On Constructing Facial Similarity Maps

Alex Holub
California Institute of Technology
Pasadena, CA
holub@caltech.edu

Yun-hsueh Liu

Pietro Perona

Abstract

Automatically determining facial similarity is a difficult and open question in computer vision. The problem is complicated both because it is unclear what facial features humans use to determine facial similarity and because facial similarity is subjective in nature: similarity judgements change from person to person. In this work we suggest a system which places facial similarity on a solid computational footing. First we describe methods for acquiring facial similarity ratings from humans in an efficient manner. Next we show how to create feature vector representations for each face by extracted patches around facial keypoints. Finally we show how to use the acquired similarity ratings to learn functional mapping which project facial-feature vectors into **Face Spaces** *which correspond to our notions of facial similarity. We use different collections of images to both create and validate the Face Spaces including: perceptual similarity data obtained from humans, morphed faces between two different individuals, and the CMU PIE collection which contains images of the same individual under different lighting conditions. We demonstrate that using our methods we can effectively create Face Spaces which correspond to human notions of facial similarity.*

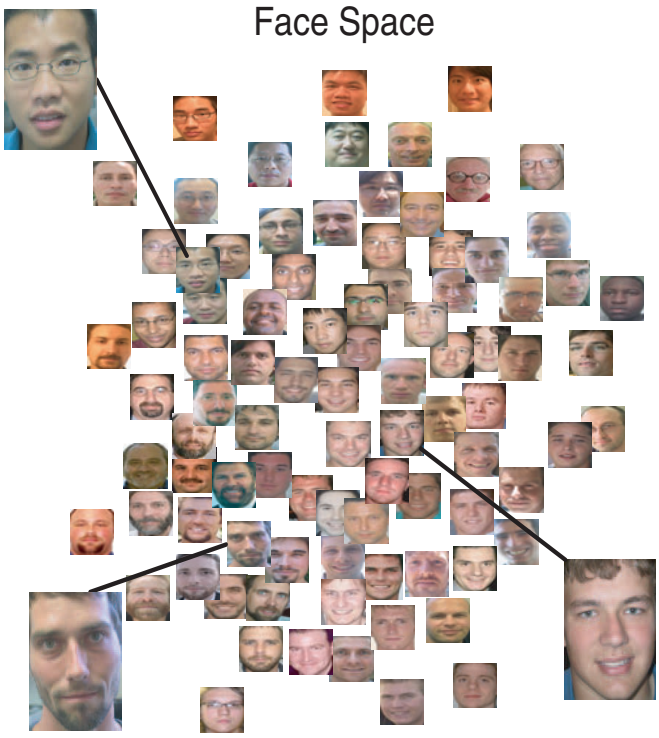

Figure 1. Example of typical Face Space. Perceptual ratings from 130 images were used to generate a linear map. $\mathcal{R}^{\mathcal{D}_{\text{MAP}}} = 100$, $D_{\text{PCA}} = 200$. MDS was performed to embed projected faces into a 2D space. Notice that some areas do not conform well to notions of similarity, such as the upper right of the space. Other areas contain groups of similar-looking faces (these areas are highlighted by enlarged images), such as 'Asian', and 'Men with Beards and Sunken Eyes'. Overall the map seems todo a good job of creating a metric space which preserves notions of facial similarity.

1. Introduction

Humans naturally perceive the similarity between different objects. Humans are especially sensitive to facial similarity and it has been suggested that individuals seek partners with similar facial attributes [6]. Facial similarity is particularly useful in social situations such as determining familial relationships or dating preferences. The goal of this work is to place facial similarity on a solid computational footing. We suggest to learn functions which map measured facial features to metric spaces in which similar looking faces are near one another.

While there is a vast amount literature devoted to facial recognition, judging similarity is a more subtle and difficult topic. Our challenges include: (A) obtaining reliable facial similarity judgments from human observers, (B) developing 'objective' approaches to similarity to supplement measurements from humans, (C) automatically mapping measured facial features to a space where the natural metric predicts human similarity judgments.

