Estimating smooth deformation models of substance and noise

Nebojsa Jojic[1], Brendan J. Frey[2], Patrice Simard[1], David Heckerman[1]

[1]Microsoft Research
Redmond, Washington

[2]Computer Science
University of Waterloo

Abstract

By representing image prototypes, or "substance", by linear subspaces spanned by deformation fields derived from low-frequecy wavelets, impressive invariance to distortion can be built into generative probability models. The prototypes representing substance and the distribution over wavelet coefficients can be estimated using EM, since exact inference is tractable in this model. While this approach works for noise-free data, it is prone to errors for noisy data, where the noise is deformed and then confused with the prototypes representing substance. We describe a generative model for smooth, nonuniform deformations, in which noise fields are deformed along with the prototypes representing substance. This prevents deformed substance from being confused with deformed noise. We show that a variational technique can be used for inference and parameter estimation in this model. We give results on a very difficult, contrived problem and on facial expression modeling.

1 Introduction

Many computer vision and image processing tasks benefit from invariances to spatial deformations in the image. Examples include handwritten character recognition, face recognition and motion estimation in video sequences. Small image deformations can often be modeled by the following simple additive model. Suppose $(\boldsymbol{\delta}_x, \boldsymbol{\delta}_y)$ is a deformation field (a vector field that specifies where to shift pixel intensity), where $(\delta_{xi}, \delta_{yi})$ is the 2-D real vector associated with pixel i. Given a vector of pixel intensities $\mathbf{f}$ for an image, we can approximate the deformed image by

$$\tilde{\mathbf{f}} = \mathbf{f} + \frac{\partial \mathbf{f}}{\partial x} \circ \boldsymbol{\delta}_x + \frac{\partial \mathbf{f}}{\partial y} \circ \boldsymbol{\delta}_y, \tag{1}$$

where $\circ$ is element-wise product and $\partial \mathbf{f}/\partial x$ is a gradient image computed by shifting the original image to the right a small amount and then subtracting off the original image. Fig. 1 shows some more complex examples of deformations computed using the above approximation.

Simard *et al.* (1992, 1993) considered a deformation field that is a linear combination of the uniform fields for translation, rotation, scaling and shearing plus the nonuniform field for line thickness. When the deformation field is parameterized by a scalar α (*e.g.*, x-translation), $\frac{\partial \mathbf{f}}{\partial x} \circ \boldsymbol{\delta}_x + \frac{\partial \mathbf{f}}{\partial y} \circ \boldsymbol{\delta}_y$ can be viewed as the gradient of $\mathbf{f}$ with respect to α. Since the above approximation holds for small α, this gradient is tangent to the true 1-D deformation manifold of $\mathbf{f}$.

The tangent approximation can also be included in generative models, including linear factor analyzer models (Hinton *et al.*, 1997) and nonlinear generative models (Jojic and Frey 2000).

Another approach to modeling small deformations is to jointly cluster the data and *learn* a locally linear deformation model for each cluster, *e.g.*, using EM in a factor analyzer (Ghahramani and Hinton 1997). An advantage of this approach over the tangent approach is that the types of deformation need not be specified beforehand. So, unknown, nonuniform types of deformation can be learned. However, a large amount of data is needed to accurately model the deformations, and learning is susceptible to local optima that confuse deformed data from one cluster with data from another cluster. (Some factors tend to "erase" parts of the image and "draw" new parts, instead of just perturbing the image.)

We describe a new probability model that can jointly cluster data, learn an appearance model and learn mixtures of nonuniform, smooth deformation fields.

2 Smooth, wavelet-based deformation fields

We ensure the deformation field $(\boldsymbol{\delta}_x, \boldsymbol{\delta}_y)$ is smooth by constructing it from low-frequency wavelets,

$$\boldsymbol{\delta}_x = \mathbf{R}\mathbf{a}_x, \qquad \boldsymbol{\delta}_y = \mathbf{R}\mathbf{a}_y, \tag{2}$$

