Semi-Supervised Learning of Multi-Factor Models for Face De-Identification

Ralph Gross, Latanya Sweeney
Data Privacy Lab, SCS
Carnegie Mellon University
{rgross,latanya}@cs.cmu.edu

Fernando de la Torre
Robotics Institute
Carnegie Mellon University
ftorre@cs.cmu.edu

Simon Baker
Microsoft Research
Microsoft Corporation
sbaker@microsoft.com

Abstract

With the emergence of new applications centered around the sharing of image data, questions concerning the protection of the privacy of people visible in the scene arise. Recently, formal methods for the de-identification of images have been proposed which would benefit from multi-factor coding to separate identity and non-identity related factors. However, existing multi-factor models require complete labels during training which are often not available in practice. In this paper we propose a new multi-factor framework which unifies linear, bilinear, and quadratic models. We describe a new fitting algorithm which jointly estimates all model parameters and show that it outperforms the standard alternating algorithm. We furthermore describe how to avoid overfitting the model and how to train the model in a semi-supervised manner. In experiments on a large expression-variant face database we show that data coded using our multi-factor model leads to improved data utility while providing the same privacy protection.

1. Introduction

Recent advances in both camera technology as well as supporting computing hardware have made it significantly easier to deal with large amounts of visual data. This enables a wide range of new usage scenarios involving the acquisition, processing and sharing of images. However, many of these applications are plagued by privacy problems concerning the people visible in the scene. Examples include the recently introduced Google Streetview service, surveillance systems to help monitor patients in nursing homes [3], and the collection and distribution of medical face databases (studying e.g. pain [1]).

In most of these applications knowledge of the identity of people in the image is not required. This makes the case for image de-identification, the removal of identifying information from images, prior to sharing of the data. Privacy protection methods are well established for field-structured data [19], however, work on images is still limited. The implicit goal of these methods is to protect privacy and preserve data utility, e.g. the ability to recognize gender or facial expressions from de-identified images. While initial methods discussed in the literature were limited to applying naïve image obfuscation methods such as blurring [15], more recent methods such as the k-Same algorithm provide provable privacy guarantees and preserve data utility [16].

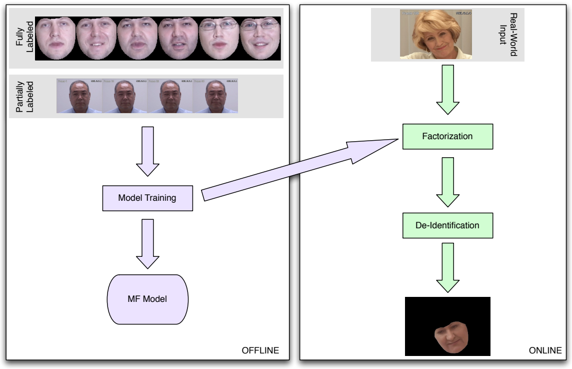

Figure 1. Illustration of the proposed multi-factor de-identification algorithm. We compute multi-factor models over fully and partially labeled training data. During runtime image data is factorized using the models prior to de-identification resulting in improved data utility.

Previously introduced methods operate directly on image data which varies both with identity as well as non-identity related factors such as facial expressions. A natural extension of these methods would use a factorization approach to separate identity and non-identity related factors to improve preservation of data utility. However, existing multi-factor models such as the bilinear models introduced by Tenenbaum and Freeman [20] or tensor models [22] require complete data labels during training which are often not available in practice. To address this problem we propose a new multi-factor framework which combines linear, bilinear, and quadratic models and enables semi-supervised learning consistent with the needs of the de-identification application. Figure 1 illustrates our approach.

We make the following contributions in this paper. We introduce a new multi-factor framework which unifies linear, bilinear, and quadratic models into one formulation (Section 3.1). We describe a fitting algorithm for the joint estimation of the model parameters which improves significantly upon the standard alternating fitting algorithm (Section 3.3). We then show how to include regularization terms into the fitting framework to prevent the model from overfitting the data (Section 4.4). On synthetic data, inclusion of the regularization term improves classification performance from 60% to 83.3% for mixed linear, bilinear, and quadratic models. We furthermore show that our framework allows for semi-supervised training of data (Section 4.5). In de-identification experiments we demonstrate improved data utility rates for a data representation using coefficients of a mixed linear and bilinear model learned in a semi-supervised manner (94.7% classification accuracy vs. 75% for a more conventional model; Section 5.3).

This paper is organized as follows. We provide background material on image de-identification in Section 2. Section 3 defines our model and introduces the joint fitting algorithm. In Section 4 we show how to enforce additional constraints within the framework and how to train the model in a semi-supervised manner. We present experimental result from applying the framework for a de-identification task in Section 5.

2. Face De-Identification

The goal of face de-identification is to remove identifying information from images, ideally while preserving most aspects of the data to enable usage after processing. Currently available image de-identification algorithms fall into one of two groups: ad-hoc distortion methods and the k-Same [16] family of algorithms implementing the k-anonymity protection model [19]. In this section we describe both approaches.

2.1. Ad-hoc De-Identification Algorithms

Across a number of different communities the problem of protecting privacy of people visible in images has been addressed. The majority of approaches employ simple obfuscation methods such as blurring (smoothing the image with e.g. a Gaussian filter with large variance) or pixelation (image subsampling) [7, 15]. While these algorithms are applicable to all images, they lack a formal privacy model. As a consequence no guarantees can be made that the privacy of people visible in the images is actually protected. Privacy protection is evaluated, if at all, only in human subject studies. It has been shown that these naïve algorithms are easy to defeat [16] and typically neither preserve privacy nor the utility of the data [11]. Phillips [17] proposed an algorithm for privacy protection of facial images through reduction of the number of eigenvectors used in reconstructing images from basis vectors. A direct trade-off between privacy protection and data utility is established through the introduction of the privacy operating characteristic (POC), a plot similiar to a receiver operating characteristic (ROC) often used in pattern classifier design.

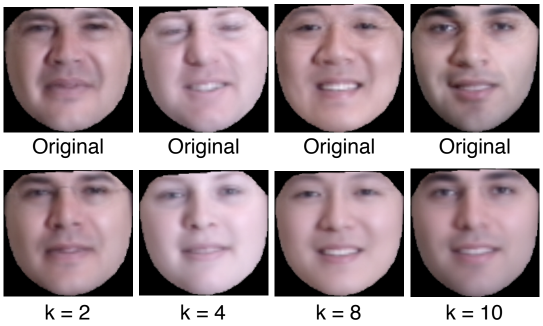

Figure 2. Examples of de-identified face images for different values of the privacy parameter k.

