ARTICLE Communicated by John Platt

Separating Style and Content with Bilinear Models

Joshua B. Tenenbaum*
Department of Brain and Cognitive Sciences, Massachusetts Institute of Technology, Cambridge, MA 02139, U.S.A.

William T. Freeman
MERL, a Mitsubishi Electric Research Lab, 201 Broadway, Cambridge, MA 02139, U.S.A.

Perceptual systems routinely separate "content" from "style," classifying familiar words spoken in an unfamiliar accent, identifying a font or handwriting style across letters, or recognizing a familiar face or object seen under unfamiliar viewing conditions. Yet a general and tractable computational model of this ability to untangle the underlying factors of perceptual observations remains elusive (Hofstadter, 1985). Existing factor models (Mardia, Kent, & Bibby, 1979; Hinton & Zemel, 1994; Ghahramani, 1995; Bell & Sejnowski, 1995; Hinton, Dayan, Frey, & Neal, 1995; Dayan, Hinton, Neal, & Zemel, 1995; Hinton & Ghahramani, 1997) are either insufficiently rich to capture the complex interactions of perceptually meaningful factors such as phoneme and speaker accent or letter and font, or do not allow efficient learning algorithms. We present a general framework for learning to solve two-factor tasks using bilinear models, which provide sufficiently expressive representations of factor interactions but can nonetheless be fit to data using efficient algorithms based on the singular value decomposition and expectation-maximization. We report promising results on three different tasks in three different perceptual domains: spoken vowel classification with a benchmark multispeaker database, extrapolation of fonts to unseen letters, and translation of faces to novel illuminants.

1 Introduction

Perceptual systems routinely separate the "content" and "style" factors of their observations, classifying familiar words spoken in an unfamiliar accent, identifying a font or handwriting style across letters, or recognizing a familiar face or object seen under unfamiliar viewing conditions. These and many other basic perceptual tasks have in common the need to pro-

* Current address: Department of Psychology, Stanford University, Stanford, CA 94305, U.S.A.

cess separately two independent factors that underlie a set of observations. This article shows how perceptual systems may learn to solve these crucial two-factor tasks using simple and tractable bilinear models. By fitting such models to a training set of observations, the influences of style and content factors can be efficiently separated in a flexible representation that naturally supports generalization to unfamiliar styles or content classes.

Figure 1 illustrates three abstract tasks that fall under this framework: *classification, extrapolation*, and *translation*. Examples of these abstract tasks in the domain of typography include classifying known characters in a novel font, extrapolating the missing characters of an incomplete novel font, or translating novel characters from a novel font into a familiar font. The essential challenge in all of these tasks is the same. A perceptual system observes a training set of data in multiple styles and content classes and is then presented with incomplete data in an unfamiliar style, missing either content labels (see Figure 1A) or whole observations (see Figure 1B) or both (see Figure 1C). The system must generate the missing labels or observations using only the available data in the new style and what it can learn about the interacting roles of style and content from the training set of complete data.

We describe a unified approach to the learning problems of Figure 1 based on fitting models that discover explicit parameterized representations of what the training data of each row have in common independent of column, what the data of each column have in common independent of row, and what all data have in common independent of row and column—the interaction of row and column factors. Such a modular representation naturally supports generalization to new styles or content. For example, we can extrapolate a new style to unobserved content classes (see Figure 1B) by combining content and interaction parameters learned during training with style parameters estimated from available data in the new style.

Several models for the underlying factors of observations have recently been proposed in the literature on unsupervised learning. These include essentially *additive* factor models, as used in principal component analysis (Mardia et al., 1979), independent component analysis (Bell & Sejnowski, 1995), and cooperative vector quantization (Hinton & Zemel, 1994; Ghahramani, 1995), and *hierarchical* factorial models, as used in the Helmholtz machine and its descendants (Hinton et al., 1995; Dayan et al., 1995; Hinton & Ghahramani, 1997).

We model the mapping from style and content parameters to observations as a bilinear mapping. Bilinear models are two-factor models with the mathematical property of separability: their outputs are linear in either factor when the other is held constant. Their combination of representational expressiveness and efficient learning procedures enables bilinear models to overcome two principal drawbacks of existing factor models that might be applied to learning the tasks in Figure 1. In contrast to additive factor models, bilinear models provide for rich factor interactions by allowing factors to modulate each other's contributions multiplicatively (see section 2).

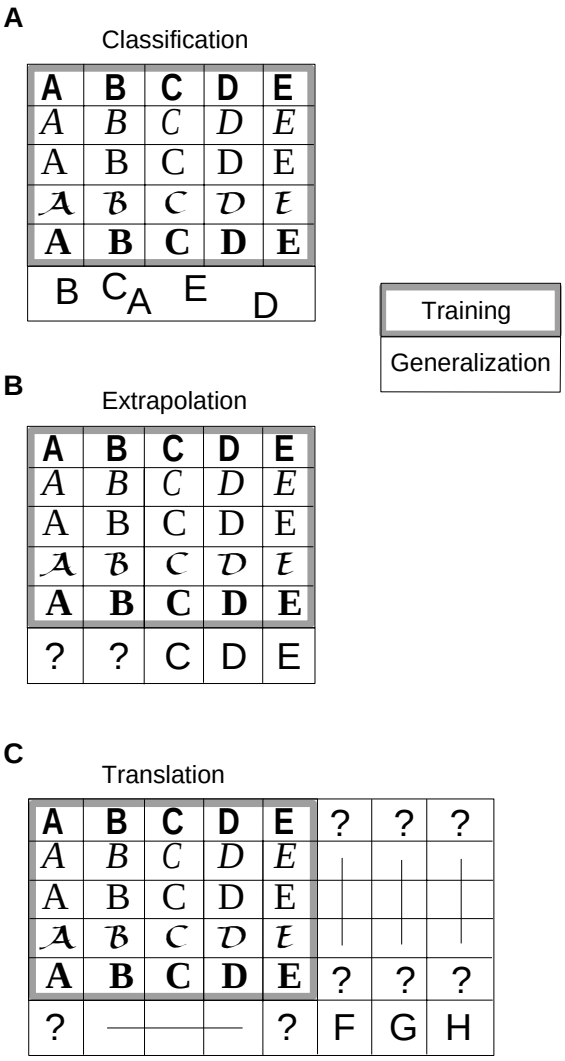

Figure 1: Given a labeled training set of observations in multiple styles (e.g., fonts) and content classes (e.g., letters), we want to (A) *classify* content observed in a new style, (B) *extrapolate* a new style to unobserved content classes, and (C) *translate* from new content observed only in new styles into known styles or content classes.

Model dimensionality can be adjusted to accommodate data that arise from arbitrarily complex interactions of style and content factors. In contrast to hierarchical factorial models, model fitting can be carried out by efficient

techniques well known from the study of linear models, such as the singular value decomposition (SVD) and the expectation-maximization (EM) algorithm, without having to invoke extensive stochastic (Hinton et al., 1995) or deterministic (Dayan et al., 1995) approximations.

Our approach is also related to the "learning-to-learn" research program (Thrun & Pratt, 1998)—also known as task transfer or multitask learning (Caruana, 1998). The central insight of "learning to learn" is that learning problems often come in clusters of related tasks, and thus learners may automatically acquire useful biases for a novel learning task by training on many related ones. Rather than families of related tasks, we focus on how learners can exploit the structure in families of related observations, bound together by their common styles, content classes, or style $\times$ content interaction, to acquire general biases useful for carrying out tasks on novel observations from the same family. Thus, our work is closest in spirit to the family discovery approach of Omohundro (1995), differing primarily in our focus on bilinear models to parameterize the style $\times$ content interaction.

Section 2 explains and motivates our bilinear modeling approach. Section 3 describes how these models are fit to a training set of observations. Sections 4, 5, and 6 present specific applications of these techniques to the three tasks of classification, extrapolation, and translation, using realistic data from three different perceptual domains. Section 7 suggests directions for future work, and section 8 offers some concluding comments.

We will use the terms *style* and *content* generically to refer to any two independent factors underlying a set of perceptual observations. For tasks that require generalization to novel classes of only one factor (see Figures 1A and 1B), we will refer to the variable factor (which changes during generalization) as style and the invariant factor (with a fixed set of classes) as content. For example, in a task of recognizing familiar words spoken in an unfamiliar accent, we would think of the words as content and the accent as style. For tasks that require generalization across both factors (see Figure 1C), the labels *style* and *content* are arbitrary, and we will use them as seems most natural.

2 Bilinear Models

We have explored two bilinear models, closely related to each other, which we distinguish by the labels *symmetric* and *asymmetric*. This section describes the two models and illustrates them on a simple data set of face images.

2.1 Symmetric Model. In the symmetric model, we represent both style s and content c with vectors of parameters, denoted $\mathbf{a}^s$ and $\mathbf{b}^c$ and with dimensionalities I and J, respectively. Let $\mathbf{y}^{sc}$ denote a K-dimensional observation vector in style s and content class c. We assume that $\mathbf{y}^{sc}$ is a bilinear

function of $\mathbf{a}^s$ and $\mathbf{b}^c$ given most generally by the form

$$y_k^{sc} = \sum_{i=1}^{I} \sum_{j=1}^{J} w_{ijk} a_i^s b_j^c. \tag{2.1}$$

Here i, j, and k denote the components of style, content, and observation vectors, respectively.[1] The w_{ijk} terms are independent of style and content and characterize the interaction of these two factors. Their meaning becomes clearer when we rewrite equation 2.1 in vector form. Letting $\mathbf{W}_k$ denote the $I \times J$ matrix with entries $\{w_{ijk}\}$, equation 2.1 can be written as

$$y_k^{sc} = \mathbf{a}^{s^T} \mathbf{W}_k \mathbf{b}^c. \tag{2.2}$$

In equation 2.2, the K matrices $\mathbf{W}_k$ describe a bilinear map from the style and content vector spaces to the K-dimensional observation space.

The interaction terms have another interpretation, which can be seen by writing the symmetric model in a different vector form. Letting $\mathbf{w}_{ij}$ denote the K-dimensional vector with components $\{w_{ijk}\}$, equation 2.1 can be written as

$$\mathbf{y}^{sc} = \sum_{i,j} \mathbf{w}_{ij} a_i^s b_j^c. \tag{2.3}$$

In equation 2.3, the w_{ijk} terms represent $I \times J$ basis vectors of dimension K, and the observation $\mathbf{y}^{sc}$ is generated by mixing these basis vectors with coefficients given by the tensor product of $\mathbf{a}^s$ and $\mathbf{b}^c$.

Of course, all of these interpretations are formally equivalent, but they suggest different intuitions which we will exploit later. As a concrete example, Figure 2 illustrates a symmetric model of face images of different people in different poses (sampled from the complete data in Figure 6). Here the basis vector interpretation of the w_{ijk} terms is most natural, by analogy to the well-known work on eigenfaces (Kirby & Sirovich, 1990; Turk & Pentland, 1991). Each pose is represented by a vector of I parameters, a_i^{pose}, and each person by a vector of J parameters, b_j^{person}. To render an image of a particular person in a particular pose, a set of $I \times J$ basis images $\mathbf{w}_{ij}$ is linearly mixed with coefficients given by the tensor product of these two parameter vectors (see equation 2.3). The symmetric model can exactly reproduce the observations when I and J equal the numbers of observed styles S and content classes C, respectively, as is the case in Figure 2. The model provides coarser but more compact representations as these dimensionalities are decreased.

[1] The model in equation 2.1 may appear trilinear, but we view the w_{ijk} terms as describing a fixed bilinear mapping from $\mathbf{a}^s$ and $\mathbf{b}^c$ to $\mathbf{y}^{sc}$.