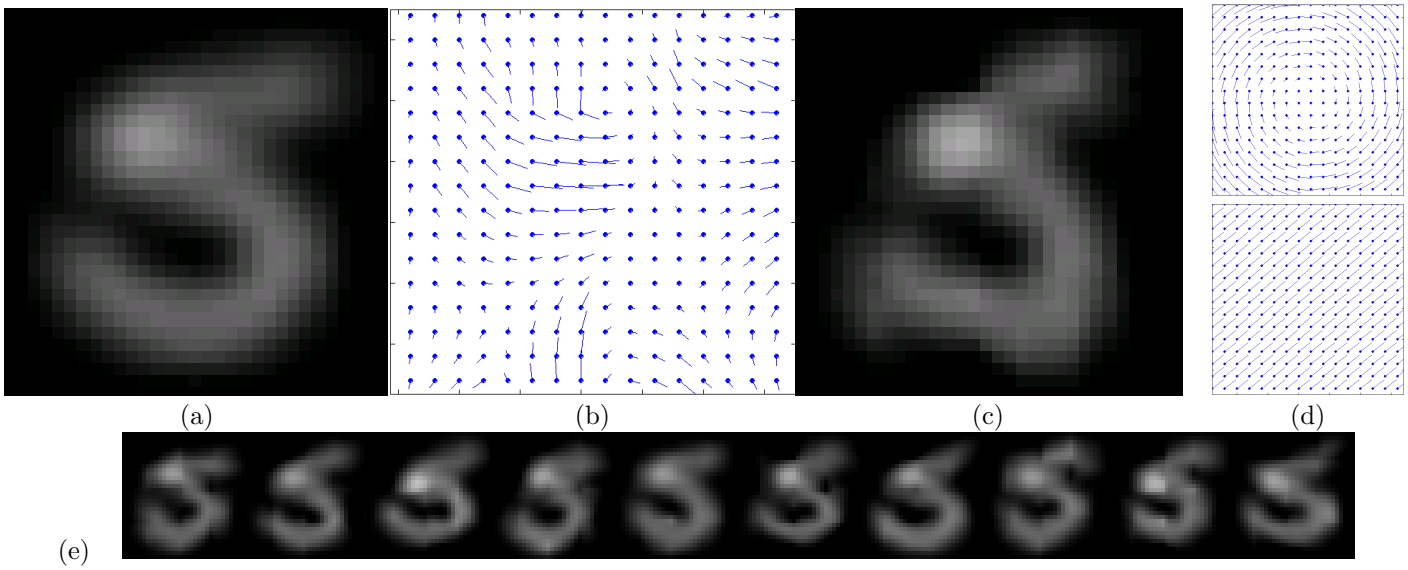


Figure 1: (a) An image of a hand-written digit. (b) A smooth, non-uniform deformation field. (c) The resulting deformed image. (d) Rotation and translation deformation fields. (e) Examples of deformed images produced by learned distributions over wavelet-based fields.

where the columns of $\mathbf{R}$ contain low-frequency wavelet basis vectors, and $\mathbf{a} = \begin{bmatrix} \mathbf{a}_x \\ \mathbf{a}_y \end{bmatrix}$ are the deformation coefficients. We use a number of deformation coefficients that is a small fraction of the number of pixels in the image. (In contrast, each factor in factor analysis has a number of coefficients that is *equal* to the number of pixels.)

An advantage of wavelets is their space/frequency localization. The global trends in the image can be captured in the low-frequency coefficients while at the same time, the deformations localized in smaller regions of the image can be expressed by more spatially localized wavelets.

The deformed image can be expressed as

$$\tilde{\mathbf{f}} = \mathbf{f} + (\mathbf{G}_x\mathbf{f}) \circ (\mathbf{R}\mathbf{a}_x) + (\mathbf{G}_y\mathbf{f}) \circ (\mathbf{R}\mathbf{a}_y), \tag{3}$$

where the derivatives in (1) are approximated by sparse matrices $\mathbf{G}_x$ and $\mathbf{G}_y$ that operate on $\mathbf{f}$ to compute finite differences.

(3) is bilinear in the deformation coefficients $\mathbf{a}$ and the original image $\mathbf{f}$, *i.e.*, it is linear in $\mathbf{f}$ given $\mathbf{a}$ and it is linear in $\mathbf{a}$ given $\mathbf{f}$. To rewrite the element-wise product as a matrix product, we convert either the vector $\mathbf{G}\mathbf{f}$ or the vector $\mathbf{R}\mathbf{a}$ to a diagonal matrix using the diag() function:

$$\tilde{\mathbf{f}} = \mathbf{f} + \mathbf{D}(\mathbf{f})\mathbf{a}, \quad \text{where } \mathbf{D}(\mathbf{f}) = [\text{diag}(\mathbf{G}_x\mathbf{f})\mathbf{R} \quad \text{diag}(\mathbf{G}_y\mathbf{f})\mathbf{R}] \tag{4}$$

$$\tilde{\mathbf{f}} = \mathbf{T}(\mathbf{a})\mathbf{f}, \quad \text{where } \mathbf{T}(\mathbf{a}) = [\mathbf{I} + \text{diag}(\mathbf{R}\mathbf{a}_x)\mathbf{G}_x + \text{diag}(\mathbf{R}\mathbf{a}_y)\mathbf{G}_y]\,. \tag{5}$$

The first equation shows by applying a simple pseudo inverse, we can estimate the coefficients of the image deformation that transforms $\mathbf{f}$ into $\tilde{\mathbf{f}}$: $\mathbf{a} = \mathbf{D}(\mathbf{f})^{-1}(\tilde{\mathbf{f}} - \mathbf{f})$. This low-dimensional vector of coefficients minimizes the distance $||\mathbf{f} - \tilde{\mathbf{f}}||$. Under easily satisfied conditions on the differencing matrices $\mathbf{G}_x$ and $\mathbf{G}_y$, $\mathbf{T}(\mathbf{a})$ in (5) can be made invertible regardless of the image $\mathbf{f}$, so that $\mathbf{f} = \mathbf{T}(\mathbf{a})^{-1}\tilde{\mathbf{f}}$.