2.2. The k-Same Framework

The k-Same family of algorithms [11, 13, 16] implement the k-anonymity protection model [19] for face images. Given a *person-specific*[1] set of images $H = \{\mathbf{I}_1, \ldots, \mathbf{I}_M\}$, k-Same computes a de-identified set of images $H^d = \{\mathbf{I}^d_1, \ldots, \mathbf{I}^d_M\}$ in which each $\mathbf{I}^d_i$ indiscriminately relates to at least k elements of H. It can be shown that the best possible success rate for a face recognition algorithm linking an element of H^d to the correct face in H (independent of the algorithm used) is $\frac{1}{k}$. See [16] for details. k-Same achieves k-anonymity protection by averaging the k closest faces for each element of H and adding k copies of the resulting average to H^d. The algorithm selects images for averaging based on raw Euclidean distances in image space or Principal Component Analysis coefficient space [16]. In order to use additional information such as gender or facial expression during image selection, k-Same-Select was introduced in [11]. The resulting algorithm provides k-anonymity privacy protection while preserving data utility as evidenced by both gender and facial expression recognition experiments. See Figure 2 for example images. All algorithms of the k-Same family work directly on images, either raw or PCA-encoded. These methods would benefit from a recoding that separates out identity and non-identity related factors. We propose a framework in the next section to achieve this goal.

3. Reconstructive Models

In this section we introduce our unified framework for linear, bilinear, and quadratic models. We define the model

[1] In a person-specific set of faces each subject is represented by no more than one image.

in Section 3.1 and discuss ambiguities inherent in the model in Section 3.2. We describe two fitting algorithms, the alternating and joint fitting algorithm, in Section 3.3. We empirically compare the performance of the two algorithms in Section 3.4.

3.1. Model Definition

We define the general model M for data dimension k as

$$M_k(\boldsymbol{\mu}, \mathbf{B}^1, \mathbf{B}^2, \mathbf{W}, \mathbf{Q}^1, \mathbf{Q}^2; \mathbf{c}_1, \mathbf{c}_2) = \begin{pmatrix} 1 & \mathbf{c}_1^T & \mathbf{c}_2^T \end{pmatrix} \underbrace{\begin{pmatrix} \mu_k & \mathbf{B}_k^2 & 0 \\ \mathbf{B}_k^1 & \mathbf{W}_k & \mathbf{Q}_k^1 \\ 0 & \mathbf{Q}_k^2 & 0 \end{pmatrix}}_{\boldsymbol{\Omega}_k} \begin{pmatrix} 1 \\ \mathbf{c}_2 \\ \mathbf{c}_1 \end{pmatrix} \tag{1}$$

with mean $\boldsymbol{\mu}$, linear bases $\mathbf{B}^1, \mathbf{B}^2$, bilinear basis $\mathbf{W}$, quadradic bases $\mathbf{Q}^1, \mathbf{Q}^2$, and coefficients $\mathbf{c}_1$ and $\mathbf{c}_2$. $\mathbf{c}^1 \in \mathbb{R}^{r_1}, \mathbf{c}^2 \in \mathbb{R}^{r_2}, \boldsymbol{\mu} \in \mathbb{R}^d, \mathbf{B}^1 \in \mathbb{R}^{d \times r_1}$ with $\mathbf{B}_k^1 \in \mathbb{R}^{1 \times r_1}$, $\mathbf{B}^2 \in \mathbb{R}^{d \times r_2}, \mathbf{W}_k \in \mathbb{R}^{r_1 \times r_2}, \mathbf{Q}_k^1 \in \mathbb{R}^{r_1 \times r_1}, \mathbf{Q}_k^2 \in \mathbb{R}^{r_2 \times r_2}$. To avoid redundancy, $\mathbf{Q}^1, \mathbf{Q}^2$ could be either symmetric or upper triangular. Here we choose upper triangular.

While Eqn. (1) defines a quadratic model, it in fact contains lower-dimensional linear, bilinear and quadratic models as special cases. To illustrate this we set $\mathbf{W} = \mathbf{Q}^1 = \mathbf{Q}^2 = 0$ and obtain

$$\begin{aligned} & M_k^{Lin}(\boldsymbol{\mu}, \mathbf{B}^1, \mathbf{B}^2, 0, 0, 0; \mathbf{c}_1, \mathbf{c}_2) = \\ = \; & \begin{pmatrix} 1 & \mathbf{c}_1^T & \mathbf{c}_2^T \end{pmatrix} \begin{pmatrix} \mu_k & \mathbf{B}_k^2 & 0 \\ \mathbf{B}_k^1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ \mathbf{c}_2 \\ \mathbf{c}_1 \end{pmatrix} \\ = \; & \mu_k + \mathbf{c}_1^T \mathbf{B}_k^1 + \mathbf{B}_k^2 \mathbf{c}_2 \end{aligned}$$

the linear model in $\mathbf{c}^1$ and $\mathbf{c}^2$. Similarly, for $\mathbf{Q}^1 = \mathbf{Q}^2 = 0$ we obtain the bilinear model

$$\begin{aligned} & M_k^{Bilin}(\boldsymbol{\mu}, \mathbf{B}^1, \mathbf{B}^2, \mathbf{W}, 0, 0; \mathbf{c}_1, \mathbf{c}_2) = \\ = \; & \begin{pmatrix} 1 & \mathbf{c}_1^T & \mathbf{c}_2^T \end{pmatrix} \begin{pmatrix} \mu_k & \mathbf{B}_k^2 & 0 \\ \mathbf{B}_k^1 & \mathbf{W}_k & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ \mathbf{c}_2 \\ \mathbf{c}_1 \end{pmatrix} \\ = \; & \mu_k + \mathbf{c}_1^T \mathbf{B}_k^1 + \mathbf{B}_k^2 \mathbf{c}_2 + \mathbf{c}_1^T \mathbf{W}_k \mathbf{c}_2 \end{aligned}$$

Mixtures of the components yield model combinations, i.e. mixed linear and bilinear, mixed bilinear and quadratic, etc.

