorithm [5] for estimating the maximum likelihood parameters of a transformed component analyzer. We refer to this procedure as transformed component analysis (TCA). The EM algorithm consists of an E-step, which accumulates statistics that are sufficient for estimating the model parameters, and an M-step, which updates the model parameters using the sufficient statistics. Each iteration increases the likelihood of the training data.

Notation: $\mathrm{diag}(\mathbf{A})$ is the vector containing the diagonal elements of matrix $\mathbf{A}$; $\mathrm{diag}(\mathbf{a})$ is the diagonal matrix whose diagonal equals the vector $\mathbf{a}$; $\mathbf{a}\circ\mathbf{b}$ is the element-wise product of vectors $\mathbf{a}$ and $\mathbf{b}$.

3.1 Expectation step

The sufficient statistics for the M-Step are computed in the E-Step during a single pass through the training set. Before making this pass, the following matrices are computed:

$$\boldsymbol{\Omega}_\ell = \mathrm{COV}(\mathbf{z}|\mathbf{x},\mathbf{y},\ell) = (\boldsymbol{\Phi}_c^{-1} + \mathbf{G}'_\ell\boldsymbol{\Psi}^{-1}\mathbf{G}_\ell)^{-1},$$
$$\boldsymbol{\beta}_\ell = (\mathbf{I} + \boldsymbol{\Lambda}'\boldsymbol{\Phi}^{-1}\boldsymbol{\Lambda} - \boldsymbol{\Lambda}'\boldsymbol{\Phi}^{-1}\boldsymbol{\Omega}_\ell\boldsymbol{\Phi}^{-1}\boldsymbol{\Lambda})^{-1}. \quad (10)$$

For each case during the pass through the training set, $P(\ell|\mathbf{x}_t)$, $\mathrm{E}[\mathbf{y}|\mathbf{x}_t,\ell]$ and $\mathrm{E}[\mathbf{y}|\mathbf{x}_t] = \sum_\ell P(\ell|\mathbf{x}_t)\mathrm{E}[\mathbf{y}|\mathbf{x}_t,\ell]$ are first computed as described in Sec. 2.1.

For each ℓ, we then compute

$$\mathrm{E}[\mathbf{z}|\mathbf{x}_t,\ell] = \boldsymbol{\mu} + \boldsymbol{\Omega}_\ell\mathbf{G}'_\ell\boldsymbol{\Psi}^{-1}(\mathbf{x_t} - \mathbf{G}_\ell\boldsymbol{\mu})$$
$$+\boldsymbol{\Omega}_\ell\boldsymbol{\Phi}^{-1}\boldsymbol{\Lambda}\boldsymbol{\beta}_\ell\boldsymbol{\Lambda}'\boldsymbol{\Phi}^{-1}\boldsymbol{\Omega}_\ell\mathbf{G}'_\ell\boldsymbol{\Psi}^{-1}(\mathbf{x_t} - \mathbf{G}_\ell\boldsymbol{\mu}) \quad (11)$$

and obtain the following expectation:

$$\mathrm{E}[\mathbf{z}-\boldsymbol{\Lambda}\mathbf{y}|\mathbf{x}_t]=\sum_{\ell=1}^{L} P(\ell|\mathbf{x}_t)(\mathrm{E}[\mathbf{z}|\mathbf{x}_t,\ell]-\boldsymbol{\Lambda}\mathrm{E}[\mathbf{y}|\mathbf{x}_t,\ell]). \quad (12)$$

Two other expectations,

$$\mathrm{E}[(\mathbf{z}-\boldsymbol{\mu})\circ(\mathbf{z}-\boldsymbol{\mu})|\mathbf{x}_t,\ell]$$
$$= (\mathrm{E}[\mathbf{z}|\mathbf{x}_t,\ell]-\boldsymbol{\mu})\circ(\mathrm{E}[\mathbf{z}|\mathbf{x}_t,\ell]-\boldsymbol{\mu})$$
$$+ \mathrm{diag}(\boldsymbol{\Omega}_\ell) + \mathrm{diag}(\boldsymbol{\Omega}_\ell\boldsymbol{\Phi}^{-1}\boldsymbol{\Lambda}\boldsymbol{\beta}_\ell\boldsymbol{\Lambda}'\boldsymbol{\Phi}^{-1}\boldsymbol{\Omega}_\ell), \quad (13)$$