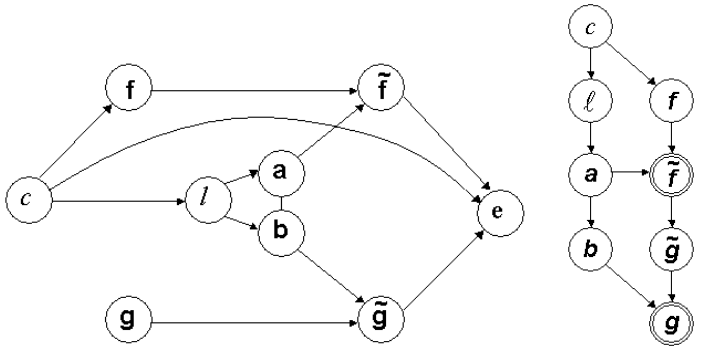
Given a test image $\mathbf{g}$, we could match $\mathbf{f}$ to $\mathbf{g}$ by computing the deformation coefficients, $\mathbf{a} = \mathbf{D}(\mathbf{f})^{-1}(\mathbf{g} - \mathbf{f})$, that minimize $||\mathbf{f} - \mathbf{g}||$. However, more extreme deformations can be successfully matched by deforming $\mathbf{g}$ as well:

$$\tilde{\mathbf{g}} = \mathbf{g} + (\mathbf{G}_x\mathbf{g}) \circ (\mathbf{R}\mathbf{b}_x) + (\mathbf{G}_y\mathbf{g}) \circ (\mathbf{R}\mathbf{b}_y), \tag{6}$$

where $\mathbf{b}$ are the deformation coefficients for $\mathbf{g}$. Again, minimizing $||\tilde{\mathbf{f}} - \tilde{\mathbf{g}}||$ is a simple quadratic optimization with respect to the deformation coefficients $\mathbf{a}$, $\mathbf{b}$. To favor some deformation fields over others, we can include a cost term that depends on the deformation coefficients.

Finally, a versatile image distance can be defined as:

$$d(\mathbf{f}, \mathbf{g}) = \min_{\mathbf{a},\mathbf{b}} \left\{ (\tilde{\mathbf{f}} - \tilde{\mathbf{g}})' \mathbf{\Psi}^{-1} (\tilde{\mathbf{f}} - \tilde{\mathbf{g}}) + [\mathbf{a}' \quad \mathbf{b}'] \mathbf{\Gamma}^{-1} \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix} \right\}. \tag{7}$$



a b

Figure 2: (a) A Bayes net for deformable image matching. (b) A generative version of the net conditioned on $\mathbf{e} = \mathbf{0}$.

Matrix $\mathbf{\Psi}$ is a diagonal matrix whose non-zero elements contain variances of appropriate pixels. This distance allows different pixels to have different importance. For example, if we are matching two images of a tree in the wind, the deformation coefficients should be capable of aligning the trunk and large branches, while the variability in the appearance of the leaves would be captured in $\mathbf{\Psi}$. As will be explained in the next section, it is even better to model the appearance variability directly in the model image $\mathbf{f}$. $\mathbf{\Gamma}$ captures the covariance structure of the wavelet coefficients of the allowed deformations.

3 Bayes net for deformable image matching

In Fig. 2a we show a Bayes net that can be used to compute the likelihood that the input image matches the images modeled by the network. For classification, we learn one of these networks for each class of data.

The generative matching process begins by clamping the test image $\mathbf{g}$. Then, an image cluster index c is drawn from $P(c)$ and given c, a latent image $\mathbf{f}$ is drawn from a Gaussian, $\mathcal{N}(\mathbf{f}; \boldsymbol{\mu}_c, \mathbf{\Phi}_c)$.

Next, a deformation type index ℓ is picked according to $P(\ell|c)$. This index determines the covariance $\mathbf{\Gamma}_l$ of the deformation coefficients for both the latent image $\mathbf{f}$ and the test image $\mathbf{g}$:

$$p(\left[\begin{array}{c}\mathbf{a}\\ \mathbf{b}\end{array}\right]|\ell) = \mathcal{N}(\left[\begin{array}{c}\mathbf{a}\\ \mathbf{b}\end{array}\right]; \mathbf{0}, \mathbf{\Gamma}_\ell). \tag{8}$$

$\mathbf{\Gamma}_\ell$ could be a diagonal matrix with larger elements corresponding to lower-frequency basis functions, to capture a wide range of smooth non-uniform deformations. However, $\mathbf{\Gamma}_\ell$ could also capture correlations among deformations in different parts of the image. The deformation coefficients for the latent image $\mathbf{a}$ and for the observed image $\mathbf{b}$ should be strongly correlated, so we model the joint distribution instead of modeling $\mathbf{a}$ and $\mathbf{b}$ separately.

