# Vignette: Reimagining the Analog Photo Album

David Eng, Andrew Lim, Pavitra Rengarajan

## 1 Abstract

Although the smartphone has emerged as the most convenient device on which to capture photos, it lacks the tools to effectively view and organize these photos. Given the casual nature of smartphone photography, we propose a system that can streamline and simplify the post-capture process. We propose Vignette, a content management system for smartphone photography that improves the visual story by automating the organization of a user's photo album. We hope to provide dynamic organization of photo albums that supports user queries including requests for images of a particular person or genre. The main task we consider is clustering photos of a particular individual.

## 2 Introduction

The basic structure of the system involves a pipeline that first sorts a photo album into images those that contains people and those do not. Further processing is then carried out on the photos of people to cluster the images into sub-albums with photos of a particular person. This would allow the user to perform some basic queries over what was a previously untagged set of photos.

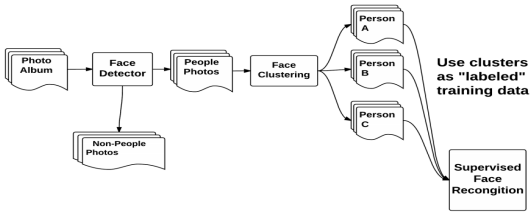


**Figure 1: Photo Organization Pipeline**

We consider a pipeline that involves both unsupervised and supervised learning. Early on, the system has no notion of any labels. Thus, on the arrival of a new image, we perform unsupervised clustering to add it to either an existing cluster or a new cluster. At some point, however, there may be a sufficient number of different clusters with high enough density that we could use as labeled training data for a supervised facial recognition algorithm.

## 4 Face Detection

At the moment, we have focused more on the facial recognition and clustering task. As our face detector, we use a Haar feature-based cascade classifier trained on various facial features including the position and shapes of eyes, the nose, and facial outline.

## 5 Feature Extraction

### 5.1 Eigenfaces

We consider the Eigenface algorithm, which performs a Principal Component Analysis (PCA) on the training set of face images to generate a set of basis vectors. Given a pixel grid with a face image, the Eigenface algorithm models faces as a composition of different Eigenfaces, which we discuss in detail below. We now describe the Eigenface algorithm, which is parametrized by $k$ where $k$ is the number of principal components to consider. We let $X = \{x^{(1)}, x^{(2)}, ..., x^{(n)}\}$ be the training set of face images, where $x^{(i)} \in \mathbb{R}^d$.

1. Compute the mean of $X$.
$$\mu = \frac{1}{n}\sum_{i=1}^{n} x^{(i)}$$
2. Compute the covariance matrix of $X$.
$$S = \frac{1}{n}\sum_{i=1}^{n}(x^{(i)} - \mu)(x^{(i)} - \mu)^T$$
3. Solve for the eigenvalues $\lambda^{(i)}$ and eigenvectors $v^{(i)}$ of $X$.
$$Sv^{(i)} = \lambda^{(i)}v^{(i)}, i = 1, 2, ..., n$$
4. Choose the eigenvectors that correspond to the $k$ largest eigenvalues.
5. We project the test images into the PCA subspace and use the features in this reduced dimensional space to train a multiclass SVM.

### 5.2 Fisherfaces

For the purposes of comparison to PCA using the Eigenfaces algorithm, we consider a second dimensionality reduction technique based on linear discriminant analysis (LDA). We now describe the Fisherface algorithm. Let us assume that our training set contains images of $n$ different people. We let $X = \{X_1, X_2, ..., X_n\}$, where $X_i$ is a set of images of person $i$. We compute the total mean of all the images:

$$\mu = \frac{1}{\sum_{i=1}^{n}|X_i|}\sum_{i=1}^{n}\sum_{x \in X_i} x.$$

We also compute a between-class scatter matrix defined as:

$$S_B = \sum_{i=1}^{n}|X_i|(\mu_i - \mu)(\mu_i - \mu)^T$$

and a within-class scatter matrix defined as:

$$S_W = \sum_{i=1}^{n}\sum_{x \in X_i}(x - \mu_i)(x - \mu_i)^T$$

The Fisherface algorithm then solves the following optimization to compute a projection $W_{opt}$ that maximizes the class separability.

$$W_{opt} = \arg\max_W \frac{|W^T S_B W|}{|W^T S_W W|} = [w_1 w_2 ... w_k]$$

The $x_i$'s correspond to the generalized eigenvectors of $S_B$ and $S_W$ that correspond to the $k$ largest eigenvalues. We note that there can be at most $n-1$ nonzero generalized eigenvalues. Thus, we have an upper bound on $k$ of $n-1$.