and

$$\mathrm{E}[(\mathbf{z}-\boldsymbol{\mu})\mathbf{y}'|\mathbf{x}_t,\ell]$$
$$= (\mathrm{E}[\mathbf{z}|\mathbf{x}_t,\ell] - \boldsymbol{\mu})\mathrm{E}[\mathbf{y}|\mathbf{x}_t,\ell]' + \boldsymbol{\Omega}_\ell\boldsymbol{\Phi}^{-1}\boldsymbol{\Lambda}\boldsymbol{\beta}_\ell, \quad (14)$$

are used to compute the expectation,

$$\mathrm{E}[(\mathbf{z}-\boldsymbol{\mu}-\boldsymbol{\Lambda}\mathbf{y})\circ(\mathbf{z}-\boldsymbol{\mu}-\boldsymbol{\Lambda}\mathbf{y})|\mathbf{x}_t] = \sum_{\ell=1}^{L} P(\ell|\mathbf{x}_t)$$
$$\cdot\big\{\mathrm{E}[(\mathbf{z}-\boldsymbol{\mu})\circ(\mathbf{z}-\boldsymbol{\mu})|\mathbf{x}_t,\ell] + \mathrm{diag}(\boldsymbol{\Lambda}\boldsymbol{\beta}_\ell\boldsymbol{\Lambda}')$$
$$- 2\mathrm{diag}(\boldsymbol{\Lambda}\mathrm{E}[(\mathbf{z}-\boldsymbol{\mu})\mathbf{y}'|\mathbf{x}_t,\ell]')$$
$$+ (\boldsymbol{\Lambda}\mathrm{E}[\mathbf{y}|\mathbf{x}_t,\ell])\circ(\boldsymbol{\Lambda}\mathrm{E}[\mathbf{y}|\mathbf{x}_t,\ell])\big\}. \quad (15)$$

The following expectations are also computed:

$$\mathrm{E}[(\mathbf{x}_t-\mathbf{G}_\ell\mathbf{z})\circ(\mathbf{x_t}-\mathbf{G}_\ell\mathbf{z})|\mathbf{x_t}] = \sum_{\boldsymbol{\ell=1}}^{L} \mathbf{P}(\ell|\mathbf{x_t})$$
$$\cdot\big\{(\mathbf{x}_t-\mathbf{G}_\ell\mathrm{E}[\mathbf{z}|\mathbf{x_t},\ell])\circ(\mathbf{x_t}-\mathbf{G}_\ell\mathrm{E}[\mathbf{z}|\mathbf{x_t},\ell])$$
$$+ \mathrm{diag}(\mathbf{G}_\ell\boldsymbol{\Omega}_\ell\mathbf{G}'_\ell)$$
$$+ \mathrm{diag}(\mathbf{G}_\ell\boldsymbol{\Omega}_\ell\boldsymbol{\Phi}^{-1}\boldsymbol{\Lambda}\boldsymbol{\beta}_\ell\boldsymbol{\Lambda}'\boldsymbol{\Phi}^{-1}\boldsymbol{\Omega}_\ell\mathbf{G}'_\ell)\big\}, \quad (16)$$

$$\mathrm{E}[(\mathbf{z}-\boldsymbol{\mu})\mathbf{y}'|\mathbf{x}_t] = \sum_{\ell=1}^{L} P(\ell|\mathbf{x}_t)\mathrm{E}[(\mathbf{z}-\boldsymbol{\mu})\mathbf{y}'|\mathbf{x}_t,\ell], \quad (17)$$

$$\mathrm{E}[\mathbf{y}\mathbf{y}'|\mathbf{x}_t]=\sum_{\ell=1}^{L} P(\ell|\mathbf{x}_t)(\boldsymbol{\beta}_\ell+\mathrm{E}[\mathbf{y}|\mathbf{x}_t,\ell]\mathrm{E}[\mathbf{y}|\mathbf{x}_t,\ell]'). \quad (18)$$