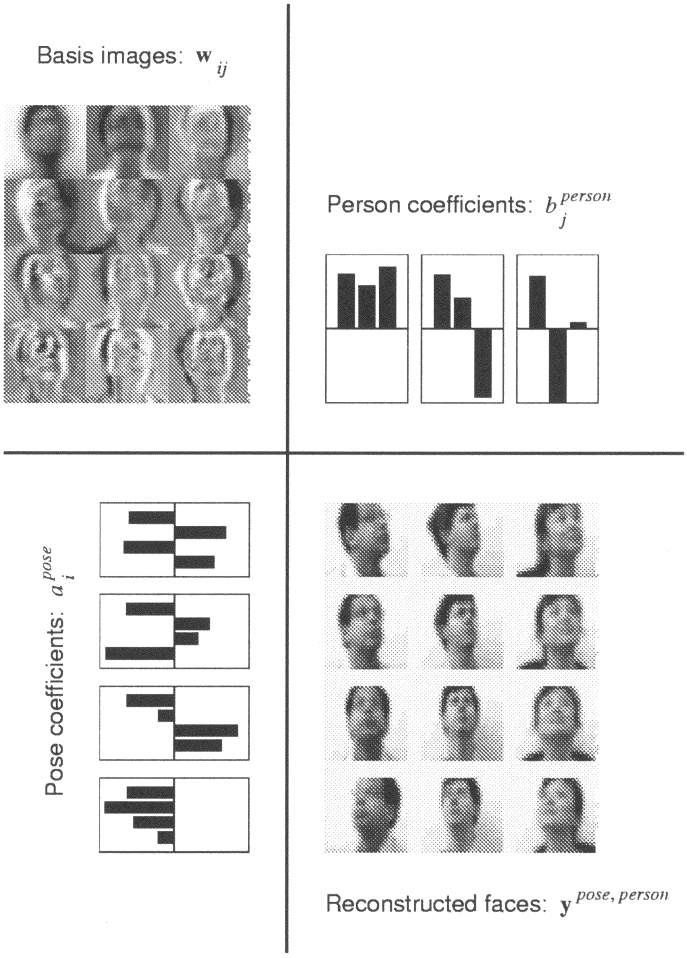

Figure 2: Illustration of a symmetric bilinear model for a small set of faces (a subset of Figure 6). The two factors for this example are person and pose. One vector of coefficients, a_i^{pose}, describes the pose, and a second vector, b_j^{person}, describes the person. To render a particular person under a particular pose, the vectors a_i^{pose} and b_j^{person} multiply along the four rows and three columns of the array of basis images $\mathbf{w}_{ij}$. The weighted sum of basis images yields the reconstructed faces $\mathbf{y}^{pose,person}$.

2.2 Asymmetric Model. Sometimes linear combinations of a few basis styles learned during training may not describe new styles well. We can obtain more flexible, *asymmetric* models by letting the interaction terms w_{ijk}

themselves vary with style. Then equation 2.1 becomes $y_k^{sc} = \sum_{i,j} w_{ijk}^s a_i^s b_j^c$. Without loss of generality, we can combine the style-specific terms of equation 2.1 into

$$a_{jk}^s = \sum_i w_{ijk}^s a_i^s, \tag{2.4}$$

giving

$$y_k^{sc} = \sum_j a_{jk}^s b_j^c. \tag{2.5}$$

Again, there are two interpretations of the model, corresponding to different vector forms of equation 2.5. First, letting $\mathbf{A}^s$ denote the $K \times J$ matrix with entries $\{a_{jk}^s\}$, equation 2.5 can be written as

$$\mathbf{y}^{sc} = \mathbf{A}^s \mathbf{b}^c. \tag{2.6}$$

Here, we can think of the a_{jk}^s terms as describing a style-specific linear map from content space to observation space. Alternatively, letting $\mathbf{a}_j^s$ denote the K-dimensional vector with components $\{a_{jk}^s\}$, equation 2.5 can be written as

$$\mathbf{y}^{sc} = \sum_j \mathbf{a}_j^s b_j^c. \tag{2.7}$$

Now we can think of the a_{jk}^s terms as describing a set of J style-specific basis vectors that are mixed according to content-specific coefficients b_j^c (independent of style) to produce the observations.

Figure 3 illustrates an asymmetric bilinear model applied to the face database, with head pose as the style factor. Each pose is represented by a set of J basis images $\mathbf{a}_j^{pose}$ and each person by a vector of J parameters b_j^{person}. To render an image of a particular person in a particular pose, the pose-specific basis images are linearly mixed with coefficients given by the person-specific parameter vector.

Note that the basis images for each pose look like eigenfaces (Turk & Pentland, 1991) in the appropriate style of each pose. However, they do not provide a true orthogonal basis for any one pose, as in Moghaddam and Pentland (1997), where a distinct set of eigenfaces is computed for each of several poses. Instead, the factorized structure of the model ensures that corresponding basis vectors play corresponding roles across poses (e.g., the first vector holds roughly the mean face for that pose, the second seems to modulate hair distribution, and the third seems to modulate head size), which is crucial for adapting to new styles. Familiar content can be easily

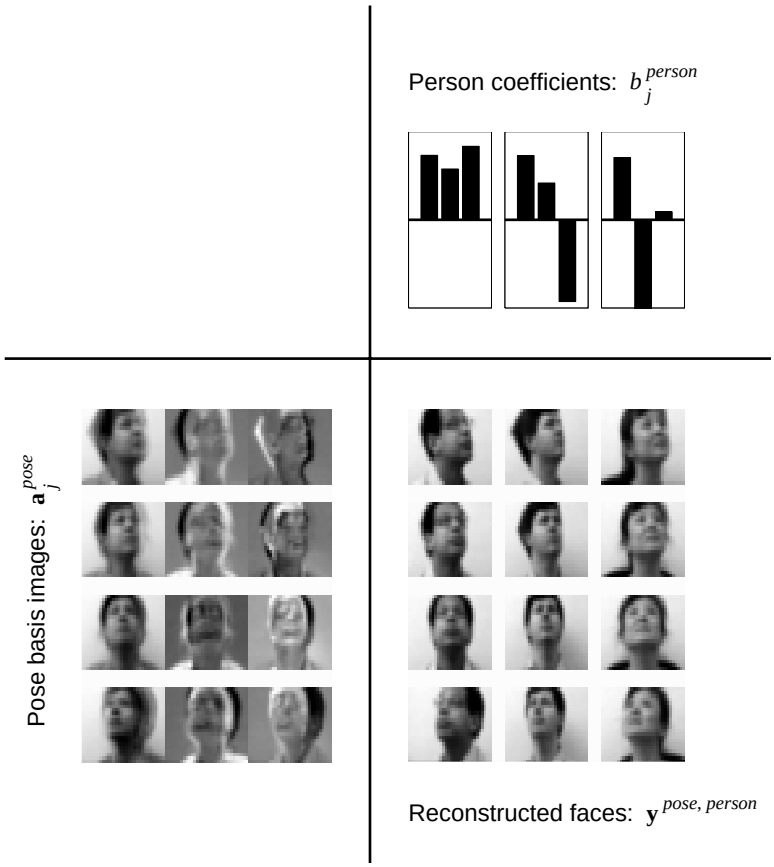

Figure 3: The images of Figure 2 represented by an asymmetric bilinear model with head pose as the style factor. Person-specific coefficients b_j^{person} multiply pose-specific basis images $\mathbf{a}_j^{pose}$; the sum reconstructs a given person in a given pose. The basis images are similar to an eigenface representation within a given pose (Moghaddam & Pentland, 1997), except that in this model the different basis images are constrained to allow one set of person coefficients to reconstruct the same face across different poses.

translated to a new style by mixing the new style-specific basis functions with the old content-specific coefficients.

Figure 4 shows the same data represented by an asymmetric model, but with the roles of style and content switched. Now the $\mathbf{a}_j^s$ parameters provide a basis set of poses for each person's face. Again, corresponding basis vectors play corresponding roles across styles (e.g., for each person's face, the first

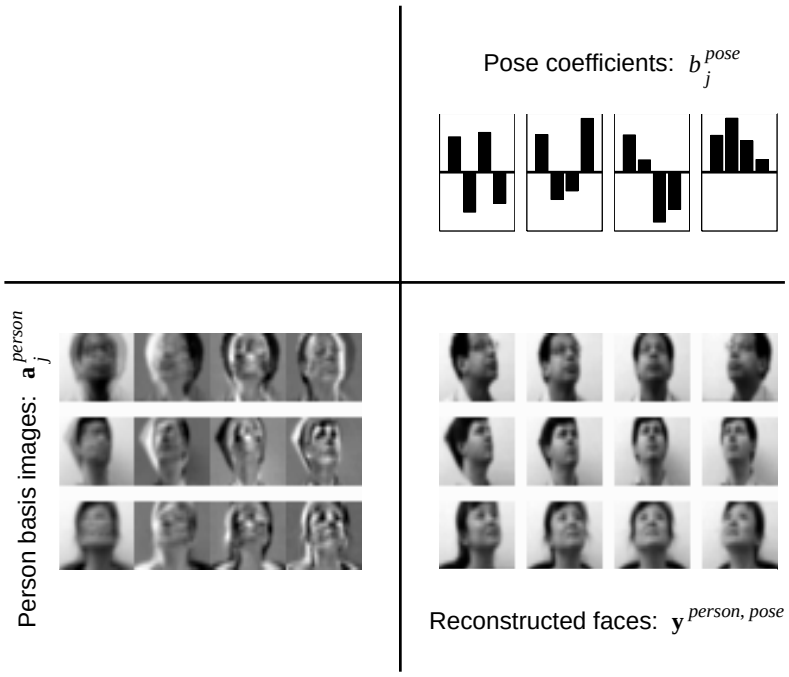

Figure 4: Asymmetric bilinear model applied to the data of Figure 2, treating identity as the style factor. Now pose-specific coefficients b_j^{pose} weight person-specific basis images $\mathbf{a}_j^{person}$ to reconstruct the face data. Across different faces, corresponding basis images play corresponding roles in rotating head position.

vector holds roughly the mean pose, the second modulates head orientation, the third modulates amount of hair showing, and the fourth adds in facial detail), allowing ready stylistic translation.

Finally, we note that because the asymmetric model can be obtained by summing out redundant degrees of freedom in the symmetric model (see equation 2.4),[2] the three sets of basis images in Figures 2 through 4 are not at all independent. Both the pose- and person-specific basis images in Figures 3 and 4 can be expressed as linear combinations of the symmetric model basis images in Figure 2, mixed according to the pose- or person-specific coefficients (respectively) from Figure 2.

The asymmetric model's high-dimensional matrix representation of style may be too flexible in adapting to data in new styles and cannot support

[2] This holds exactly only when the dimensionalities of style and content vectors are equal to the number of observed styles and content classes, respectively.

translation tasks (see Figure 1C) because it does not explicitly model the structure of observations that is independent of both style and content (represented by w_{ijk} in equation 2.1). However, if overfitting can be controlled by limiting the model dimensionality J or imposing some additional constraint, asymmetric models may solve classification and extrapolation tasks (see Figures 1A and 1B) that could not be solved using symmetric models with a realistic number of training styles.

3 Model Fitting

In conventional supervised learning situations, the data are divided into complete training patterns and incomplete (e.g., unlabeled) test patterns, which are assumed to be sampled randomly from the same distribution (Bishop, 1995). Learning then consists of fitting a model to the training data that allows the missing aspects of the test patterns (e.g., class labels in a classification task) to be filled in given the available information. The tasks in Figure 1, however, require that the learner generalize from training data sampled according to one distribution (i.e., in one set of styles and content classes) to test data drawn from a different but related distribution (i.e., in a different set of styles or content classes).

Because of the need to adapt to a different but related distribution of data during testing, our approach to these tasks involves model fitting during both training and testing phases. In the training phase, we learn about the interaction of style and content factors by fitting a bilinear model to a complete array of observations of C content classes in S styles. In the testing or generalization phase, we adapt the same model to new observations that have something in common with the training set, in either content or style or in their interaction. The model parameters corresponding to the assumed commonalities are fixed at the values learned during training, and new parameters are estimated for the new styles or content using algorithms similar to those used in training. New and old parameters are then combined to accomplish the desired classification, extrapolation, or translation task.

This section presents the basic algorithms for model fitting during training. The algorithms for model fitting during *testing* are essentially variants of these training algorithms but are task-specific; they will be presented in the appropriate sections below.