Once the deformation coefficients $\mathbf{a}$, $\mathbf{b}$ have been generated, the deformed latent image $\tilde{\mathbf{f}}$ and the deformed test image $\tilde{\mathbf{g}}$ are produced from $\mathbf{f}$ and $\mathbf{g}$ according to (3) and (6). Using the functions $\mathbf{D}()$ and $\mathbf{T}()$ introduced above, we have

$$p(\tilde{\mathbf{f}}|\mathbf{f}, \mathbf{a}) = \delta(\tilde{\mathbf{f}} - \mathbf{f} - \mathbf{D}(\mathbf{f})\mathbf{a}) = \delta(\tilde{\mathbf{f}} - \mathbf{T}(\mathbf{a})\mathbf{f}), \tag{9}$$

$$p(\tilde{\mathbf{g}}|\mathbf{g}, \mathbf{b}) = \delta(\tilde{\mathbf{g}} - \mathbf{g} - \mathbf{D}(\mathbf{g})\mathbf{b}) = \delta(\tilde{\mathbf{g}} - \mathbf{T}(\mathbf{b})\mathbf{g}). \tag{10}$$

As an illustration of the generative process up to this point, in Fig. 1 we show several images produced by randomly selecting 8 deformation coefficients from a unit-covariance Gaussian and applying the resulting deformation field to an image.

The last random variable in the model is an error image $\mathbf{e}$ (called a "reference signal" in control theory), which is formed by adding a small amount of diagonal Gaussian noise to the difference between the deformed images $\tilde{\mathbf{f}}$ and $\tilde{\mathbf{g}}$:

$$p(\mathbf{e}|\tilde{\mathbf{f}}, \tilde{\mathbf{g}}, c) = \mathcal{N}(\mathbf{e}; \tilde{\mathbf{f}} - \tilde{\mathbf{g}}, \mathbf{\Psi}_c). \tag{11}$$

For good model parameters, it is likely that one of the cluster means can be slightly deformed to match a slightly deformed observed image. However, due to the constrained nature of these deformations, an exact match may not be achievable. Thus, to allow an exact match, the model helps the image difference with a small amount of non-uniform, possibly cluster dependent noise. $\mathbf{\Psi}_c$ is diagonal and the non-zero elements contain the pixel variances (as described in the previous section). A natural place to include cluster dependence is in fact in the cluster noise $\mathbf{\Phi}_c$. However, it is possible to simplify the model by setting $\mathbf{\Phi}_c = 0$, so that $p(\mathbf{f}|c) = \delta(\mathbf{f} - \boldsymbol{\mu}_c)$, in which case it is helpful to at least add cluster dependence into $\mathbf{\Psi}_c$. Such a model can be trained by an exact EM algorithm (Jojic et al, 2000).

If the model has both levels of noise, the model is intractable but we will describe an efficient variational approximation that allows tractable inference and learning.

The described model can now be used to evaluate how likely it is to achieve a zero error image $\mathbf{e}$ by randomly selecting hidden variables conditioned on their parents in the fashion described above. If the model has the right cluster means, right noise levels and the right variability in the deformation coefficients, then the likelihood $p(\mathbf{e} = 0|\mathbf{g})$ will be high. Thus, this likelihood can be used for classification of images when the parameters of the models for different classes are known. Also, we can use the EM algorithm to estimate the parameters of the model that will maximize this likelihood for all observed images $\mathbf{g}_t$ in a training data set (see the Appendix).

Note that by conditioning on $\mathbf{e} = \mathbf{0}$, we can transform the network into the generative network shown in Fig. 2b.[1]

After collapsing the deterministic nodes in the network, the joint distribution conditioned on the input $\mathbf{g}$ is

$$p(c, l, \mathbf{a}, \mathbf{b}, \mathbf{f}, \mathbf{e}|\mathbf{g}) = P_{c,l}\mathcal{N}(\begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix}; 0, \mathbf{\Gamma}_\ell)\mathcal{N}(\mathbf{f}; \boldsymbol{\mu}_c, \mathbf{\Phi}_c)\mathcal{N}(\mathbf{e}; \mathbf{T}(\mathbf{a})\mathbf{f} - \mathbf{T}(\mathbf{b})\mathbf{g}, \mathbf{\Psi}_c) \tag{12}$$

With $\mathbf{\Phi}_c = 0$, the above expression would simplify enough to allow computation of $P(\mathbf{e}|c, \ell, g)$ by analytical integration of the deformation coefficients followed by normalization. Unfortunately, due to the multiplicative term $\mathbf{T}(\mathbf{a})\mathbf{f}$, the same cannot be done when a non-deterministic prior is assumed on both $\mathbf{f}$ and $\mathbf{a}$. To deal with such a model, we bound $\log P(\mathbf{e}, c, \ell|\mathbf{g})$ by:

$$\log P(\mathbf{e}, c, \ell|\mathbf{g}) \geq \int_{\mathbf{a},\mathbf{b},\mathbf{f}} q(\mathbf{f}, \mathbf{a}) \log \frac{p(c, l, \mathbf{a}, \mathbf{b}, \mathbf{f}, \mathbf{e}|\mathbf{g})}{q(\mathbf{f}, \mathbf{a})}. \tag{13}$$