3.2 Maximization step

We use $\langle\cdot\rangle = \frac{1}{T}\sum_{t=1}^{T}(\cdot)$ to indicate a statistic computed by averaging the above expectations over the training set and "$\tilde{}$" to denote the updated parameters:

$$\tilde{\boldsymbol{\mu}} = \langle\mathrm{E}[\mathbf{z} - \boldsymbol{\Lambda}\mathbf{y}|\mathbf{x}_t]\rangle, \quad (19)$$
$$\tilde{\boldsymbol{\Phi}}=\mathrm{diag}\big(\langle\mathrm{E}[(\mathbf{z}-\boldsymbol{\mu}-\boldsymbol{\Lambda}\mathbf{y})\circ(\mathbf{z}-\boldsymbol{\mu}-\boldsymbol{\Lambda}\mathbf{y})|\mathbf{x}_t]\rangle\big), \quad (20)$$
$$\tilde{\boldsymbol{\Psi}} = \mathrm{diag}\big(\langle\mathrm{E}[(\mathbf{x}_t-\mathbf{G}_\ell\mathbf{z})\circ(\mathbf{x_t}-\mathbf{G}_\ell\mathbf{z})|\mathbf{x_t}]\rangle\big), \quad (21)$$
$$\tilde{\boldsymbol{\Lambda}} = \langle\mathrm{E}[(\mathbf{z} - \boldsymbol{\mu})\mathbf{y}'|\mathbf{x}_t]\rangle\langle\mathrm{E}[\mathbf{y}\mathbf{y}'|\mathbf{x}_t]\rangle^{-1}, \quad (22)$$
$$\tilde{p}_l = \langle P(\ell=l|\mathbf{x}_t)\rangle. \quad (23)$$

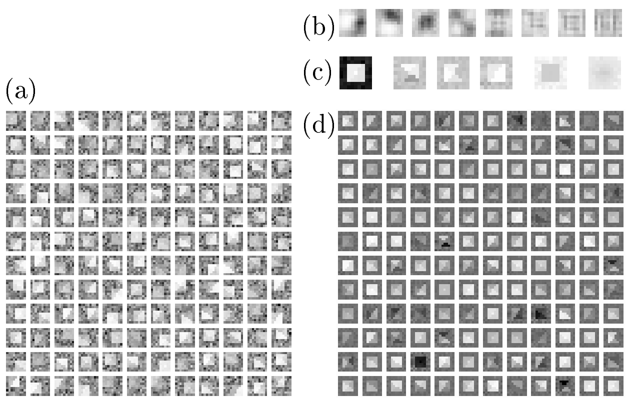

Figure 1: (a) Noisy images of a pyramid at different locations and under different lighting conditions. (b) The first 8 scaled principal components. (c) The mean, components, and noise deviation of a TCA with 3 components, after 10 iterations of EM. (d) Examples simulated from the TCA, without noise and without transformations.

In order to avoid overfitting the noise variances, it is sometimes useful to set the diagonal elements of $\mathbf{\Phi}$ and $\mathbf{\Psi}$ that are below some ϵ equal to ϵ.

4 Learning shape and lighting representations from noisy unaligned images of an object

Fig. 1a shows a training set of 144 9×9 noisy images of a uniformly colored pyramid (gray) at randomly selected positions and illuminated by parallel light rays with randomly selected angle and intensity. A cluttered background was simulated by randomly selecting pixel values from a uniform distribution.

The first 8 principal components of the training data, scaled by the standard deviation of the projected data, are shown in Fig. 1b. It appears the components implement a multiresolution approximation to model shifts of the object.

We trained a TCA with 3 components and 81 transformations implementing 9 horizontal and 9 vertical shifts using 10 iterations of the EM algorithm. To initialize the parameters, the mean and variance of each pixel was first computed from the training data. The parameters were then initialized to random values, using the mean and variance as a 1st order guide. The transformation probabilities p_ℓ were set equal.

Fig. 1c shows the mean latent image $\boldsymbol{\mu}$, the 3 columns of $\mathbf{\Lambda}$ (shown as 3 images), the latent image noise $\mathbf{\Phi}$ (shown as an image where the pixel intensity is equal to 4 times the standard deviation) and the observed image noise $\mathbf{\Psi}$. The mean clearly shows that the outline of the object has been determined and that the uniform coloring has been determined (except at the point of the pyramid). Linear combinations of the 3 components produce different lighting conditions (see the following paragraph) which implies that the 3-element *rows* of $\mathbf{\Lambda}$ are proportional to the object surface normals, up to some rotation in 3-dimensional space. The variance map for the latent image shows that the model predicts low variance for pixels belonging to the object, but high variance for other pixels (the background clutter). Finally, the variance map for the observed image accounts for the small amount of noise that is present in the images.