3.2. Model Ambiguities

The model as defined in Eqn. (1) is ambiguous, i.e. there exists transformations of $\mathbf{c}^1, \mathbf{c}^2$, and $\boldsymbol{\Omega}_k$ that produce identical data vectors. In the linear case the ambiguity is well known:

$$\boldsymbol{\mu}^T + \mathbf{c}_1^T \mathbf{B}^{1^T} = \boldsymbol{\mu}^T + \mathbf{c}_1^T \mathbf{R} \mathbf{R}^{-1} \mathbf{B}^{1^T} \tag{2}$$

for any invertible $\mathbf{R}$. So for $\bar{\mathbf{B}}^{1^T} = \mathbf{R}^{-1} \mathbf{B}^{1^T}, \bar{\mathbf{c}}_1^T = \mathbf{c}_1^T \mathbf{R}$ it holds that

$$\boldsymbol{\mu} + \bar{\mathbf{c}}_1^T \bar{\mathbf{B}}^{1^T} = \boldsymbol{\mu} + \mathbf{c}_1^T \mathbf{B}^{1^T} \tag{3}$$

This ambiguity is broken in the case of PCA due to the ordering of the basis vectors according to the corresponding eigenvalues. In the case of the general model defined in Eqn. (1) arbitrary linear reparametarizations are possible:

$$\begin{aligned} & M_k(\boldsymbol{\Omega}_k; \mathbf{c}_1, \mathbf{c}_2) = \\ = \; & \begin{pmatrix} 1 & \mathbf{c}_1^T & \mathbf{c}_2^T \end{pmatrix} \begin{pmatrix} \mu_k & \mathbf{B}_k^2 & \mathbf{0} \\ \mathbf{B}_k^{1^T} & \mathbf{W}_k & \mathbf{Q}_k^1 \\ \mathbf{0} & \mathbf{Q}_k^2 & \mathbf{0} \end{pmatrix} \begin{pmatrix} 1 \\ \mathbf{c}_2 \\ \mathbf{c}_1 \end{pmatrix} \\ = \; & \begin{pmatrix} 1 & \mathbf{c}_1^T & \mathbf{c}_2^T \end{pmatrix} \boldsymbol{\Phi}_l \boldsymbol{\Phi}_l^{-1} \boldsymbol{\Omega}_k \boldsymbol{\Phi}_r \boldsymbol{\Phi}_r^{-1} \begin{pmatrix} 1 \\ \mathbf{c}_2 \\ \mathbf{c}_1 \end{pmatrix} \end{aligned} \tag{4}$$

with

$$\boldsymbol{\Phi}_l = \begin{pmatrix} 1 & \mathbf{o}_1 & \mathbf{o}_2 \\ 0 & \mathbf{R}_{11} & \mathbf{R}_{12} \\ 0 & \mathbf{R}_{21} & \mathbf{R}_{22} \end{pmatrix}, \boldsymbol{\Phi}_r = \begin{pmatrix} 1 & \mathbf{o}_2 & \mathbf{o}_1 \\ 0 & \mathbf{R}_{22} & \mathbf{R}_{21} \\ 0 & \mathbf{R}_{12} & \mathbf{R}_{11} \end{pmatrix}$$

and $\mathbf{o}_1 \in \mathbb{R}^{r_1}, \mathbf{o}_2 \in \mathbb{R}^{r_2}$, $\mathbf{R}_{11} \in \mathbb{R}^{r_1 \times r_1}$, $\mathbf{R}_{12}, \mathbf{R}_{21} \in \mathbb{R}^{r_1 \times r_2}$, and $\mathbf{R}_{22} \in \mathbb{R}^{r_2 \times r_2}$. The first column of both matrices $\boldsymbol{\Phi}_l$ and $\boldsymbol{\Phi}_r$ must be $\begin{pmatrix} 1 & \mathbf{0} & \mathbf{0} \end{pmatrix}^T$ due to the structure of the coefficient vectors $\begin{pmatrix} 1 & \mathbf{c}_1^T & \mathbf{c}_2^T \end{pmatrix}$ and $\begin{pmatrix} 1 & \mathbf{c}_2 & \mathbf{c}_1 \end{pmatrix}^T$, each with a leading 1. As a consequence of these ambiguities the model parameters obtained during fitting are not unique. Therefore special care must be taken to normalize parameters during the synthetic experiments described in Section 3.4.

3.3. Model Fitting

The goal of fitting is to compute the parameters that minimize the model reconstruction error for a given training data set $\mathbf{D} = [\mathbf{d}_1 \dots \mathbf{d}_n]$:

$$\underset{\boldsymbol{\Gamma}, \mathbf{C}_1, \mathbf{C}_2}{\arg\min} \sum_{l=1}^{n} \| M(\boldsymbol{\Gamma}; \mathbf{c}_1(l), \mathbf{c}_2(l)) - \mathbf{d}_l \|_2^2 \tag{5}$$

with the bases $\boldsymbol{\Gamma} = (\boldsymbol{\mu}, \mathbf{B}^1, \mathbf{B}^2, \mathbf{W}, \mathbf{Q}^1, \mathbf{Q}^2)$ and coefficients $\mathbf{C}_1 = (\mathbf{c}_1(1), \dots, \mathbf{c}_1(n)), \mathbf{C}_2 = (\mathbf{c}_2(1), \dots, \mathbf{c}_2(n))$.

For the linear model M^{Lin} the corresponding minimization problem is

$$\underset{\mathbf{B}, \mathbf{C}}{\arg\min} \sum_{l=1}^{n} \| M^{Lin}(\mathbf{B}; \mathbf{c}(l)) - \mathbf{d}_l \|_2^2 \tag{6}$$