We note, however, that the within-class scatter matrix $S_w$ will always be singular. Thus, we perform PCA and reformulate the optimization problem as:

$$W_{PCA} = \arg\max_W |W^T S_T W|$$

$$W_{FLD} = \arg\max_W \frac{|W^T W_{PCA}^T S_B W_{PCA} W|}{|W^T W_{PCA}^T S_W W_{PCA} W|}$$

The transformation matrix $W$ is given by:

$$W = W_{FLD}^T W_{PCA}^T$$

### 5.3 Gabor Wavelets

To provide a computationally inexpensive means to extract relevant features from our image, we applied various Gabor wavelets, a selective filter for both scale and orientation of the face. Since it convolves a Gaussian kernel and sinusoid FFT, the filter is parameterized by the Gaussian $\sigma$, sinusoid phase $\phi$, orientation $w$, and wavelength $\lambda$.
We took a variety of approaches to clustering these filtered images. Let us assume that our training set contains $n$ images of different people.

1. Generate $g$ Gabor kernels by varying the values of $\sigma, \phi, w$, and $\lambda$.

2. For each image 1 to $n$: Represent the image as a list of $g$ convolutions of each Gabor kernel with the original image. To reduce the feature space, we also attempted to represent the image as a list of $g$ means and variances, based on least squared error for simplicity.

3. During $k$-means clustering, centroids take on the dimensionality of the filtered images.

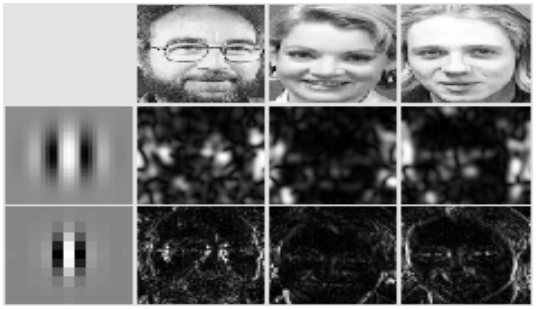

**Figure 2: Gabor Filters & Feature Extraction**

### 5.4 Neural Networks

In addition to the more engineered feature extraction techniques discussed, recent work has shown the validity of applying the Deep Learning framework to vision tasks. We now consider how the Deep Learning framework can be applied to the face recognition problem in Facebook's DeepFace system. DeepFace feature extraction pipeline consists of two phases: an alignment phase and a representation phase. The alignment phase involves explicit 3D modeling of the face based on fiducial points in order to warp an input facial image into a cropped 3D frontal mode, allowing DeepFace to learn from raw pixel RGB values. The network architecture of DeepFace is nine-layers deep and involves more than 120 million parameters. The first three layers of the neural network are used to extract low-level features such as edges and texture. The middle layers are locally connected and apply various filters to different regions based on local patterns. The final layers of the neural network are fully connected and are able to capture correlations between features in distant parts of the images. The output of running an image through the neural network is then used as the feature representation [6].

## 6 Unsupervised Clustering

In order to provide the best user experience, we will begin with unsupervised facial clustering techniques in order to provide automatic photo organization without requiring several manually annotated or tagged photos.

### 6.1 $k$-means Clustering

#### 6.1.1 Model Selection with CV using AIC

One metric we consider for determining the number of clusters (used as a parameter) for $k$-means clustering in the set of images is a modified form of the Akaike Information Criterion (AIC). We note that the reconstruction cost is a monotonically decreasing function in $k$, which is minimized when $k = n$. In other words, if we minimize reconstruction cost, the optimal clustering will place each image in its own cluster, which is clearly not desirable. Using AIC, however, we can impose a penalty for each additional cluster to penalize more complex models. Cross-validation using AIC optimizes:

$$k = \arg\min_k RC(k) + \lambda k$$

We note that a larger value of $\lambda$ will favor solutions with fewer clusters. Using the AIC metric, we have that $\lambda = 2M$, where $M$ is the number of features used to represent an image. For our application, however, we have found that experimentally using $\lambda \sim 0.05M$ yields a more appropriate penalty term due to the larger feature space $\mathbb{R}^{10000}$ used to represent the pixel grid of an image.