Within the psychological literature there have been nu-

1-4244-1180-7/07/$25.00 ©2007 IEEE

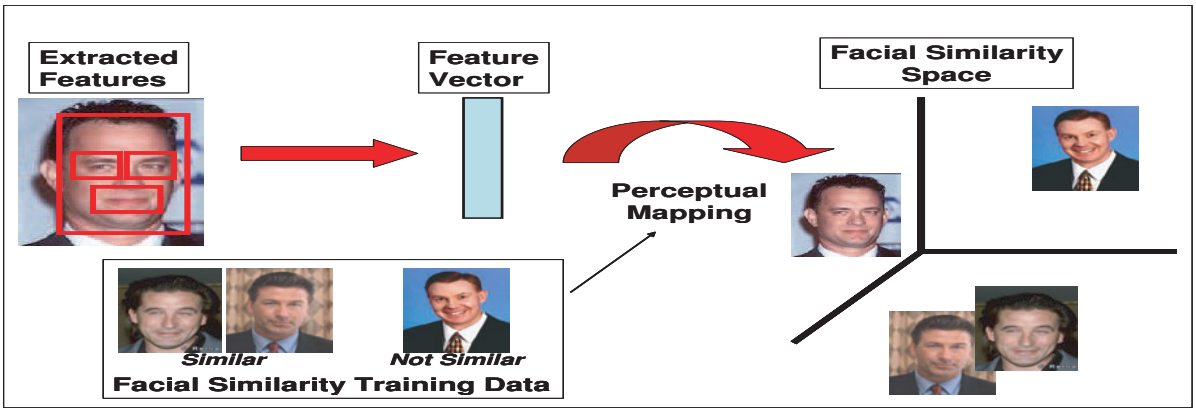

Figure 2. Outline of proposed facial similarity algorithm. Patches are extracted from the eyes and mouth and converted to a feature vector. A mapping function is generated using perceptual similarity training data which is used to project the feature vectors to a metric space which reflects human similarity judgements.

merous studies on how to create metric spaces for faces which reflect perceptual notions of similarity (see for instance [1]). Multi-Dimensional Scaling (MDS) [7] is often used to embed faces into a metric space based on perceptual judgements of facial similarity. Within these embedded metric spaces, faces which appear similar are nearby one another and thus these spaces are often referred to as *Faces Spaces*. These methods have two major drawbacks: (1) MDS methods are not useful when presented with new faces as MDS does not create an explicit mapping function, (2) The faces used were always in standard poses and constant lighting conditions and do not exhibit the same statistical variations found in real-world images.

The computer vision community has a long history of mapping faces into lower dimensional representations for recognition, such as the 'unsupervised' Eigenface [11] and 'supervised' FisherFace [3] methods. More recently LeCun et al. [4] proposed an interesting supervised non-linear mapping using contrastive divergence learning and a convolutional neural network which they apply to numerous data-sets in addition to face data-sets. The goal of our work is to explore in a principled manner the creation of facial similarity spaces which reflect perceptual notions of similarity. We generate explicit maps from extracted facial features and use mostly *real-world* images obtained both from the web and personal photo collections. Figure 2 gives an overview of our proposed algorithm.

The paper is organized as follows: In Section 2 we describe and compare methods for effectively obtaining facial similarity data. In Section 3 we describe the various data-collections used. In section 4 we show how to use this training data to construct explicit functional mappings from feature space to face spaces. Finally, in Sections 5 and 6 we will present and discuss our experiments.

2. Obtaining Facial Similarity Data

A difficult requirement inherent in creating and evaluating a facial similarity space is acquiring large amounts of facial similarity data in an efficient manner. It is desirable that: (1) the similarity measurements be obtained quickly, (2) the measurements be accurate, and (3) the ratings be collected on a large set of faces. Authors have suggested numerous methods for comparing and rating the similarity of faces (see for instance [8]) and here we evaluate variants of two common paradigms which we call *Absolute* and *Relative* rating methods. Figure 3 describes and compares both the methods.

We compared the two rating methods by asking 5 subjects to rate the facial similarity of 127 face images: 100 'random' face images and 9 sets of 3 images of the same person (the images of the same were photos of minor celebrities). We included images of the same person in order to ensure that each subject was accurately performing the rating tasks (i.e. that when the subjects were presented with two images of the same individual, they would indicate that these images were very similar to one another). It took subjects on average 3s to make an absolute rating and 12s to make a relative rating.

We wish to understand how consistent subjects were in assessing facial similarity. That is, if the subject were asked to make the same similarity judgement twice would they make the same judgement? For space considerations we have included the analysis of consistency within the Supplementary Materials, but note that consistency within a subject was slightly higher than consistency between subjects. However, consistency between subjects was surprisingly high on this data-set.

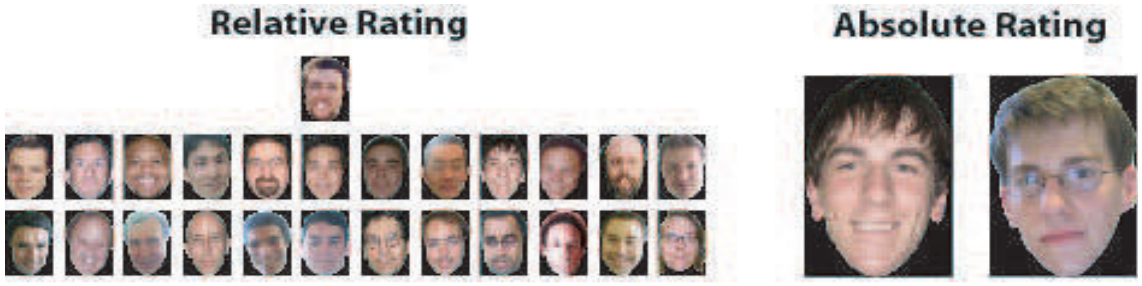

Figure 3. (Left) Relative Rating experiment. Subjects are asked to select which of the 24 faces are most similar to the target face located on top. (Right) Subjects are asked to rate, from 1-7, how similar the two faces are. Subjects were given a precise definition of each numerical value, from (7) 'Same Person' to (1) 'Completely Different'.