where we combined the separate bases $\mathbf{B}^1, \mathbf{B}^2$ into $\mathbf{B}$ and the coefficients $\mathbf{C}_1, \mathbf{C}_2$ into $\mathbf{C} = (\mathbf{c}(1), \dots, \mathbf{c}(n))$. Eqn. (6) can be minimized efficiently by using PCA (see

```
The Alternating Fitting Algorithm
Initialization
  Randomly initialize $\boldsymbol{\mu}, \mathbf{B}^1, \mathbf{B}^2, \mathbf{W}, \mathbf{Q}^1, \mathbf{Q}^2; \mathbf{C}_1, \mathbf{C}_2$
Iterate
  (I1) Compute $\Delta\mathbf{\Gamma}$ in
       $\arg\min_{\Delta\mathbf{\Gamma}} \|M(\mathbf{\Gamma} + \Delta\mathbf{\Gamma}; \mathbf{C}_1, \mathbf{C}_2) - \mathbf{D}\|_2^2$
       Update $\mathbf{\Gamma} \leftarrow \mathbf{\Gamma} + \Delta\mathbf{\Gamma}$
  (I2) Compute $\Delta\mathbf{C}_1$ in
       $\arg\min_{\Delta\mathbf{C}_1} \|M(\mathbf{\Gamma}; \mathbf{C}_1 + \Delta\mathbf{C}_1, \mathbf{C}_2) - \mathbf{D}\|_2^2$
       Update $\mathbf{C}_1 \leftarrow \mathbf{C}_1 + \Delta\mathbf{C}_1$
  (I3) Compute $\Delta\mathbf{\Gamma}\mathbf{C}_2$ in
       $\arg\min_{\Delta\mathbf{C}_2} \|M(\mathbf{\Gamma}; \mathbf{C}_1, \mathbf{C}_2 + \Delta\mathbf{C}_2) - \mathbf{D}\|_2^2$
       Update $\mathbf{C}_2 \leftarrow \mathbf{C}_2 + \Delta\mathbf{C}_2$
```

Figure 3. The alternating fitting algorithm.

```
The Joint Fitting Algorithm
Initialization
  Randomly initialize $\boldsymbol{\mu}, \mathbf{B}^1, \mathbf{B}^2, \mathbf{W}, \mathbf{Q}^1, \mathbf{Q}^2; \mathbf{C}_1, \mathbf{C}_2$
Iterate
  (I1) Compute $\Delta = (\Delta\mathbf{\Gamma}, \Delta\mathbf{C}_1, \Delta\mathbf{C}_2)$ in
       $\arg\min_{\Delta} \|M(\mathbf{\Gamma} + \Delta\mathbf{\Gamma}; \mathbf{C}_1 + \Delta\mathbf{C}_1, \mathbf{C}_2 + \Delta\mathbf{C}_2) - \mathbf{D}\|_2^2$
       Update $\mathbf{\Gamma} \leftarrow \mathbf{\Gamma} + \Delta\mathbf{\Gamma}$
       Update $\mathbf{C}_1 \leftarrow \mathbf{C}_1 + \Delta\mathbf{C}_1$
       Update $\mathbf{C}_2 \leftarrow \mathbf{C}_2 + \Delta\mathbf{C}_2$
```

Figure 4. The joint fitting algorithm.

e.g. [5]). This, however, is not the only way. Assuming initial parameter estimates $\mathbf{B}_0$ and $\mathbf{C}_0$ we can minimize the expression in Eqn. (6) by alternating between computing updates $\Delta\mathbf{B}$ that minimize $\|M^{Lin}(\mathbf{B}_0+\Delta\mathbf{B}; \mathbf{C})-\mathbf{D}\|_2^2$ and updates $\Delta\mathbf{C}$ that minimize $\|M^{Lin}(\mathbf{B}_0; \mathbf{C}_o + \Delta\mathbf{C}) - \mathbf{D}\|_2^2$ [21]. Both equations are linear in their unknowns and can therefore be solved directly. In the case of linear models this alternated least squares algorithm has been shown to always converge to the global minimium [2].

PCA does not generalize to bilinear or quadratic models, however the alternating algorithm does. (Note that for bilinear models and fully labeled data, the iterative Tenenbaum-Freeman algorithm can be used [20]). We can minimize Eqn. (5) by solving separately in turn for updates $\Delta\mathbf{\Gamma}$, $\Delta\mathbf{C}_1$, and $\Delta\mathbf{C}_2$. See Figure 3. In each case the corresponding minimization problem is linear in its unknowns and can therefore be solved directly. In order to e.g. compute the basis update $\Delta\mathbf{\Gamma}$ we compute $\arg\min_{\Delta\mathbf{\Gamma}} \|\mathbf{E} - \mathbf{T}_{\Delta\mathbf{\Gamma}}\Delta\mathbf{\Gamma}\|_2^2$, with the current reconstruction error $\mathbf{E} = \mathbf{D} - M(\mathbf{\Gamma}; \mathbf{C}_1, \mathbf{C}_2)$ and the constraint matrix $\mathbf{T}_{\mathbf{\Gamma}}$. $\Delta\mathbf{\Gamma}$ can be computed in closed form as $\Delta\mathbf{\Gamma} = (\mathbf{T}_{\mathbf{\Gamma}}^T\mathbf{T}_{\mathbf{\Gamma}})^{-1}\mathbf{T}_{\mathbf{\Gamma}}^T\mathbf{E}$. $\Delta\mathbf{C}_1$ and $\Delta\mathbf{C}_2$ are computed in a similar manner.

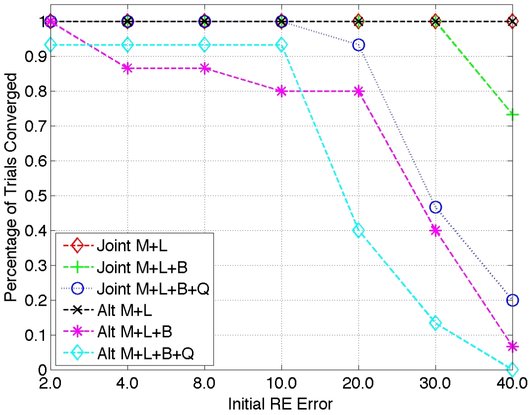

Figure 5. Comparison of the covergence frequency for the alternating and joint fitting algorithm on synthetic data. The fitting algorithms are initialized with ground-truth data perturbed by noise of varying magnitude. Results are shown for different model configurations combining the mean (M), linear (L), bilinear (B), and quadratic (Q) components. The joint fitting algorithm is more robust as shown by higher frequencies of convergence across models and initial perturbations.