#### 6.1.2 Model Selection with Tuned Regularization Term

Since the cost function incurred by $k$-means monotonically decreases with the number of clusters, we append the following tuned regularization term which varies with the number of clusters:

$$R(k) = \theta^T \phi(k) \; for \; \phi(k) = [1 \; k \; k^2].$$

To tune the parameters $\theta$ for our regularization term, we first selected 100 random samples of 10 images from our dataset of images. Therefore, the correct number of clusters in each of these 100 examples ranged from 1 to 10. Then, we defined the following objective function to minimize by stochastic gradient descent:

$$min_\theta \sum_{i=1}^{m} \tfrac{1}{2} \arg\min_k (C(k) + R(k)) - k_i)^2 + \tfrac{1}{2}\lambda||\theta||_2^2$$

However, the presence of the argmin resulted in a discontinuous function for which the gradient could not be computed. Therefore, we reformulated our objective function to a continuous function minimized at the same $\theta$:

$$min_\theta \sum_{i=1}^{m} \tfrac{1}{2}(min_k[(C(k)+R(k))-(C(k_i)+R(k_i))]^2+\tfrac{1}{2}\lambda||\theta||_2^2$$

With this objective function, we applied stochastic gradient descent on the set of examples to tune our parameters $\theta$.

#### 6.1.3 Unsupervised Clustering with Eigenfaces

As an unsupervised clustering approach, we consider $k$-means with Eigenfaces. We first use the PCA algorithm known as Eigenfaces, described in Section 5.1, to extract principal components from a set of examples images that are independent of the actual images we wish to cluster. In our application, we use 7000 images from the Labeled Faces in the Wild face database to extract the eigenfaces that can be considered as representative of the space of all possible faces. We then project the face images that we wish to cluster into the space specified by the extracted eigenfaces.

Thus, each image is reduced to a feature vector of weights representing the contribution of each eigenface to the particular image. We can then run the $k$-means algorithm on the images using the reduced dimension feature vectors.

### 6.2 LBP

We also considered an approach using the local binary patterns histograms (LBPH) algorithm, which extracts local features of the image and is rooted in two-dimensional texture analysis.

#### 6.2.1 Model

We now describe the model of the LBPH algorithm. The algorithm functions by comparing each pixel with its neighborhood; if the center pixel'intensity is greater than or equal to that of its neighbors, then denote this relationship with a 0. Otherwise, denote it with a 1. More formally, this can be written as follows:

$$LBP(x_c, y_c) = \sum_{p=0}^{P-1} 2^P s(i_p - i_c),$$

where $(x_c, y_c)$ is the central pixel with intensity $i_c$, with $i_n$ being the intensity of the neighbor pixel. Then, $s$ would be the sign function defined as follows:

$$s(x) = \begin{cases} 1 & \text{if x} \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

The LBP image is divided into $m$ local regions (in our implementation, the LBP image is divided into 8x8 local regions), and a histogram is extracted from each; the spatially enhanced feature vector is obtained by concatenating the local histograms.

#### 6.2.2 Algorithm

We employ the following iterative algorithm:

1. The first face image is used to initialize the first cluster.
2. We manually preset a threshold for a confidence value at which a new cluster is created.
3. Subsequent images are then run through the face recognizer. If the confidence value is greater than our threshold, we instantiate a new cluster, otherwise we update the existing clusters.
4. The images within each cluster that obtained the highest confidence are considered representative samples for the clusters and are used to re-initialize the face recognizer model.
5. The algorithm repeats steps 2-5 for a few iterations until convergence.

Since the LBP algorithm extracts local features, we note that it is quite robust against monotonic gray scale transforms and thus limits the effects of confounding factors such as lighting.

## 7 Supervised Recognition

We imagine that at some point, once the system has clustered a fair number of face images, the previously unsupervised task of facial clustering could be transitioned into one of supervised facial recognition. Assuming that the system, has accurately clustered the previous stream of images, we can use the cluster assignments as labels to create a training set for supervised learning. We have considered two canonical supervised face recognition algorithms, namely Eigenfaces and Fisherfaces. For Eigenfaces, we use an SVM with eigenface features, and for Fisherfaces, we extract fisherfaces features and run nearest neighbors.

## 8 Results

### 8.1 Model Selection

We first begin by considering the efficacy of model selection for $k$-means clustering using AIC.