The goal of model fitting during training is to minimize the total squared error over the training set for the symmetric or asymmetric models. This goal is equivalent to maximum likelihood estimation of the style and content parameters given the training data, under the assumption that the data were generated from the models plus independently and identically distributed gaussian noise.

3.1 Asymmetric Model. Because the procedure for fitting the asymmetric model is simpler, we discuss it first. Let $\mathbf{y}(t)$ denote the t^{th} training observation ($t = 1, \ldots, T$). Let the indicator variable $h^{sc}(t) = 1$ if $\mathbf{y}(t)$ is in style s and content class c, and 0 otherwise. Then the total squared error E over the training set for the asymmetric model (in the form of equation 2.6) can be written as

$$E = \sum_{t=1}^{T} \sum_{s=1}^{S} \sum_{c=1}^{C} h^{sc}(t) ||\mathbf{y}(t) - \mathbf{A}^s \mathbf{b}^c||^2. \tag{3.1}$$

If the training set contains equal numbers of observations in each style and in each content class, there exists a closed-form procedure to fit the asymmetric model using the SVD. Although we are the first to use this procedure as the basis for a learning algorithm, it is mathematically equivalent to a family of computer vision algorithms (Koenderink & van Doorn, 1997) best known in the context of recovering structure from motion of tracked points under orthographic projection (Tomasi & Kanade, 1992).

Let

$$\bar{\mathbf{y}}^{sc} = \frac{\sum_t h^{sc}(t) \mathbf{y}(t)}{\sum_t h^{sc}(t)},$$

the mean observation in style s and content class c. These observations are most naturally represented in a three-way array, but in order to work with standard matrix algorithms, we must stack these SC K-dimensional (column) vectors into a single $(SK) \times C$ observation matrix:

$$\bar{\mathbf{Y}} = \begin{bmatrix} \bar{\mathbf{y}}^{11} & \cdots & \bar{\mathbf{y}}^{1C} \\ \vdots & \ddots & \\ \bar{\mathbf{y}}^{S1} & & \bar{\mathbf{y}}^{SC} \end{bmatrix}. \tag{3.2}$$

We can then write the asymmetric model (see equation 2.6) in compact matrix form,[3]

$$\bar{\mathbf{Y}} = \mathbf{A}\mathbf{B}, \tag{3.3}$$

identifying the $(SK) \times J$ matrix $\mathbf{A}$ and $J \times C$ matrix $\mathbf{B}$ as the stacked style and content parameters, respectively:

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}^1 \\ \vdots \\ \mathbf{A}^S \end{bmatrix}, \tag{3.4}$$

[3] Strictly speaking, equation 3.3 is a model of the *mean* observations. However, when the data are evenly distributed across styles and content classes, the parameter values that minimize the total squared error for equation 3.3 will also minimize E in equation 3.1.

$$\mathbf{B} = \left[\mathbf{b}^1 \cdots \mathbf{b}^C\right]. \tag{3.5}$$

To find the least-squares optimal style and content parameters for equation 3.3, we compute the SVD of $\bar{\mathbf{Y}} = \mathbf{USV}^T$. (By convention, we always take the diagonal elements of S to be ordered by decreasing eigenvalue.) We then define the style parameter matrix $\mathbf{A}$ to be the first J columns of $\mathbf{US}$ and the content parameter matrix $\mathbf{B}$ to be the first J rows of $\mathbf{V}^T$. The model dimensionality J can be chosen in various ways: from prior knowledge, by requiring a sufficiently good approximation to the data, or by looking for an "elbow" in the singular value spectrum.

If the training data are not distributed equally across the different styles and content classes, we must minimize equation 3.1 directly. There are many ways to do this. We use a quasi-newton method (BFGS; Press, Teukolsky, Vetterling, & Flannery, 1992) with initial parameter estimates determined by the SVD of the mean observation matrix $\bar{\mathbf{Y}}$, as described above. If there happen to be no observations in a particular style s and content class c, $\bar{\mathbf{Y}}$ will have some indeterminate (0/0) entries. Before taking the SVD of $\bar{\mathbf{Y}}$, we replace any indeterminate entries by the mean of the observations in the appropriate style s (across all content classes) or content class c (across all styles). In our experience, this method has yielded satisfactory results, although it is at least an order of magnitude slower than the closed-form SVD solution. Note that if the training data are almost equally distributed across styles and content classes, then the closed-form SVD solution found by assuming that the data are exactly balanced will almost minimize equation 3.1, and improving this solution using quasi-newton optimization may not be worth the much greater effort involved. Because all of the examples presented below have equally distributed training observations, we will need only the closed-form procedure for the remainder of this article.

3.2 Symmetric Model. The total squared error E over the training set for the symmetric model (in the form of equation 2.2) can be written as

$$E = \sum_{t=1}^{N}\sum_{s=1}^{S}\sum_{c=1}^{C}\sum_{k=1}^{K} h^{sc}(t)(y_k(t) - \mathbf{a}^{s^T}\mathbf{W}_k\mathbf{b}^c)^2. \tag{3.6}$$

Again, if we assume that the training set consists of an equal number of observations in each style and content class, there are efficient matrix algorithms for minimizing E. The algorithm we use was described for scalar observations by Magnus and Neudecker (1988) and adapted to vector observations by Marimont and Wandell (1992), in the context of characterizing color surface and illuminant spectra. Essentially, we repeatedly apply the SVD algorithm for fitting the asymmetric model, alternating the role of style and content factors within each iteration until convergence.

First we need a few matrix definitions. Recall that $\bar{\mathbf{Y}}$ consists of the SC K-dimensional mean observation vectors $\bar{\mathbf{y}}^{sc}$ stacked into a single $SK \times C$

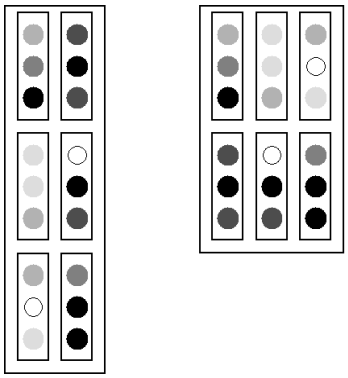

Figure 5: Schematic illustration of the vector transpose (following Marimont & Wandell, 1992). For a matrix $\mathbf{Y}$ considered as a two-dimensional array of stacked vectors (A), its vector-transpose $\mathbf{Y}^{\mathrm{VT}}$ is defined to be the same stacked vectors with their row and column positions in the array transposed (B).

matrix (see equation 3.2). In general, for any $AK \times B$ matrix $\mathbf{X}$ constructed by stacking AB K-dimensional vectors A down and B across, we can define its *vector transpose* $\mathbf{X}^{\mathrm{VT}}$ to be the $BK \times A$ matrix consisting of the same K-dimensional vectors stacked B down and A across, where the vector a across and b down in $\mathbf{X}$ becomes the vector b across and a down of $\mathbf{X}^{\mathrm{VT}}$ (see Figure 5). In particular, $\bar{\mathbf{Y}}^{\mathrm{VT}}$ consists of the means $\bar{\mathbf{y}}^{sc}$ stacked into a single $(KC) \times S$ matrix:

$$\bar{\mathbf{Y}}^{\mathrm{VT}} = \begin{bmatrix} \bar{\mathbf{y}}^{11} & \cdots & \bar{\mathbf{y}}^{1S} \\ \vdots & \ddots & \\ \bar{\mathbf{y}}^{C1} & & \bar{\mathbf{y}}^{CS} \end{bmatrix}. \tag{3.7}$$

Finally, we define the $IK \times J$ stacked weight matrix $\mathbf{W}$, consisting of the IJ K-dimensional basis functions $\mathbf{w}_{ij}$ (see equation 2.3) in the form

$$\mathbf{W} = \begin{bmatrix} \mathbf{w}_{11} & \cdots & \mathbf{w}_{1J} \\ \vdots & \ddots & \\ \mathbf{w}_{I1} & & \mathbf{w}_{IJ} \end{bmatrix}. \tag{3.8}$$

Its vector transpose $\mathbf{W}^{\mathrm{VT}}$ is also defined accordingly.

We can then write the symmetric model (see equation 2.6) in either of these two equivalent matrix forms,[4]

$$\bar{\mathbf{Y}} = \left[\mathbf{W}^{\mathrm{VT}}\mathbf{A}\right]^{\mathrm{VT}}\mathbf{B}, \tag{3.9}$$

$$\bar{\mathbf{Y}}^{\mathrm{VT}} = [\mathbf{WB}]^{\mathrm{VT}}\mathbf{A}, \tag{3.10}$$

identifying the $I \times S$ matrix $\mathbf{A}$ and the $J \times C$ matrix $\mathbf{B}$ as the stacked style and content parameter vectors, respectively:

$$\mathbf{A} = \left[\mathbf{a}^1 \cdots \mathbf{a}^S\right], \mathbf{B} = \left[\mathbf{b}^1 \cdots \mathbf{b}^C\right]. \tag{3.11}$$

The iterative procedure for estimating least-squares optimal values of $\mathbf{A}$ and $\mathbf{B}$ proceeds as follows. We initialize $\mathbf{B}$ using the closed-form SVD procedure described above for the asymmetric model (see equation 3.5). Note that this initial $\mathbf{B}$ is an orthogonal matrix ($\mathbf{BB}^{\mathrm{T}}$ is the $J \times J$ identity matrix), so that $[\bar{\mathbf{Y}}\mathbf{B}^{\mathrm{T}}]^{\mathrm{VT}} = \mathbf{W}^{\mathrm{VT}}\mathbf{A}$ (from equation 3.9). Thus, given this initial estimate for $\mathbf{B}$, we can compute the SVD of $[\bar{\mathbf{Y}}\mathbf{B}^{\mathrm{T}}]^{\mathrm{VT}} = \mathbf{USV}^T$ and update our estimate of $\mathbf{A}$ to be the first I rows of $\mathbf{V}^T$. This $\mathbf{A}$ is also orthogonal, so that $[\bar{\mathbf{Y}}^{\mathrm{VT}}\mathbf{A}^{\mathrm{T}}]^{\mathrm{VT}} = \mathbf{WB}$ (from equation 3.10). Thus, given this estimate of $\mathbf{A}$, we can compute the SVD of $[\bar{\mathbf{Y}}^{\mathrm{VT}}\mathbf{A}^{\mathrm{T}}]^{\mathrm{VT}} = \mathbf{USV}^T$ and update our estimate of $\mathbf{B}$ to be the first J rows of $\mathbf{V}^T$. This completes one iteration of the algorithm. Typically, convergence occurs within five iterations (around 10 SVD operations). Convergence is also guaranteed. (See Magnus & Neudecker, 1988, for a proof for the scalar case, $K = 1$, which can be extended to the vector case considered here.) Upon convergence, we solve for $\mathbf{W} = [[\bar{\mathbf{Y}}\mathbf{B}^{\mathrm{T}}]^{\mathrm{VT}}\mathbf{A}^{\mathrm{T}}]^{\mathrm{VT}}$ to obtain the basis vectors independent of both style and content. As with the asymmetric model, if the training data are not distributed equally across the different styles and content classes, we minimize equation 3.1 starting from the same initial estimates for $\mathbf{A}$ and $\mathbf{B}$ but using a more costly quasi-Newton method.

4 Classification

Many common classification problems involve multiple observations likely to be in one style, for example, recognizing the handwritten characters on an envelope or the accented speech of a telephone voice. People are significantly better at recognizing a familiar word spoken in an unfamiliar voice or a familiar letter character written in an unfamiliar font when it is embedded in the context of other words or letters in the same novel style

[4] As with the asymmetric model, when the data are evenly distributed across styles and content classes, the parameter values that minimize the total squared error for the mean observations in equations 3.9 and 3.10 will also minimize E in Equation 3.6.

(van Bergem, Pols, & van Beinum, 1988; Sanocki, 1992; Nygaard & Pisoni, 1998), presumably because the added context allows the perceptual system to build a model of the new style and factor out its influence.

In this section, we show how a perceptual system may use bilinear models and assumptions of style consistency to factor out the effects of style in content classification and thereby significantly improve classification performance on data in novel styles. We first describe the two concrete tasks investigated. We then present the general classification algorithm and the results of several experiments comparing this algorithm to standard techniques from the pattern recognition literature, such as nearest neighbor classification, which do not explicitly model the effects of style on content classification.