2.1. Synthetic Experiments using MDS

Which method, the relative or absolute rating method, is more efficient in creating *Face Spaces*? The relative method yields relative rating information: e.g., Face $\mathcal{A}$ is closer to Face $\mathcal{B}$ than Face $\mathcal{A}$ is to Face $\mathcal{C}$. The absolute method gives the absolute distance between two faces as judged by the subject: e.g. Face $\mathcal{A}$ and $\mathcal{B}$ are distance 2 apart. How can we compare the efficiency of these two rating methods? We proceed by using Multi-Dimensional Scaling (MDS) [7] on both relative and absolute rating data.

First we generate a set of synthetic vectors, where each vector represents a face. From this synthetic data we generate artificial absolute and relative ratings. We then use these artificial ratings to re-create the original vector space using MDS. Next we describe the steps more explicitly.

(Step 1) Generate a random set of N vectors of dimensionality D. Each vector represents a face and the perceptual information available to the subject. (Step 2) Generate the pairwise distances between all vectors which correspond to perceptual distances between faces. (Step 3a) *Absolute Measurements*: distances are discretized into 7 discrete values, $[1..7]$. An absolute measurement is indicated by the discrete value between two vectors. I.e. if vector (image) $\mathcal{A}$ and $\mathcal{B}$ are 2 apart, then this corresponds to an absolute rating of 2 between these two vectors (images). (Step 3b) *Relative Measurements*: Randomly generate 25 images and set one as the target image as in Figure 3. Sort the remaining images by their Euclidean distance to the target image and chose the closest Euclidean image.

From the Supplementary Material we know that subjects are not always consistent in their ratings (i.e when presented with the same set of faces they will not always make the same similarity judgement). We add appropriate noise to the synthetic analysis to ensure that the synthetic responses have the same amount of uncertainty as the subject responses.

Once the sets of absolute and relative ratings have been generated from the synthetic data we perform MDS on both sets of of ratings. For the relative ratings we assign a distance of 1 and 2 to close and far images respectively. We use the Sammon [9] stress criteria for creating the MDS spaces which penalizes most points which are measured as being close to one another (i.e. faces which are perceptually similar) and which are far apart in the embedded space. Explicitly it minimizes the stress $\mathcal{E}$, where

$$\mathcal{E} = \sum_{k \neq l} \frac{[d(k,l) - d'(k,l)]^2}{d(k,l)} \tag{1}$$

and $d(k,l)$ and $d'(k,l)$ is the distance between points k and l in both the original and embedded space respectively. Other stress functions yielded qualitatively similar results. Figure 4 compares the two rating techniques and we find relative ratings re-create the original target space more accurately than absolute ratings. We thus chose to use relative ratings to obtain perceptual facial similarity data.

Collection	Num Sets	Total Num Images
Perceptual	-	180
Celebrity Morphs	-	52
Celebrity	62	400
PIE	10	270

Table 1. Different data collections and the number of images/sets of images in each collection.

3. Data Collections

We wish to create a Face Space by training a mapping function using a set of perceptual training data. After the perceptual space has been generated we would like to evaluate its performance. We use 4 different collections of images, shown in Figure 5, to evaluate and create Face Spaces. We describe each below:

Perceptual Measurements We collect perceptual data from subjects by asking two subjects (one Caucasian male and female US native) to rate the similarity of 180 face images using the relative rating technique shown in Figure 3.

Figure 5. Different data collections used to train perceptual mappings. (Top Row) Images from two individuals from the PIE collection, note the controlled variations in lighting but little pose variation. (Middle Row) Images of two celebrities, they exhibit some pose, lighting, and facial affect variations. (Bottom Row) Example of three sets of morphed images, center is the morphed image, the sides are the images used to generate the morph. Note that the morphed image appears perceptually similar to the faces used to generate the morph, making this collection potentially useful for training a perceptual similarity map.

CMU Pie Collection We selected 10 unique individuals from the CMU PIE Collection [10] and used 27 frontal poses for each individual, which contain variations in lighting and some facial emotions in a controlled environment.

Celebrity Images We wished to obtain multiple images of individuals in more diverse lighting and pose conditions than available in the PIE collection. For this we downloaded 62 sets of images from the web. Each collection had between 3-11 images. These individuals tended to be Celebrities.