The TCA can be simulated in a noise-free transformation-free fashion, by drawing a subspace representation from $p(\mathbf{y})$, computing $\mathbf{z} = \boldsymbol{\mu} + \mathbf{\Lambda}\mathbf{y}$ and then computing $\mathbf{x} = \mathbf{G_1}\mathbf{z}$. Fig. 1d shows 144 examples simulated in this way. These "fantasies" show that the TCA can simulate the different lighting conditions.

Figure 2: Imperfectly aligned images of faces with different expressions.

5 Learning a subspace representation of facial expressions from imperfectly aligned images

Fig. 2 shows a training set of 100 16×24 images of automatically aligned faces with different expressions. The accuracy of the face detection algorithm used to align the images is +/-2 pixels in each direction.

Figure 3: (a) The mean and first 10 scaled principal components of the face data. (b) The mean, 10 components and noise deviations found by FA (TCA with only the identity transformation). (c) The mean, 10 components and noise deviations found by TCA.

Fig. 3a shows the mean of the training data and the first 10 principal components, scaled by the standard deviation of the projected data. The first 5 components obviously account for vertical, horizontal and diagonal shifts in the data and the remaining components are very blurred.

Fig. 3b shows the parameters for a FA model (a TCA with only the identity transformation) trained using 70 iterations of EM. The parameters were initialized using the mean and variance of each pixel in the training data. The sum of the two images on the far right of Fig. 3b gives the variance map for FA. In contrast to PCA, different components represent similar amounts of energy (variance). This is because FA does not find a preferred set of basis vectors (factors) for the subspace. Like PCA, FA finds very blurred components.

We trained a TCA with 10 components and 25 transformations implementing 5 horizontal and 5 vertical shifts using 70 iterations of the EM algorithm. The parameters were initialized using the mean and variance of each pixel in the training data. The transformation probabilities p_ℓ were set equal.

Fig. 3c shows the mean, components and variance maps. Unlike PCA and FA, TCA extracts clear components. The first component appears to expose some teeth, the second component appears to raise the eyebrows, raise the upper lip and expose a "tongue", and so on. The components found by TCA are unique up to a unitary transformation, so each component often includes more than one feature. A further processing step can be applied to find a unitary transformation that produces components with spatially localized energy.

To see how well the PCA subspace represents the data, we can draw a subspace point from an axis-aligned Gaussian with variances determined from the projected training data, and then use the principal components to map the point to image space. 100 ex-

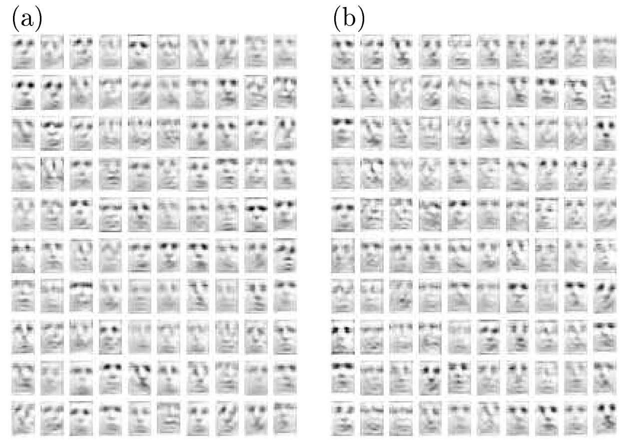

Figure 4: Examples of the face data simulated using (a) the PCA subspace and (b) the factor analyzer.

Figure 5: Examples of the face data simulated using the TCA model.

amples simulated in this manner are shown in Fig. 4a. Although the "faces" do appear to be shifted around the field of vision, they are also severely blurred. Fig. 4b shows examples simulated using the FA model, without adding sensor noise. They appear similar to the examples simulated using the PCA model.