4.1 Task Specifics. We report experiments with two data sets: a benchmark speech data set and a face data set that we collected ourselves. The speech data consist of six samples of each of 11 vowels (content classes) uttered by 15 speakers (styles) of British English.[5] Each data vector consists of $K = 10$ log area parameters, a standard vocal tract representation computed from a linear predictive coding analysis of the digitized speech. The specific task we must learn to perform is classification of spoken vowels (content) for new speakers (styles).

The face data were introduced in section 2 (see Figures 2–4). Figure 6 shows the complete data set, consisting of images of 11 people's faces (styles) viewed under 15 different poses (content classes). The poses span a grid of three vertical positions (up, level, down) and five horizontal positions (far left, left, straight ahead, right, far right). The pictures were shifted to align the nose tip position, found manually. The images were then blurred and cropped to 22×32 pixels, and represented simply as vectors of $K = 704$ pixel values each. The specific task we must learn to perform is classification of head pose (content) for new people's faces (styles).

4.2 Algorithm. For both speech and face data sets, we train on observations in all content classes but in a subset of the available styles (the "training" styles). We fit asymmetric bilinear models (see equation 3.3) to these training data using the closed-form SVD procedure described in section 3.1. This yields a $K \times J$ matrix $\mathbf{A}^s$ representing each style s and a J-dimensional vector $\mathbf{b}^c$ representing each content class c. The model dimensionality J is a free parameter, which we discuss in depth at the end of this section. Fitting these asymmetric models to either data set required less than 1 second. This and all other execution times reported below were obtained for nonoptimized MATLAB code running on a Silicon Graphics O2 workstation.

[5] The data were originally collected by David Deterding and are now available online from the UC Irvine machine learning repository (http://www.ics.uci.edu/~mlearn).

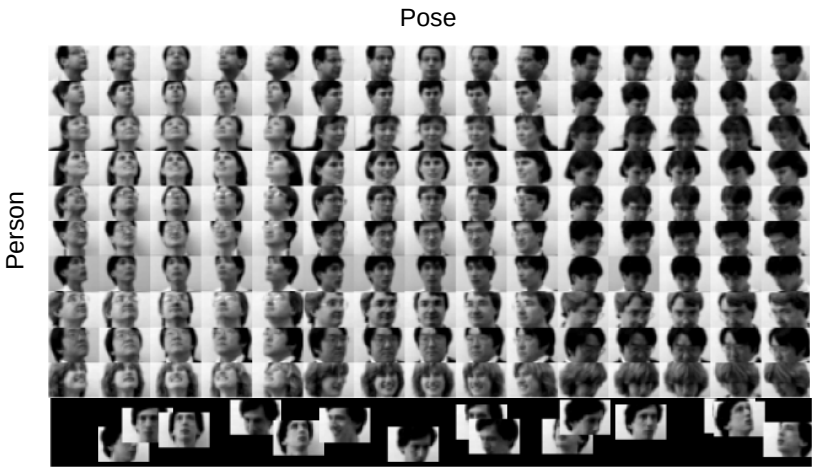

Figure 6: Classification of familiar content in a new style, illustrated on a data set of face images. During training, we observe the faces of different people (style) in different poses (content), resulting in the matrix of faces shown. During generalization, we observe a new person's face in each of these familiar poses, and we seek to classify the pose of each new image. The separable mixture model allows us to build up a model for the new face at the same time that we classify the new poses, thereby improving classification performance.

The generalization task is to classify observations in the remaining styles (the "test" styles) by filling in the missing content labels for these novel observations using the style-invariant content vectors $\mathbf{b}^c$ fixed during training (see Figure 6). Trying to estimate both content labels and style parameters for the new data presents a classic chicken-and-egg problem, very much like the problems of k-means clustering or mixture modeling (Duda & Hart, 1973). If the content class assignments were known, then it would be easy to estimate parameters for a new style $\tilde{s}$ by simply inserting into the basic asymmetric bilinear model (i.e., equation 2.6) all the observation vectors in style $\tilde{s}$, along with the appropriate content vectors $\mathbf{b}^c$, and solving for the style matrix $\mathbf{A}^{\tilde{s}}$. Similarly, if we had a model $\mathbf{A}^{\tilde{s}}$ of new style $\tilde{s}$, then we could classify any test observation from this new style based simply on its distance to each of the known content vectors $\mathbf{b}^c$ multiplied by $\mathbf{A}^{\tilde{s}}$.[6] Initially, however, both style models and content class assignments are unknown for the test data.

To handle this uncertainty, we embed the bilinear model within a gaussian mixture model to yield a *separable mixture model* (SMM) (Tenenbaum &

[6] This is equivalent to maximum likelihood classification, assuming gaussian likelihood functions centered on the predictions of the bilinear model $\mathbf{A}^{\tilde{s}}\mathbf{b}^c$.

Freeman, 1997), which can then be fit efficiently to new data using the EM algorithm (Dempster, Laird, & Rubin, 1977). The mixture model has $S \times C$ gaussian components—one for each pair of S styles and C content classes—with means given by the predictions of the bilinear model. However, only on the order of $(S + C)$ parameters are needed to represent the means of these $S \times C$ gaussians, because of the bilinear model's separable structure. To classify known content in new styles and estimate new style parameters simultaneously, the EM algorithm alternates between calculating the probabilities of all content labels given current style parameter estimates (E-step) and estimating the most likely style parameters given current content label probabilities (M-step), with likelihood determined by the gaussian mixture model. If, in addition, the test data are not segmented according to style, style label probabilities can be calculated simultaneously as part of the E-step.

More formally, after training on labeled data from S styles and C content classes, we are given test data from the same C content classes and $\tilde{S}$ new styles, with labels for content (and possibly also style) missing. The new style matrices $\mathbf{A}^{\tilde{s}}$ are also unknown, but the content vectors $\mathbf{b}^c$ are known, having been fixed in the training phase. We assume that the probability of a new unlabeled observation $\mathbf{y}$ being generated in new style $\tilde{s}$ and old content c is given by a gaussian distribution with variance σ^2 centered at the prediction of the bilinear model: $p(\mathbf{y}|\tilde{s}, c) \propto \exp\{-\|\mathbf{y} - \mathbf{A}^{\tilde{s}}\mathbf{b}^c\|^2/(2\sigma^2)\}$. The total probability of $\mathbf{y}$ is then given by the mixture distribution $p(\mathbf{y}) = \sum_{\tilde{s},c} p(\mathbf{y}|\tilde{s}, c)p(\tilde{s}, c)$. Here we assume equal prior probabilities $p(\tilde{s}, c)$, unless the observations are otherwise labeled.[7] The EM algorithm (Dempster et al., 1977) alternates between two steps in order to find new style matrices $\mathbf{A}^{\tilde{s}}$ and style-content labels $p(\tilde{s}, c|\mathbf{y})$ that best explain the test data. In the E-step, we compute the probabilities $p(\tilde{s}, c|\mathbf{y}) = p(\mathbf{y}|\tilde{s}, c)p(\tilde{s}, c)/p(\mathbf{y})$ that each test vector $\mathbf{y}$ belongs to style $\tilde{s}$ and content class c, given the current style matrix estimates. In the M-step, we estimate new style matrices by setting $\mathbf{A}^{\tilde{s}}$ to maximize the total log-likelihood of the test data, $L^* = \sum_{\mathbf{y}} \log p(\mathbf{y})$. The M-step can be computed in closed form by solving the equations $\partial L^*/\partial \mathbf{A}^{\tilde{s}} = \mathbf{0}$, which are linear in $\mathbf{A}^{\tilde{s}}$ given the probability estimates from the E-step and the quantities $\mathbf{m}^{\tilde{s}c} = \sum_{\mathbf{y}} p(\tilde{s}, c|\mathbf{y})\mathbf{y}$ and $n^{\tilde{s}c} = \sum_{\mathbf{y}} p(\tilde{s}, c|\mathbf{y})$:

$$\mathbf{A}^{\tilde{s}} = \left[\sum_c \mathbf{m}^{\tilde{s}c}\mathbf{b}^{c^{\mathrm{T}}}\right]\left[\sum_c n^{\tilde{s}c}\mathbf{b}^c\mathbf{b}^{c^{\mathrm{T}}}\right]^{-1}. \tag{4.1}$$

The EM algorithm is guaranteed to converge to a local maximum of L^*, and this typically takes around 20 iterations for our problems. After each

[7] That is, if the style identity s^* of $\mathbf{y}$ is labeled but the content class is unknown, we let $p(\tilde{s}, c) = 1/C$ if $\tilde{s} = s^*$ and 0 if $\tilde{s} \neq s^*$. If both the style and content identities of $\mathbf{y}$ are unlabeled, we let $p(\tilde{s}, c) = 1/(\tilde{S}C)$ for all $\tilde{s}$ and c.

E-step, test vectors in new styles can be classified by selecting the content class c that maximizes $p(c|\mathbf{y}) = \sum_{\tilde{s}} p(\tilde{s}, c|\mathbf{y})$.[8] Classification performance is determined by the percentage of test data for which the probability of content class c, as given by EM, is greatest for the actual content class. Note that because content classification is determined as a by-product of the E-step, the standard practice of running EM until convergence to a local maximum in likelihood will not necessarily lead to optimal classification performance. In fact, we have often observed overfitting behavior, in which optimal classification is obtained after only two or three iterations of EM, but the likelihood continues to increase in subsequent iterations as observations are assigned to incorrect content classes.

This classification algorithm thus has three free parameters for which good values must somehow be determined. In addition to the model dimensionality J mentioned above, these parameters include the model variance σ^2 and the maximum number of iterations for which EM is run, $t_{\max}$. In general, we set these parameters using a leave-one-style-out cross-validation procedure with the training data. That is, given S complete training styles, we train separate bilinear models on each of the S subsets of $S - 1$ styles and evaluate these models' classification performance on the one style that each was not trained on. For each of these S training set splits, we try a range of values for the parameters J, σ^2, and $t_{\max}$. Those values that yield the best average performance over the S training set splits are then used in fitting a new bilinear model to the full training data and generalizing to the designated test set. The main drawback of this cross-validation procedure is that it can be time-consuming. Although each run of EM is quite fast—on the order of seconds—an exhaustive search of the full parameter space in a leave-one-out design may take hours. In real applications, such a search would hopefully need to be done only once in a given domain, if at all, and might also be guided by domain-specific knowledge that could narrow the search space significantly.

Initialization is an important factor in determining the success of the EM algorithm. Because we are primarily interested in good classification, we initialize EM in the E-step, using the results of a simple 1-nearest-neighbor classifier to set the content-class assignments $p(c|\mathbf{x})$. That is, we initially assign each test observation to the content class of the most similar training observation (for which the content labels are known).

4.3 Results. We conducted four experiments: three using the benchmark speech data to investigate different aspects of the algorithm's behavior and one using the face data to provide evidence from a separate domain. In each case, we report results with all three free parameters set using the

[8] If the style s^* of y is known, then $p(\tilde{s}, c|\mathbf{y}) = 0$ for all $\tilde{s} \neq s^*$ (see previous note). In that case, $p(c|\mathbf{y})$ is simply equal to $p(s^*, c|\mathbf{y})$.

Table 1: Accuracy of Classifying Spoken Vowels from a Benchmark Multispeaker Database.