Celebrity Morphs [1] We wish to artificially create similar faces using a morphing program. We randomly chose two images of different individuals from our celebrity collection and morphed the images together. We chose the perceptual middle point of the morph when the morph face looked equally similar to the two faces (see Figure 5 for examples of the morphs). The generated morphs do indeed look very similar to the two faces which generated them.

4. Creating a Perceptual Mapping

Next we discuss how to use to create an explicit mapping from features extracted from face images to a space which conforms to facial similarity.

4.1. Facial Feature Representation

First we describe how to obtain a feature (vector) representation for each face. We manually annotated points at the corners and above/below the eyes and mouth. We note that automatic detectors for these particular facial locations have been shown to be successful on similar data [5], but for this study we preferred more controlled data. We extracted patches around both eyes and the mouth at the size shown in Figure 6 and resized them to be 17×24 and 25×14 respectively. The patches were combined into one vector and used to represent each face. Each patch was normalized to have zero mean and unit variance. We also conducted experiments which included the area around the nose, but found no noticeable change in performance. The dimensionality of all feature vectors was reduced to $D_{\text{PCA}} = 200$ dimensions using PCA which typically encompassed $> 99\%$ of the variance. We also experimented using kPCA with an RBF kernel to reduce dimensionality and found no noticeable change in performance.

4.2. Representing Ratings using 'Triplets'

In Section 2 and Figure 3 we discussed how to obtain relative facial similarity measurements. Here we discuss how to represent relative measurements so they can be used to optimize a perceptual map. Each relative rating tells us that the image which is picked (i.e. image $\mathcal{B}$) is more similar to the target image $\mathcal{A}$ than any other image in the set of 24 images, indexed $\mathcal{C}$. We can represent this information using a triplet $[\mathcal{A}, \mathcal{B}, \mathcal{C}]$. For any triplet, the subject has indicated that $\mathcal{D}(\mathcal{A}, \mathcal{B}) < \mathcal{D}(\mathcal{A}, \mathcal{C})$ where $\mathcal{D}$ is some perceptual distance metric used by the subject. It is easy to see that each relative rating generates 23 such triplets.

We can represent data collections containing images of the same person, such as the Celebrities and PIE collections, using triplets as well. We consider that images of each individual are more similar to one another than to any of the other images in any collection. In this case, in the triplet

[1] We used the program FaceMorpher Multi by Luxand Development to create facial morphs.

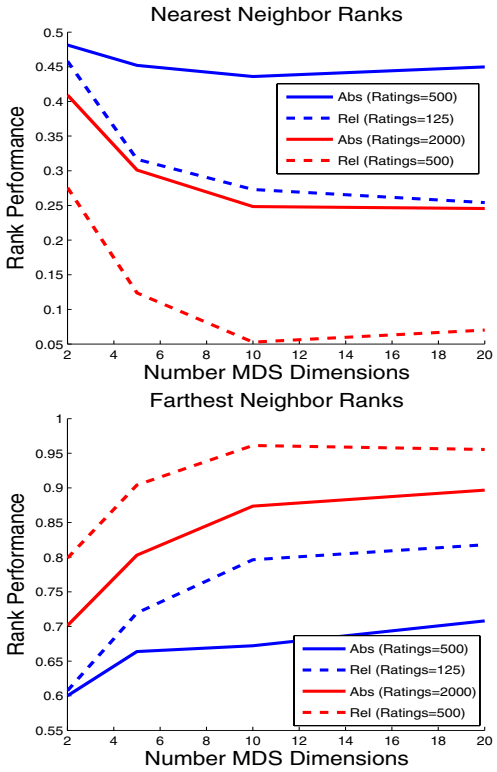

Figure 4. Synthetic comparison of Absolute and Relative measurements using MDS. (Top) Nearest Neighbors. For a vector (image) $\mathcal{A}$ we find the nearest neighbor to $\mathcal{A}$ (let this be $\mathcal{B}$). These two vectors (images) are embedded into an MDS space which is created using a set of either absolute or relative ratings. We then calculate the rank distance of these two vectors in the embedded MDS space: $r = \text{rank}(\mathcal{A}, \mathcal{B})$. The y-axis is the average rank distance between all images in the MDS space. The best possible performance would be a value of 1 as this would indicate every nearest neighbor in the original space was a nearest neighbor in the embedded space. A lower rank distance is better. Solid lines: absolute ratings, dashed lines: relative ratings. Different colors indicate different numbers of acquired ratings. The number of absolute/relative ratings were chosen such that they took an equivalent amount of time. The dimensionality of the embedded space is varied on the x-axis. (Bottom) Farthest Neighbor. Same as top but instead of looking at the rank of the *closest* point we look at the rank of the *furthest* point. In this case larger ranks r indicate that the MDS embedding is performing better. 200 random vectors of dimensionality 10 were generated for these experiments. Relative ratings seem to re-create the vector space more effectively over all parameter initializations.