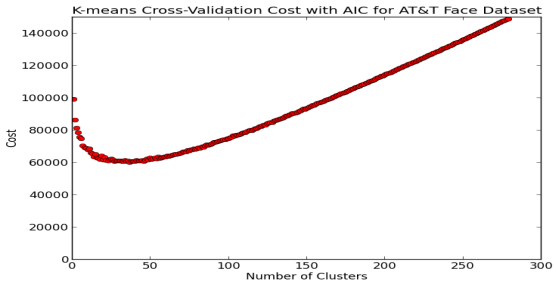


**Figure 3: $k$-means Cross-Validation Cost with AIC**

We find that running cross-validation with the modified cost function based on AIC generates representative models for the number of clusters in the training image set. We run the cross-validation algorithm described in Section 6.1.1 on two training sets. The training sets are generated from randomly sampling 70 percent of the AT&T and Yale face datasets. The AT&T dataset consists of 40 subjects and the Yale dataset consists of 15 subjects. Model selection using the modified cost function based on AIC estimates 36 subjects for the AT&T dataset and 18 subjects for the Yale dataset. We thus find that the AIC provides a reasonable and simple heuristic for initially tuning the unsupervised clustering of the initial photo album.

### 8.2 Unsupervised Clustering

To gain a better understanding of the efficacy of the LBPH algorithm, we consider two primary metrics: homogeneity, which describes the purity of a cluster, and completeness, which describes the purity of a class. We first consider the homogeneity and completeness scores when performing 30 trials of entirely unsupervised learning (with a training set of size 1) and testing on 50, 75, 100, 150, and 200 images from the AT&T Face Dataset.

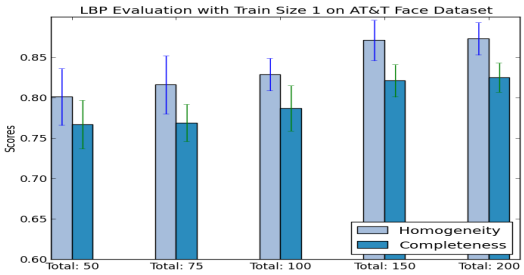


**Figure 4: LBP Evaluation with Train Size 1 on AT&T Face Dataset**

When performing entirely unsupervised clustering, we were able to achieve a high homogeneity score of 0.873 and completeness score of 0.825. We now consider the option in which we have the user correctly tag a small subset of photos; when performing 30 trials of LBP with a training set of size 25 correctly tagged images and testing on 50, 75, 100, 150, and 200 different images from the AT&T Face Dataset, we obtain the homogeneity and completeness scores pictured in the plot below.

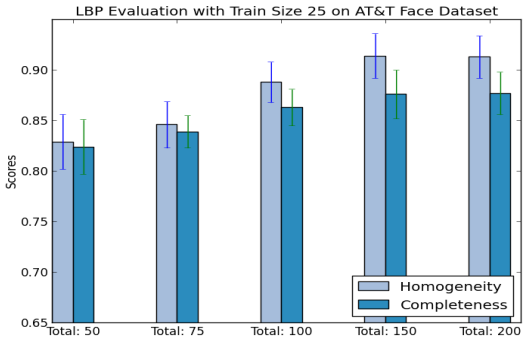


**Figure 5: LBP Evaluation with Train Size 25 on AT&T Face Dataset**

We see a general increase in LBP efficacy with a larger dataset, after which the scores seem to plateau. As expected, when correctly tagging 25 images, LBP performs achieves higher homogeneity and completeness scores; we were able to achieve a high homogeneity score of 0.914 and completeness score of 0.877.

As a means of unsupervised learning, we consider how the $k$-means algorithm performs on the Yale face dataset using different features.

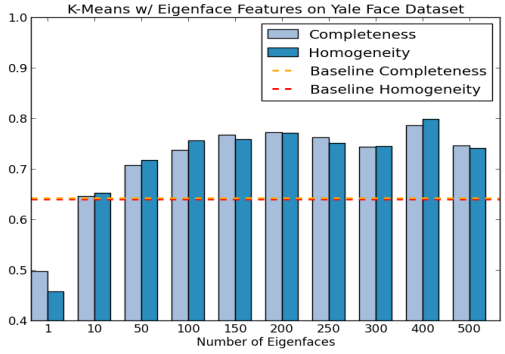


**Figure 6: $k$-means Baseline vs. Eigenface Features**

We assume that during the model selection phase that we are able to determine the correct number of subjects in the dataset. The results from the cross-validation experiment with AIC show that this assumption is not unreasonable. We then consider using $k$-means with Eigenface features. Using Eigenfaces, we are able to improve the clustering to a maximum homogeneity of 0.799 and completeness of 0. The 400 Eigenface features selected are representative of the LFW dataset and are assumed to generalize to the facial images that comprise the Yale dataset.
When we compare the LBP algorithm with unsupervised $k$-means clustering with eigenfaces, we see that the homogeneity and completeness scores are slightly higher with LBP for the AT&T and Yale Face Datasets, likely due to the algorithm's robustness against monotonic grayscale transformations.