Classifier	Percentage Correct on Test Data
Multilayer perceptron (MLP)	51%
Radial basis function network (RBF)	53%
1-nearest neighbor (1-NN)	56%
Discriminant adaptive nearest neighbor (DANN)	
Generic parameter settings	59.7%
Optimal parameter settings	61.7%
Separable mixture models (SMM)—test speakers labeled	
All parameters set by CV ($J = 3, \sigma^2 = 1/64, t_{\max} = 2$)	75.8%
$t_{\max} = \infty$; J, σ^2 set by CV ($J = 3, \sigma^2 = 1/32$)	68.2%
$t_{\max} = \infty$; optimal J, σ^2 ($J = 4, \sigma^2 = 1/16$)	77.3%
Separable mixture models (SMM)—test speakers *not* labeled	
All parameters set by CV ($J = 3, \sigma^2 = 1/64, t_{\max} = 2$)	$69.8\% \pm .3\%$
$t_{\max} = \infty$; J, σ^2 set by CV ($J = 3, \sigma^2 = 1/32$)	$59.9\% \pm 1.1\%$
$t_{\max} = \infty$; optimal J, σ^2 ($J = 3, \sigma^2 = 1/64$)	$63.2\% \pm 1.5\%$

Notes: MLP, RBF, and 1-NN results were obtained by Robinson (1989). The DANN classifier (Hastie & Tibshirani, 1996) achieves the best performance we know of for an approach that does not adapt to new speakers. The SMM classifiers perform significantly better vowel classification by simultaneously modeling speaker style. "CV" denotes the cross-validation procedure for parameter setting described in the text.

cross-validation procedure described above, as well as for two conditions in which EM was run until convergence ($t_{\max} = \infty$): both J and σ^2 determined by cross validation and both J and σ^2 set to their optimal values (as an indicator of best in-principle performance for the maximum likelihood solution).

4.3.1 Train 8, Test 7 on Speech Data—Speakers Labeled. The first experiment with the speech data was the standard benchmark task described in Robinson (1989). Robinson compared many learning algorithms trained to categorize vowels from the first eight speakers (four male and four female) and tested on samples from the remaining seven speakers (four male and three female). The variety and the small number of styles make this a difficult task. Table 1 shows the best results we know of for standard approaches that do not adapt to new speakers. Of the many techniques that Robinson (1989) tested, 1-nearest neighbor (1-NN) performs the best, with 56.3% correct; chance is approximately 9% correct. The discriminant adaptive nearest neighbor (DANN) classifier (Hastie & Tibshirani, 1996) slightly outperforms 1-NN, obtaining 59.7% correct with its generic parameter settings and 61.7% correct for optimal parameter settings.

After fitting an asymmetric bilinear model to the training data, we tested classification performance using our SMM on the seven new speakers' data. We first assumed that style labels were available for the test data (indicating only a change of speaker, but no information about the new speaker's style). Running EM until convergence ($t_{\max} = \infty$), our best results of 77.3% correct were obtained with $J = 4$ and $\sigma^2 = 1/16$. Almost comparable results of 75.8% correct were obtained using cross-validation to set all parameters ($J, \sigma^2, t_{\max}$) automatically (see Table 1). The SMM clearly outperforms the many nonadaptive techniques tested, exploiting extra information available in the speaker labels that nonadaptive techniques make no use of.

4.3.2 Train 8, Test 7 on Speech Data—Speakers Not Labeled. We next repeated the benchmark experiment without the assumption that any style labels were available during testing. Thus, our SMM algorithm and the nonadaptive techniques had exactly the same information available for each test observation (although the SMM had style labels available during training). We used EM to figure out both the speaker assignments and the vowel class assignments for the test data. EM requires that the number of style components S in the mixture model be set in advance; we chose $S = 7$ (the actual number of distinct speakers) for consistency with the previous experiment. In initializing EM in the E-step, we assigned each test observation equally to each new style component, plus or minus 10% random noise to break symmetry and allow each style component to adapt to a distinct subset of the new data. Using cross-validation to set J, σ^2, and $t_{\max}$ automatically, we obtained $69.8\% \pm 0.3\%$ correct. Table 1 presents results for other parameter settings. These scores reflect average performance over 10 random initializations. Not surprisingly, SMM performance here was worse than in the previous section, where the correct style labels were assumed to be known. Nonetheless, the SMM still provided a significant improvement over the best nonadaptive approaches tested.

4.3.3 Train 14, Test 1 on Speech Data. We noticed that the performance of the SMM on the speech data varied widely across different test speakers and also depended significantly on the particular speakers chosen for training and testing. In some cases, the SMM did not perform much better than 1-NN, and in other cases the SMM actually did worse. Thus, we decided to conduct a more systematic study of the effects of individual speaker style on generalization. Specifically, we tested the SMM's ability to classify the speech of each of the 15 speakers in the database individually when the other 14 speakers were used for training. Because only one speaker was presented during testing, we used only a single style model in EM, and thus there was no distinction between the "speakers labeled" and "speakers not labeled" conditions investigated in the previous two sections.

Averaged over all 15 possible test speakers, 1-NN obtained $63.9\% \pm 3.4\%$ correct. Using cross-validation to set J, σ^2, and $t_{\max}$ automatically, our SMM

obtained 74.3% ± 4.2% correct. Running EM until convergence ($t_{\max} = \infty$), we obtained 75.6% ± 4.0% using the best parameter settings of $J = 11$, $\sigma^2 = .01$, and 73.5% ± 4.4% correct using cross-validation to select J and σ^2 automatically. These SMM scores are not significantly different from each other but are all significantly higher than 1-NN as measured by paired t-tests ($p < .01$ in all cases). The results suggest that the superior performance of SMM over nonadaptive classifiers such as nearest neighbor will hold in general over a range of different test styles.

This experiment also provided an opportunity to assess systematically the performance details of EM on the task of speech classification. For all 15 possible test speakers, EM converged within 30 iterations, taking less than 5 seconds. The optimal number of iterations for EM as determined by cross-validation was always between 1 and 10 iterations, with a mean of 3.2 iterations. We can assess the chance of falling into a bad local minimum with EM by comparing the classification performance of the SMM fit using EM with the performance of 1-NN on a speaker-by-speaker basis. For 14 of 15 test speakers, the SMM consistently obtained better classification rates than 1-NN, regardless of which of the three parameter-setting procedures described above was used. The remaining speaker was always classified better by 1-NN. Evidently the vast majority of stylistic variations in this domain can be successfully modeled by the SMM and separated from the relevant content to be classified using the EM algorithm.

4.3.4 Train 10, Test 1 on Face Data. To provide further support for the general advantage of SMM over nonadaptive approaches, we replicated the previous experiment using the face database instead of the speech data. Although the two databases are of roughly comparable size, the nature of the observations is quite different: 704-dimensional vectors of pixel values versus 10-dimensional vectors of vocal tract log area coefficients. Specifically, we tested the SMM's ability to classify head pose for each of the 11 faces in the database individually when the other 10 people's faces were used for training. There were 15 possible poses, with one image of each face in each pose.

Averaged over all 11 possible test faces, 1-NN obtained 53.9% ± 4.3% correct. Using cross-validation to set J, σ^2, and $t_{\max}$ automatically, our SMM obtained 73.9% ± 6.7% correct. Running EM until convergence ($t_{\max} = \infty$), we obtained 80.6% ± 7.5% using the best parameter settings of $J = 6$, $\sigma^2 = 10^5$, and 75.8% ± 6.4% correct using cross-validation to select J and σ^2 automatically. As on the speech data, these SMM scores are not significantly different from each other but do represent significant improvements over 1-NN as measured by paired t-tests ($p < .05$ in all cases).

The performance details of EM on this face classification task were quite similar to those on the speech classification task reported above. For all 11 possible test faces, EM converged within 30 iterations, taking less than 20 seconds. (The longer convergence times here reflect the higher-dimensional

vectors involved.) The optimal number of iterations for EM as determined by cross-validation was between 9 and 20 iterations, with a mean of 12.6 iterations. For 9 of 11 test faces, the SMM consistently obtained better classification rates than 1-NN over all three parameter-setting procedures we used, indicating that local minima do not prevent the EM algorithm from factoring out the majority of stylistic variations in this domain.

4.4 Discussion. Our approach to style-adaptive content classification involves two significant modeling choices: the use of a bilinear model of the mean observations and the use of a gaussian mixture model—centered on the predictions of the bilinear model—for observations whose content or style assignments are unknown. The mixture model provides a principled probabilistic framework, allowing us to use the EM algorithm to solve the chicken-and-egg problem of simultaneously estimating style parameters for the new data and labeling the data according to content class (and possibly style as well). The bilinear structure of the model allows the M-step to be computed in closed form by solving systems of linear equations. In this sense, bilinear models represent the content of observations independent of their style in a form that can be generalized easily to model data in new styles. To summarize our results, we found that separating style and content with bilinear models improves content classification in new styles substantially over the best nonadaptive approaches to classification, even when no style information is available during testing, and dramatically so when style demarcation is available. Although our SMM classifier has several free parameters that must be chosen a priori, we showed that near-optimal values can be determined automatically using a cross-validation training procedure. We obtained good results on two very different data sets, low-dimensional speech data and high-dimensional face image data, suggesting that our approach may be widely applicable to many two-factor classification tasks that can be thought of in terms of recognizing invariant content elements under variable style.

5 Extrapolation

The ability to draw analogies across observations in different contexts is a hallmark of human perception and cognition (Hofstadter, 1995; Holyoak & Barnden, 1994). In particular, the ability to produce analogous content in a novel style—and not just recognize it, as in the previous section—has been taken as a severe test of perceptual abstraction (Hofstadter, 1995; Grebert, Stork, Keesing, & Mims, 1992). The domain of typography provides a natural place to explore these issues of analogy and production. Indeed, Hofstadter (1995) has argued that the question, "What is the letter 'a'?" may be "the central problem of AI" (p. 633). Following Hofstadter, we study the task of extrapolating a novel font from an incomplete set of letter observations in that font to the remaining unobserved letters. We first describe the

task specifics and our shape representation for letter observations. We then present our algorithm for stylistic extrapolation based on bilinear models and show results on extrapolating a natural font.

5.1 Task Specifics. Given a training set of $C = 62$ characters (content) in $S = 5$ standard fonts (style), the task is to generate characters that are stylistically consistent with letters in a novel sixth font. The initial data were obtained by digitizing the uppercase letters, lowercase letters, and digits 0–9 of the six fonts at 38×38 pixels using Adobe Photoshop. Successful shape modeling often depends on having an image representation that makes explicit the appropriate structure that is only implicit in raw pixel values. Specifically, the need to represent shapes of different topologies in comparable forms motivates using a particle-based representation (Szeliski & Tonnesen, 1992). We also want the letters in our representation to behave like a linear vector space, where linear combinations of letters also look like letters. Beymer and Poggio (1996) advocate a dense warp map for related problems. Combining these two ideas, we chose to represent each letter shape by a $2 \times 38 \times 38 = 2888$-dimensional vector of (horizontal and vertical) displacements that a set of $38 \times 38 = 1444$ ink particles must undergo to form the target shape from a reference grid.

With identical particles, there are many possible such warp maps. To ensure that similarly shaped letters are represented by similar warps, we use a physical model. We give each particle of the reference shape (taken to be the full rectangular bitmap) unit positive charge, and each pixel of the target letter negative charge proportional to its gray level intensity. The total charge of the target letter is set equal to the total charge of the reference shape. We track the electrostatic force lines from each particle of the reference shape to where they intersect the plane of the target letter (see Figure 7A). The target shape is positioned opposite the reference shape, 100 pixel units away in depth. The force lines land in a uniform density over the target letter, creating a correspondence between each ink particle of the reference shape and a position on the target letter. The result is a smooth, dense warp map specifying a translation that each ink particle of the reference shape must undergo to reassemble the collection of particles into the target letter shape (see Figure 7B). The electrostatic forces used to find the correspondences are easily calculated from Coulomb's law (Jackson, 1975). We call this a *Coulomb warp* representation. To render a warp map representation of a shape, we first translate each particle of the reference shape by its warp map value, using a grid at four times the linear pixel resolution. We then blur with a gaussian (standard deviation equal to five pixels) threshold the ink densities at 60% of their maximum and subsample to the original font resolution. By allowing noninteger charge values and subpixel translations, we can preserve font antialiasing information.