$[\mathcal{A}, \mathcal{B}, \mathcal{C}]$, $\mathcal{A}$ and $\mathcal{B}$ are images of the same person and $\mathcal{C}$ is an image of a different individual. For example if we have 2 images of the same individual, and 100 images of other individuals, we can generate $2 * 100 = 200$ triplets.

We follow similar logic for the morphed images. In this case the morphed image is similar to both of the individuals

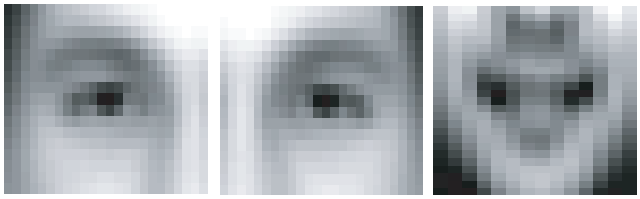

Figure 6. Examples of the average visual features extracted from a set of faces. Note the relatively large size of the extracted feature.

used to generate the morph, but dissimilar to other individuals. For instance, if we generate a morph from two individuals where we have 2 images of each individual, as well as 100 images of other people, we can generate $4 * 100 = 400$ different triplets.

4.3. Perceptual Map

How can we generate a perceptual map which projects the feature vectors extracted in Section 4.1 to a space conforming to our notions of similarity? Explicitly, we would like a mapping f which takes feature vectors representing each face of dimensionality D_{PCA} and projects them into a space of dimensionality D_{MAP} such that $f : \mathcal{R}^{\mathcal{D}_{\text{PCA}}} \rightarrow \mathcal{R}^{\mathcal{D}_{\text{MAP}}}$, where D_{MAP} is typically 100. Results are not particular sensitive to small changes in the dimensionality of D_{MAP}: we found qualitatively similar results for $D_{\text{MAP}} = 50-200$. Here we assume a linear map, although the framework is applicable to any differentiable function f. We can represent our mapping function as $\vec{y} = f(\vec{x})$, or, since we are assuming a linear map, $\vec{y} = M\vec{x}$, where M is a matrix of $\mathcal{R}^{\mathcal{D}_{\text{MAP}}} \times \mathcal{R}^{\mathcal{D}_{\text{PCA}}}$, and $\vec{x}$ is a feature representation for a face, and $\vec{y}$ is the projected representation after the linear map.

Consider a particular triplet $t = [\mathcal{A}, \mathcal{B}, \mathcal{C}]$ as described in the previous section. Now consider a feature representation for the images $\mathcal{A}, \mathcal{B}, \mathcal{C}$ which we denote by $\vec{a}, \vec{b}, \vec{c}$. For a particular triplet, we would like that the distance between $\vec{a}$ and $\vec{b}$ be less than the distance between $\vec{a}$ and $\vec{c}$ in the projected space. Thus we would like a cost function which penalizes inequalities when the following is true: $\mathcal{D}(M\vec{a}, M\vec{b}) > \mathcal{D}(M\vec{a}, M\vec{c})$ where M is the projection matrix to be optimized. We use the squared L2 metric to calculate distances in the projected space and penalize inequalities using:

$$S_t = ||M\vec{a}_t - M\vec{b}_t||^2 - ||M\vec{a}_t - M\vec{c}_t||^2 \quad (2)$$

$$S_t = (\vec{a}_t - \vec{b}_t)M(\vec{a}_t - \vec{b}_t)^t - (\vec{a}_t - \vec{c}_t)M(\vec{a}_t - \vec{c}_t)^t \quad (3)$$

The mapping is linear, so the derivative of the penalty term w.r.t. the linear mapping is a matrix and can be written as:

$$\frac{\partial S_t}{\partial M} = (\vec{a}_t - \vec{b}_t)^t(\vec{a}_t - \vec{b}_t) - (\vec{a}_t - \vec{c}_t)^t(\vec{a}_t - \vec{c}_t) \quad (4)$$

	Exponential	Sigmoid	Rank [2]
20 Celeb Train	$.25 \pm .04$	$.25 \pm .03$	$.27 \pm .04$
40 Celeb Train	$.2 \pm .03$	$.21 \pm .03$	$.23 \pm .04$
60 Celeb Train	$.17 \pm .04$	$.18 \pm .05$	$.20 \pm .04$

Table 2. Comparison of different cost functions when different numbers of celebrities are used for training. Rank performance on celebrity data-sets shown, see Figure 9 for details on calculating rank performance. Lower is better. Averaged over 20 iterations. The exponential cost seems to perform the best.