This follows from Jensen's inequality and the bound becomes tight when $q(\mathbf{f}, \mathbf{a}) = p(\mathbf{f}, \mathbf{a}|\mathbf{e}, c, \ell, \mathbf{g})$, i.e., when the above expectation is performed with respect to the posterior. To make this integration tractable, we impose the following simple form on the variational posterior:

$$q(\mathbf{f}, \mathbf{a}) = \mathcal{N}(\mathbf{f}; \tilde{\boldsymbol{\mu}}_c, \tilde{\mathbf{\Phi}}_c)\mathcal{N}(\begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix}; \tilde{\boldsymbol{\gamma}}_\ell, \tilde{\mathbf{\Gamma}}_\ell) \tag{14}$$

This simplifies the above bound to:

$$\begin{aligned} B = 1 + \frac{1}{2}\log\left|\frac{\tilde{\mathbf{\Phi}}_c}{2\pi}\right| + \frac{1}{2}\log\left|\frac{\tilde{\mathbf{\Gamma}}_c}{2\pi}\right| - \frac{1}{2}\log\left|\frac{\mathbf{\Phi}_c}{2\pi}\right| - \frac{1}{2}\log\left|\frac{\mathbf{\Gamma}_c}{2\pi}\right| - \frac{1}{2}tr[\mathbf{\Phi}_c^{-1}\tilde{\mathbf{\Phi}}_c] - \frac{1}{2}tr[\mathbf{\Gamma}_c^{-1}\tilde{\mathbf{\Gamma}}_c] - \frac{1}{2}(\tilde{\boldsymbol{\mu}}_c - \boldsymbol{\mu}_c)'\mathbf{\Phi}_c^{-1}(\tilde{\boldsymbol{\mu}}_c - \boldsymbol{\mu}_c) - \frac{1}{2}\tilde{\boldsymbol{\gamma}}_\ell'\mathbf{\Gamma}_\ell\tilde{\boldsymbol{\gamma}}_\ell \\ - \log\left|\frac{\mathbf{\Psi}_c}{2\pi}\right| - \frac{1}{2}tr\left\{E\left[\mathbf{T}(\mathbf{a})'\mathbf{\Psi}_c^{-1}\mathbf{T}(\mathbf{a})\right](\tilde{\mathbf{\Phi}}_c + \tilde{\boldsymbol{\mu}}_c\tilde{\boldsymbol{\mu}}_c')\right\} + \mathbf{g}'E\left[\mathbf{T}(\mathbf{b})'\mathbf{\Psi}_c^{-1}\mathbf{T}(\mathbf{a})\right]\tilde{\boldsymbol{\mu}}_c' - \frac{1}{2}\mathbf{g}'E\left[\mathbf{T}(\mathbf{b})'\mathbf{\Psi}_c^{-1}\mathbf{T}(\mathbf{b})\right]\mathbf{g} \end{aligned} \tag{15}$$

Then, the approximate likelihood of the data can be computed by maximizing the bound with respect to the variational parameters $\tilde{\boldsymbol{\mu}}_c, \tilde{\mathbf{\Phi}}_c, \tilde{\boldsymbol{\gamma}}_\ell, \tilde{\mathbf{\Gamma}}_\ell$. This maximization can be done efficiently as equating to zero the derivatives of the bound with respect to the variational parameters leads to a system of equations which are linear in the elements

By integrating out the deformation coefficients we obtain $p(c, \ell, \mathbf{e}|\mathbf{g}) = P_{c,\ell}\mathcal{N}\left(\mathbf{e}; \boldsymbol{\mu}_c - \mathbf{g}, [\mathbf{\Psi}_c^{-1} - \mathbf{\Psi}_c^{-1}\mathbf{M}_c\mathbf{\Omega}_{c,\ell}\mathbf{M}_c'\mathbf{\Psi}_c^{-1}]^{-1}\right)$, where $\mathbf{M}_c = \begin{bmatrix}\mathbf{D}(\boldsymbol{\mu}_c) & -\mathbf{D}(\mathbf{g})\end{bmatrix}$ and $\mathbf{\Omega}_{c,\ell} = (\mathbf{\Gamma}_\ell^{-1} + \mathbf{M}_c'\mathbf{\Psi}_c^{-1}\mathbf{M}_c)^{-1}$. This density function can be normalized over c, ℓ to obtain $P(c, \ell|\mathbf{e}, \mathbf{g})$. The likelihood can be computed by summing over the class and transformation indices:

$$p(\mathbf{e}|\mathbf{g}) = \sum_{c=1}^{C}\sum_{l=1}^{L} P_{c,l}\mathcal{N}\left(\mathbf{e}; \boldsymbol{\mu}_c - \mathbf{g}, [\mathbf{\Psi}_c^{-1} - \mathbf{\Psi}_c^{-1}\mathbf{M}_c\mathbf{\Omega}_{c,\ell}\mathbf{M}_c'\mathbf{\Psi}_c^{-1}]^{-1}\right). \tag{16}$$

By using this likelihood instead of the distance measure in (7), we are integrating over all possible deformations instead of finding the optimal deformation (which is given by (21) in the Appendix).