Figure 8 shows three pairs of shapes of different topologies, and the average of each pair in a pixel representation and in a Coulomb warp rep-

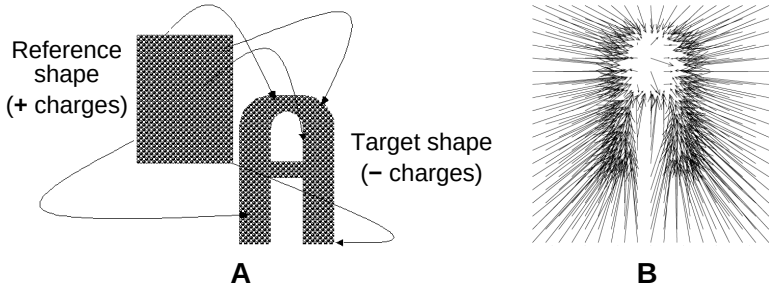

Figure 7: Coulomb warp representation for shape. A target shape is represented by the displacements that a collection of ink particles would undergo to assemble themselves into the target shape. (A) Correspondences between ink particles of the reference shape and those of the target shape are found by tracing the electrostatic force lines between positively and negatively charged ink particles. (B) The vector from the start to the end of the field line specifies the translation for each particle. The result is a smooth warp, with similar shapes represented by similar warps.

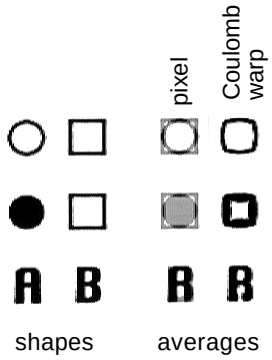

Figure 8: The result of averaging shapes together under different representations. Averaging in a pixel representation always gives a "double exposure" of the two shapes. Under the Coulomb warp representation, a circle averaged with a square gives a rounded square; a filled circle averaged with a square gives a very thick rounded square. The letter A averaged with the letter B yields a shape that is arguably intermediate to the shapes of the two letters. This property of the representation makes it well suited to linear algebraic manipulations with bilinear models.

resentation. Averaging the shapes in a pixel representation simply yields a "double exposure" of the two images; averaging in a Coulomb warp representation results in a shape intermediate to the two being averaged.

5.2 Algorithm. During training, we fit the asymmetric bilinear model (see equation 3.3) to five full fonts using the closed-form SVD procedure described in section 3.1. This yields a $K \times J$ matrix $\mathbf{A}^s$ representing each font s and a J-dimensional vector $\mathbf{b}^c$ representing each letter class c, with the observation dimensionality $K = 2888$. Training on this task took approximately 48 seconds.

Adapting the model to an incomplete new style $\tilde{s}$ can be carried out in closed form using the content vectors $\mathbf{b}^c$ learned during training. Suppose we observe M samples of style $\tilde{s}$, in content classes $C = \{c_1, \ldots, c_M\}$. We find the style matrix $\mathbf{A}^{\tilde{s}}$ that minimizes the total squared error over the test data,

$$E^* = \sum_{c \in C} \|\mathbf{y}^{\tilde{s}c} - \mathbf{A}^{\tilde{s}}\mathbf{b}^c\|^2. \tag{5.1}$$

The minimum of E^* is found by solving the linear system $\partial E^*/\partial \mathbf{A}^{\tilde{s}} = \mathbf{0}$. Missing observations in the test style $\tilde{s}$ and known content class c can then be synthesized from $\mathbf{y}^{\tilde{s}c} = \mathbf{A}^{\tilde{s}}\mathbf{b}^c$.

In order to allow the model sufficient expressive range to produce natural-looking letter shapes, we set the model dimensionality J as high as possible. However, such a flexible model led to overfitting on the available letters of the test font and consequently poor synthesis of the missing letters in that font. To regularize the style fit to the test data and thereby avoid overfitting, we add a prior term to the squared-error cost of equation 5.1 that encourages $\mathbf{A}^{\tilde{s}}$ to be close to the linear combination of training style parameters $\mathbf{A}^1, \ldots, \mathbf{A}^S$, that best fits the test font. Specifically, let $\mathbf{A}^{OLC}$ be the value of $\mathbf{A}^{\tilde{s}}$ that minimizes equation 5.1 subject to the constraint that $\mathbf{A}^{OLC}$ is a linear combination of the training style parameters $\mathbf{A}^s$, that is, $\mathbf{A}^{OLC} = \sum_{s=1}^{S} \alpha_s \mathbf{A}^s$ for some values of α_s. (OLC stands for "optimal linear combination.") Without loss of generality, we can think of the α_s coefficients as the best-fitting style parameters of a symmetric bilinear model with dimensionality I equal to the number of styles S. We then redefine E^* to include this new cost,

$$E^* = \sum_{c \in C} \|\mathbf{y}^{\tilde{s}c} - \mathbf{A}^{\tilde{s}}\mathbf{b}^c\|^2 + \lambda \|\mathbf{A}^{\tilde{s}} - \mathbf{A}^{OLC}\|^2, \tag{5.2}$$

and again minimize E^* by solving the linear system $\partial E^*/\partial \mathbf{A}^{\tilde{s}} = \mathbf{0}$. The tradeoff between these two costs is determined by the free parameter λ, which we set by eye to yield results with the best appearance. For this example, we used a model dimensionality of 60 and $\lambda = 2 \times 10^4$.[9] All model fitting during this testing phase took less than 10 seconds.

[9] This large value of λ was necessary to compensate for the differences in scale between

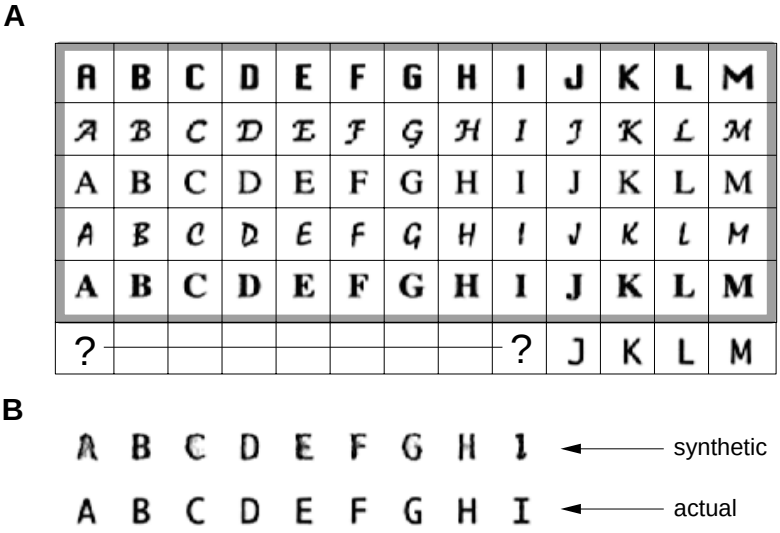

Figure 9: Style extrapolation in typography. (A) Rows 1–5: The first 13 (of 62) letters of the training fonts. Row 6: The novel test font, with A–I unseen by the model. (B) Row 1: Font extrapolation results. Row 2: The actual unseen letters of the test font.

5.3 Results. Figure 9 shows the results of extrapolating the unseen letters "A"–"I" of a new font, Monaco, using the asymmetric model with a symmetric model (OLC) prior as described above. All characters in the Monaco font except the uppercase letters "A" through "I" were used to estimate its style parameters (via equation 5.2). In contrast to the objective performance scores on the classification tasks reported in the previous section, here the evaluation of our results is necessarily subjective. Given examples of "A" through "I" in only the five training fonts, the model nonetheless has succeeded in rendering these letters in the test font, with approximately correct shapes for each letter class and with the distinctive stylistic features of Monaco: strict upright posture and uniformly thin strokes. Note that each of these stylistic features appears separately in one or more of

the two error norms in equation 5.2 and does not directly represent the relative importance of these two terms. To assess the relative contributions of the asymmetric and symmetric (OLC) terms to the new style matrix $\mathbf{A}^{\tilde{s}}$, we can compare the distances from $\mathbf{A}^{\tilde{s}}$ to the two style matrices that minimize equation 5.2 for $\lambda = 0$ and $\lambda = \infty$. Call these two matrices $\mathbf{A}^{\tilde{s}}_{asym}$ and $\mathbf{A}^{\tilde{s}}_{sym} (= \mathbf{A}^{OLC})$, respectively. For $\lambda = 2 \times 10^4$, the distance $\|\mathbf{A}^{\tilde{s}} - \mathbf{A}^{\tilde{s}}_{asym}\|$ is approximately 2.5 times greater than the distance $\|\mathbf{A}^{\tilde{s}} - \mathbf{A}^{\tilde{s}}_{sym}\|$, suggesting that the symmetric (OLC) term makes a greater—but not vastly greater—contribution to the new style matrix.

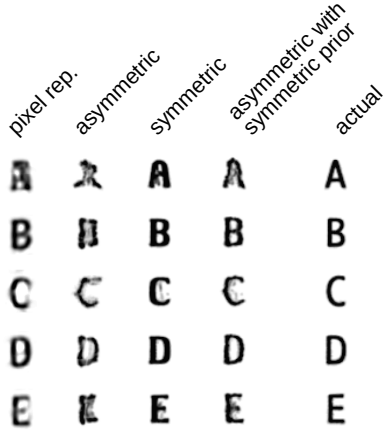

Figure 10: Result of different methods applied to the font extrapolation problem (Figure 9), where unseen letters in a new font are synthesized. The asymmetric bilinear model has too many parameters to fit, and generalization to new letters is poor (second column). The symmetric bilinear model has only 5 degrees of freedom for our data and fails to represent the characteristics of the new font (third column). We used the symmetric model result as a prior to constrain the flexibility of the asymmetric model, yielding the results shown here (fourth column) and in Figure 9. All of these methods used the Coulomb warp representation for shape. Performing the same calculations in a pixel representation requires blurring the letters so that linear combinations can modify shape, and yields barely passable results (first column). The far right column shows the actual letters of the new font.

the training fonts, but they do not appear together in any one training font.

5.4 Discussion. We have shown that it is possible to learn the style of a font from observations and extrapolate that style to unseen letters, using a hybrid of asymmetric and symmetric bilinear models. Note that the asymmetric model uses 173,280 degrees of freedom (the 2888×60 matrix $\mathbf{A}^{s}$) to describe the test style, while the optimal linear combination style model $\mathbf{A}^{OLC}$ uses only five degrees of freedom (the coefficients α_s of the five training styles). Results using only the high-dimensional asymmetric model without the low-dimensional OLC term are far too unconstrained and fail to look like recognizable letters (see Figure 10, second column).

Results using only the low-dimensional OLC model without the high-dimensional asymmetric model are clearly recognizable as the correct letters, but fail to capture the distinctive style of Monaco (see Figure 10, third column). The combination of these two terms, with a flexible high-

dimensional model constrained to lie near the subspace of known style parameters, is capable of successful stylistic extrapolation on this example (see Figure 9 and Figure 10, fourth column). Perceptually, the hybrid model results appear more similar to the low-dimensional OLC results than to the high-dimensional asymmetric model results, which is consistent with the fact that the hybrid style matrix is somewhat closer in Euclidean distance to the OLC matrix than to the pure asymmetric model (see note 9).

Using an appropriate representation, such as our Coulomb warp, was also important in obtaining visually satisfying results. Applying the same modeling methodology to a pixel space representation of letters resulted in significantly less appealing output (see Figure 10, first column). Previous models of extrapolation and abstraction in typography have been restricted to artificial grid-based fonts, for which the grid elements provide a reasonable distributed representation (Hofstadter, 1995; Grebert et al., 1992), or even simpler "grandmother cell" representations of each letter (Polk & Farah, 1997). In contrast, our shape representation allowed us to work directly with natural fonts.

Why did this extrapolation task require a hybrid modeling strategy and a specially designed input representation, while the classification tasks of the previous section did not? Two general differences between classification and extrapolation tasks make the latter kind of task objectively more difficult. First, successful classification requires only that the outputs of the bilinear model be close to the test data under the generic metric of mean squared error, while success in extrapolation is judged by the far more subtle metric of visual appearance (Teo & Heeger, 1994). Second, successful classification requires that the model outputs be close to the data only in a relative sense—the test data must be closer to the model outputs using the correct content classes than to the model outputs using incorrect content classes—while extrapolation tasks require model and data to be close in an absolute sense—the model outputs must actually look like plausible instances of the correct content classes.