Fig. 5 shows 100 examples of images simulated using the TCA, without the latent image and observed image noise and without randomly selected transformations. The images are much clearer than those simulated using the PCA subspace and the factor analyzer. The expressions in the training set are reproduced and the model also generates novel realistic expressions that are not present in the training set, such

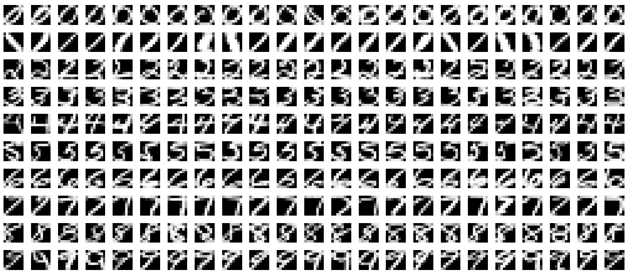

Figure 6: Example images from the training set of handwritten digits show a wide variety in writing style.

(a) (b)

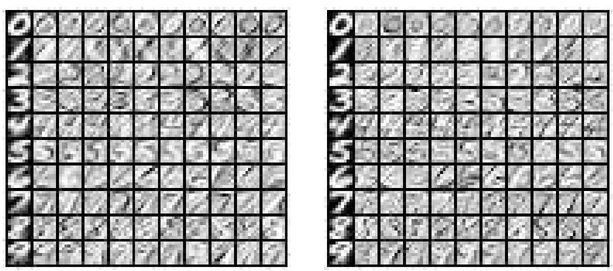

Figure 7: The means and components learned by (a) FA and (b) TCA applied to 200 images of each handwritten digit. Pixels for the means are colored using the same scale as the training images; pixels for the components are colored using intermediate gray to represent 0.

as the one in the 5th column of the 1st row and the one in the 1st column of the 3rd row.

6 Using shear transformations to improve handwritten digit recognition

Fig. 6 shows 230 images of handwritten digits taken from envelopes mailed by people in Buffalo [6]. A wide variety of handwriting styles are evident. The original images were affinely transformed to fit snugly in an 8×8 gray level image. Although this preprocessing helps to normalize for uniform scaling in the horizontal and vertical directions, it does not normalize for local scaling (*e.g.*, some 8s have smaller loops than others) or for shearing (*e.g.*, the vertical stroke of the 7 is at different angles).

For each class of digit, we trained a 10 component factor analyzer on 200 images using 30 iterations of EM. The parameters of each factor analyzer were initialized using the mean and variance of each pixel in the respective class. Fig. 7a shows the means and components found by FA and Fig. 8a shows images

(a) (b)

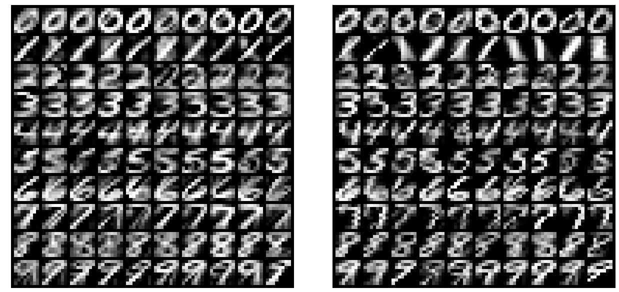

Figure 8: Images simulated from (a) the FA models and (b) the TCA models.

simulated from the trained models, without adding noise.

Some components account for interesting variability in writing style, but many clearly account for simple spatial transformations. For example, most of the components for 7 simply negate the vertical stroke in the image mean and produce a new vertical stroke at a different angle.

Shearing is a simple spatial transformation to model using a sparse matrix. Using 30 iterations of EM, we trained one TCA on each class of 200 digits using 29 transformations that included horizontal shearing plus horizontal translations. The first row of images in Fig. 9 shows how the 29 transformations modify a contrived pattern. Some transformations are quite severe and erase a large number of pixels, so in this experiment the transformation probabilities, p_ℓ, were learned. The parameters were initialized using the mean and variance of each pixel in the training set.

Fig. 7b shows the means and components found by TCA. The 2nd component for 7 models the accent on the nose of the 7; the 3rd component for 7 rounds out the angle in the upper right. These features are not clearly represented in the components found by FA. Fig. 8b shows digits simulated from the TCA models, without adding noise onto the latent image and the observed image. These images show a greater variation in style and fewer inappropriate artifacts than the images simulated from the FA model.