While the alternating algorithm works well for linear models it has issues for higher-order models. The linearization into separate component updates ignores the coupling between the bases $\mathbf{\Gamma}$ and coefficients $\mathbf{C}_1, \mathbf{C}_2$. As a consequence the algorithm is more prone to local minima (see results in Section 3.4). To improve performance we propose to *jointly* solve for updates to all parameters at the same time. By dropping second order terms and reorganizing components we can transform the minimization problem $\arg\min_{\Delta\mathbf{\Gamma},\Delta\mathbf{C}_1,\Delta\mathbf{C}_2} \|M(\mathbf{\Gamma} + \Delta\mathbf{\Gamma}; \mathbf{C}_1 + \Delta\mathbf{C}_1, \mathbf{C}_2 + \Delta\mathbf{C}_2) - \mathbf{D}\|_2^2$ into a similar form as above:

$$\arg\min_{\Delta\mathbf{\Gamma},\Delta\mathbf{C}_1,\Delta\mathbf{C}_2} \|E - \mathbf{T}_{\mathbf{\Gamma},\mathbf{C}_1,\mathbf{C}_2} \begin{pmatrix} \Delta\mathbf{\Gamma} \\ \Delta\mathbf{C}_1 \\ \Delta\mathbf{C}_2 \end{pmatrix} \|_2^2 \qquad (7)$$

with $\mathbf{E} = \mathbf{D} - M(\mathbf{\Gamma}; \mathbf{C}_1, \mathbf{C}_2)$ and the constraint matrix $\mathbf{T}_{\mathbf{\Gamma},\mathbf{C}_1,\mathbf{C}_2}$. Figure 4 summarized the algorithm. See [10] for details.

3.4. Experiments

In order to compare the performance of the alternating and joint fitting algorithms we use synthetic data with known ground-truth. We randomly generate bases and coefficient matrices (drawn from a zero mean, unit variance normal distribution) and perturb both with varying amounts of noise before initializing the fitting algorithm. For each noise level the bases and coefficients are then normalized to ensure that all models are initialized at the same reconstruction error. We evaluate the fitting algorithms by comparing the ground-truth models with the fitted models.

In all experiments we report results averaged over five different ground-truth settings with three different initialization settings each for a total of 15 experiments for every model and fitting algorithm. We run every algorithm until convergence (normalized ground-truth error falls below a threshold) or a maximum of 150 iterations, whichever comes first. Figure 5 compares the frequency of convergence for different variations of the joint and alternating fitting algorithms for different initial reconstruction errors. Across all conditions, the joint fitting algorithm performs better than the alternating algorithm. For the combined linear, bilinear and quadratic model (M+L+B+Q) the joint algorithm converges in 80% of all cases whereas the alternating algorithm only converges in 61% of trials. The difference is even larger for the combined linear and bilinear model (M+L+B) where the joint algorithm converges in 96.2% of all trials compared to 68.6% for the alternating algorithm. The joint fitting algorithm also converges faster, requiring on average 8.7 iterations in comparison to 86.7 iterations for the alternating algorithm (for an initial ground-truth error of 20.0).

4. Multi-Factor Models with Constraints

The joint fitting algorithm described in Section 3.3 computes bases and coefficients iteratively by minimizing the model reconstruction error for a given training dataset. See Eqn. (5). While the resulting model succeeds at reconstructing the data, no other properties (e.g. affinity of class coefficients, basis orthonormality) are enforced. In order to accomplish this we add further constraints to the energy function on the coefficients, the bases or both. We then strive to compute

$$\begin{aligned} \arg\min_{\mathbf{\Gamma},\mathbf{C}_1,\mathbf{C}_2} & \sum_{l=1}^{n} \|M(\mathbf{\Gamma};\mathbf{c}_1(l),\mathbf{c}_2(l)) - \mathbf{d}_l\|_2^2 + \\ + & \lambda_1\Theta_1(\mathbf{C}_1,\mathbf{C}_2) + \lambda_2\Theta_2(\mathbf{\Gamma}) \end{aligned} \tag{8}$$

where Θ_1 and Θ_2 refer to sets of constraints. The parameters λ_1 and λ_2 balance the magnitude of the terms. In this section we describe how to enforce coefficient equality constraints (Section 4.1) and basis normality constraints (Section 4.2).

4.1. Coefficient Constraints

Let $S^1 = \{s_1^1,\ldots,s_{m_1}^1\}$, $S^2 = \{s_1^2,\ldots,s_{m_2}^2\}$ be sets of coefficient indices of elements in $\mathbf{C}_1$ and $\mathbf{C}_2$, respectively, for which we want to enforce equality. We then strive to compute

$$\begin{aligned} \arg\min_{\mathbf{\Gamma},\mathbf{C}_1,\mathbf{C}_2} & \sum_{l=1}^{n} \|M(\mathbf{\Gamma};\mathbf{c}_1(l),\mathbf{c}_2(l)) - \mathbf{d}_l\|_2^2 + \\ + & \lambda_{11} \sum_{\substack{s_i^1,s_j^1\in S^1 \\ s_i^1\neq s_j^1}} \|\mathbf{c}_1(s_i^1) - \mathbf{c}_1(s_j^1)\|_2^2 + \\ + & \lambda_{12} \sum_{\substack{s_i^2,s_j^2\in S^2 \\ s_i^2\neq s_j^2}} \|\mathbf{c}_2(s_i^2) - \mathbf{c}_2(s_j^2)\|_2^2 \end{aligned} \tag{9}$$

Linearalizing the expression in Eqn. (9) as described in Section 3.3 leads to

$$\begin{aligned} \arg\min_{\Delta\mathbf{\Gamma},\Delta\mathbf{C}_1,\Delta\mathbf{C}_2} & \|\mathbf{E}_{RE} - \mathbf{T}_{\mathbf{\Gamma},\mathbf{C}_1,\mathbf{C}_2}\begin{pmatrix}\Delta\mathbf{\Gamma}\\ \Delta\mathbf{C}_1\\ \Delta\mathbf{C}_2\end{pmatrix}\|_2^2 + \\ + & \lambda_{11}\|\mathbf{E}_{C_1} - \mathbf{T}_{S_1}\Delta\mathbf{C}_1\|_2^2 \\ + & \lambda_{12}\|\mathbf{E}_{C_2} - \mathbf{T}_{S_2}\Delta\mathbf{C}_2\|_2^2 \end{aligned} \tag{10}$$

with the reconstruction error $\mathbf{E}_{RE} = \mathbf{D} - M(\mathbf{\Gamma};\mathbf{C}_1,\mathbf{C}_2)$, the coefficient constraint error for $\mathbf{C}_1$ ($\mathbf{C}_2$ is defined analogously)