Although the choice of model and representation turned out to be essential in this example, our results were obtained without any detailed knowledge or processing specific to the domain of typography. Hofstadter (1995) has been critical of approaches to stylistic extrapolation that minimize the role of domain-specific knowledge and processing, in particular, the connectionist model of Grebert et al. (1992), arguing that models that "don't know anything about what they are doing" (p. 408) cannot hope to capture the subtleties and richness of a human font designer's productions. We agree with Hofstadter's general diagnosis. There are many aspects of typographical competence, and we have tried to model only a subset of those. In particular, we have not tried to model the higher-level creative processes of expert font designers, who draw on elaborate knowledge bases, reflect on the results of their work, and engage in multiple revisions of each synthesized character. However, we do think that our approach captures two

essential aspects of the human competence in font extrapolation: people's representations of letter and font characteristics are modular and independent of each other, and people's knowledge of letters and fonts is abstracted from the ability to perform the particular task of character synthesis. Generic connectionist approaches to font extrapolation such as Grebert et al. (1992) do not satisfy these constraints. Letter and font information is mixed together inextricably in the extrapolation network's weights, and an entirely different network would be needed to perform recognition or classification tasks with the same stimuli. Our bilinear modeling approach, in contrast, captures the perceptual modularity of style and content in terms of the mathematical property of separability that characterizes equations 2.1 through 2.7. Knowledge of style s (in equation 2.6, for example) is localized to the matrix parameter $\mathbf{A}^s$, while knowledge of content class c is localized to the vector parameter $\mathbf{b}^c$, and both can be freely combined with other content or style parameters, respectively. Moreover, during training, our models acquire knowledge about the interaction of style and content factors that is truly abstracted from any particular behavior, and thus can support not only extrapolation of a novel style, but also a range of other synthesis and recognition tasks as shown in Figure 1.

6 Translation

Many important perceptual tasks require the perceiver to recover simultaneously two unknown pieces of information from a single stimulus in which these variables are confounded. A canonical example is the problem of separating the intrinsic shape and texture characteristics of a face from the extrinsic lighting conditions, which are confounded in any one image of that face. In this section, we show how a perceptual system may, using a bilinear model, learn to solve this problem from a training set of faces labeled according to identity and lighting condition. The bilinear model allows a novel face viewed under a novel illuminant to be translated to its appearance under known lighting conditions, and the known faces to be translated to the new lighting condition. Such translation tasks are the most difficult kind of two-factor learning task, because they require generalization across both factors at once. That is, what is common across both training and test data sets is not any particular style or any particular content class, but only the manner in which these two factors interact. Thus, only a symmetric bilinear model (see equations 2.1–2.3) is appropriate, because only it represents explicitly the interaction between style and content factors, in the $\mathbf{W}_k$ parameters.

6.1 Task Specifics. Given a training set of $S = 23$ faces (content) viewed under $C = 3$ different lighting conditions (style) and a novel face viewed under a novel light source, the task is to translate the new face to known lighting conditions and the known faces to the new lighting condition. The

face images, provided by Y. Moses of the Weizmann Institute, were cropped to remove nonfacial features and blurred and subsampled to 48×80 pixels. Because these faces were aligned and lacked sharp edge features (unlike the typographical characters of the previous section), we could represent the images directly as 3840-dimensional vectors of pixel brightness values.

6.2 Algorithm. We first fit the symmetric model (see equation 2.2) to the training data using the iterated SVD procedure described in section 3.2. This yields vector representations $\mathbf{a}^s$ and $\mathbf{b}^c$ of each face c and illuminant s, and a matrix of interaction parameters $\mathbf{W}$ (defined in equation 3.8). The dimensionalities for $\mathbf{a}^s$ and $\mathbf{b}^c$ were set equal to S and C, respectively, allowing the bilinear model maximum expressivity while still ensuring a unique solution. Because these dimensionalities were set to their maximal values, only one iteration of the fitting algorithm (taking approximately 10 seconds) was required for complete training.

For generalization from a single test image $\tilde{\mathbf{y}}$, we adapt the model simultaneously to both the new face identity $\tilde{c}$ and the new illuminant $\tilde{s}$, while holding fixed the face $\times$ illuminant interaction term $\mathbf{W}$ learned during training. Specifically, we first make an initial guess for the new face identity vector $\mathbf{b}^{\tilde{c}}$ (e.g., the mean of the training set style vectors) and solve for the least-squares optimal estimate of the illuminant vector $\mathbf{a}^{\tilde{s}}$:

$$\mathbf{a}^{\tilde{s}} = \left[\left[\mathbf{W}\mathbf{b}^{\tilde{c}}\right]^{\mathrm{VT}}\right]^{-1} \tilde{\mathbf{y}}. \tag{6.1}$$

Here $[\ldots]^{-1}$ denotes the pseudoinverse function. Given this new value for $\mathbf{a}^{\tilde{s}}$, we then reestimate $\mathbf{b}^{\tilde{c}}$ from

$$\mathbf{b}^{\tilde{c}} = \left[\left[\mathbf{W}^{\mathrm{VT}}\mathbf{a}^{\tilde{s}}\right]^{\mathrm{VT}}\right]^{-1} \tilde{\mathbf{y}}, \tag{6.2}$$

and iterate equations 6.1 and 6.2 until both $\mathbf{a}^{\tilde{s}}$ and $\mathbf{b}^{\tilde{c}}$ converge. In the example presented in Figure 11, convergence occurred after 14 iterations (approximately 16 seconds). We can then generate images of the new face under known illuminant s (from $y_k^{s\tilde{c}} = \mathbf{a}^{s\mathrm{T}}\mathbf{W}_k\mathbf{b}^{\tilde{c}}$) and of known face c under the new illuminant (from $y_k^{\tilde{s}c} = \mathbf{a}^{\tilde{s}\mathrm{T}}\mathbf{W}_k\mathbf{b}^{c}$).

6.3 Results. Figure 11 shows results, with both the old faces translated to the new illuminant and the new face translated to the old illuminants. For comparison, the true images are shown next to the synthetic ones. Again, evaluation of these results must necessarily be subjective. The lighting and shadows for each synthesized image appear approximately correct, as do the facial features of the old faces translated to the new illuminant. The facial features of the new face translated to the old illuminants appear slightly

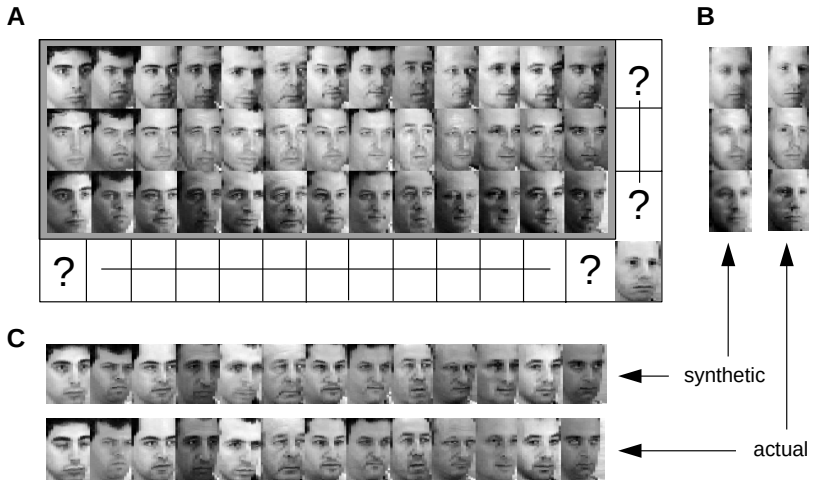

Figure 11: Translation across style and content in shape-from-shading. (A) Row 1–3: The first 13 (of 24) faces viewed under the three illuminants used for training. Row 4: The single test image of a new face viewed under a new light source. (B) Column 1: Translation of the new face to known illuminants. Column 2: The actual (unseen) images. (C) Row 1: Translation of known faces to the new illuminant. Row 2: The actual (unseen) images.

blurred, but otherwise resemble the new face more than any of the old faces. One reason the synthesized images of the new face are not as sharp as the synthesized images of the old faces is that the latter are produced by averaging images of a single face under several lighting conditions—across which all the facial features are precisely aligned—while the former are produced by averaging images of many faces under a single lighting condition—across which the facial features vary significantly in their positions.

6.4 Discussion. A history of applying linear models to face images motivates our bilinear modeling approach. The original work on eigenfaces (Kirby & Sirovich, 1990; Turk & Pentland, 1991) showed that images of many different faces taken under fixed lighting and viewpoint conditions occupy a low-dimensional linear subspace of pixel space. Subsequent work by Hallinan (1994) established the complementary result that images of a single face taken under many different lighting conditions also occupy a low-dimensional linear subspace. Thus, the factors of facial identity and illumination have already been shown to satisfy approximately the definition of bilinearity—the effects of one factor are linear when the other is held constant—so it is natural to integrate them into a bilinear model.

While the general problem of separating shape, texture, and illumination features in an image is underdetermined (Barrow & Tenenbaum, 1978), here the bilinear model learned during training provides sufficient constraint to approximately recover both face and illumination parameters from a single novel image. Atick, Griffin, & Redlich (1996) proposed a related approach to learning shape-from-shading for face images, based on a linear model of head shape in three dimensions and a nonlinear physical model of the image formation process. In contrast, our bilinear model uses only two-dimensional (i.e., image-based) information and requires no prior knowledge about the physics of image formation. Of course, we have not solved the general shape-from-shading recovery problem for arbitrary objects under arbitrary illumination. Our solution (as well as that of Atick et al., 1996) depends critically on the assumption that the new image, like the images in the training set, depicts an upright face under reasonable lighting conditions. In fact, there is evidence that the brain often does not solve the shape-from-shading problem in its most general form, but rather has learned (or evolved) solutions to important special cases such as face images (Cavanagh, 1991). So-called Mooney faces—brightness-thresholded face images in which shading is the only cue to shape—can be easily recognized as images of three-dimensional surfaces when viewed in upright position, but cannot be discriminated from two-dimensional ink blotches when viewed upside down so that the shading conventions are atypical (Shepard, 1990), or when the underlying three-dimensional structure has been distorted away from a globally facelike shape (Moore & Cavanagh, 1998). More generally, the ability to learn constrained solutions to a priori underconstrained inference problems may turn out to be essential for perception (Poggio & Hurlbert, 1994; Nayar & Poggio, 1996). Bilinear models offer one simple and general framework for how biological and artificial perceptual systems may learn to solve a wide range of such tasks.

7 Directions for Future Work

The most obvious extension of our work is to observations and tasks with more than two underlying factors, via multilinear models (Magnus & Neudecker, 1988). For example, a symmetric trilinear model in three factors q, r, and s would take the form:

$$y_l^{qrs} = \sum_{i,j,k} w_{ijkl} a_i^q b_j^r c_k^s. \tag{7.1}$$

The procedures for fitting these models are direct generalizations of the learning algorithms for two-factor models that we describe here. As in the two-factor case, we iteratively apply linear matrix techniques to solve for the parameters of each factor given parameter estimates for the other factors, until all parameter estimates converge (Magnus & Neudecker, 1988).

As with any other learning framework, the success of bilinear models depends on having suitable input representations. Further research is needed to determine what kinds of representations will endow specific kinds of observations with the most nearly bilinear structure. In particular, the font extrapolation task might benefit from a representation that is better tailored to the important features of letter shapes. Also, it would be of interest to develop general procedures for incorporating available domain-specific knowledge into bilinear models, for example, via a priori constraints on the model parameters (Simard, LeCun, & Denker, 1993).