and if we impose an exponential cost function, C_e and sum over all triplets t:

$$\mathcal{C}_e = \sum_t exp(\frac{-S_t}{\beta}) \tag{5}$$

$$\frac{\partial \mathcal{C}}{\partial M} = \frac{1}{\beta} \sum_t exp(\frac{-S_t}{\beta}) \frac{\partial S_t}{\partial M} \tag{6}$$

where β controls the sensitivity of the penalty (we usually set $\beta = 1$ for our experiments), i.e. the steeper the exponential distribution (smaller β), the more we penalize triplets in which the inequality is not respected. We considered other cost functions including a sigmoid function, C_s:

$$C_s = \sum_t Sig(-S_t) \tag{7}$$

where $Sig(\mathrm{x}) = \frac{1}{1+exp^{-x}}$. We also considered the cost function proposed by the authors of [2] derived from the statistical estimate of the rank correlation between two variables. We found the exponential cost function to, in general, yield slightly better results for most simulations (see Table 2 for a comparison of different cost functions). In addition we experimented using a non-linear one-layer Radial Basis Function network as our mapping function. These mappings exhibited severe problems with over-fitting due to the large number of parameters in our map and the relatively small amount of training data available and hence we used the linear map M for our experiments.

We optimize the map using the conjugate gradient algorithm. We initialize the matrix M at multiple starting points to avoid local minima and chose the experiment resulting in the lowest cost. We find that convergence usually occurs after 100 iterations. We witnessed only minor fluctuations in cost due to local minima during optimizations.

5. Experiments

In the previous sections we described how to acquire perceptual judgements efficiently and how to generate a perceptual map. In this section we describe our experiments. First, we explicitly indicate how we divide up our training and test set of data so as not to contaminate training data with test data.

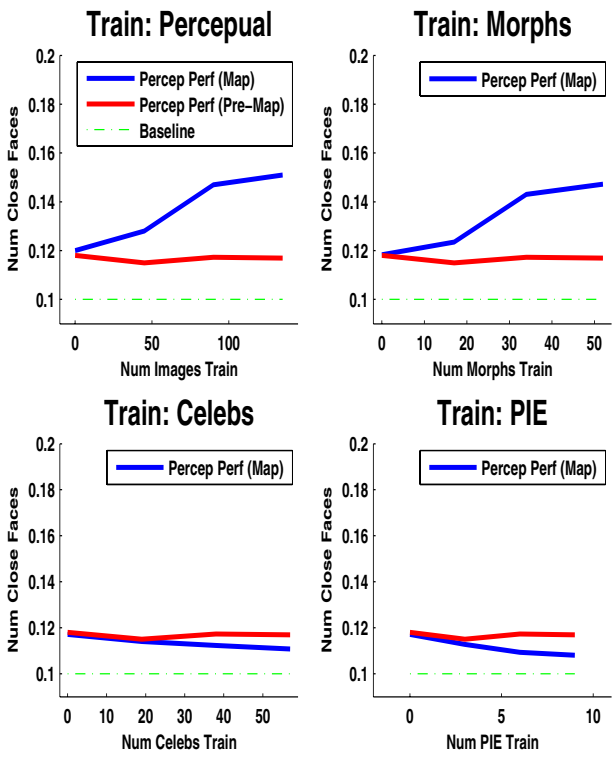

Figure 7. How accurately can we reproduce human perceptual judgements? We train mappings using 4 different collections of images (the training collection used to create the mapping is indicated in the title of each sub-plot). x-axis: clockwise from the top left, number of images for which perceptual data was acquired, the number of morph images, the number celebrity individuals, and the number of pie individuals used to train the mapping. y-axis: performance on human perceptual measurements using the metric described in Section 5.3. Higher is better. Red line is the performance expected by chance. The best performance is obtained when the map is trained with perceptual data (about $4\times$ better than baseline performance), although morphed images perform well as well. Training a mapping using either the Celebrity and PIE data-sets results in poor performance: these collections do not seem appropriate for training mappings which predict human perceptual judgements.

5.1. Creating Train / Test Sets

Perceptual Ratings: We collected relative ratings on 180 face images which we call this set $\mathcal{E}$. For training, we selected a subset $\mathcal{E}$ as a train set and found all the triplets indexing these training images. The test set consisted of the remaining images and the triplets associated to those test images.