$$\mathbf{E}_{C_1} = \begin{pmatrix} \mathbf{c}_1(s_{i_1}^1) - \mathbf{c}_1(s_{i_2}^1) \\ \ldots \\ \mathbf{c}_1(s_{i_{m-1}}^1) - \mathbf{c}_1(s_{i_m}^1) \end{pmatrix} \tag{11}$$

and the coefficient constraint matrices $\mathbf{T}_{S_1}, \mathbf{T}_{S_2}$. The problem defined in Eqn. (10) can be solved as constraint least squares problem with linear equality constraints (see e.g. [6]). To do so we stack the components of Eqn. (10) and compute

$$\begin{aligned} \arg\min_{\Delta\mathbf{\Gamma},\Delta\mathbf{C}_1,\Delta\mathbf{C}_2} \| & \begin{pmatrix} \mathbf{E}_{RE} \\ \lambda_{11} * \mathbf{E}_{C_1} \\ \lambda_{12} * \mathbf{E}_{C_2} \end{pmatrix} - \\ - & \begin{pmatrix} & \mathbf{T}_{\mathbf{\Gamma},\mathbf{C}_1,\mathbf{C}_2} & \\ 0 & \lambda_{11} * \mathbf{T}_{S_1} & 0 \\ 0 & 0 & \lambda_{12} * \mathbf{T}_{S_2} \end{pmatrix} \begin{pmatrix}\Delta\mathbf{\Gamma}\\ \Delta\mathbf{C}_1\\ \Delta\mathbf{C}_2\end{pmatrix} \|_2^2 \end{aligned} \tag{12}$$

The solution to Eqn. (12) can be computed in the same way as the solution to the unconstrained least squares problem. Since the coefficient constraints are added individually and independently for the factors $\mathbf{c}_1$ and $\mathbf{c}_2$, the framework enables semi-supervised learning (see Section 2).

4.2. Basis Constraints

While PCA produces orthonormal basis vectors for linear models, direct minimization of Eqn. (7) typically results in non-orthogonal basis vectors of length greater than one. In order to enforce orthonormality on the linear components

$\mathbf{B}^1$ and $\mathbf{B}^2$ we set $\Theta_2 = \|\mathbf{B}^{1^T}\mathbf{B}^1 - \mathbf{I}\|_2^2 + \|\mathbf{B}^{2^T}\mathbf{B}^2 - \mathbf{I}\|_2^2$ and compute

$$\begin{aligned} & \underset{\mathbf{\Gamma},\mathbf{C}_1,\mathbf{C}_2}{\arg\min} \|M(\mathbf{\Gamma};\mathbf{C}_1,\mathbf{C}_2) - \mathbf{D}\|_2^2 + \\ + \quad & \lambda_2(\|\mathbf{B}^{1^T}\mathbf{B}^1 - \mathbf{I}\|_2^2 + \|\mathbf{B}^{2^T}\mathbf{B}^2 - \mathbf{I}\|_2^2) \quad (13) \end{aligned}$$

Linearalizing the expression leads to the same constrained least squares problem as above in Eqn. (12)

$$\begin{aligned} & \underset{\Delta\mathbf{\Gamma},\Delta\mathbf{C}_1,\Delta\mathbf{C}_2}{\arg\min} \| \begin{pmatrix} \mathbf{E}_{RE} \\ \lambda_2 * \mathbf{E}_B \end{pmatrix} - \quad (14) \\ - & \begin{pmatrix} & \mathbf{T}_{\mathbf{\Gamma},\mathbf{C}_1,\mathbf{C}_2} & \\ \lambda_2 * \mathbf{T}_B & 0 & 0 \end{pmatrix} \begin{pmatrix} \Delta\mathbf{\Gamma} \\ \Delta\mathbf{C}_1 \\ \Delta\mathbf{C}_2 \end{pmatrix} \|_2^2 \end{aligned}$$

with the basis constraint error $\mathbf{E}_B$ and the basis constraint matrix $\mathbf{T}_B$.

4.3. Coefficient and Basis Constraints

The constraints for coefficients and basis vectors as described in Sections 4.1 and 4.2 can be directly combined resulting in the minimization problem

$$\begin{aligned} & \underset{\Delta\mathbf{\Gamma},\Delta\mathbf{C}_1,\Delta\mathbf{C}_2}{\arg\min} \| \begin{pmatrix} \mathbf{E}_{RE} \\ \lambda_{11} * \mathbf{E}_{C_1} \\ \lambda_{12} * \mathbf{E}_{C_2} \\ \lambda_2 * \mathbf{E}_B \end{pmatrix} - \\ - & \begin{pmatrix} & \mathbf{T}_{\mathbf{\Gamma},\mathbf{C}_1,\mathbf{C}_2} & \\ 0 & \lambda_{11} * \mathbf{T}_{S_1} & 0 \\ 0 & 0 & \lambda_{12} * \mathbf{T}_{S_2} \\ \lambda_2 * \mathbf{T}_B & 0 & 0 \end{pmatrix} \begin{pmatrix} \Delta\mathbf{\Gamma} \\ \Delta\mathbf{C}_1 \\ \Delta\mathbf{C}_2 \end{pmatrix} \|_2^2 \end{aligned}$$

4.4. Dealing with Overfitting

Given enough degrees of freedom, every model is prone to overfitting the training data and subsequently generalize poorly to unseen data [9]. To address overfitting in our framework we include a penality term on the magnitude of the bilinear and quadratic bases into the energy function:

$$\begin{aligned} & \underset{\mathbf{\Gamma},\mathbf{C}_1,\mathbf{C}_2}{\arg\min} \sum_{l=1}^{n} \|M(\mathbf{\Gamma};\mathbf{c}_1(l),\mathbf{c}_2(l)) - \mathbf{d}_l\|_2^2 + \\ + \quad & \lambda_1\Theta_1(\mathbf{C}_1,\mathbf{C}_2) + \lambda_2\Theta_2(\mathbf{\Gamma}) + \\ + \quad & \lambda_3(\|\mathbf{W}^T\mathbf{W}\|_2^2 + \|\mathbf{Q}^{1^T}\mathbf{Q}^1\|_2^2 + \|\mathbf{Q}^{2^T}\mathbf{Q}\|_2^2)(15) \end{aligned}$$