Finally, we would like to explore the relevance of bilinear models for research in neuroscience and psychophysics. The essential representational feature of bilinear models is their multiplicative factor interactions. At the physiological level, multiplicative neuronal interactions (Andersen, Essick, & Siegel, 1985; Olshausen, Anderson, & van Essen, 1993; Riesenhuber & Dayan, 1997), arising from nonlinear synaptic (Koch, 1997) or population-level (Salinas & Abbott, 1993) mechanisms, have been proposed for visual computations that require the synergistic combination of two inputs, such as modulating spatial attention (Andersen et al., 1985; Olshausen et al., 1993; Riesenhuber & Dayan, 1997; Salinas & Abbott, 1993) or estimating motion (Koch, 1997). May these same kinds of circuits be co-opted to perform some of the tasks we study here? It has been suggested that SVD, the essential computational procedure for learning in our framework, can be implemented naturally in neural networks using Hebb-like learning rules (Sanger, 1989, 1994; Oja, 1989). Could these or similar mechanisms support a biologically plausible learning algorithm for bilinear models? At the level of psychophysics, many studies have demonstrated that the ability of the human visual and auditory systems to factor out contextually irrelevant dimensions of variation, such as lighting conditions or speaker accent, is neither perfect nor instantaneous. Observations in unusual styles are frequently more time-consuming to process and more likely to be processed incorrectly (van Bergem et al., 1988; Sanocki, 1992; Nygaard & Pisoni, 1998; Moore & Cavanagh, 1998). Do bilinear models have difficulty on the same kinds of stimuli that people do? Do the dynamics of adaptation in bilinear models (e.g., EM for classification, or the iterative SVD-based procedure for translation) take longer to converge on stimuli that people are slower to process? Can our approach provide a framework for designing and modeling perceptual learning experiments, in which subjects learn to separate the effects of style and content factors in a novel stimulus domain? These are just a few of the empirical questions motivated by a bilinear modeling approach to studying perceptual inference.

8 Conclusions

In one sense, bilinear models are not new to perceptual research. It has previously been shown that several core vision problems, such as the recovery

of structure from motion under orthographic projection (Tomasi & Kanade, 1992) and color constancy under multiple illuminants (Brainard & Wandell, 1991; Marimont & Wandell, 1992; D'Zmura, 1992) are solvable efficiently because they are fundamentally bilinear at the level of geometry or physics (Koenderink & van Doorn, 1997). These results are important but of limited usefulness, because many two-factor problems in perception do not have this true bilinear character. More commonly, perceptual inferences based on accurate physical models of factor interactions are either very complex (as in speech recognition), underdetermined (as in shape-from-shading), or simply inappropriate (as in typography).

Here we have proposed that perceptual systems may often solve such challenging two-factor tasks without detailed domain knowledge, using bilinear models to learn approximate solutions rather than to describe explicitly the intrinsic geometry or physics of the problem. We presented a suite of simple and efficient learning algorithms for bilinear models, based on the familiar techniques of SVD and EM. We then demonstrated the scope of this approach with applications to three two-factor tasks—classsification, extrapolation, and translation—using three kinds of signals—speech, typography, and face images. With their combination of broad applicability and ready learning algorithms, bilinear models may prove to be a generally useful component in the tool kits of engineers and brains alike.

Acknowledgments

We thank E. Adelson, B. Anderson, M. Bernstein, D. Brainard, P. Dayan, W. Richards, and Y. Weiss for helpful discussions. Joshua Tenenbaum was a Howard Hughes Medical Institute predoctoral fellow. Preliminary reports of this material appeared in Tenenbaum and Freeman (1997) and Freeman and Tenenbaum (1997).

References

Andersen, R., Essick, G., & Siegel, R. (1985). The encoding of spatial location by posterior parietal neurons. *Science, 230*, 456–458.

Atick, J. J., Griffin, P. A., & Redlich, A. N. (1996). Statistical approach to shape from shading: Reconstruction of 3d face surfaces from single 2d images. *Neural Computation, 8*, 1321–1340.

Barrow, H. G., & Tenenbaum, J. M. (1978). Recovering intrinsic scene characteristics from images. In A. R. Hanson & E. M. Riseman (Eds.), *Computer vision systems* (pp. 3–26). New York: Academic Press.

Bell, A., & Sejnowski, T. (1995). An information–maximization approach to blind separation and blind deconvolution. *Neural Computation, 7*(6), 1129–1159.

Beymer, D., & Poggio, T. (1996). Image representations for visual learning. *Science, 272*, 1905–1909.

Bishop, C. M. (1995). *Neural networks for pattern recognition*. New York: Oxford University Press.

Brainard, D., & Wandell, B. (1991). A bilinear model of the illuminant's effect on color appearance. In M. S. Landy and J. A. Movshon (Eds.), *Computational models of visual processing* (chap. 13). Cambridge, MA: MIT Press.

Caruana, R. (1998). Multitask learning. In S. Thrun and L. Pratt (Eds.), *Learning to learn* (pp. 95–134). Norwell, MA: Kluwer.

Cavanagh, P. (1991). What's up in top-down processing? In A. Gorea (Ed.), *Representations of vision: Trends and tacit assumptions in vision research* (pp. 295–304). Cambridge: Cambridge University Press.

Dayan, P., Hinton, G., Neal, R., & Zemel, R. (1995). The Helmholtz machine. *Neural Computation, 7(5)*, 889–904.

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of Royal Statist. Soc. B, 39*, 1–38.

Duda, R. O., & Hart, P. E. (1973). *Pattern classification and scene analysis*. New York: Wiley-Interscience.

D'Zmura, M. (1992). Color constancy: Surface color from changing illumination. *Journal of the Optical Society of America A, 9*, 490–493.

Freeman, W. T., & Tenenbaum, J. B. (1997). Learning bilinear models for two-factor problems in vision. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition* (pp. 554–560). San Juan, PR.

Ghahramani, Z. (1995). Factorial learning and the EM algorithm. In G. Tesauro, D. Touretzky, and T. Leen (Eds.), *Advances in neural information processing systems* (Vol. 7, pp. 617–624). Cambridge, MA: MIT Press.

Grebert, I., Stork, D. G., Keesing, R., & Mims, S. (1992). Connectionist generalization for production: An example from gridfont. *Neural Networks, 5*, 699–710.

Hallinan, P. W. (1994). A low-dimensional representation of human faces for arbitrary lighting conditions. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition* (pp. 995–999).

Hastie, T., & Tibshirani, R. (1996). Discriminant adaptive nearest neighbor classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 18*, 607–616.

Hinton, G., Dayan, P., Frey, B., & Neal, R. (1995). The wake-sleep algorithm for unsupervised neural networks. *Science, 268*, 1158–1161.

Hinton, G., & Ghahramani, Z. (1997). Generative models for discovering sparse distributed representations. *Phil. Trans. Royal Soc. B, 352*, 1177–1190.

Hinton, G. E., & Zemel, R. (1994). Autoencoders, minimum description length, and Helmholtz free energy. In J. Cowan, G. Tesauro, and J. Alspector (Eds.), *Advances in neural information processing systems, 6* (pp. 3–10). San Mateo, CA: Morgan Kauffman.

Hofstadter, D. (1985). *Metamagical themas*. New York: Basic Books.

Hofstadter, D. (1995). *Fluid concepts and creative analogies*. New York: Basic Books.

Holyoak, K., & Barnden, J. (Eds.). (1994). *Advances in connectionist and neural computation theory*. Norwood, NJ: Ablex.

Jackson, J. D. (Ed.). (1975). *Classical electrodynamics* (2nd ed.). New York: Wiley.

Kirby, M., & Sirovich, L. (1990). Application of the Karhunen-Loeve procedure for the characterization of human faces. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 12(1)*, 103–108.

Koch, C. (1997). Computation and the single neuron. *Nature, 385*, 207–211.

Koenderink, J. J., & van Doorn, A. J. (1997). The generic bilinear calibration–estimation problem. *International Journal of Computer Vision, 23*(3), 217–234.

Magnus, J. R., & Neudecker, H. (1988). *Matrix differential calculus with applications in statistics and econometrics*. New York: Wiley.

Mardia, K., Kent, J., & Bibby, J. (1979). *Multivariate analysis*. London: Academic Press.

Marimont, D. H., & Wandell, B. A. (1992). Linear models of surface and illuminant spectra. *Journal of the Optical Society of America A, 9*(11), 1905–1913.

Moghaddam, B., & Pentland, A. P. (1997). Probabilistic visual learning for object representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 19*(7), 696–710.

Moore, C., & Cavanagh, P. (1998). Recovery of 3d volume from 2-tone images of novel objects. *Cognition, 67* (1,2), 45–71.

Nayar, S., & Poggio, T. (Eds.). (1996). *Early visual learning*. New York: Oxford University Press.

Nygaard, L. C., & Pisoni, D. B. (1998). Talker-specific learning in speech perception. *Perception and Psychophysics, 60*(3), 355–376.

Oja, E. (1989). Neural networks, principal components, and subspaces. *International Journal of Neural Systems, 1*, 61–68.

Olshausen, B., Anderson, C., & van Essen, D. (1993). A neural model of visual attention and invariant pattern recognition. *Journal of Neuroscience, 13*, 4700–4719.

Omohundro, S. M. (1995). Family discovery. In D. Touretzky, M. Moser, & M. Hasselmo (Eds.), *Advances in neural information processing systems, 8* (pp. 402–408). Cambridge, MA: MIT Press.

Poggio, T., & Hurlbert, A. (1994). Observations on cortical mechanisms for object recognition and learning. In C. Koch & J. Davis (Eds.), *Large scale neuronal theories of the brain* (pp. 152–182). Cambridge, MA: MIT Press.

Polk, T. A., & Farah, M. J. (1997). A simple common contexts explanation for the development of abstract letter identities. *Neural Computation, 9*(6), 1277–1289.

Press, W. H., Teukolsky, S. A., Vetterling, W. T., & Flannery, B. P. (1992). *Numerical recipes in C*. New York: Cambridge University Press.

Riesenhuber, M., & Dayan, P. (1997). Neural models for part-whole hierarchies. In M. Mozer, M. Jordan, & T. Petsche (Eds.), *Advances in neural information processing systems, 9* (pp. 17–23). Cambridge, MA: MIT Press.

Robinson, A. (1989). *Dynamic error propagation networks*. Unpublished doctoral dissertation, Cambridge University.

Salinas, E., & Abbott, L. (1993). A model of multiplicative neural responses in parietal cortex. In *Proc. Nat. Acad. Sci. USA* (pp. 11956–11961).

Sanger, T. D. (1989). Optimal unsupervised learning in a single-layer linear feedforward neural network. *Neural Networks, 2*, 459–473.

Sanger, T. (1994). Two algorithms for iterative computation of the singular value decomposition from input/output samples. In J. Cowan, G. Tesauro, &

J. Alspector (Eds.), *Advances in neural information processing systems, 6* (pp. 144–151). San Mateo, CA: Morgan Kauffman.

Sanocki, T. (1992). Effects of font- and letter-specific experience on the perceptual processing of letters. *American Journal of Psychology, 105*(3), 435–458.

Shepard, R. N. (1990). *Mind sights*. New York: Freeman.

Simard, P. Y., LeCun, Y., & Denker, J. (1993). Efficient pattern recognition using a new transformation distance. In S. Hanson, J. Cowan, & L. Giles (Eds.), *Advances in neural information processing systems, 5*. San Mateo, CA: Morgan Kaufman.

Szeliski, R., & Tonnesen, D. (1992). Surface modeling with oriented particle systems. In *Proc. SIGGRAPH 92* (Vol. 26, pp. 185–194).

Tenenbaum, J. B., & Freeman, W. T. (1997). Separating style and content. In M. Mozer, M. Jordan, & T. Petsche (Eds.), *Advances in neural information processing systems, 9* (pp. 662–668). Cambridge, MA: MIT Press.

Teo, P., & Heeger, D. (1994). Perceptual image distortion. In *First IEEE International Conference on Image Processing* (Vol. 2, pp. 982–986). New York: IEEE.

Thrun, S., & Pratt, L. (Eds). (1998). *Learning to learn*. Norwell, MA: Kluwer.

Tomasi, C., & Kanade, T. (1992). Shape and motion from image streams under orthography: A factorization method. *International Journal of Computer Vision, 9*(2), 137–154.

Turk, M., & Pentland, A. (1991). Eigenfaces for recognition. *Journal of Cognitive Neuroscience, 3*(1), 71–86.

van Bergem, D. R., Pols, L. C., & van Beinum, F. J. K. (1988). Perceptual normalization of the vowels of a man and a child in various contexts. *Speech Communication, 7*(1), 1–20.

Received October 27, 1998; accepted June 8, 1999.