Celebrity/Pie Images: Of the 62 sets of celebrities we chose a subset to train with and the remainder was used as a test set. 120 additional images from the set $\mathcal{E}$ were used as 'far' images, e.g. in the triplet $[\mathcal{A}, \mathcal{B}, \mathcal{C}]$, $\mathcal{A}$ and $\mathcal{B}$ referenced

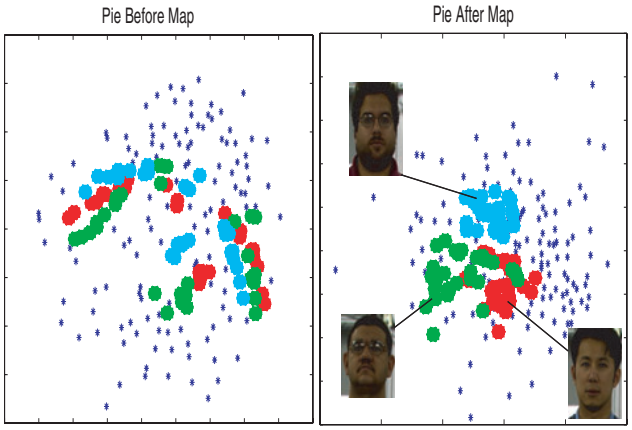

Figure 8. Distance between PIE individuals before and after mapping. Note the tighter clusters after the mapping, the right image, than before the mapping, the left image. This mapping was generated by training on 54 sets of *Celebrities*, no PIE images were used for training the map. (Left) The blue stars represent non-PIE images. Each color represents a single PIE individual, and each dot represents 1 of 27 different images of this person under different with different lighting and facial affects. The embedding was generated using MDS on the PCA reduced representations of each feature vector. (Right) The same set of points after being mapped into face space and embedded using MDS. Pictures indicate the identity of each cluster. Note that each PIE individual is now clustered in a particular area of space and that the points representing each individual are now closer to one another. The mapping has learned invariance to lighting and facial affect. This is somewhat remarkable considering that the map was trained on only Celebrities and generalized similarity information across data-sets to the PIE images.

an image of the same celebrity while $\mathcal{C}$ indexed one of the images from the set $\mathcal{E}$. For PIE experiments, we used 10 sets of PIE people in a paradigm identical to the Celebrity Images.

Celebrity Morphs: Each morph image was generated from two celebrity images and the triplets, $[\mathcal{A}, \mathcal{B}, \mathcal{C}]$ were generated such that $\mathcal{A}$ corresponded to the morphed image, $\mathcal{B}$ to an image of a celebrity from which the morph was generated, and $\mathcal{C}$ an image from $\mathcal{E}$. We used morphs and their associated celebrities for training and the rest of the images for testing.

5.2. Assessing Map Performance: Rank

How should we measure performance before and after the perceptual mapping(s)? Since the learned map M can scale the space arbitrarily, a Euclidean distance metric is not appropriate. We chose instead to measure performance using the *rank* between images. The rank r between two images is the number of images in between two images. Consider a set of N face images. Let n_i^c indicate that image i is of celebrity c. Finally let M^c be the number of images of celebrity c in the set N. We can measure the average rank of celebrity c as:

$$r^c = \frac{1}{M^c(M^c-1)\times N} \sum_{i,j\in c, i\neq j} \text{rank}(\vec{n}_i^c, \vec{n}_j^c) \qquad (8)$$

5.3. Assessing Map Performance: Closest

We would also like to quantify the performance of the perceptual ratings obtained from humans. This is a bit trickier thank using the rank distance. We proceed as follows. For each image $\mathcal{A}$, find the 10% closest L2 distance images. Now consider all triplets of the form $[\mathcal{A}, \mathcal{B}, \mathcal{C}]$. Calculate the percentage of times $\mathcal{B}$ is in the set of closest images. The higher this percentage, the better the metric space is at predicted perceptual judgements. By chance we would expect on average 10% of triplets to have an image $\mathcal{B}$ in the top 10% closest images (a perfect map would yield roughly 19%). See Figure 7 for experiments evaluated using this performance metric.

6. Results

The different collections of data exhibit different statistical variations as shown in Figure 5. Not surprisingly, we found that a mapping had the highest success when tested on the same collection as it was trained with (see Figures 7 and 9): the map learned robustness to the statistical variations within the *training* data-set which it generalized to the *testing* set. Figure 1 shows a nice example of a face-space generated using perceptual data.

There are several interesting observations from our experiments: (1) Training using the morphed images generated Face Spaces which predicted user similarity judgements well (although not as well as training on facial similarity judgements from subjects). However training using either the Celebrity or PIE collections did not generalize to good performance on the human data. The latter two collections exhibit mostly variations in lighting and facial affect and the variability inherent there did not generalize to facial similarity. (2) Our performance curves, although leveling out, do not seem to have saturated, indicating that by acquiring more training data we might be able to improve performance further (see Figures 7 and 9. (3) The Celebrity collection generalized well to the PIE collection although the PIE did not generalize well to the Celebrity collection (Figures 8 and 9). Presumably the PIE collection, which was obtained in a highly controlled environment, did not exhibit enough variability to generalize to the other collections.