In order to show the benefit of including regularization terms into Eqn. (15) we conduct synthetic classification experiments. We first randomly generate synthetic linear (ML), mixed linear and bilinear (MLB), and mixed linear, bilinear, and quadratic models (MLBQ). We then randomly generate class means and class members distributed around the class means for both the $\mathbf{c}_1$ and $\mathbf{c}_2$ coefficients, producing both "training" as well as "testing" coefficients. Instantiating the different models with the training coefficients we obtain linear, mixed linear and bilinear, and mixed linear, bilinear and quadratic data. We compare different fitting algorithms by initializing the algorithms with a perturbed version of the training data and ground-truth model, fitting a model to the data, and classifying the extracted testing coefficients against the training coefficients. In Figure 6 we compare the recognition rates for fitting all types of models (ML, MLB, or MLBQ) to all types of data (again ML, MLB, and MLBQ). For fitting MLB and MLBQ models we also show results for the regularized version of the energy function. Overall fitting the MLBQ model with regularization (MLBQ/R) performs best across all three data models (83.3% accuracy vs. 60% for the non-regularized version). The non-regularized fitting algorithms perform well only in matched conditions (i.e. with fitted to data of equal complexity). In the case of mismatches (e.g. fitting MLBQ to linear data), performance decreases noticeably.

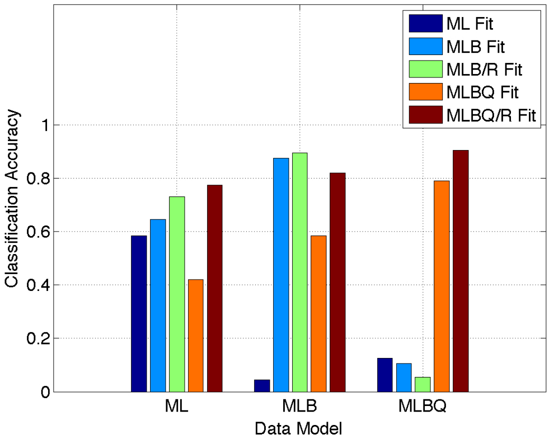

Figure 6. Comparison of regularized and non-regularized fitting algorithms for different data complexities. We synthetically generate linear (ML), mixed linear/bilinear (MLB), and mixed linear/bilinear/quadratic (MLBQ) data and fit ML, MLB, and MLBQ models to it. We measure classification accuracies for extracted test coefficients against training coefficients for all combinations of fitted models and data complexities. Across all data models, fitting the regularized MLBQ model (shown as MLBQ/R) performs best (83.3% accuracy). Non-regularized fitting algorithms perform well only in matched conditions but degrate substantially in non-matched conditions (e.g. fitting MLBQ to linear data).

4.5. Semi-Supervised Training

With the growing availability of large numbers of images from e.g. photo sharing sites such as Flickr, there is increasing interest in semi-supervised learning methods which are able to use incompletely labeled data [23]. Previously proposed algorithms to fit multi-factor models such as the Tenenbaum-Freeman algorithm [20] require complete labels for all training examples. In our framework, coefficient constraints are added individually per training vector *and* factor. Constraints can be included for only $\mathbf{c}_1$, or only $\mathbf{c}_2$, or both. See Eqn. (9) in Section 4.1. As a consequence we can include training vectors were only e.g. identity information but not expression information is available.

5. Experiments

In this section we provide experimental results for de-identifying images encoded with the multi-factor framework introduced in this paper. In Section 5.1 we describe the data used in the experiments. We then show results of de-identification experiments in Section 5.2. In Section 4.5 we demonstrate that usage of partially labeled data improves the quality of the de-identification results.

5.1. Data Set

We use a subset of the CMU Multi-PIE face database [12] containing 100 subjects displaying neutral, smile and disgust expressions in frontal pose and with frontal illumination. The images were captured within minutes of each other as part of a multi-camera, multi-flash recording. We normalize the face images by manually establishing facial feature point labels, computing an Active Appearance Model [8, 14] over the dataset, and extracting the appearance parameters for all images.

5.2. Multi-Factor Representation

In the first experiment we compare privacy protection and data utility of the (ϵ, k)-map algorithm [10] (an extension of the k-Same algorithm described in Section 2.2) using two different data representations: the original AAM appearance parameters and the combined $\mathbf{c}_1$ and $\mathbf{c}_2$ parameters extracted from a combined linear and quadratic model. For both representations we de-identify the data, reconstruct the (normalized) image and compute recognition rates using a whitened cosine distance PCA classifier [4] with the de-identified images as probe and the original images as gallery. We evaluate the utility of the de-identified images by computing facial expression classification rates using a SVM classifier (trained on independent original images) [18]. Figure 7 plots the results of both experiments for varying values of k for the original and multi-factor representations. Across all values of k, expression classification on de-identified images based on the multi-factor representation yields better recognition rates while providing the same privacy protection. As comparison, results for simple blurring of the images are included as well. Figure 8 shows examples of smile images de-identified using the proposed framework.

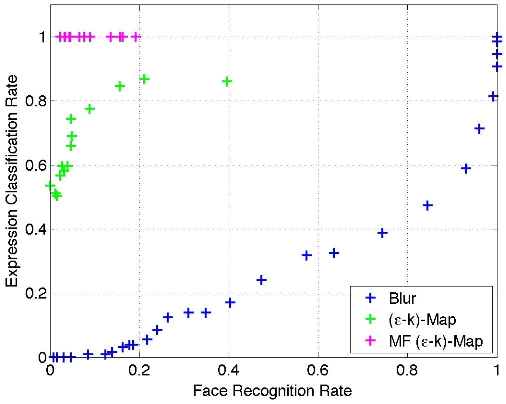

Figure 7. Privacy-Data Utility map of the (ϵ, k)-map algorithm using original and multi-factor representations. We show PCA face recognition and SVM facial expression classification rates for different values of the privacy parameter k. Usage of the multi-factor representation (MF (ϵ, k)-map) results in higher expression classification accuracies than the original representation while providing similar privacy protection. As comparison we also show results for image blurring.

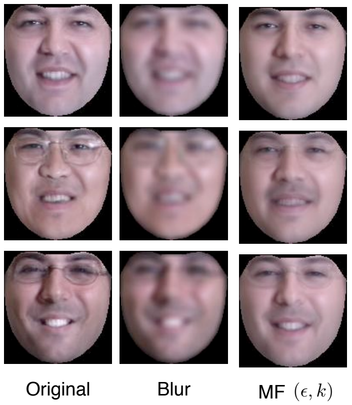

Figure 8. Examples of smile images de-identified using the multi-factor (ϵ, k)-map algorithm.