### 8.3 Supervised Clustering

We first evaluate the eigenfaces algorithm. From the AT&T Face Database of 400 images containing 40 different subjects, we uniformly sample 70% of the images to construct our training data set and we test on the remaining 30% of the images. Preliminary results using a Gaussian kernel yield a precision of 96% and a recall of 93%. We perform a similar evaluation of Fisherfaces as that performed on Eigenfaces. From the AT&T Face Database of 400 images containing 40 different subjects, we uniformly sample 70% of the images to construct our training data set and we test on the remaining 30% of the images. Preliminary results yield a classification accuracy of 89%.

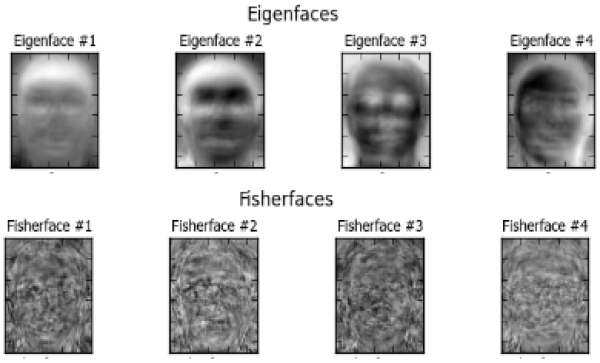


**Figure 7: 4 Eigenfaces & 4 Fisherfaces generated from AT&T Face Database**

For the supervised task of face recognition, we considered Eigenfaces and Fisherfaces. The experimental setup considered the AT&T Face Dataset and compared the Eigenface and Fisherface features on a hold out set that comprised a random sampling of 30 percent of the dataset.

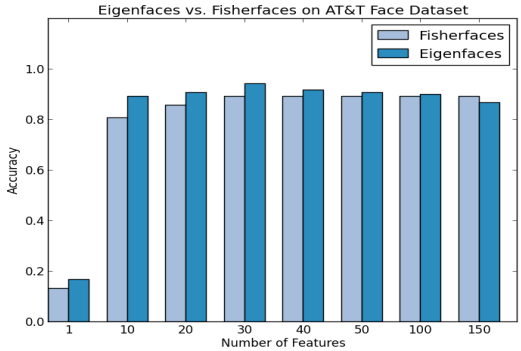


**Figure 8: Eigenfaces vs. Fisherfaces Comparison**

We note that the Fisherface accuracy plateaus after 40 features. This results from the fact that the AT&T dataset contains images of only 40 unique subjects and Fisherfaces uses a number of features that is at most the number of distinct labels. We decided to consider the Eigenface SVM for our supervised experiments since its high-mark accuracy

of 94.167% is higher than that of Fisherface, which is 89.1667%. To explore whether it would be possible to transition to supervised learning with inaccurate clusters, we randomly mislabel 10 percent of the dataset.

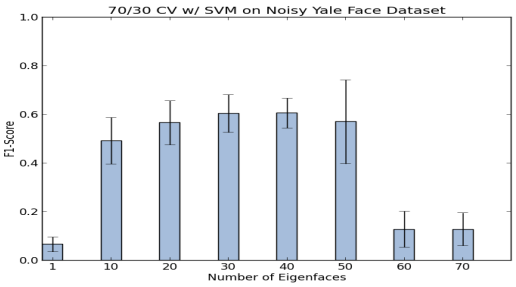


**Figure 9: SVM with Eigenface Features on Noisy Dataset**

We note that 10 percent noise is less than what would be generated by the unsupervised clustering algorithms we considered. Thus, it offers a reasonable baseline to consider whether supervised recognition on noisy clusters is feasible. The result of training on a randomly sampled set of 70 percent of the training data set and testing on the remaining 30 percent peaks at about an F1-Score of 60 percent. The F1-Score is the harmonic mean of precision and recall and we use it as a measure of a test's accuracy. From this result, we conclude that with the existing techniques considered, we cannot perform supervised recognition on noisy data. Alternatively, the user could provide the necessary feedback to correct the clusters.