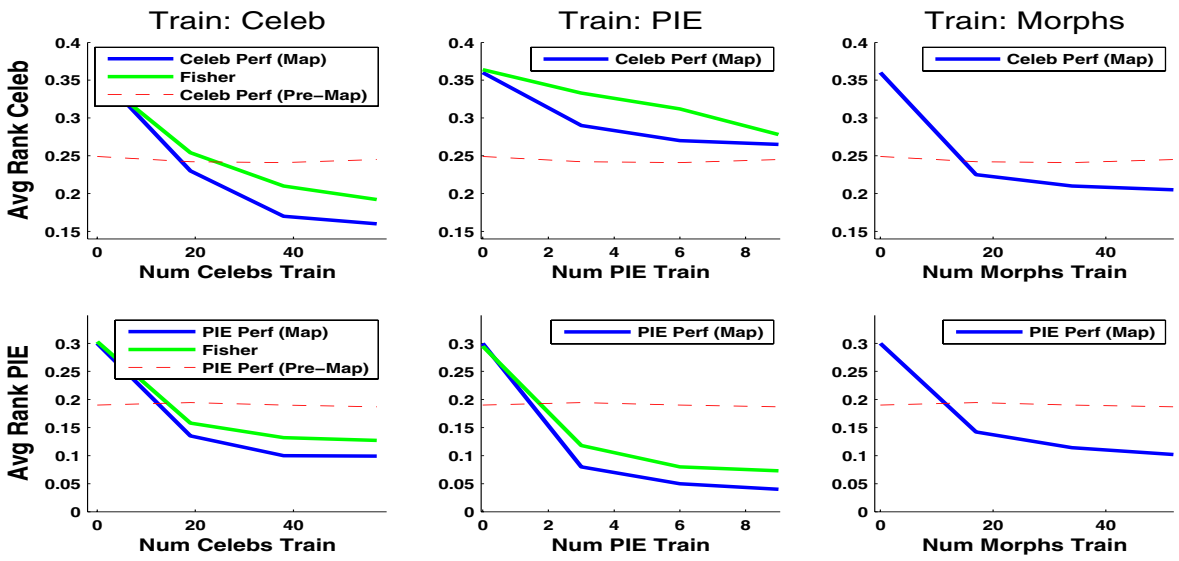

Figure 9. Rank performance of both the Celebrity and PIE collections after training mappings using different data. (Top Row) Rank performance of the celebrity collection. X-axis: the number of celebrities/PIE individuals/morphs used for training. Y-axis: the rank performance on the celebrity data-set (lower=bettter). Training with celebrities results in the best performance. Dotted red line is the rank performance before the mappings are applied. Green line is the performance when we train a mapping using Fisher Linear Discriminants (see [3]). Fisher seems to yield slightly worse performance when compared to our mapping. Note that increasing the number of individuals used for training yields better rank performance as the mapping over-fits less and generalizes more. (Bottom Row) Same but measuring rank performance on the PIE data-set. Training on PIE individuals results in the best performance. For both the top and bottom row, training on the same data-set as one tests on yields the best performance. Training using morph images seems to create good mappings for both PIE and Celebrity collections. Note that we create distinct train/test sets for all experiments (see Section 5.1). All results averaged 25 times.

7. Discussion

We have shown how to construct and evaluate facial similarity spaces which mimic human perceptual judgements on real data. In addition we show the flexibility of the approach: training the mapping with different data-collections results in different *Face Spaces*. This work takes the first steps towards creating metrics and mappings for faces which correspond to human perceptual judgements.

References

[1] F. Ashby. *Multidimensional models of Perception and Cognition.* Erlbaum, Hillsdale, NJ., 1992. 2

[2] B. Bartell, G. Cottrell, and R. Belew. Optimizing similarity using multi-query relevane feedback. In *American Society for Information Science*, volume 49, pages 742–761, 1998. 6

[3] P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 19(7):711–720, 1997. 2, 8

[4] S. Chopra, R. Hadsell, and Y. LeCun. Learning a similarity metric discriminatively, with application to face verification. In *CVPR*, 2006. 2

[5] M. Everingham and A. Zisserman. Regression and classification approaches to eye localization in face images. In *FG*, pages 441–448, 2006. 4

[6] V. Hinsz. Facial resemblance in engaged and married couples. *Journal of Social and Personal Relationships*, 6:223–229, 1989. 1

[7] J. Kruskal. In *Psychometrika*, pages 115–129, 1964. 2, 3

[8] G. Rhodes. Looking at faces: first-order and second-order features as determinants of facial appearance. *Perception*, 17:43–63, 1988. 2

[9] J. Sammon. A nonlinear mapping for data structure analysis. *IEEE Transactions on Compututation*, C-18:401–409, 1969. 3

[10] T. Sim, S. Baker, and M. Bsat. The cmu pose, illumination, and expression database. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(12):1615–1618, 2003. 4

[11] M. Turk and A. Pentland. Face recognition using eigenfaces. In *In Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 1991. 2