5.3. Semi-Supervised Learning

In the second experiment we evaluate the influence of semi-supervised learning as described in Section 4.5. For this we split images into two sets of 50 subjects each (set 1 and set 2). We compare the performance of the k-Same-Select algorithm on coefficients extracted in one of three means : a) after training on set 1 and set 2 using both identity and expression labels for all elements, b) after training on set 1 using both labels and set 2 using only identity labels, and c) after training on only the subjects in set 1 using both labels. In all cases we then de-identify images of subjects in set 2. Figure 9 shows the resulting Privacy/Data Utility map. Performance of facial expression classification with an SVM classifier on coefficients trained without

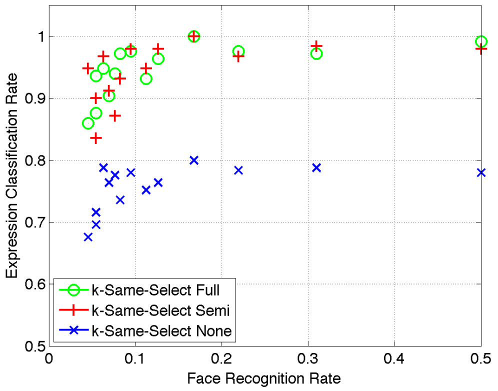

Figure 9. Privacy-Data Utility map of the k-Same-Select algorithm applied to different data representations for different values of k. Expression classification performance for coefficients extracted from a model trained without expression labels (94% on average, labeled “Semi”) is very close to the performance on coefficients trained with full labels (95% on average, labeled “Full”). Performance on coefficients trained in a tradional manner is noticeably worse (75% on average, labeled “None”).

expression labels is very close to the performance on coeficients trained with full labels (95% for full labels, 94% for semi labels). Performance for coefficients trained on set 1 is noticeably worse (75%). For all representations the same privacy protection is achieved.

6. Conclusion

In this paper we proposed a new framework for the combined learning of linear, bilinear, and quadratic models. We described a new fitting algorithm which jointly estimates all model parameters and showed how to use it for semi-supervised learning. In experiments on the CMU Multi-PIE database we demonstrated that usage of our framework improves the quality of results for algorithms de-identifying facial images. In future work we plan to apply the framework to video sequences in medical face databases.

7. Acknowledgements

This work was supported by the National Institute of Justice, Fast Capture Initiative, under award number 2005-IJ-CX-K046.

References

[1] A. Ashraf, S. Lucey, J. Cohn, T. Chen, Z. Ambadar, and K. Prkachin. The painful face - pain expression recognition using active appearance models. In *ICMI*, 2007.

[2] P. Baldi and K. Hornik. Neural networks and principal component analysis: learning from examples without local minimia. *Neural Networks*, 2(1):53–58, 1989.

[3] A. Barucha, C. Atkeson, S. Stevens, D. Chen, H. Wactlar, B. Pollock, and M. Dew. Caremedia: Automated video and sensor analysis for geriatric care. In *Annual Meeting of the American Association for Geriatric Psychiatry*, 2006.

[4] R. Beveridge, D. Bolme, B. Draper, and M. Teixeira. The CSU face identification evaluation system. *Machine Vision and Applications*, 16:128–138, 2005.

[5] C. Bishop. *Neural networks for pattern recognition*. Oxford University Press, 1995.

[6] A. Björck. *Numerical methods for least squares problems*. SIAM, 1996.

[7] M. Boyle, C. Edwards, and S. Greenberg. The effects of filtered video on awareness and privacy. In *ACM Conference on Computer Supported Cooperative Work*, pages 1–10, Philadelphia, PA, Dec 2000.

[8] T. Cootes, G. Edwards, and C. Taylor. Active appearance models. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 23(6), 2001.

[9] R. Duda, P. Hart, and D. Stork. *Pattern Classification*. Wiley-Interscience, 2000.

[10] R. Gross. *Face de-identification using multi-factor active appearance models*. PhD thesis, Carnegie Mellon University, 2008.

[11] R. Gross, E. Airoldi, B. Malin, and L. Sweeney. Integrating utility into face de-identification. In *Workshop on Privacy Enhancing Technologies (PET)*, June 2005.

[12] R. Gross, I. Matthews, J. Cohn, T. Kanade, and S. Baker. The CMU multi-pose, illumination, and expression (Multi-PIE) face database. Technical Report TR-07-08, Carnegie Mellon University, Robotics Institute, 2007.

[13] R. Gross, L. Sweeney, F. de la Torre, and S. Baker. Model-based face de-identification. In *IEEE Workshop on Privacy Research in Vision*, 2006.

[14] I. Matthews and S. Baker. Active appearance models revisited. *IJCV*, 60(2):135–164, 2004.

[15] C. Neustaedter and S. Greenberg. Balancing privacy and awareness in home media spaces. In *Workshop on Ubicomp Communities: Privacy as Boundary Negotiation*, 2003.

[16] E. Newton, L. Sweeney, and B. Malin. Preserving privacy by de-identifying facial images. *IEEE Transactions on Knowledge and Data Engineering*, 17(2):232–243, 2005.

[17] P. J. Phillips. Privacy operating charateristic for privacy protection in surveillance applications. In *AVBPA*, 2005.

[18] B. Schoelkopf and A. Smola. *Learning with Kernels: support vector machines, regularization, optimization, and beyond*. MIT Press, 2001.

[19] L. Sweeney. k-anonymity: a model for protecting privacy. *International Journal on Uncertainty, Fuzziness, and Knowledge-Based Systems*, 10(5):557–570, 2002.

[20] J. Tenenbaum and W. Freeman. Separating style and content with bilinear models. *Neural Computation*, 12(6):1247–1283, 2000.

[21] F. d. l. Torre and M. Black. A framework for robust subspace learning. *IJCV*, 54(1-3):117–142, 2003.

[22] M. Vasilescu and D. Terzopoulous. Multilinear subspace analysis of image ensembles. In *CVPR*, 2003.

[23] X. Zhou. Semi-supervised learning literature survey. Technical Report Computer Sciences TR 1530, University of Wisconsin - Madison, 2007.