4 Experiments and conclusions

We tested our algorithm on 20x28 greyscale images of people with different facial expressions and 8x8 greyscale images of handwritten digits from the CEDAR CDROM (Hull, 1994). To compare our method with other generative models, we used a training set of 2000 images to learn 10 digit models using the EM algorithm and tested the algorithms on a test set of 1000 digit images.

Deformable image matching. In Fig. 3a we estimate the optimal deformation fields necessary to match two images of a face of the same person but with different facial expression. We set the $\mathbf{\Psi}$ matrix to identity and we set $\mathbf{\Gamma}$ by hand to allow a couple of pixels of deformations. See Section 2 for nomenclature.

[1] To do so in a straightforward fashion, we assume that $|\mathbf{T}(\mathbf{b})| = 1$.

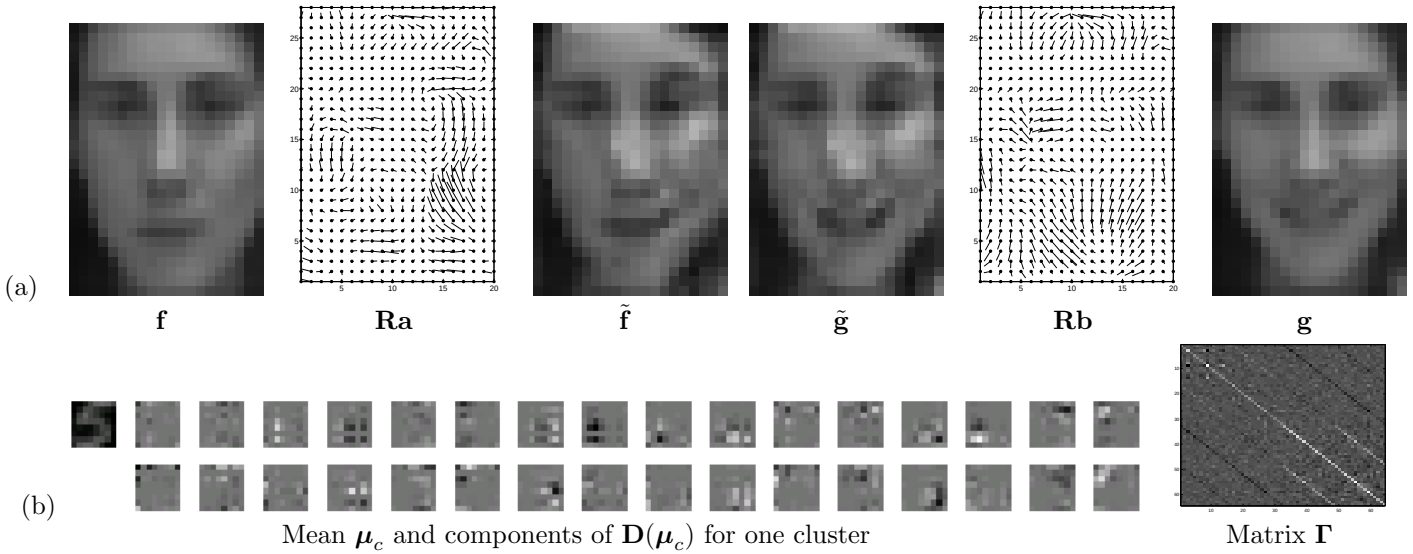


Figure 3: Estimating the image deformation due to a change in facial expression and a subset of the learned parameters for the model of handwritten digits

Comparison with the mixture of diagonal Gaussians (MDG). MDG needs 10-20 classes per digit to achieve the optimal error rate of only about 8% (Frey and Jojic 1999a) on the handwritten digit recognition task. Note that our network reduces to MDG when $\boldsymbol{\Gamma}_\ell$ is set to zero. To demonstrate the effectiveness of adding a deformation model to MDG, we trained our model with 15 classes per digit and only a single transformation model ($L = 1$) for all digits, with a total of 64 deformation coefficients (8 for each dimension in the latent and the observed images). In Fig. 3b we show one of the learned cluster means, the components in the corresponding deformation matrix $\mathbf{D}$ and the learned covariance matrix $\boldsymbol{\Gamma}$. $\boldsymbol{\Gamma}$ shows anticorrelation among the deformation coefficients for the latent and the observed image, as the network usually applies opposite deformations on these two images to achieve the match. However, there is also strong correlation between $\mathbf{b}_x$ and $\mathbf{b}_y$ and less correlation between $\mathbf{a}_x$ and $\mathbf{a}_y$ as the network uses mostly a rotational adjustment on the input image, while the latent image is more freely deformed (Fig. 1e). Our model achieved the error rate of **3.6%**. Even if we keep only the diagonal elements in $\boldsymbol{\Gamma}$, the model achieves a 5% error rate.