To measure the performance of the SVM trained on Eigenface features, we trained on a randomly sampled set of 70 percent of the Yale data set and tested on the remaining 30 percent. We varied the number of Eigenface features extracted and experimented with both a linear kernel and a Gaussian kernel.

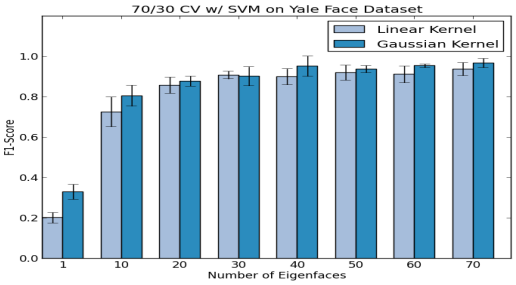


**Figure 10: SVM with Eigenface Features on Pure Dataset**

Experimentally, we find that the Gaussian kernel outperforms the linear kernel. We note that the high-mark F1-Score for the SVM using a Gaussian kernel incorporated 70 Eigenface features and achieved an F1-Score of 0.968. The high-mark for the linear kernel was 0.938 using 70 features as well. These results confirm the potential for transitioning to supervised recognition if the likelihood of adding a photo of an unseen person is low. If the user, however, is likely to add photos of new people, the supervised recognition could be used as an additional metric of confidence when evaluating whether a new photo belongs to an existing cluster. In our SVM, we note that the penalty parameter term C is weighted such that the penalty term is inversely proportional to class frequencies.

## 9 Literature

To date, Facebook's DeepFace algorithm, as described in Section 5.4, is one of the most accurate face recognition algorithms, achieving 97.35% accuracy on the Labeled Faces in the Wild (LFW) dataset [7]. Prior to Facebook's DeepFace algorithm, the most successful system using a large labeled face dataset adapted a joint generative Bayesian model learned on a dataset containing 99,773 images from 2,995 different subjects to the LFW image domain [6]. Another approach involves the use of a multi-task deep convolutional neural network which simultaneously learns the face-nonface decision, the face pose estimation problem, and the facial landmark localization problem (locating major features or landmarks of a face) [2]. Multi-task learning was applied to the neural network using a shared representation for the different problems by learning multiple targets and making them share the common lower layers. On the unsupervised front, a state-of-the-art algorithm that has been used for facial clustering is Over-Complete Local Binary Patterns (OCLBP), a multi-scale modified version of the LBP algorithm [4]. While LBP capitalizes on the locality of its feature extraction, OCLBP is computed with overlapping blocks and improves the robustness of classification systems by using richer descriptors.

## 10 Future Work

For the purposes of this project, we have focused primarily on the task of clustering photos that contain people; in the future, we would love to expand on this work by supporting user queries over non-people photos as well and clustering on criteria such as genre. We are also interested in investigating the tradeoffs between $k$-means clustering and LBP clustering for online unsupervised learning, using the SVM as an additional measure of confidence. Furthermore, we would like to attempt to extract other features including edges or corners. Lastly, we would like to experiment with local PCA and local ICA as well.

## 11 References

1. C. Zhang, Z. Zhang. Improving Multiview Face Detection with Multi-Task Deep Convolutional Neural Networks. In Applications of Computer Vision (WACV), 2014.

2. H. Akaike. Information theory and an extension of the maximum likelihood principle. Selected Papers of Hirotugu Akaike. Springer New York, 1998. 199-213.

3. O. Barkan, J. Weill, L. Wolf, and H. Aronowitz. Fast high dimensional vector multiplication face recognition. In ICCV, 2013.

4. P. Belhumeur, J. Hespanha, D. Kriegman. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. Pattern Analysis and Machine Intelligence, IEEE Transactions on 19.7 (1997): 711-720.

5. X. Cao, D. Wipf, F. Wen, G. Duan, and J. Sun. A practical transfer learning algorithm for face verification. In ICCV, 2013.

6. Y. Taigman, M. Yang, M. Ranzato, L. Wolf. DeepFace: Closing the Gap to Human-Level Performance in Face Verification. In Conference on Computer Vision and Pattern Recognition (CVPR), 2014.