The 10 bottom rows of images in Fig. 9 show the means and corresponding sheared and translated images from the TCA models. Those transformed images whose average expected responsibilities were less than 0.01 in the training set are dimmed. A wide variation in writing style is captured by the transformations. For example, whereas FA uses components to model the different angles of vertical strokes in images of the

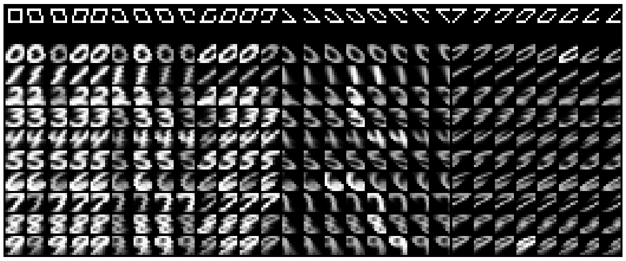

Figure 9: The sheared and translated versions of the mean image learned by TCA for each class of digit. The first row of images uses a contrived pattern to illustrate the 29 shearing and translation transformations that were used by TCA. Those versions whose average expected responsibilities were less than 0.01 in the training set are dimmed.

Table 1: Handwritten digit recognition rates, for a training set of 2000 8×8 images and a test set of 1000 images.

Method	Error rate
k-nearest neighbors	7.6%
Factor analysis	3.2%
Transformed component analysis	2.7%

digit 7, TCA models the different angles using the transformations. This allows TCA to use its components to model more subtle variations in writing style, such as the accent on the nose of the 7.

6.1 Improvement in recognition rate

We performed FA (20 components) and TCA (20 components) on 200 training cases from each class of handwritten digit using 50 iterations of EM. The noise variances were not allowed to drop below 10^{-2} to prevent overfitting a pixel that happens to always be off in the training data. Bayes rule was then used to classify 1000 test patterns. The results are summarized in Table 1 and compared with a standard feedforward method, k-nearest neighbors, where k was chosen using leave-one-out cross validation. TCA has a lower error rate than the other two methods.

The probability of each transformation p_ℓ in TCA was learned and we believe there was some overfitting. For example, some of the sheared image means in Fig. 9a that are faded (have average responsibilities less than 0.01) are good generalizations. We are currently running experiments that regularize p_ℓ and we are also running experiments to find the performance of support vector machines on this data.

7 Summary

We introduced a technique called "transformed component analysis" (TCA), that can jointly learn components from data and normalize for transformations. We explored transformations for translation and shearing, but the method can be applied to other transformations, such as rotation, scale and warping. When applied to noisy images of a uniformly colored pyramid at randomly selected positions and illuminated by parallel light rays with randomly selected angle and intensity, TCA is able to extract representations of shape and lighting, despite a significant level of background clutter. TCA also found a clear representation of facial expression from imperfectly aligned images, whereas PCA and factor analysis (FA) found severely blurred components. By including shearing transformations in the TCA model, we showed that handwritten digit recognition performance was improved over FA and that the images simulated from the TCA model contained greater variation in style and fewer inappropriate artifacts than the images simulated from FA.

Acknowledgements

We used grants from CITO (28838), NSERC (RGPIN-217657-99), NSF (IRI-9634618, CDA-9624396) and the Beckman Foundation.

References

[1] B. J. Frey and N. Jojic, "Estimating mixture models of images and inferring spatial transformations using the EM algorithm," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, June 1999, pp. 416–422.

[2] I. T. Jolliffe, *Principal Component Analysis*, Springer-Verlag, New York NY., 1986.

[3] A. J. Bell and T. J. Sejnowski, "An information maximization approach to blind separation and blind deconvolution," *Neural Computation*, vol. 7, pp. 1129–1159, 1995.

[4] B. S. Everitt, *An Introduction to Latent Variable Models*, Chapman and Hall, New York NY., 1984.

[5] D. Rubin and D. Thayer, "EM algorithms for ML factor analysis," *Psychometrika*, vol. 47, no. 1, pp. 69–76, 1982.

[6] J. J. Hull, "A database for handwritten text recognition research," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, no. 5, pp. 550–554, 1994.