Comparison with factor analysis. In factor analysis (FA) or in a mixture of factor analyzers (MFA), the deformation matrix $\mathbf{D}$ is called factor loading matrix and is not tied to the mean $\boldsymbol{\mu}_c$ as in our model (Fig. 3b). The factor covariance matrix is set to the identity matrix, as the extra freedom in the choice of the factor variances can be captured in the factor loading matrix. So, while FA/MFA try to capture the variability in the data by learning the components in the factor loading matrix and keeping the distribution over the factors fixed, our model does the opposite by tying the factor loading matrix to the mean image and learning the distribution over the factors (deformation coefficients). By doing this, we we are able to expand other images using the same deformation model. This allows us to share the deformation model across clusters and also to deform the input images. The comparable error rate in classification of handwritten digits for FA/MFA (3.3%) and our model (3.6%) indicates that most of the variability in images of handwriten digits can be captured by modeling smooth, non-uniform deformations without allowing full FA learning.

Our deformable image matching network could be used for a variety of computer vision tasks such as optical flow estimation, deformation invariant recognition and modeling correlations in deformations. For example, our learning algorithm could learn to jointly deform the mouth and eyes when modeling facial expressions.

References

A. P. Dempster, N. M. Laird and D. B. Rubin 1977. Maximum likelihood from incomplete data via the EM algorithm. *Proceedings of the Royal Statistical Society* **B-39**, 1–38.

B. J. Frey and N. Jojic 1999a. Estimating mixture models of images and inferring spatial transformations using the EM algorithm. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Ft. Collins, CO. IEEE Computer Society Press, Los Alamitos, CA.

Z. Ghahramani and G. E. Hinton 1997. The EM algorithm for mixtures of factor analyzers. University of Toronto Technical Report CRG-TR-96-1. Available at www.gatsby.ucl.ac.uk/∼zoubin.

G. E. Hinton, P. Dayan and M. Revow 1997. Modeling the manifolds of images of handwritten digits. *IEEE Trans. on Neural Networks* **8**, 65–74.

N. Jojic and B. J. Frey 2000. Topographic transformation as a discrete latent variable. In S.A. Solla, T. K. Leen, and K.-R. Müller (eds) *Advances in Neural Information Processing Systems 12*, MIT Press, Cambridge, MA.

P. Y. Simard, Y. Le Cun and J. Denker 1993. Efficient pattern recognition using a new transformation distance. In S. J. Hanson, J. D. Cowan and C. L. Giles, *Advances in Neural Information Processing Systems 5*, Morgan Kaufmann, San Mateo, CA.

N. Vasconcelos and A. Lippman 1998. Multiresolution tangent distance for affine invariant classification. In M. I. Jordan and M. I. Kearns and S. A. Solla (eds) *Advances in Neural Information Processing Systems 10*, MIT Press, Cambridge, MA.

Appendix: EM for deformable image matching network

To fit the network to a set of training data, we assume that the error images for the training cases are zero and estimate the maximum likelihood parameters using EM (Dempster *et al.* 1977). In deriving the M-step, both forms of the deformation equations (4) and (5) are useful, depending on which parameters are being optimized. Using $\langle\rangle$ to denote an average over the training set, the update equations are:

$$P_{c,\ell} = \langle P(c,\ell|\mathbf{e}_t = \mathbf{0}, \mathbf{g}_t)\rangle \tag{17}$$

$$\begin{aligned}\hat{\boldsymbol{\mu}}_c =& \langle \sum_\ell P(c,\ell|\mathbf{e}_t = \mathbf{0}, \mathbf{g}_t) \mathrm{E}[\mathbf{T}(\mathbf{a})' \boldsymbol{\Psi}_c^{-1} \mathbf{T}(\mathbf{a}) | c, \ell, \mathbf{e}_t = \mathbf{0}, \mathbf{g}_t]\rangle^{-1} \\ & \cdot \langle \sum_\ell P(c,\ell|\mathbf{e}_t = \mathbf{0}, \mathbf{g}_t) \mathrm{E}[\mathbf{T}(\mathbf{a})' \boldsymbol{\Psi}_c^{-1} \mathbf{T}(\mathbf{b}) \mathbf{g}_t | c, \ell, \mathbf{e}_t = \mathbf{0}, \mathbf{g}_t]\rangle,\end{aligned} \tag{18}$$

$$\hat{\boldsymbol{\Gamma}}_\ell = \frac{\left\langle \sum_c P(c,\ell|\mathbf{e}_t = \mathbf{0}, \mathbf{g}_t) \mathrm{E}\left\{ \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix} \begin{bmatrix} \mathbf{a}' & \mathbf{b}' \end{bmatrix} \middle| c, \ell, \mathbf{e}_t = \mathbf{0}, \mathbf{g}_t \right\} \right\rangle}{\langle \sum_c P(c,\ell|\mathbf{e}_t = \mathbf{0}, \mathbf{g}_t)\rangle} \tag{19}$$

$$\hat{\boldsymbol{\Psi}}_c = \mathrm{diag}\left(\frac{\langle \sum_\ell P(c,l|\mathbf{e}_t = \mathbf{0}, \mathbf{g}_t) \mathrm{E}[(\tilde{\mathbf{f}} - \tilde{\mathbf{g}}_t) \circ (\tilde{\mathbf{f}} - \tilde{\mathbf{g}}_t) | c, l, \mathbf{e}_t = \mathbf{0}, \mathbf{g}_t]\rangle}{\langle \sum_\ell P(c,l|\mathbf{e}_t = \mathbf{0}, \mathbf{g}_t)\rangle} \right) \tag{20}$$

The expectations needed to evaluate the above update equations are given by:

$$\boldsymbol{\Omega}_{c,\ell} = \mathrm{cov}\left\{ \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix} \middle| c, \ell, \mathbf{e}_t = \mathbf{0}, \mathbf{g}_t \right\} = (\boldsymbol{\Gamma}_\ell^{-1} + \mathbf{M}_c' \boldsymbol{\Psi}_c^{-1} \mathbf{M}_c)^{-1}$$

$$\boldsymbol{\gamma}_{c,\ell} = \mathrm{E}\left\{ \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix} \middle| c, l, \mathbf{e}_t = \mathbf{0}, \mathbf{g}_t \right\} = \boldsymbol{\Omega}_{c,\ell}^{-1} \mathbf{M}_c' \boldsymbol{\Psi}_c^{-1} (\boldsymbol{\mu}_c - \mathbf{g}_t) \tag{21}$$

$$\mathrm{E}\left\{ \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix} \begin{bmatrix} \mathbf{a}' & \mathbf{b}' \end{bmatrix} \middle| c, \ell, \mathbf{e}_t = \mathbf{0}, \mathbf{g}_t \right\} = \boldsymbol{\Omega}_{c,\ell} + \boldsymbol{\gamma}_{c,\ell} \boldsymbol{\gamma}_{c,\ell}' \tag{22}$$

$$\mathrm{E}[(\tilde{\mathbf{f}} - \tilde{\mathbf{g}}_t) \circ (\tilde{\mathbf{f}} - \tilde{\mathbf{g}}_t) | c, l, \mathbf{e}_t = \mathbf{0}, \mathbf{g}_t] = \left(\boldsymbol{\mu}_c - \mathbf{g}_t + \mathbf{M}_c \boldsymbol{\gamma}_{c,\ell}\right) \circ \left(\boldsymbol{\mu}_c - \mathbf{g}_t + \mathbf{M}_c \boldsymbol{\gamma}_{c,\ell}\right) + \mathrm{diag}(\mathbf{M}_c (\boldsymbol{\Omega}_{c,\ell}) \mathbf{M}_c')$$

Expectations in (18) are computed using

$$\begin{aligned}\mathbf{T}(\mathbf{a})' \boldsymbol{\Psi}_c^{-1} \mathbf{T}(\mathbf{a}) =& \boldsymbol{\Psi}_c^{-1} + \sum_{d \in \{x,y\}} \mathbf{G}_d' \mathrm{diag}(\mathbf{R}\mathbf{a}_d) \boldsymbol{\Psi}_c^{-1} \\ & + \sum_{d \in \{x,y\}} \boldsymbol{\Psi}_c^{-1} \mathrm{diag}(\mathbf{R}\mathbf{a}_d) \mathbf{G}_d + \sum_{d_1, d_2 \in \{x,y\}} \mathbf{G}_{d_1}' \boldsymbol{\Psi}_c^{-1} \mathrm{diag}(\mathbf{R}\mathbf{a}_{d_1} \mathbf{a}_{d_2}' \mathbf{R}') \mathbf{G}_{d_2}\end{aligned} \tag{23}$$

$$\begin{aligned}\mathbf{T}(\mathbf{a})' \boldsymbol{\Psi}_c^{-1} \mathbf{T}(\mathbf{b}) \mathbf{g}_t =& \boldsymbol{\Psi}_c^{-1} \mathbf{g}_t + \sum_{d \in \{x,y\}} \mathbf{G}_d' \mathrm{diag}(\mathbf{R}\mathbf{a}_d) \boldsymbol{\Psi}_c^{-1} \mathbf{g}_t \\ & + \sum_{d \in \{x,y\}} \boldsymbol{\Psi}_c^{-1} \mathrm{diag}(\mathbf{R}\mathbf{b}_d) \mathbf{G}_d \mathbf{g}_t + \sum_{d_1, d_2 \in \{x,y\}} \mathbf{G}_{d_1}' \boldsymbol{\Psi}_c^{-1} \mathrm{diag}(\mathbf{R}\mathbf{a}_{d_1} \mathbf{b}_{d_2}' \mathbf{R}') \mathbf{G}_{d_2} \mathbf{g}_t.\end{aligned} \tag{24}$$

Then, the expectations $\mathrm{E}[\mathbf{a}]$ and $\mathrm{E}[\mathbf{b}]$ are the two halves of the vector $\boldsymbol{\gamma}_{c,\ell}$, while $\mathrm{E}[\mathbf{a}_{d_1} \mathbf{a}_{d_2}']$ and $\mathrm{E}[\mathbf{a}_{d_1} \mathbf{b}_{d_2}']$, for $d_1, d_2 \in \{x, y\}$, are square blocks of the matrix in (22).